LNAI 14946

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

Machine Learning and Knowledge Discovery in Databases

Research Track

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

6 Part VI

ECML PKDD 2024

Springer

MOREMEDIA

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 VI

Springer

Editors
Albert Bifet
LTCI
Télécom Paris
Palaiseau Cedex, France

Jesse Davis
KU Leuven
Leuven, Belgium

Tomas Krilavičius
Faculty of Informatics
Vytautas Magnus University
Akademija, Lithuania

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

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

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

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-031-70364-5 ISBN 978-3-031-70365-2 (eBook)
https://doi.org/10.1007/978-3-031-70365-2

LNCS Sublibrary: SL7 – Artificial Intelligence

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

Linas Petkevičius	Vilnius University, Lithuania

Workshop and Tutorial Chairs

Mantas Lukoševičius	Kaunas University of Technology, Lithuania
Mykola Pechenizkiy	Technische Universiteit Eindhoven, the Netherlands

Demo Chairs

Povilas Daniušis	Vytautas Magnus University, Lithuania
Kai Puolamäki	University of Helsinki, Finland

Proceedings Chairs

Wouter Duivesteijn	Technische Universiteit Eindhoven, the Netherlands
Rianne Schouten	Technische Universiteit Eindhoven, the Netherlands

PhD Forum Chairs

Virginijus Marcinkevičius	Vilnius University, Lithuania
Simona Ramanauskaitė	Vilnius Tech, Lithuania

Discovery Track Chairs

Peter van der Putten	Universiteit Leiden, the Netherlands
Jan N. van Rijn	Universiteit Leiden, the Netherlands

Workshop Proceedings Chairs

Danguole Kalinauskaite	Vytautas Magnus University, Lithuania
Kristina Šutiene	Kaunas Technology University, Lithuania

Social Media and Web Chairs

Julija Vaitonytė	Tilburg University, the Netherlands
Kamilė Dementavičiūtė	Vilnius University, Lithuania

Sponsorship Chairs

Mariam Barry	BNP Paribas, France
Dalia Breskuvienė	Vilnius University, Lithuania
Daniele Apiletti	Politecnico di Torino, Italy

Diversity and Inclusion Chair

Rūta Binkytė-Sadauskienė	Inria, France

Industry Track Chairs

Pieter Van Hertum	ASML, the Netherlands
Bjoern Bringmann	Deloitte, Germany

Nectar Track Chairs

Heitor Murilo Gomes	Victoria University of Wellington, New Zealand
Jesse Read	École Polytechnique, France

Awards Chairs

Michele Sebag	CNRS, France
João Gama	University of Porto, Portugal

ECML PKDD Steering Committee

Tijl De Bie	Ghent University, Belgium
Francesco Bonchi	ISI Foundation, Italy
Albert Bifet	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 Allmendringer	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

Andrea Paudice	University of Milan, Italy
Benjamin Noack	Otto-von-Guericke University Magdeburg, Germany
Slawomir Nowaczyk	Halmstad University, Sweden
Vincenzo Pasquadibisceglie	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 Diethe	AstraZeneca, UK
Kurt Driessens	Maastricht University, the Netherlands
Wouter Duivesteijn	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 Krempl	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

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 Baltsou	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 Beigaite	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	Universtiät 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 Coquenet	IRISA, France
Luca Corbucci	University of Pisa, Italy

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 Masseglia	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 Mercorio	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
Linas 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

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 Staufer	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 Zemmari	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	Malmo 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
Lingyang 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 Grahn	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

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

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 VI

Research Track

Rejection Ensembles with Online Calibration

Sebastian Buschjäger(✉)

The Lamarr Institute for Machine Learning and Artificial Intelligence, TU Dortmund University, Dortmund, Germany
sebastian.buschjaeger@tu-dortmund.de

Abstract. As machine learning models become increasingly integrated into various applications, the need for resource-aware deployment strategies becomes paramount. One promising approach for optimizing resource consumption is rejection ensembles. Rejection ensembles combine a small model deployed to an edge device with a large model deployed in the cloud with a rejector tasked to determine the most suitable model for a given input. Due to its novelty, existing research predominantly focuses on ad-hoc ensemble design, lacking a thorough understanding of rejector optimization and deployment strategies. This paper addresses this research gap by presenting a theoretical investigation into rejection ensembles and proposing a novel algorithm for training and deploying rejectors based on these novel insights. We give precise conditions of when a good rejector can improve the ensemble's overall performance beyond the big model's performance and when a bad rejector can make the ensemble worse than the small model. Second, we show that even the perfect rejector can overuse its budget for using the big model during deployment. Based on these insights, we propose to ignore any budget constraints during training but introduce additional safeguards during deployment. Experimental evaluation on 8 different datasets from various domains demonstrates the efficacy of our novel rejection ensemble outperforming existing approaches. Moreover, compared to standalone large model inference, we highlight the energy efficiency gains during deployment on a Nvidia Jetson AGX board.

Keywords: Ensemble Learning · Learning with Rejection · Resource-aware Machine Learning

1 Introduction

In recent years, the pervasive integration of machine learning models into various applications has underscored the importance of resource-aware deployment. Most

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_1.

famously, Deep Learning is one of the most resource-hungry technologies available, and therefore, a large body of the literature tries to improve the resource usage of Deep Learning (see [19] for a recent overview). Similarly, approaches for improving the resource usage of non-Deep Learning approaches, such as Random Forests [3,4] or graphical models [21], have also been discussed in the literature. Last, as ML models find their way into critical decision-making processes across diverse domains, there is a growing need for strategies that balance fast model application and the opportunity for human model inspection. This, again, leads to a resource-accuracy trade-off. For example, consider a medical scenario in which a machine-learning model autonomously diagnoses patients. Naturally, such a model will not always be correct, and human supervision and intervention are sometimes necessary. Hence, we have to balance human supervision and autonomous predictions during deployment.

One area of research reduces resource consumption through the fusion of small and large models into an ensemble coupled with a rejector (sometimes called a router) that determines the most suitable model for a given input [2,8,12]. Such a rejection ensemble first applies the small model alongside the rejector and, if the rejector accepts the output of the small model, serves it. If the rejector rejects the small model's prediction, it also queries the big model (e.g., the doctor in the previous example) for additional help. Such an approach can reduce the overall resource consumption of the system if the small model is used most of the time while maintaining competitive predictive performance due to the big model.

The current literature on rejection ensembles mostly focuses on the ad-hoc design of rejection ensembles. The design and training of a good rejector is difficult and poorly understood. Moreover, while a budget is typically introduced during training to capture how often the big model can be queried, it is no longer considered during deployment. Hence, a deployed system might query the big model too often (i.e., overuse its budget) and – in the worst case – only query the big model if no additional safeguards are employed.

To address these issues, this paper presents the first theoretical investigation of learning rejection ensembles and derives practical insights from it. To this end, we propose a novel algorithm for training the rejector based on our theoretical insights and introduce a novel algorithm that ensures that the budget is always kept during deployment. More precisely, our contributions are as follows:

– **Theoretical Investigation:** We offer a thorough theoretical investigation of the impact of the rejector on the ensemble. We give precise conditions of when a good rejector can improve the ensemble's overall performance beyond the big model's performance and when a bad rejector can make the ensemble worse than the small model. Second, we show that even the perfect rejector can overuse its budget during deployment. Third, we give an example of when a rejector should not trust the outputs of the small and big models but learn its own decision boundary based on the input data.
– **Novel Algorithm:** Based on our theoretical investigation, we propose to ignore any budget constraints during training but introduce additional

safeguards during deployment. For training the rejector, we introduce a novel training algorithm based on so-called virtual labels capturing when to use the small and when to use the big model. For deployment, we introduce safeguards that essentially rank the rejector's output during deployment and ensure we always adhere to the prediction budget.
- **Experimental Evaluation:** We experimentally evaluate our proposed algorithm on 8 datasets from various domains and execute it on a Nvidia Jetson AGX board. We show that our novel rejection ensemble outperforms other ensembles while keeping the budget during deployment. Moreover, we highlight that these rejection ensembles use less energy during deployment compared to simply running the big model. The code for these experiments is available under https://github.com/sbuschjaeger/rewoc.

This paper is organized as the following: Sect. 2 introduces the notation and related work. Section 3 presents our main theoretical findings, whereas Sect. 4 translates these into a practical algorithm. Section 5 then discusses the experiments, whereas Sect. 6 concludes the paper.

2 Notation and Related Work

We consider a supervised classification setting in which training and test points are drawn i.i.d. according to some distribution $\mathcal{D}$ over the input space $\mathcal{X} \subseteq \mathbb{R}^d$ of d-dimensional feature vectors and labels $\mathcal{Y} = \{1, \dots, C\}$, where C is the number of classes. We are interested in a classifier triplet that we call *Rejection Ensemble*:

$$f(x) = (f_s, f_b, r)\,(x) = \begin{cases} f_s(x) & \text{if } r(x) = 0 \\ f_b(x) & \text{else} \end{cases} \tag{1}$$

Conceptually, $f_s\colon \mathcal{X} \to \mathcal{Y}$ is a small model whose predictions can be easily explained by a human and/or a model that does not use many resources, e.g. a Decision Tree. Similarly, $f_b\colon \mathcal{X} \to \mathcal{Y}$ is a big model whose predictions cannot be easily explained, and its execution might require many resources, such as e.g. a large neural network running in the cloud. Naturally, we want to use the small model as often as possible to make predictions explainable and the overall system more resource-efficient. At the same time, a small model might not be powerful enough to provide (good) predictions for certain inputs. Hence, we use a rejection function $r\colon \mathcal{X} \to \{0, 1\}$ that outputs 1 if we should reject the small model's prediction and use the big model instead. For training the triplet (f_s, f_b, r), we have given a (user-defined) budget[1] $p \in [0, 1]$, which defines how often we can query the big model. For example, a budget of $p = 1$ means we can always query the big model, a budget of $p = 0$ means we should never query

[1] Sometimes this is called the coverage, if there is no big model available and the small model abstains from a prediction.

it, and everything in between allows for some queries of the small and the big model. Given a loss $\ell : \mathbb{R}^C \times \mathcal{Y} \rightarrow \mathbb{R}_{\geq 0}$ our goal is to find a model such that

$$f^* = (f_s^*, f_b^*, r^*) = \arg\min_f \mathbb{E}_{\mathcal{D}}[\ell(f(x), y)] \text{ s.t. } \mathbb{E}_{\mathcal{D}}[r(x)] \leq p \tag{2}$$

Since $\mathcal{D}$ is typically unknown, we use a labeled training dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^m$ to approximate Eq. 2 with its empirical counterpart:

$$f^* = (f_s^*, f_b^*, r^*) = \arg\min_f \frac{1}{m} \sum_{i=1}^m \ell(f(x_i), y_i) \text{ s.t. } \frac{1}{m} \sum_{i=1}^m r(x_i) \leq p \tag{3}$$

For convenience, we further define the predictions of f_s, f_b, f as y_s, y_b, y_f and the corresponding confidences as c_s, c_b, c_f:

$$y_s(x) = \arg\max_{j=1,\ldots,C} f_s(x)_j, \; y_b(x) = \arg\max_{j=1,\ldots,C} f_b(x)_j, \; y_f(x) = \arg\max_{j=1,\ldots,C} f(x)_j$$
$$c_s(x) = \max_{j=1,\ldots,C} f_s(x)_j, \; c_b(x) = \max_{j=1,\ldots,C} f_b(x)_j, \; c_f(x) = \max_{j=1,\ldots,C} f(x)_j$$

2.1 Related Work

Several approaches in the literature focus on classification with a reject option. Arguably, the largest collection of works focuses on training a classifier tuple (f, r) where f is the prediction model and r is the rejector. The rejector can output a designated `REJECT` token, meaning that the model's prediction should be ignored. In this setting, the rejector is chosen such that f covers a certain percentage of the input space, and f is chosen to maximize the classification performance on the covered subspace. This way, a trade-off between the (likely better) performance of f on smaller subspaces and maximizing the coverage through r is introduced. The first works [5,6] in this area introduce a cost model (c.f. [23]) that balances the costs of rejection and its miss-classification. Subsequent theoretical works in this direction further refine this idea by developing new loss functions based on the hinge loss [1], introducing Bayes consistent loss functions [18], and studying the Rademacher complexity of such a classifier pair [7]. A second line of research is introduced in [22], sometimes called the bounded improvement model (c.f. [23]), which does not assign specific costs to rejection but views training a pair (f, r) as a min-max problem in which the goal is to maximize coverage while minimizing the error. A recent example is presented in [10], which introduces a novel 'selection with a guaranteed risk' algorithm that dynamically adjusts the bounds for confidence scoring of a pre-trained classifier. Similarly, SelectiveNet [11] is a neural network architecture that includes a reject option and is trained based on a convex combination of classification and coverage loss. Finally, some works in the literature also discuss the joint training of the model and the rejector without explicitly considering costs or coverage. For example, [15] studies the joint learning of the rejector and classification model by drawing inspiration from portfolio theory. Here, the authors introduce the

REJECT token as an additional class and a novel information-centric loss function that uses the REJECT token for better optimization. Last, Madras et al. study an edge case similar to our setting in [17], in which the small model can PASS an observation to a domain expert. However, their study focuses on fairness and does not introduce a budget constraint for optimization.

The framework presented in this paper extends prior work by using a classifier triplet (f_s, f_b, r) instead of a classifier tuple. Clearly, this generalization recovers the classification with rejection framework by setting the big model to output a constant reject value $f_b(x) = \texttt{REJECT}$ so that whenever $r(x) = 1$, we output the REJECT token. Training such a triplet is much more difficult because we cannot rely on the coverage as a guideline: When training a classifier with a reject option, we can essentially ignore parts of the input space by training the rejector r accordingly. However, when training a triplet, rejecting an observation means that the big model is tasked to provide a prediction. Hence, the rejector must take the shortcomings of the small and the big models into account and *only* transfer those samples to the big model when it can be sure that the big model will likely answer correctly to minimize the overall resource consumption. To our knowledge, only three articles in the literature utilize this more general framework. In [12], the authors propose a novel hybrid learning method in which a triplet is trained using a Frank-Wolfe-style algorithm. First, they start with a random rejector, which assigns training examples to a large and a small Deep Neural Network. After these two models are trained, the rejector is updated based on the overall performance, and the process is then repeated until convergence. Notably, the rejector only receives the outputs of the small model as input to minimize resource consumption. The resulting hybrid system achieves better ImageNet accuracy over inference latency and energy used. In [2,8], the authors deal with the problem of Human Activity Recognition by using a similar hybrid system that combines models from different model classes, i.e., a decision tree (DT) and a CNN. In [8], the authors introduce a multi-step learning process that first fits a decision tree on the entire task (i.e., all samples with all labels) and then iteratively merges the labels of samples that are too difficult in a common FALLBACK class. Finally, a DT is trained on all unchanged 'easy' samples and all difficult samples with the novel FALLBACK class assigned to them, whereas a CNN is trained on all available difficult samples with their original classes. While [8] shows the feasibility of this approach on a microcontroller unit (MCU), [2] goes one step further and deploys the DT model to the sensor directly by leveraging in-sensor computation and only executing the CNN on the MCU when necessary.

3 A Theoretical Investigation of Rejection

The adaption of Rejection Ensembles shows promising successes in practice [2,8,12]. We extend this work by investigating the theoretical properties of Rejection Ensembles with a particular focus on the rejector r. To this end, we assume that the small model f_s and the big model f_b are already trained and given to us for deployment. We assume that f_b generally performs better than f_s, but we do not have any special requirements towards f_s and f_b. In particular, our discussion here does not assume any special model class or training algorithm for the small and big models. Last, we assume that during deployment we receive data in batches $\mathcal{T} = \{x_1, \ldots, x_N\}$ of N data points for prediction and our task is to provide a labeled set $\{(x_1, f(x_1)), \ldots, (x_M, f(x_N))\}$. We highlight three theoretical insights about the rejector in this setting.

3.1 Three Distinct Situations Can Occur When Training the Rejector

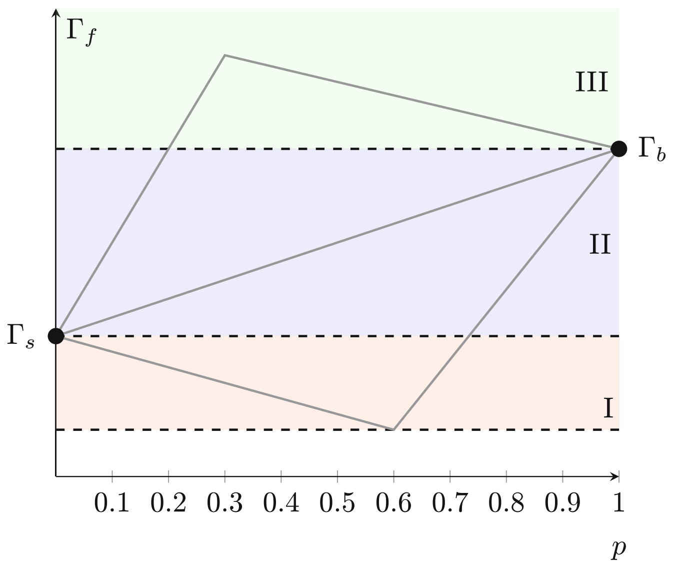


Fig. 1. Number of correctly classified samples over the rejection rate p for three archetypical examples. $\Gamma_{\{f,s,b\}}$ denotes the number of correctly classified samples of the ensemble and the small and big model respectively. The green area III marks improved performance in which the ensemble outperforms the big model while being more resource-efficient. The blue area II marks an accuracy-resource trade-off in which the ensemble underperforms compared to the big model but uses fewer resources. Finally, the red area I marks where the combined model performs worse than the small model and is more resource-hungry. The example curves in grey illustrate three different archetypical behaviors, although they highly depend on the specific models, data, and task. (Color figure online)

We identify three distinct situations that can occur when training a rejector: 1) A bad rejector can destroy the performance of both models by always choosing the wrong model for a given sample. For example, if the small model is correct, it might choose the big one instead, and if the big one is correct, it chooses the small model. 2) If the big model is always better than the small model, i.e., $\forall x, y \sim \mathcal{D} : \ell(f_b(x), y) \leq \ell(f_s(x), y)$ then the Rejection Ensemble will never be better than the big model. However, the rejector can try to find situations in which the small model makes the same prediction $(y_s(x) = y_b(x))$ as the big model. Compared to f_b this will not increase the overall accuracy, but it decreases resource consumption while maintaining a comparable performance. 3) Both models complement each other through the rejector. Whenever f_s would be wrong, the rejector uses f_b instead, and whenever f_s is correct (and f_b might be wrong), f_s is used. This way, the overall accuracy might exceed the accuracy of f_b, resulting in a better *and* more resource-friendly ensemble. An illustration of these three cases is depicted in Fig. 1 and Theorem 1 formally establishes the conditions for each case.

Theorem 1. *Consider a binary classification problem with $C = 2$. Let (f_s, f_b, r) be a rejection ensemble with budget p and let $\mathcal{S} \sim \mathcal{D}^m$ be a sample with m data points. Define the following:*

$$\varGamma_s = \sum_{(x,y)\in\mathcal{S}} \mathbb{1}\{y_s(x) = y\}, N_{s\bar{b}} = \sum_{(x,y)\in\mathcal{S}} \mathbb{1}\{y_s(x) = y \neq y_b(x)\}$$

$$\varGamma_b = \sum_{(x,y)\in\mathcal{S}} \mathbb{1}\{y_b(x) = y\}, N_{\bar{s}b} = \sum_{(x,y)\in\mathcal{S}} \mathbb{1}\{y_b(x) = y \neq y_s(x)\}$$

$$\varGamma_f = \sum_{(x,y)\in\mathcal{S}} \mathbb{1}\{y_f(x) = y\}, N_{sb} = \sum_{(x,y)\in\mathcal{S}} \mathbb{1}\{y_s(x) = y_b(x) = y\}$$

Let $P = \lfloor p \cdot m \rfloor$, then the following holds:

1. *For all rejectors the following lower bound holds: $N_{sb} + \max\{N_{s\bar{b}} - P, 0\} \leq \varGamma_f$*
2. *If $\varGamma_b - P \geq \varGamma_s$, then $\varGamma_s$ is a lower bound for $\varGamma_f$, i.e. $\varGamma_s \leq \varGamma_f$*
3. *If $\varGamma_b \leq \varGamma_s + P$, then $\varGamma_b$ is a lower bound for $\varGamma_f$, i.e. $\varGamma_b \leq \varGamma_f$*

Proof. It holds that $\varGamma_s = N_{sb} + N_{s\bar{b}}$ and $\varGamma_b = N_{sb} + N_{\bar{s}b}$. Further, it holds that

$$\varGamma_f = N_{sb} + N_{s\bar{b}} + \min\{P, N_{\bar{s}b}\} = N_{sb} + N_{s\bar{b}} + \min\{P, \varGamma_b - N_{sb}\}$$

since we can only use the big model up to P times and have to revert to the small model otherwise. Given this, we proof each statement separately:

1. A worst-case rejector would always choose the big model when the small model is correct and vice versa. For those samples on which both models agree this is impossible, i.e., N_{sb} is a trivial lower bound for the performance of the worst rejector. Moreover, due to the budget, the rejector can only choose the big model up to P times, leading to

$$N_{sb} + \max\{N_{s\bar{b}} - P, 0\} \leq \varGamma_f$$

2. We want to show that $\varGamma_s$ is a lower bound for $\varGamma_f$ given $\varGamma_b - P \geq \varGamma_s$:

$$\begin{aligned}&\varGamma_s \leq \varGamma_f \Leftrightarrow \varGamma_s < N_{sb} + N_{s\bar{b}} + \min\{P, \varGamma_b - N_{sb}\} = \varGamma_s + \min\{P, \varGamma_b - N_{sb}\}\\ &\Leftrightarrow 0 \leq \min\{P, \varGamma_b - N_{sb}\}\end{aligned}$$

Note that $0 \leq P$ is true by definition of P. Similarly, $0 \leq \Gamma_b - N_{sb} \Leftrightarrow N_{sb} \leq \Gamma_b$ is true due to the assumption $\Gamma_b - P \geq \Gamma_s$.

3. We want to show that Γ_b is a lower bound for Γ_f given $\Gamma_b < \Gamma_s + P$:

$$\Gamma_b \leq \Gamma_f \Leftrightarrow \Gamma_b < N_{sb} + N_{s\bar{b}} + \min\{p, \Gamma_b - N_{sb}\} = \Gamma_s + \min\{P, \Gamma_b - N_{sb}\}$$

We check both min$-$ cases individually:

$$\Gamma_b \leq \Gamma_s + \Gamma_b - N_{sb} \Leftrightarrow N_{sb} \leq \Gamma_s$$
$$\Gamma_b \leq \Gamma_s + P \Leftrightarrow \Gamma_b - P \leq \Gamma_s$$

The first case $N_{sb} \leq \Gamma_s$ always holds by definition of N_{sb} and Γ_s, whereas the second case $\Gamma_b - P \leq \Gamma_s$ holds by assumption. Hence, if $\Gamma_b - P \leq \Gamma_s$ holds, then $\Gamma_b \leq \Gamma_f$. □

Theorem 1 shows that both a good rejector and well-trained models are crucial for good performance. Surprisingly, it also implies that the small model is much more important for the overall performance because it will be queried most of the time. The big model only needs to perform well on those $P = \lfloor p \cdot m \rfloor$ data points on which the small model underperforms and hence has a much smaller impact overall. Therefore, a good rejector should always favor the small model as much as possible and carefully pick those P samples.

3.2 Even a Perfect Rejector Will Overuse Its Budget

Recall that the rejector in [2,8,12] is trained to pick the big model at most p times on average during training. Unfortunately, there is no guarantee that the rejector will satisfy this constraint during deployment without further safeguards. Consider a deployed model (f_s, f_b, r), and a given batch $\mathcal{T}$ of N samples that should be classified. Since during training $\mathbb{E}_{\mathcal{S}}[r(x)] \leq p$ we hope that during deployment it also holds that $\mathbb{E}_{\mathcal{T}}[r(x)] \leq p$ but clearly we can construct corner cases in which this is not true. More critically, even if the estimation of $\mathbb{E}_{\mathcal{D}}[r(x)] = \mathbb{E}_{\mathcal{S}}[r(x)]$ during training is perfect and we have the perfect rejector with $\mathbb{E}_{\mathcal{D}}[r(x)] \leq p$, then there is still a non-zero chance to find a sample $\mathcal{T}$ that forces us to overuse the big model breaking our budget p, i.e., a sample with $\mathbb{E}_{\mathcal{T}}[r(x)] > p$. Theorem 2 formalizes this insight. Its proof utilizes the fact that $\mathbb{E}_{\mathcal{T} \sim \mathcal{D}^N}[r(x)]$ is normally distributed around $\mathbb{E}_{\mathcal{D}}[r(x)]$ due to the central limit theorem. Conversely, for $N < \infty$, the standard deviation of this normal distribution is larger than 0 so that there is a non-zero chance to find values above the mean $\mathbb{E}_{\mathcal{D}}[r(x)]$.

Theorem 2. *Assume we have given the Bayes optimal classifier* $(f_s^*, f_b^*, r^*) = \arg\min_f \mathbb{E}_{\mathcal{D}}[\ell(f(x), y)]$ *s.t.* $\mathbb{E}_{\mathcal{D}}[r(x)] \leq p$ *and* $\mathbb{E}_{\mathcal{D}}[r(x)] \in (0, 1)$. *Then there exists a sample* $\mathcal{T} \sim \mathcal{D}^N$ *such that* $\mathbb{E}_{\mathcal{T}}[r(x)] = \frac{1}{N}\sum_{i=1}^{N} r(x_i) > p$.

Proof. Consider a sample $\mathcal{T} \sim \mathcal{D}^N$ and the empirical mean $\widehat{p} = \frac{1}{N}\sum_{i=1}^{N} r(x_i)$. For a sufficiently large N the empirical mean $\widehat{p}$ is normal distributed due to the central limit theorem with $\widehat{p} \sim \mathcal{N}(\mu, \sigma)$ where

$$\mu = \mathbb{E}_{\mathcal{D}}[r(x)]$$
$$\sigma = \frac{\mathbb{V}_{\mathcal{D}}[r(x)]}{\sqrt{N}} = \frac{\mu(1-\mu)}{\sqrt{N}}$$

Let Φ denote the CDF of a Gaussian distribution. Then there is a non-zero probability $P(\widehat{p} > p) = \Phi(\widehat{p}) > 0$, given $M < +\infty$ and $\mu(1-\mu) \neq 0$ which holds due to the assumption that $\mu = \mathbb{E}_{\mathcal{D}}[r(x)] \in (0,1)$. Hence, there exists a sample $\mathcal{T}$ such that $r(x)$ is queried more often than p. □

3.3 A Rejector Should Not Trust f_s and f_b

Arguably, the most straightforward rejector that always adheres to the budget p only selects the big model up to $P = \lfloor N \cdot p \rfloor$ times during deployment. In this case, we do not necessarily need to train a rejector, as we could simply trust the small model's outputs to determine if it is in doubt about a sample or not. More formally, we query the small model for all N data points, sort them according to a confidence score (e.g., the model's uncertainty), and then select those P data points with the smallest scores to be predicted by the big model. The pseudocode for this algorithm is depicted in Algorithm 3.1, where $\mathcal{T}$ is the current batch and p is the budget. For general applicability (we will re-visit Algorithm 3.1 in the next section), we explicitly include the rejector r as a parameter. However, note that using the confidence scores of the small model as rejector means we set $r = f_s$ in the parameters, i.e., use `confidence_thresholding`$(f_s, f_b, f_s, \mathcal{T}, p)$ so that no rejector r is necessary.

Implicitly, this algorithm trusts that the small model can express its confidence accurately. Unfortunately, this is not guaranteed without further assumptions on f_s, and we argue that any rejector that blindly trusts the output probabilities of a model can be fooled. And indeed, we can easily construct a simple counter-example in which the big model is a decision tree of depth 2, which is overly confident (but wrong), and the small model is a decision tree of depth 1, which is uncertain (but correct) for some samples. Theorem 3 formally established this argument and shows that – in the worst case – the lower bound in Theorem 2 can be realized.

Algorithm 3.1: Confidence Thresholding.

```
1 Function confidence_thresholding($f_s$, $f_b$, $r$, $\mathcal{T}$, $p$):
2     $s \leftarrow (c_r(x_1), \ldots, c_r(x_N))$
        // Confidences of $r$
3     $s, x \leftarrow \text{sorted}(s, x)$      // Sort confidences ascending
4     $P \leftarrow \lfloor N \cdot p \rfloor$              // Set cutoff
5     return $\{(x_i, y_s(x_i)\mathbb{1}\{c_r(x_i) > s_P\} + y_b(x_i)\mathbb{1}\{c_r(x_i) \leq s_P\})\}_{i=1}^{N}$
```

Theorem 3. *Given two models f_s, f_b, a batch $\mathcal{T}$ of N data points, and a budget $p \in [0,1]$, then confidence thresholding of the small model f_s, i.e., using* `confidence_thresholding`*($f_s, f_b, f_s, \mathcal{T}$,$p$), can have a performance that matches the lower bound 1 in Theorem 1, i.e.*

$$\Gamma_f = N_{sb} + \max\{N_{s\bar{b}} - P, 0\}$$

Proof. We give an example situation with two decision trees. Consider a one-dimensional example $x \in [0,10]$ with two classes $y \in \{0,1\}$ that are assigned by the following rule:

$$y = \begin{cases} 0 & \text{if } x \le 5 \\ 1 & \text{if } 5 < x \le 7.5 \\ 0 & \text{if } 7.5 < x \end{cases}$$

We have gathered the following training data: $\mathcal{S} = \{(3, 0.2), (3.5, 0.2), (4, 0.2), (6, 0.2), (6.5, 0.2), (8, 0.2), (8.5, 0.2), (9, 0.2), (9.5, 0.2)\}$. We trained two trees f_b and f_s as depicted in Fig. 2 using the CART algorithm. Clearly, f_b performs better most of the time than f_s. However, for any $x \in (7.25, 7.5)$, it is very confident in its prediction ($f_b(x) = 1$) and f_s is comparably unconfident ($f_s = 4/6$). Now consider a batch of N data points that fall exactly in the region of $x \in (7.25, 7.5)$. Here, we should always pick the small model because – although unconfident – it is correct. However, `confidence_thresholding` picks at least $P = \lfloor Np \rfloor$ predictions from the big model, leading to a performance of $\Gamma_f = N_{sb} + N_{s\bar{b}} - P$.

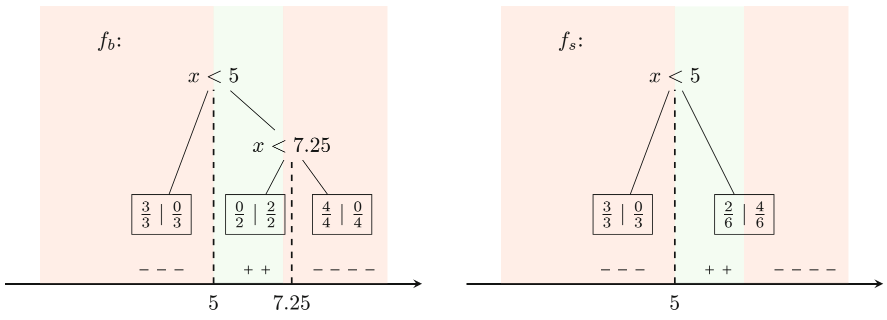

f_b:
$x < 5$
$x < 7.25$
$\frac{3}{3} \mid \frac{0}{3}$
$\frac{0}{2} \mid \frac{2}{2}$
$\frac{4}{4} \mid \frac{0}{4}$
$- - -$
$+ +$
$- - - -$
5
7.25
f_s:
$x < 5$
$\frac{3}{3} \mid \frac{0}{3}$
$\frac{2}{6} \mid \frac{4}{6}$
$- - -$
$+ +$
$- - - -$
5

Fig. 2. Two decision trees trained on the sample $\mathcal{S} = \{(3, 0.2), (3.5, 0.2), (4, 0.2), (6, 0.2), (6.5, 0.2), (8, 0.2), (8.5, 0.2), (9, 0.2), (9.5, 0.2)\}$. The color-shaded area marks the correct class, and the samples are depicted as '-' and '+'. Each leaf node shows the empirical class probabilities for both classes, where the left entry represents the negative '-' class and the right entry represents the positive class '+'.

□

4 Training a Rejector for a Rejection Ensemble

The previous section discussed the theoretical properties of a (good) rejector, which we summarize as follows: First, a good rejector should not (blindly) use the confidence scores of the small model (c.f. Theorem 3). Hence, we argue against training the rejector on the outputs of the small model (i.e., $f_s(x)$) but propose using the input data x or intermediate transformations such as, e.g., embeddings of a Neural Network derived from f_s directly. Second, a rejector will likely overuse its budget during deployment. Hence, if the budget is a hard constraint, we must employ additional safeguards (c.f. Theorem 2). Therefore, we argue that we can simplify the training of r by ignoring any budget constraints during training, but we add additional safeguards during deployment that handle these constraints. Third, a rejector can improve the overall performance over f_b if f_s performs well and is sufficiently different from f_b. Hence, the training of r should favor the small model in all cases where it is correct and only use f_b if f_s is incorrect (c.f. Theorem 1). To this end, we propose Algorithm 4.1 for training a rejection ensemble. It receives the models f_s and f_b and applies them to the given training set $\mathcal{S}$. Then, it generates virtual labels that are 1 if both models disagree and the big model is correct. Otherwise, it assigns the label 0 to a sample. Finally, the rejector is trained on the original observations (or intermediate representations if available) with the new labels.

Algorithm 4.1: Training of Rejection Ensembles via virtual labels.

```
Function fit($f_s$, $f_b$, $\mathcal{S}$):
    $\mathcal{V} = \emptyset$                        // Virtual Labels for $r$
    for $i = 1, \dots, m$ do
        if $y_s(x_i) = y_b(x_i)$ then
            $v_i \leftarrow 0$              // Both models agree. Pick the small model
        else
            if $y_b(x_i) = y_i$ then
                $v_i \leftarrow 1$          // Big model is correct, pick it.
            else
                $v_i \leftarrow 0$          // Big model is wrong. Use the small model.
            end if
        end if
        $\mathcal{V} \leftarrow \mathcal{V} \cup \{(x_i, v_i)\}$
    end for
    // Train $r$ by minimizing $\ell$ over $\mathcal{V}$. $f_s, f_b$ do not change
    $r \leftarrow \arg\min_r \frac{1}{m} \sum_{(x,v)\in\mathcal{V}} \ell(r(x), v)$
    return $(f_s, f_b, r)$
```

Ensuring that the rejector satisfies the budget constraint during deployment is more challenging. For clarity, we now assume that the rejector outputs a confidence score, i.e., it is a function $r\colon \mathcal{X} \rightarrow [0, 1]$. Then we can use Algorithm 3.1 to sort the confidence scores of the rejector in ascending order and only use the big model up to $\lfloor Np \rfloor$ times, i.e. using `confidence_thresholding`$(f_s, f_b, r, \mathcal{T}, p)$. Notably, this algorithm now assumes that $r(x)$ reflects the propensity of the rejector to favor the big model instead of trusting f_s. We argue that this is a more favorable scenario, as we can focus our energy entirely on training a good rejector instead of training three models at once.

5 Experiments

We now experimentally validate our findings in Theorem 1 and Theorem 2. Further, we show that training via virtual labels and online calibration outperforms existing methods. To do so, we perform two sets of experiments on the datasets listed in Table 1. The first experiment uses Deep Learning models evaluated on CIFAR100 and ImageNet, whereas the second experiment uses decision trees evaluated on several UCI datasets. We compare four different methods: For our baseline, we follow the established approaches of [2,8,12] by using confidence scores for training the rejector. More formally, we apply the small model to the training data and sort it according to the confidence score of the small model. Then, we assign a 1-label to the $\lfloor Np \rfloor$ examples with the smallest scores, whereas the remaining samples receive a 0-label. Finally, we train the rejector with these new labels. Note that, by construction, this approach satisfies the budget for the training data if the rejector is sufficiently accurate. We call this approach *confidence labels*. As a variation of this baseline, we combine a rejector trained via confidence labels with Algorithm 3.1, i.e., with confidence calibration, and call this method *confidence calibrated*. Third, we use Algorithm 4.1 to train the rejector and call this approach *virtual labels*, and finally, we combine Algorithm 4.1 and Algorithm 3.1 into *virtual labels calibrated*. The code for these experiments is available under https://github.com/sbuschjaeger/rewoc. Additional ablation studies on the UCI datasets can be found in the full version of the paper available in the code repository.

Table 1. Datasets used for the experiments.

Dataset	# Samples	Dimensionality	# Classes
ImageNet [24]	50 000	$3 \times 224 \times 224$	1 000
CIFAR 100 [14]	10 000	$3 \times 32 \times 32$	100
Anuran [13]	7 195	22	10
Covertype [13]	581 012	54	7
EEG [13]	14 980	14	2
Elec [13]	45 312	14	2
Gas Drift [13]	13 910	128	6
Weather [13]	18 159	8	2

As mentioned earlier, we are interested in an energy-efficient deployment in real-world scenarios. To measure the energy improvement under real-world circumstances, we perform all experiments on an Nvidia Jetson Orin AGX board. The Nvidia AGX board is a high-performance system-on-module (SoM) tailored for AI applications and marketed explicitly for model deployment. It offers 12 ARM CPU cores, 2048 CUDA cores, and 64 tensor cores combined with 64 GB of main memory. Its maximum power usage is 50 W, although we measured significantly less than that during our experiments. In total, *all* experiments can be run in roughly under 6 hours on the AGX, and we estimate a total energy consumption of around 540 kJ on this platform for all experiments, which equates to around 0.0444 KG CO2 given an average European energy mix. We are interested in the following questions:

1. Out of the four methods, which method performs the best overall?
2. Can a Rejection Ensemble improve over the performance of the big model?
3. How severely will the budget be overused if no calibration is done?
4. Does a Rejection Ensemble use less energy than the big model f_b?

5.1 Experiments with Deep Learning Models

For the Deep Learning experiments, we use the following setup: For the ImageNet experiment, we employed ShuffleNetV2 x0.5 [16] as the small model and Efficientnet-B4 [25] as the big model[2]. To evaluate the rejection ensemble's performance, we conducted a 5-fold cross-validation over the validation dataset of ImageNet, i.e., in each fold, we used one part of the validation dataset to train the rejector and the other part to test the ensemble. For CIFAR100, we also use ShuffleNetV2 x0.5 [16] as the small model, while RepVGG-A2 [9] acted as the big model[3]. Similar to the ImageNet experiment, we conducted a 5-fold cross-validation over the test dataset of CIFAR100. We tested different rejectors during pre-experiments but could not find meaningful differences. Hence, we use a Logistic Regression as the rejector trained via scikit-learn [20] trained on intermediate representations of the small model. In all experiments, we use $N = 32$ as the batch size during deployment and vary $p \in \{0, 0.1, \ldots, 1.0\}$.

Figure 3 shows the results for CIFAR100 (left column) and ImageNet (right column). As expected, the accuracy improves with increasing budget for all methods except *virtual labels without calibration*. Here, the method chooses always to use the small model, so it does not increase its performance. *Virtual labels calibrated* seems to be the best method, offering the highest accuracy, although it does not outperform the big model. Looking at the power consumption, we see a similar trend: As expected, with increasing usage of the big model, the power consumption increases but never exceeds the power consumption of the big model. However, on Imagenet, a notable plateau is visible for $p > 0.5$, in which the energy consumption is already close to the big model. Second, we see

[2] Obtained from https://pytorch.org/vision/stable/models.html.
[3] Obtained from https://github.com/chenyaofo/pytorch-cifar-models.

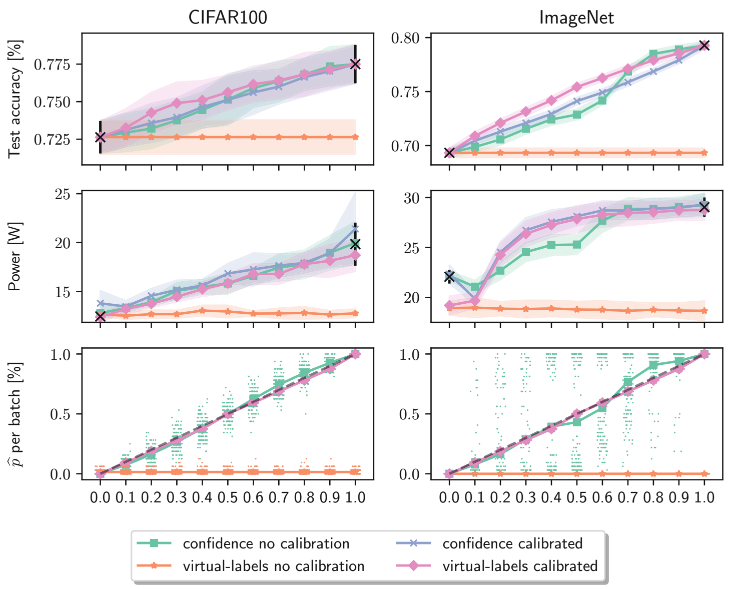


Fig. 3. Experimental results on CIFAR100 (left column) and ImageNet (right column). The first row shows the test accuracies, and the second row shows the energy consumption. The standard deviation is computed over the 5 cross-validation folds, and the crosses mark the small (left cross) and big model's (right cross) accuracy and power consumption. The third row shows how often the big model is queried for each batch $\widehat{p} = \frac{1}{N}\sum_{x,y \in \mathcal{T}} r(x)$, with one marker representing one batch. In all plots, the x-axis represents the given budget p.

that *confidence no calibration* seems to use less energy for some budget constraints. While we are not entirely sure why this is the case, we assume that our implementation of `confidence_thresholding` is sub-optimal[4] and we expect a more evolved implementation to have a better energy consumption. Looking at the relative use of the big model (third row), we see a mixed picture: First, we see that *virtual-labels no calibration* does not use the big model at all as expected from its test accuracy. Second, we see that all other methods increase their usage of the big model with a growing budget. Please note that the plots for *confidence calibrated* and *virtual-labels calibrated* overlap due to the calibration step here, so it is difficult to distinguish them in these plots. Both methods do not overuse their budget and choose the big model close to as often as the budget allows. For better interpretability, the gray line depicts the maximum usage allowed of the big model for a given budget: Anything above the gray line means we are overusing the budget, whereas everything below means we could have picked the big model more often. Most interestingly, we see that *confidence no calibration* does not overuse its budget on average, but there are many batches (a single

[4] Due to sorting, data needs to be transferred between the CPU and GPU.

marker represents one batch) in which the budget is not kept. In particular, on the ImageNet dataset, there are batches in which the big model is used for almost all data points, although the budget is close to zero. We conclude that *virtual-labels calibrated* is overall the best approach: It always keeps the budget while having a better test accuracy than the other methods with similar power consumption.

5.2 Experiments with Decision Trees

For the decision tree experiments, we use the following setup: Theorem 1 suggests that if the performance of the small model is close to the performance of the big model, then a Rejection Ensemble can improve its accuracy over the big model. To test this hypothesis, we use a small decision tree of depth three as the big model and a decision stump as the small model trained via scikit-learn [20]. Contrary to before, we now perform a 5-fold cross-validation and use the training data to train the initial small and big models. Then, we further split the testing data in each cross-validation run 50:50 into the training set for the rejector and the actual test set. Similar to before, we tested different rejectors during pre-experiments but could not find meaningful differences. Hence, we use a Logistic Regression as the rejector in all experiments, now trained on the original raw data. Similar to before, we use $N = 32$ as the batch size during deployment and vary $p \in \{0, 0.1, \dots, 1.0\}$. For space reasons, we now focus on classification accuracy and refer interested readers to the full version of the paper for additional results.

Figure 4 shows the test accuracies for our UCI experiment. We highlight three observations here: First, *virtual labels no calibration* and *confidence calibrated* both do not seem to respect the budget at all as they have nearly a constant usage of the big model. Second, on the gast-drift, covertype, and weather datasets, we see an increase in the performance of the ensemble over the big model. Most notably, on the gas-drift dataset, we see an increase of nearly 10% points in accuracy. Third, on the weather dataset, we also see a decrease in performance below the small model when using *confidence no calibration*. We conclude that our analysis in Theorem 1 is correct and that the three different scenarios can occur in practical settings. Overall, we conclude that *virtual-labels calibrated* is the best method as it seems to offer the best accuracy over different budgets compared to the other methods.

5.3 Conclusion from the Experiments

Indeed, a Rejection Ensemble can improve its performance over the big model if the small and the big models have similar performances as predicted by Theorem 1. In many scenarios, even when the rejector keeps the budget on average, there are batches on which the big model is over- or underutilized, as implied by Theorem 2. Overall, we find that *virtual-labels calibrated* seems to be the best method, as it achieves the best accuracies while keeping the budget in all scenarios. Last, we also find that, while Rejection Ensembles always use less

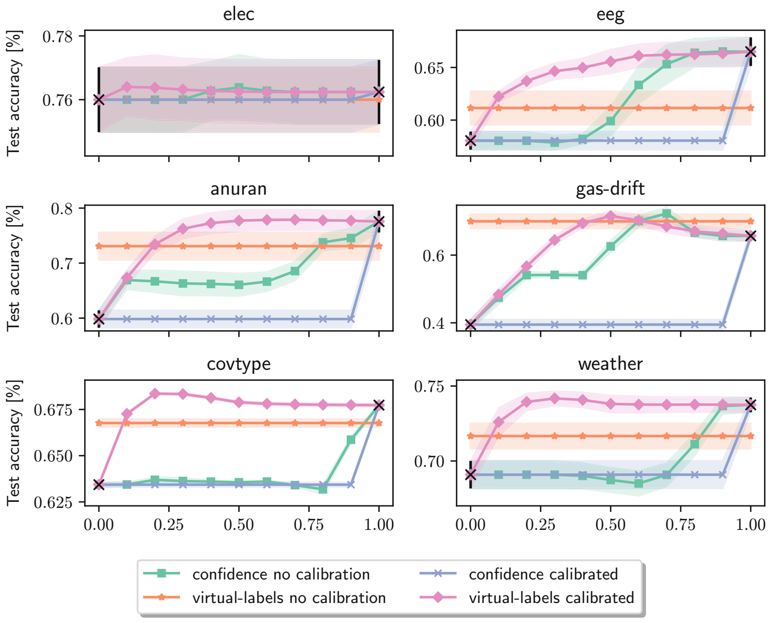


Fig. 4. Experimental results on the UCI datasets. All plots show the test accuracies over different budgets p. The standard deviation is computed over the 5 cross-validation folds, and the crosses mark the small (left cross) and big model's (right cross) accuracy.

energy than simply using the big model, there is some room for more improved implementations of the calibration step.

6 Conclusion

In this paper, we have presented a comprehensive investigation into rejection ensembles, addressing the crucial challenge of resource-aware deployment in machine learning systems. By leveraging a combination of small and large models with a rejector, Rejection Ensembles offer a promising approach to reducing resource consumption while maintaining competitive predictive performance. Our theoretical investigation has shed light on key aspects of rejector optimization and deployment, providing precise conditions under which rejection ensembles can outperform standalone models. Furthermore, we have proposed a novel algorithm for training and deploying rejectors based on theoretical insights that adhere to a given budget during deployment in all cases.

In addition to our theoretical investigation, we experimentally evaluated our novel algorithm. Our experimental evaluation on multiple datasets across various domains, executed on an Nvidia Jetson AGX board, has demonstrated the efficacy of our proposed rejection ensemble approach. In particular, we showed that our theoretical investigation in Theorem 1 and Theorem 2 capture the real-world characteristics of Rejection Ensembles. Moreover, we also showed that our novel algorithm performs better than existing approaches.

Looking ahead, future research may focus on refining and extending our theoretical framework to the real-time setting in which we have given a single sample for which we immediately need to provide a prediction (i.e., $N = 1$). This scenario is somewhat ill-defined at the moment because it is unclear what the budget should encapsulate here, i.e., under what time horizon should we consider the budget? Once a precise definition is available, we can adapt our proposed calibration algorithm to the more challenging online scenario.

Acknowledgements. This research has partly been funded by the Federal Ministry of Education and Research of Germany and the state of North-Rhine Westphalia as part of the Lamarr-Institute for Machine Learning and Artificial Intelligence.

References

1. Bartlett, P.L., Wegkamp, M.H.: Classification with a reject option using a hinge loss. J. Mach. Learn. Res. **9**, 1823–1840 (2008)
2. Brehler, M., Camphausen, L.: Combining decision tree and convolutional neural network for energy efficient on-device activity recognition. In: 2023 IEEE 16th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), pp. 179–185 (2023)
3. Buschjäger, S., Morik, K.: Joint leaf-refinement and ensemble pruning through l_1 regularization. Data Min. Knowl. Discov. **37**(3), 1230–1261 (2023)
4. Chen, K.H., et al.: Efficient realization of decision trees for real-time inference. ACM Trans. Embed. Comput. Syst. (2021)
5. Chow, C.: On optimum recognition error and reject tradeoff. IEEE Trans. Inf. Theory **16**(1), 41–46 (1970)
6. Chow, C.K.: An optimum character recognition system using decision functions. IRE Trans. Electron. Comput. **6**(4), 247–254 (1957)
7. Cortes, C., DeSalvo, G., Mohri, M.: Learning with rejection. In: Ortner, R., Simon, H.U., Zilles, S. (eds.) ALT 2016. LNCS (LNAI), vol. 9925, pp. 67–82. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46379-7_5
8. Daghero, F., Pagliari, D.J., Poncino, M.: Two-stage human activity recognition on microcontrollers with decision trees and CNNs. In: 2022 17th Conference on Ph.D. Research in Microelectronics and Electronics (PRIME), pp. 173–176 (2022)
9. Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., Sun, J.: RepVGG: making VGG-style convnets great again. In: IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2021, Virtual, 19–25 June 2021, pp. 13733–13742. Computer Vision Foundation/IEEE (2021)
10. Geifman, Y., El-Yaniv, R.: Selective classification for deep neural networks. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, pp. 4878–4887 (2017)
11. Geifman, Y., El-Yaniv, R.: Selectivenet: a deep neural network with an integrated reject option. In: Proceedings of the 36th International Conference on Machine Learning. ICML 2019, vol. 97, pp. 2151–2159. PMLR (2019)

12. Kag, A., Fedorov, I., Gangrade, A., Whatmough, P.N., Saligrama, V.: Efficient edge inference by selective query. In: The Eleventh International Conference on Learning Representations. ICLR 2023, Kigali, Rwanda, 1–5 May 2023 (2023)
13. Kelly, M., Longjohn, R., Nottingham, K.: UCI machine learning repository. https://archive.ics.uci.edu/
14. Krizhevsky, A.: Cifar-10 and cifar-100 datasets. https://www.cs.toronto.edu/~kriz/cifar.html
15. Liu, Z., Wang, Z., Liang, P.P., Salakhutdinov, R., Morency, L., Ueda, M.: Deep gamblers: learning to abstain with portfolio theory. In: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019. NeurIPS 2019, pp. 10622–10632 (2019)
16. Ma, N., Zhang, X., Zheng, H.-T., Sun, J.: ShuffleNet V2: practical guidelines for efficient CNN architecture design. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018, Part XIV. LNCS, vol. 11218, pp. 122–138. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_8
17. Madras, D., Pitassi, T., Zemel, R.S.: Predict responsibly: improving fairness and accuracy by learning to defer. In: Bengio, S., Wallach, H.M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018. NeurIPS 2018, 3–8 December 2018, Montréal, Canada, pp. 6150–6160 (2018). https://proceedings.neurips.cc/paper/2018/hash/09d37c08f7b129e96277388757530c72-Abstract.html
18. Mao, A., Mohri, M., Zhong, Y.: Theoretically grounded loss functions and algorithms for score-based multi-class abstention. CoRR abs/2310.14770 (2023)
19. Menghani, G.: Efficient deep learning: a survey on making deep learning models smaller, faster, and better. ACM Comput. Surv. **55**(12), 259:1–259:37 (2023)
20. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
21. Piatkowski, N., Lee, S., Morik, K.: Integer undirected graphical models for resource-constrained systems. Neurocomputing **173**, 9–23 (2016)
22. Pietraszek, T.: Optimizing abstaining classifiers using ROC analysis. In: Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, 7–11 August 2005. ACM International Conference Proceeding Series, vol. 119, pp. 665–672. ACM (2005)
23. Pugnana, A., Ruggieri, S.: A model-agnostic heuristics for selective classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, no. 8, pp. 9461–9469 (2023). https://doi.org/10.1609/aaai.v37i8.26133
24. Stanford Vision Lab, S.U.: Imagenet. https://www.image-net.org/
25. Tan, M., Le, Q.V.: Efficientnet: rethinking model scaling for convolutional neural networks. In: Proceedings of the 36th International Conference on Machine Learning. ICML 2019, vol. 97, pp. 6105–6114. PMLR (2019)

Lighter, Better, Faster Multi-source Domain Adaptation with Gaussian Mixture Models and Optimal Transport

Eduardo Fernandes Montesuma(✉), Fred Ngolè Mboula, and Antoine Souloumiac

Université Paris-Saclay, CEA, LIST, 91120 Palaiseau, France
eduardo.fernandesmontesuma@cea.fr

Abstract. In this paper, we tackle Multi-Source Domain Adaptation (MSDA), a task in transfer learning where one adapts multiple heterogeneous, labeled source probability measures towards a different, unlabeled target measure. We propose a novel framework for MSDA, based on Optimal Transport (OT) and Gaussian Mixture Models (GMMs). Our framework has two key advantages. First, OT between GMMs can be solved efficiently via linear programming. Second, it provides a convenient model for supervised learning, especially classification, as components in the GMM can be associated with existing classes. Based on the GMM-OT problem, we propose a novel technique for calculating barycenters of GMMs. Based on this novel algorithm, we propose two new strategies for MSDA: GMM-Wasserstein Barycenter Transport (WBT) and GMM-Dataset Dictionary Learning (DaDiL). We empirically evaluate our proposed methods on four benchmarks in image classification and fault diagnosis, showing that we improve over the prior art while being faster and involving fewer parameters (Our code is publicly available at https://github.com/eddardd/gmm_msda).

Keywords: Domain Adaptation · Optimal Transport · Gaussian Mixture Models

1 Introduction

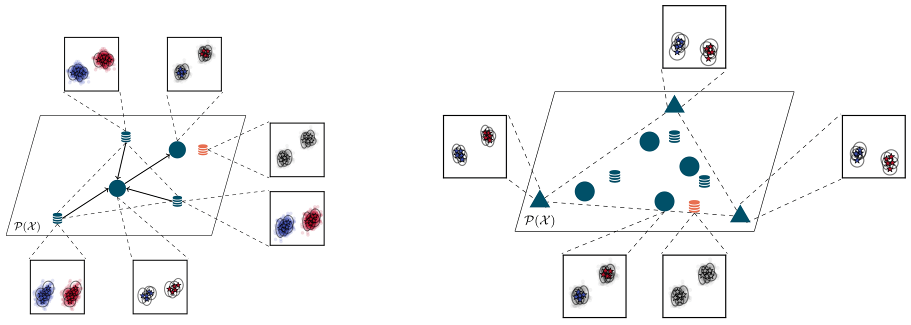
Supervised learning models, especially deep neural nets, rely on large amounts of labeled data to learn a function that reliably predicts on unseen data. This property is known as *generalization*. However, these models are subject to performance degradation, when the conditions upon which test data is acquired changes. This issue is known in the literature as distributional, or dataset shift [25].

Under distributional shift, a possible solution is to acquire a new labeled dataset under the new conditions. This solution is, in many cases such as fault

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_2.

diagnosis [18], costly or infeasible. A different approach, known as Domain Adaptation (DA), consists of collecting an unlabeled *target domain* dataset, for which the knowledge in the *source domain* dataset is transferred to [22]. A way to further enhance this adaptation is to consider multiple related, but heterogeneous sources, which is known as Multi-Source DA (MSDA) [5].

In the context of DA, a prominent framework is Optimal Transport (OT) [33], which is a field of mathematics concerned with the displacement of mass at least effort. This theory has been applied for DA in multiple ways, especially by (i) mapping samples between domains [4] and (ii) learning invariant representations [28]. For MSDA, OT has been used for aggregating the multiple source domains into a barycentric domain [15,16], which is later transported to the target domain, or by weighting source domain measures [30]. Our work considers the problem of Wasserstein Dictionary Learning (WDL), initially proposed by [27] for histogram data. This problem was later generalized by [14], for empirical measures, which allowed its application to MSDA. In [14], one expresses domains in MSDA as a barycenter of atom measures, which have a free, learnable support. As a result, the work of [14] *learns how to interpolate distributional shift* between the measures in MSDA.



(a) GMM-WBT (b) GMM-DaDiL

Fig. 1. Overview of proposed methods. 🛢 represent datasets, circles represent barycenters and triangles represent learned measures. Blue and orange elements represent labeled and unlabeled measures respectively. In Gaussian Mixture Model (GMM)-Wasserstein Barycenter Transport (WBT), a labeled GMM is determined for the target domain by transporting the barycenter of sources. In GMM-Dataset Dictionary Learning (DaDiL), we learn to express each domain as a barycenter of learned GMMs, called atoms, through dictionary learning.

However, previous algorithms relying on Wasserstein barycenters, such as WBT [15,16] and DaDiL [14], are limited in scale, since the number of points the support of the empirical measures scale with the number of samples in the original datasets. As a consequence, previous works such as [15,16] are limited to small scale datasets, or rely on mini-batch optimization [14], which introduces artifacts in the OT. To tackle these limitations, in this paper we propose a novel, parametric framework for barycentric-based MSDA algorithms. based on OT between GMMs [7]. We present an overview of our methods in Fig. 1.

Our contributions are threefold: 1. We propose a novel strategy for mapping the parameters of GMMs using OT (Sect. 3.1 and Theorem 1); 2. We propose a novel algorithm for computing mixture-Wasserstein barycenters of GMMs (Algorithm 1 in Sect. 3.3); 3. We propose an efficient parametric extension of the WBT and DaDiL algorithms based on GMMs (Sect. 3.4). We highlight that, while GMMs were previously employed in single source DA [10,20], to the best of our knowledge this is the first work to leverage GMM-OT for MSDA.

The rest of this paper is divided as follows. Section 2 covers the background behind our method. Section 3 covers our methodological contributions. Section 4 explores the empirical validation of our method with respect other OT-based MSDA algorithms, where we show that our methods significantly outperform prior art. Finally, Sect. 5 concludes this paper.

2 Preliminaries

2.1 Gaussian Mixtures

We denote the set of probability measures over a set $\mathcal{X}$ as $\mathbb{P}(\mathcal{X})$. A Gaussian measure corresponds to $P_\theta \in \mathbb{P}(\mathcal{X})$ with density,

$$f_\theta(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d \det(\mathbf{C}^{(P)})}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{m}^{(P)})^T(\mathbf{C}^{(P)})^{-1}(\mathbf{x}-\mathbf{m}^{(P)})\right),$$

where $\theta = (\mathbf{m}^{(P)}, \mathbf{C}^{(P)})$ are the mean vector $\mathbf{m}^{(P)} \in \mathbb{R}^d$ and the covariance matrix $\mathbf{C}^{(P)} \in \mathbb{S}^d = \{\mathbf{C} \in \mathbb{R}^{d\times d} : \mathbf{C} = \mathbf{C}^T \text{ and } \mathbf{x}\mathbf{C}\mathbf{x}^T > 0, \forall \mathbf{x} \in \mathbb{R}^d \setminus \{\mathbf{0}\}\}$. We generally denote $P_\theta = \mathcal{N}(\mathbf{m}^{(P)}, \mathbf{C}^{(P)})$. In addition, let $K \geq 1$ be an integer. A GMM over $\mathbb{R}^d$ is a probability measure $P_\theta \in \mathbb{P}(\mathbb{R}^d)$ such that,

$$P_\theta = \sum_{k=1}^{K} p_k P_k, \text{ where } P_k = \mathcal{N}(\mathbf{m}_k^{(P)}, \mathbf{C}_k^{(P)}), \text{ and } \mathbf{p} \in \Delta_K, \tag{1}$$

where $\Delta_K = \{\mathbf{p} \in \mathbb{R}_+^K : \sum_{k=1}^K p_k = 1\}$. Following [7], we denote the subset of $\mathbb{P}(\mathbb{R}^d)$ of probability measures which can be written as Gaussian mixtures with less than K components by $\text{GMM}_d(K)$, and $\text{GMM}_d(\infty) = \cup_{k\geq 0}\text{GMM}_d(K)$.

Given data points $\{\mathbf{x}_i^{(P)}\}_{i=1}^n$ i.i.d. from P, one can determine the parameters θ through maximum likelihood,

$$\theta^\star = \underset{\theta\in\Theta}{\text{argmax}} \sum_{i=1}^{n} \log P_\theta(\mathbf{x}_i^{(P)}), \tag{2}$$

where $\Theta = \{\{p_k, \mathbf{m}_k^{(P)}, \mathbf{C}_k^{(P)}\}_{k=1}^K : \mathbf{m}_k^{(P)} \in \mathbb{R}^d \text{ and } \mathbf{C}_k^{(P)} \in \mathbb{S}^d\}$. While Eq. 2 has no closed-form solution, one can solve this optimization problem through the celebrated Expectation-Maximization (EM) algorithm [8].

2.2 Domain Adaptation

In this paper, we focus on the problem of classification. Given a feature space $\mathcal{X} = \mathbb{R}^d$ and a label space $\mathcal{Y} = \{1, \cdots, n_{cl}\}$, this problem corresponds to finding $h \in \mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ that correctly classifies data $\{(\mathbf{x}_i^{(Q)}, y_i^{(Q)})\}_{i=1}^n$.

We use the Empirical Risk Minimization (ERM) framework [31], as it is useful for domain adaptation theory. As follows, one assumes $\mathbf{x}_i^{(Q)} \overset{iid}{\sim} Q$, for a measure $Q \in \mathbb{P}(\mathcal{X})$, and $h_0 : \mathcal{X} \rightarrow \mathcal{Y}$ such that $y_i^{(Q)} = h_0(\mathbf{x}_i^{(Q)})$. h_0 is called *ground-truth labeling function*. Given a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$, a classifier may be defined through risk minimization, i.e., $h^\star = \text{argmin}_{h \in \mathcal{H}} \mathcal{R}_Q(h)$, for $\mathcal{R}_Q(h) = \mathbb{E}_Q[\mathcal{L}(h(\mathbf{x}), h_0(\mathbf{x}))]$. This strategy is oftentimes impractical as Q and h_0 are unknown. As a result, one resorts to the minimization of the empirical risk, i.e., $\hat{h} = \text{argmin}_{h \in \mathcal{H}} \hat{\mathcal{R}}_Q(h)$, where $\hat{\mathcal{R}}_Q(h) = \frac{1}{n}\sum_{i=1}^n \mathcal{L}(h(\mathbf{x}_i^{(Q)}), y_i^{(Q)})$.

From a theoretical standpoint, this framework is useful because $\mathcal{R}_Q$ is bounded by $\hat{\mathcal{R}}_Q$ and a complexity term depending on the number of samples n, and the Vapnik-Chervonenkis dimension of $\mathcal{H}$ [31, Section 6]. As a result, $\hat{h}$ minimizing the empirical risk is guaranteed to generalize to unseen samples of Q. Nevertheless, the assumption that unseen examples come from a fixed measure Q is seldom verified in practice [25], since the conditions upon which data is acquired may change. In this case, models are required to adapt to new data, but at the same time re-training a model from the scratch is likely costly and data intensive. A solution consists of using transfer learning [22], in which one re-uses knowledge from a source domain or task to facilitate the learning on a target domain or task.

In transfer learning, a domain is a pair $(\mathcal{X}, Q(X))$ of a feature space and a (marginal) probability measure. Likewise, a task is a pair $(\mathcal{Y}, Q(Y|X))$ of a label space and a conditional probability measure. Domain adaptation is a case in which one has two domains $(\mathcal{X}, Q_S(X))$, $(\mathcal{X}, Q_T(X))$, a single task $(\mathcal{Y}, Q(Y|X))$, and $Q_S(X) \neq Q_T(X)$. Furthermore, multi-source domain adaptation supposes multiple source domain measures, i.e., $Q_{S_1}, \cdots, Q_{S_N}$, with $Q_{S_i} \neq Q_{S_j}$, and $Q_{S_i} \neq Q_T$. To reflect the idea that acquiring new data is costly, we have an unsupervised scenario. In this case, we have N labeled source datasets $\{(\mathbf{x}_i^{(Q_{S_\ell})}, y_i^{(Q_{S_\ell})})\}_{i=1}^{n_\ell}$, and an unlabeled target dataset $\{\mathbf{x}_i^{(Q_T)}\}_{i=1}^{n_T}$. Our goal is to learn a classifier on Q_T by leveraging the knowledge from the source domains.

2.3 Optimal Transport

Optimal transport is a field of mathematics concerned with the displacement of mass at least effort [17,24]. Given probability measures $P, Q \in \mathbb{P}(\mathcal{X})$, the Monge formulation [24, Section 2.2.] of OT seeks for a mapping T,

$$T^\star = \underset{T_\sharp P = Q}{\text{arginf}} \int_{\mathcal{X}} c(x, T(x)) dP(x), \tag{3}$$

where $T_\sharp$ is the push-forward mapping of T, i.e., $T_\sharp P(A) = P(T^{-1}(A))$, and $c : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is a ground-cost, that is, a measure of transportation effort.

Nonetheless, this problem poses technical difficulties, mainly due the constraint $T_\sharp P = Q$. A more tractable formulation was proposed by Kantorovich [24, Section 2.3.], and relies on OT plans,

$$\gamma^\star = \underset{\gamma \in \Gamma(P,Q)}{\text{arginf}} \int_{\mathcal{X}} \int_{\mathcal{X}} c(x,z) d\gamma(x,z), \tag{4}$$

where $\Gamma(P,Q) = \{\gamma \in \mathbb{P}(\mathcal{X} \times \mathcal{X}) : \int_{\mathcal{X}} \gamma(A,z) = P(A), \text{ and } \int_{\mathcal{X}} \gamma(x,B) = Q(B)\}$ is called the transportation polytope. There is a metric between probability measures, associated with OT, called Wasserstein distance [33]. As such, let $c(x,z) = d(x,z)^\alpha$ for $\alpha \in [1, \infty)$, where d is a metric on $\mathcal{X}$, then,

$$\mathcal{W}_{c,\alpha}(P,Q) = \left(\inf \gamma \in \Gamma(P,Q) \int_{\mathcal{X}} \int_{\mathcal{X}} c(x,z) d\gamma(x,z) \right)^{1/\alpha}. \tag{5}$$

When $\mathcal{X} = \mathbb{R}^d$, a common choice is $c(\mathbf{x}, \mathbf{z}) = \|\mathbf{x} - \mathbf{z}\|_2^\alpha$, for which we omit the subscript c. Furthermore, common values for α include 1 and 2. Throughout this paper we adopt the Euclidean metric and $\alpha = 2$.

While Eq. 4 is hard to solve for general P and Q, it has closed-form solution for Gaussian measures [29]. As such, let $P = \mathcal{N}(\mathbf{m}^{(P)}, \mathbf{C}^{(P)})$ (resp. Q). Under these conditions, for $\mathbf{C}^{(P)} = \mathbf{S}^{(P)}(\mathbf{S}^{(P)})^T$,

$$\mathcal{W}_2(P,Q)^2 = \|\mathbf{m}^{(P)} - \mathbf{m}^{(Q)}\|_2^2 + \text{Tr}\left(\mathbf{C}^{(P)} + \mathbf{C}^{(Q)} - 2(\mathbf{S}^{(P)}\mathbf{C}^{(Q)}\mathbf{S}^{(P)})^{1/2} \right),$$

This expression can be further simplified for axis-aligned Gaussians, i.e., $\mathbf{S}^{(P)} = \text{diag}(\mathbf{s}^{(P)})$, with $\mathbf{s}^{(P)} \in \mathbb{R}_+^d$,

$$\mathcal{W}_2(P,Q)^2 = \|\mathbf{m}^{(P)} - \mathbf{m}^{(Q)}\|_2^2 + \|\mathbf{s}^{(P)} - \mathbf{s}^{(Q)}\|_2^2. \tag{6}$$

Henceforth, we assume axis-aligned Gaussian measures.

Remark. Here, we give further insight into the hypothesis of using axis-aligned Gaussian measures. We use this assumption for numerical stability purposes, i.e., estimating the covariance matrix of GMMs in high dimensions is much more difficult than estimating the standard deviation vector $\mathbf{s}^{(P)}$. Here, one has two choices. First, it is possible to introduce a transformation so as to force features to be uncorrelated (e.g., through principal components analysis). This approach, nonetheless, requires more data points per domain than features, which is not always feasible. Conversely, one can increase the number of components for expressing the shape of the data (see Fig. 2). As we show in our experiments sections, we achieve good adaptation performance, even while sampling points from axis-aligned GMMs.

We use the GMM-OT framework of [7], which is convenient to our setting for 2 reasons. First, they are able to represent measures with sub-populations, such as those commonly encountered in classification and domain adaptation. Second, they yield a tractable OT problem, when γ is further restricted to be a GMM itself, that is,

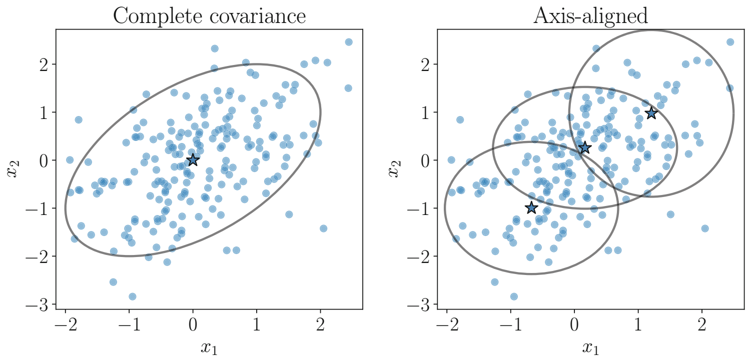


Fig. 2. Illustration of axis-aligned GMMs. This hypothesis leads to GMMs that need more components to express the underlying data distribution.

$$\omega^{\star} = \text{GMMOT}(P, Q) = \underset{\omega \in \Gamma(\mathbf{p}, \mathbf{q})}{\text{argmin}} \sum_{i=1}^{K_P} \sum_{j=1}^{K_Q} \omega_{ij} \mathcal{W}_2(P_i, Q_j)^2, \tag{7}$$

where the OT plan is given by $\gamma^{\star} = \sum_{i=1}^{n} \sum_{j=1}^{m} \omega_{ij}^{\star} f_{\theta}(\mathbf{x}) \delta(\mathbf{y} - T_{ij}(\mathbf{x}))$ and $\mathcal{W}_2(P_i, Q_j)^2$ is the Wasserstein distance between components P_i and Q_j (c.f., Eq. 6). Furthermore, the GMMOT problem defines the Mixture-Wasserstein distance [7],

$$\mathcal{MW}_2(P, Q)^2 = \sum_{i=1}^{K_P} \sum_{j=1}^{K_Q} \omega_{ij}^{\star} \mathcal{W}_2(P_i, Q_j)^2. \tag{8}$$

DaDiL [14] is an OT-based framework for expressing probability measures as barycenters of synthetic measures, called atoms. In this case, the authors use empirical measures, i.e., $\hat{P} = n^{-1} \sum_{i=1}^{n} \delta_{(\mathbf{x}_i^{(P)}, \mathbf{y}_i^{(P)})}$. The framework is inspired by dictionary learning literature [27]. The authors introduce *atoms* $\mathcal{P} = \{\hat{P}_c\}_{c=1}^{C}$ and *barycentric coordinates* $\Lambda = \{\lambda_{\ell}\}_{\ell=1}^{N}$, such that each measure in MSDA is expressed as a Wasserstein barycenter $\hat{Q}_{\ell} = \mathcal{B}(\lambda_{\ell}, \mathcal{P})$. This framework leads to the following optimization problem,

$$(\Lambda^{\star}, \mathcal{P}^{\star}) = \underset{\Lambda, \mathcal{P}}{\text{argmin}}\, \mathcal{W}_2(\hat{Q}_T, \mathcal{B}(\lambda_{\ell}; \mathcal{P})) + \sum_{\ell=1}^{N} \mathcal{W}_{c,2}(\hat{Q}_{\ell}, \mathcal{B}(\lambda_{\ell}; \mathcal{P}))^2, \tag{9}$$

where $c\left((\mathbf{x}^{(Q_{\ell})}, \mathbf{y}^{(Q_{\ell})}), (\mathbf{x}^{(B_{\ell})}, \mathbf{y}^{(B_{\ell})})\right) = \|\mathbf{x}^{(Q_{\ell})} - \mathbf{x}^{(B_{\ell})}\|_2^2 + \beta \|\mathbf{y}^{(Q_{\ell})} - \mathbf{y}^{(B_{\ell})}\|_2^2$, and $\beta \geq 0$ is a constant expressing how costly it is to move samples from different classes. This framework makes it easier to express the distributional shift between the different $\mathcal{Q} = \{\hat{Q}_1, \cdots, \hat{Q}_{N_S}, \hat{Q}_T\}$. Especially, since $\hat{B}_{\ell} = \mathcal{B}(\lambda_{\ell}, \mathcal{P})$ is labeled, one can synthesize labeled target domain data by reconstructing the target measure with $\lambda_T := \lambda_{N+1}$.

3 Methodological Contributions

3.1 First Order Analysis of $\mathcal{MW}_2$

In this section we analyze $P \mapsto \mathcal{MW}_2(P,Q)^2$, for a fixed Q. We are particularly interested on how to map the components of P towards Q, while minimizing this distance. The following theorem provides us a strategy,

Theorem 1. *Let P and Q be two GMMs with components $P_i = \mathcal{N}(\mathbf{m}_i^{(P)}, (\mathbf{s}_i^{(P)})^2)$ (resp. Q_j) and $\omega^\star$ be the solution of Eq. 7. The first-order optimality conditions of $\mathcal{MW}_2^2$, with respect $\mathbf{m}_i$ and $\mathbf{s}_i$ are given by,*

$$\hat{\mathbf{m}}_i = T_{\omega^\star}(\mathbf{m}_i^{(P)}) = \sum_{j=1}^{K_Q} \frac{\omega^\star_{ij}}{p_i}\mathbf{m}_j^{(Q)}, \text{ and } \hat{\mathbf{s}}_i = T_{\omega^\star}(\mathbf{s}_i^{(P)}) = \sum_{j=1}^{K_Q} \frac{\omega^\star_{ij}}{p_i}\mathbf{s}_j^{(Q)}, \tag{10}$$

where $\omega^\star$ is the solution of Eq. 7.

Proof. Our proof relies on the analysis of $\{(\mathbf{m}_i, \mathbf{s}_i)\}_{i=1}^{K_P} \mapsto \mathcal{MW}_2(P,Q)^2$ (c.f., Eq. 8). Given $\omega^\star = \text{GMMOT}(P,Q)$,

$$\frac{\partial \mathcal{MW}_2^2}{\partial \mathbf{m}_i} = 2\sum_{j=1}^{K_Q}\omega^\star_{ij}(\mathbf{m}_i - \mathbf{m}_j^{(Q)}) = 2\left(p_i\mathbf{m}_i - \sum_{j=1}^{K_Q}\omega^\star_{ij}\mathbf{m}_j^{(Q)}\right),$$

and, by equating this last term to 0, one gets the desired equality.

Equation 10 is similar to the barycentric mapping in Empirical Optimal Transport (EOT) [4, eq. 13], which serves as an approximation for the Monge mapping between P and Q. In our case, the barycentric mappings act on the parameters of the GMM, rather than on its samples. Theorem 1 will be useful in the calculation of $\mathcal{MW}_2$ barycenters.

3.2 Supervised Mixture-Wasserstein Distances

In this paper, we consider supervised learning problems. As such, it is necessary to equip the components of GMMs with labels that represent the classes in the datasets. We propose doing so through a simple heuristic, especially, we model $P(\mathbf{x}|y)$ through a GMM. We then concatenate the n_c obtained GMMs, and assign, for the $k-$th GMM of the $y-$th class, $v_{k,y'}^{(P)} = \delta(y'-y)$, i.e., a vector of n_c components, and 1 on the $y-$th entry. We can assure that the resulting weights sum to 1 by dividing their value by $\sum_{y=1}^{n_c}\sum_{k=1}^{K} p_{k,y}$, where $p_{k,y}$ corresponds to the weight of the $k-$th component of the $y-$th GMM.

Given a GMM $\{p_k, \mathbf{m}_k^{(P)}, \mathbf{s}_k^{(P)}, \mathbf{v}_k^{(P)}\}_{k=1}^{K}$, we define a classifier through Maximum a Posteriori (MAP) estimation. This strategy is carried out through,

$$\hat{h}_{MAP}(\mathbf{x}) = \underset{y=1,\cdots,n_c}{\text{argmax}}\ P(y|\mathbf{x}) = \sum_{k=1}^{K} \underbrace{P_\theta(k|\mathbf{x})}_{p_k P_k(\mathbf{x})/\sum_{k'} p_{k'} P_{k'}(\mathbf{x})} \underbrace{P(y|k)}_{v_{k,y}^{(P)}}, \tag{11}$$

we use this classifier in a few illustrative examples in Sect. 4.3.

Remark. In Eq. 11, we are implicitly assuming that the component k is conditionally independent with y given $\mathbf{x}$. This remark is intuitive, as $\mathbf{x}$ explains, at the same time, the component and the label.

Similarly to EOT, when the mixtures P and Q are labeled, one needs to take into account the labels in the ground-cost. Given $\beta > 0$, we propose the following distance between labeled GMMs,

$$\mathcal{SMW}_2(P, Q)^2 = \min_{\omega \in \Gamma(\mathbf{p},\mathbf{q})} \sum_{i=1}^{K_P} \sum_{j=1}^{K_Q} \omega_{ij}(\mathcal{W}_2(P_i, Q_j)^2 + \beta \|\mathbf{v}_i^{(P)} - \mathbf{v}_j^{(Q)}\|_2^2). \tag{12}$$

While simple, using an Euclidean distance for the soft-labels allows us to derive similar first-order conditions for $\mathcal{SMW}_2$,

Theorem 2. *Under the same conditions of Theorem 1, let P_i and Q_j be equipped with labels $\mathbf{v}_i^{(P)}$ and $\mathbf{v}_j^{(Q)}$. The first order optimality conditions of $\mathcal{SMW}_2$ with respect $\mathbf{m}_i$ and $\mathbf{s}_i$ are given by Eq. 10. Furthermore, for $\mathbf{v}_i$,*

$$\hat{\mathbf{v}}_i = T_\omega(\mathbf{v}_i^{(P)}) = \sum_{j=1}^{K_Q} \frac{\omega_{ij}^\star}{p_i} \mathbf{v}_j^{(Q)}. \tag{13}$$

Proof. The label distance term in $\mathcal{SMW}$ is independent of $\mathbf{m}_i$ and $\mathbf{s}_i$, hence the optimality conditions of these variables remain unchanged. Therefore, the first-order optimality condition with respect $\mathbf{v}_i$ is,

$$\frac{\partial \mathcal{SMW}_2^2}{\partial \mathbf{v}_i} = 2\beta \sum_{j=1}^{K_Q} \omega_{ij}^\star (\mathbf{v}_i - \mathbf{v}_j^{(Q)}) = 2\beta \left(p_i \mathbf{v}_i - \sum_{j=1}^{K_Q} \omega_{ij}^\star \mathbf{v}_j^{(Q)} \right),$$

which, for $\beta > 0$, is zero if and only if $\mathbf{v}_i = T_\omega(\mathbf{v}_i^{(P)})$.

Remark. In Eq. 12, we are heuristically adding a label regularization term to the $\mathcal{MW}_2$ distance. The actual continuous counterpart (between samples, rather than components) is currently beyond the scope of this paper, but methodologically, this choice remains valid, and is closer to the contributions of [3].

3.3 Mixture Wasserstein Barycenters

In this section, we detail a new algorithm for computing barycenters of GMMs under the $\mathcal{MW}_2$ and $\mathcal{SMW}_2$ metrics. As such, we adapt the definition of [1],

Definition 1. *Given $C \geq 1$ GMMs $\mathcal{P} = \{P_c\}_{c=1}^C$, $K_B \geq 1$, and a vector of barycentric coordinates $\lambda \in \Delta_C$, the $\mathcal{SMW}_2$ barycenter is given by,*

$$B^\star = \mathcal{B}(\lambda, \mathcal{P}) = \underset{B \in GMM_d(K_B)}{\mathit{argmin}} \left\{ \mathcal{L}(B) = \sum_{c=1}^{C} \lambda_c \mathcal{SMW}_2(B, P_c)^2 \right\}. \tag{14}$$

When the GMMs in $\mathcal{P}$ are unlabeled, one may define, by analogy, a barycenter under the $\mathcal{MW}_2$. Henceforth we describe an algorithm for labeled GMMs, but its extension for unlabeled GMMs is straightforward. Inspired by previous results in empirical Wasserstein barycenters [6,14], we propose a novel strategy for computing $\mathcal{B}(\lambda, \mathcal{P})$. Our method relies on the analysis of $\theta_B = \{(\mathbf{m}_i^{(B)}, \mathbf{s}_i^{(B)}, \mathbf{v}_i^{(B)})\}_{i=1}^{K_B} \mapsto \sum_{c=1}^{C} \lambda_c \mathcal{SMW}_2(B, P_c)^2$. First, for a fixed θ_B, we find $\omega_1^\star, \cdots, \omega_C^\star$ transport plans. Then, for fixed transport plans, we solve,

$$\begin{aligned}\underset{\theta_B}{\text{argmin}}\, \mathcal{L}(\theta_B) &= \sum_{c=1}^{C} \lambda_c \sum_{i=1}^{K_B} \sum_{j=1}^{K_P} \omega_{c,i,j}^\star C_{c,i,j},\\ \text{where } C_{c,i,j} &= \|\mathbf{m}_i^{(B)} - \mathbf{m}_j^{(P_c)}\|_2^2 + \|\mathbf{s}_i^{(B)} - \mathbf{s}_j^{(P_c)}\|_2^2 + \beta\|\mathbf{v}_i^{(B)} - \mathbf{v}_j^{(P_c)}\|_2^2\end{aligned}$$

which can be optimized by taking derivatives with respect $\mathbf{m}_i^{(B)}$, $\mathbf{s}_i^{(B)}$ and $\mathbf{v}_i^{(B)}$. For instance, taking the derivative of $\mathcal{L}(\theta_B)$ with respect $\mathbf{m}_i^{(B)}$,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{m}_i^{(B)}} = 2\sum_{c=1}^{C} \lambda_k \sum_{j=1}^{K_P} \omega_{c,i,j}^\star (\mathbf{m}_i^{(B)} - \mathbf{m}_j^{(P_c)}) = \frac{2}{K_B}\mathbf{m}_i^{(B)} - 2\sum_{c=1}^{C} \lambda_c \sum_{j=1}^{K_P} \omega_{c,i,j}^\star \mathbf{m}_j^{(P_c)}$$

setting the derivative to 0, one has, $\mathbf{m}_i^{(B)} = \sum_{c=1}^{C} \lambda_c T_{\omega_c^\star}(\mathbf{m}_i^{(B)})$. Similar results can be acquired for $\mathbf{s}_i^{(B)}$ and $\mathbf{v}_i^{(B)}$ by taking the appropriate derivatives. Our strategy is shown in Algorithm 1.

3.4 Multi-source Domain Adaptation Through GMM-OT

In this section, we detail two contributions for MSDA based on GMM-OT: GMM-WBT and GMM-DaDiL. In both cases, we suppose access to N labeled source GMMs $\mathcal{Q}_S = \{Q_{S_\ell}\}_{\ell=1}^{N}$ and an unlabeled target GMM Q_T. Contrary to the empirical versions of these algorithms [14–16], we assume that an axis-aligned GMM has been learned for each domain, including the target.

GMM-WBT. The intuition of this algorithm is transforming the MSDA scenario into a single-source one, by first calculating a Wasserstein barycenter of $B = \mathcal{B}(\mathbb{1}_N/N; \mathcal{Q}_S)$. After this step, WBT solves a single-source problem between B and Q_T. When each Q_{S_ℓ} is a GMM, the parameters of B are estimated through Algorithm 1. Next, one solves for $\omega^{(T)} = \text{GMMOT}(B, Q_T)$, so that the parameters of B are transported towards Q_T using Theorems 1 and 2,

$$\hat{\mathbf{m}}_i^{(Q_T)} = K_B \sum_{j=1}^{K_T} \omega_{ij}^{(T)} \mathbf{m}_j^{(Q_T)}, \text{ and, } \hat{\mathbf{s}}_i^{(Q_T)} = K_B \sum_{j=1}^{K_T} \omega_{ij}^{(T)} \mathbf{s}_j^{(Q_T)}. \tag{15}$$

With a labeled GMM, $\{\hat{\mathbf{m}}_i^{(Q_T)}, \hat{\mathbf{s}}_i^{(Q_T)}, \mathbf{v}_i^{(B)}\}_{i=1}^{K_B}$, on the target domain, we can learn a classifier on the target domain as explained in Sect. 3.2.

Algorithm 1: $\mathcal{SMW}_2$ Barycenter of GMMs

```
1  function smw_barycenter($\{(\mathbf{M}^{(P_c)}, \mathbf{S}^{(P_c)}, \mathbf{V}^{(P_c)})\}_{c=1}^{C}$, $\tau$, $N_{it}$)
2      $\mathbf{m}_i^{(B)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, $\mathbf{s}_i^{(B)} = 1$ and $\mathbf{y}_i^{(B)} = \mathbb{1}_{n_c}/n_c$
3      while $|L_{it} - L_{it-1}| \geq \tau$ and $it \leq N_{it}$ do
           // Compute GMM-OT plans
4          for $c = 1, \cdots, C$ do
5              $\omega^{(c,it)} = \text{GMMOT}(B, P_c)$
           // Note: $\mathcal{W}_2(B_i, P_{c,j})^2 = \|\mathbf{m}_i^{(B)} - \mathbf{m}_j^{(P_c)}\|_2^2 + \|\mathbf{s}_i^{(B)} - \mathbf{s}_j^{(P_c)}\|_2^2$
6          $L_{it} = \sum_{c=1}^{C} \lambda_c \sum_{i=1}^{K_B} \sum_{j=1}^{K_P} \omega_{ij}^{(c,it)} \left( (\mathcal{W}_2(B_i, P_{c,j})^2 + \beta \|\mathbf{v}_i^{(B)} - \mathbf{v}_j^{(P_c)}\|_2^2) \right)$
           // Update barycenter parameters
7          $\mathbf{m}_i^{(B)} = \sum_{c=1}^{C} \lambda_c T_{\omega^{(c,it)}}(\mathbf{m}_i^{(B)})$
8          $\mathbf{s}_i^{(B)} = \sum_{c=1}^{C} \lambda_c T_{\omega^{(c,it)}}(\mathbf{s}_i^{(B)})$
9          $\mathbf{v}_i^{(B)} = \sum_{c=1}^{C} \lambda_c T_{\omega^{(c,it)}}(\mathbf{v}_i^{(B)})$
10     return $\mathbf{M}^{(B)}$, $\mathbf{S}^{(B)}$, $\mathbf{V}^{(B)}$
```

GMM-DaDiL. Our second algorithm consists of a parametric version for the DaDiL algorithm of [14]. The idea is to replace the atoms in $\mathcal{P} = \{\hat{P}_c\}_{c=1}^{C}$ by GMMs parametrized through $\Theta_P = \{(\mathbf{M}^{(P_c)}, \mathbf{S}^{(P_c)}, \mathbf{V}^{(P_c)})\}_{c=1}^{C}$. Learning a dictionary is thus equivalent to estimating these parameters, that is,

$$(\Lambda^\star, \Theta_P^\star) = \underset{\Lambda, \Theta_P}{\text{argmin}}\, \mathcal{MW}_2(Q_T, \mathcal{B}(\lambda_T, \mathcal{P}))^2 + \sum_{\ell=1}^{N} \mathcal{SMW}_2(Q_\ell, \mathcal{B}(\lambda_\ell; \mathcal{P}))^2. \quad (16)$$

While Eq. 16 does not have a closed-form solution, we optimize it through gradient descent. An advantage of the GMM modeling is that this optimization problem involves far less variables than DaDiL, hence we do not resort to mini-batches. We detail our strategy in Algorithm 2. Note that we need to enforce 3 kinds of constraints: (i) $\mathbf{s}_i^{(P_c)} \in \mathbb{R}_+^d$, (ii) $\lambda_\ell \in \Delta_C$ and (iii) $\mathbf{y}_i^{(P_c)} \in \Delta_{n_{cl}}$. For (i) and (ii), we use orthogonal projections into $\mathbb{R}_+^d$ and Δ_C respectively. We additionally set $\mathbf{s}_i^{(P_c)} \geq s_{min}$ for numerical stability. For (iii), we perform a change of variables $\mathbf{y}_i^{(P_c)} = \text{softmax}(\mathbf{u}_i^{(P_c)})$.

Once the dictionary $(\Lambda, \mathcal{P})$ is learned, we are able to reconstruct the domains in MSDA via the barycenter $\mathcal{B}(\lambda; \mathcal{P})$. We are especially interested in the target reconstruction λ_T, i.e., $\mathcal{B}(\lambda_T, \mathcal{P})$. This barycenter is a labeled GMM (as we show in Fig. 4b). As a result, we can obtain labeled samples from this GMM, then use them to train a classifier that works on the target domain.

The computational complexity of an optimization step of Algorithm 2 corresponds to $\mathcal{O}(N \times N_{it} \times C \times K^3 \log K)$, i.e., we calculate N barycenters of C atoms. One should compare this complexity with that of DaDiL, i.e., $\mathcal{O}(N \times N_{it} \times M \times C \times n_b^3 \log n_b)$, where $M = \lceil n/n_b \rceil$ is the number of mini-batches sampled at each iteration. In our experiments in Sect. 4, we show that

Algorithm 2: GMM-Dataset Dictionary Learning

```
1  function gmm_dadil({($\mathbf{M}^{(Q_{S_\ell})}, \mathbf{S}^{(Q_{S_\ell})}, \mathbf{V}^{(Q_{S_\ell})}$)}$_{\ell=1}^{N}$, {($\mathbf{M}^{(Q_T)}, \mathbf{S}^{(Q_T)}$)}, $N_{it}$, $\eta$)
       // Initialization.
2      $\mathbf{m}_i^{(P_k)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, $\mathbf{s}_i^{(P_k)} := 1$, $\mathbf{u}_i^{(P_k)} := 1/n_c$, and $\lambda_\ell = 1/K$
3      for $it = 1, \cdots, N_{it}$ do
4          $L \leftarrow 0$
           // Change of variables
5          $\mathbf{v}_i^{(P_k)} \leftarrow \text{softmax}(\mathbf{u}_i^{(P_k)})$
           // Evaluate supervised loss on sources
6          for $\ell = 1, \cdots, N$ do
7              $L \leftarrow L + \mathcal{SMW}_2(Q_{S_\ell}, \mathcal{B}(\lambda_\ell, \mathcal{P}))^2$
           // Evaluate unsupervised loss on targets
8          $L \leftarrow L + \mathcal{MW}_2(Q_T, \mathcal{B}(\lambda_T, \mathcal{P}))^2$
           // Gradient step
9          $\mathbf{m}_j^{(P_k)} \leftarrow \mathbf{m}_j^{(P_k)} - \eta \partial L / \partial \mathbf{m}_j^{(P_k)}$
10         $\mathbf{u}_j^{(P_k)} \leftarrow \mathbf{u}_j^{(P_k)} - \eta \partial L / \partial \mathbf{u}_j^{(P_k)}$
           // Note: we project variables s and $\lambda$.
11         $\mathbf{s}_j^{(P_k)} \leftarrow \text{proj}_{\mathbb{R}_+^d}(\mathbf{s}_j^{(P_k)} - \eta \partial L / \partial \mathbf{s}_j^{(P_k)})$
12         $\lambda_\ell \leftarrow \text{proj}_{\Delta_C}(\lambda_\ell - \eta \partial L / \partial \lambda_\ell)$
13     return $\Lambda, \mathcal{P}$
```

we achieve state-of-the-art performance with K on the same order of magnitude as n_b (e.g., a few hundred Gaussian components). As a result, we achieve a speed-up on the order of M while solving an exact OT problem (see Fig. 5 (c) below).

4 Experiments

4.1 Toy Example

In this section, we explore GMM-WBT and GMM-DaDiL in the context of a toy example. We generate 4 datasets over $\mathbb{R}^d$, by gradually shifting and deforming initial measure through an affine mapping. In Fig. 3 (a) we show the generated datasets. Starting with GMM-WBT, Fig. 3 (b) shows the learned GMMs for each dataset, in which the target GMM is not labeled. The barycenter of $\mathcal{Q}_S = \{Q_{S_\ell}\}_{\ell=1}^N$ is shown in Fig. 3c (bottom-left). This barycenter is labeled. As a result, we may transfer its parameters to the target domain through GMM-OT (upper part), which leads to a labeled GMM in the target domain.

Next, we show in Fig. 4 a summary for the GMM-DaDiL optimization process (Fig. 4a), and the reconstruction of target domain GMMs (Fig. 4b). Note that, as the training progresses, the reconstruction error and the negative log-likelihood of the GMMs decrease. As a result, GMM-DaDiL produces accurate, labeled

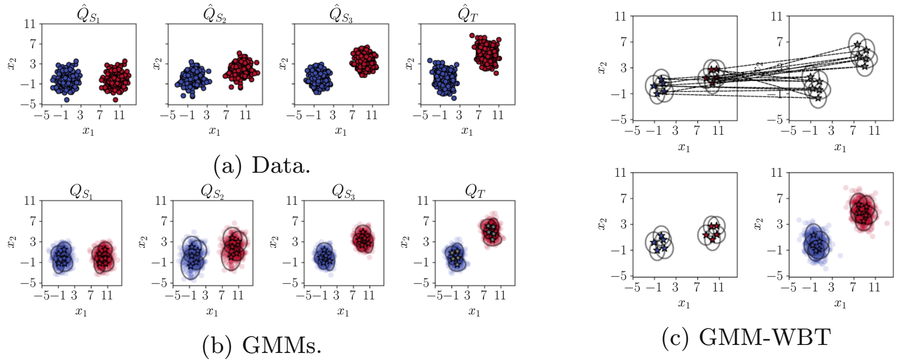


Fig. 3. Data and GMMs used in the toy experiment. In (a) Each of these datasets was generated by applying an affine transformation to an initial dataset. In (b), we show an axis-aligned GMM fitted to the data via EM. In (c), we show a summary of GMM-WBT, where show the OT plan between components (upper part) between B (left) and Q_T (right). The resulting labeled GMM is shown in the lower right part of (c).

GMMs for each domain. We provide further examples on the GMMs-DaDiL optimization in the supplementary materials. Next, we present our results on MSDA benchmarks.

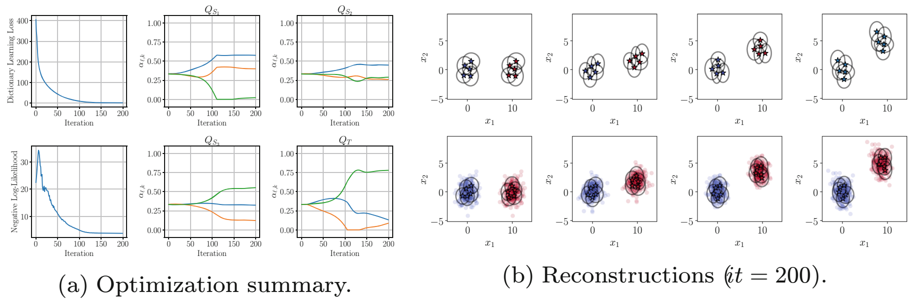


Fig. 4. Optimization and reconstruction summaries using GMM-DaDiL. In (a), we show the evolution of loss, negative log-likelihood and barycentric coordinates (i.e., λ_ℓ) over the course of optimization. In (b), we show the reconstructed GMMs (i.e., $\mathcal{B}(\lambda_\ell, \mathcal{P})$) when the algorithm converges.

4.2 Multi-source Domain Adaptation

We compare our method to prior art. We focus on OT-based methods, such as WJDOT [30], WBT [15,16] and DaDiL [14]. For completeness, we include recent strategies on deep MSDA that update the encoder network during the adaptation process, rather than using pre-extracted features. These are M^{3}SDA [23], LtC-MSDA [34], KD3A [9] and Co-MDA [12]. We establish our comparison on 4 benchmarks, divided between visual domain adaptation (Office31 [26], Office-Home [32]) and cross-domain fault diagnosis (TEP [19] and CWRU). See Table 1 for further details.

Table 1. Overview of benchmarks used in our experiments.

Benchmark	Backbone	Problem	# Samples	# Domains	# Classes	# Features
Office 31	ResNet50	Object Recognition	3287	3	31	2048
Office-Home	ResNet101	Object Recognition	15500	4	65	2048
TEP	CNN	Fault Diagnosis	17289	6	29	128
CWRU	MLP	Fault Diagnosis	24000	3	10	256

As with previous works on OT-based MSDA, we perform domain adaptation on pre-extracted features. As such, we pre-train a neural network (called backbone) on the concatenation of source domain data, then we use it to extract the features from each domain. For visual adaptation tasks, we use ResNets [11], while for fault diagnosis, we use a CNN and a multi-layer perceptron, as in [14,19]. We summarize our results in Table 2.

First, OT-based methods generally outperform other methods in MSDA. Overall, shallow DA methods solve a simpler task compared to deep DA methods, as they do not need to update the encoder network during adaptation. Second, the GMM-OT framework generally improves over using empirical OT. For instance, in the CWRU benchmark, GMM-WBT largely outperforms WBT [15,16]. Furthermore, GMM-DaDiL outperforms its empirical counterpart on all benchmarks, as well as GMM-WBT. This point further illustrates the power of dictionary learning in MSDA. Note that, in Table 2 (d), GMM-DaDiL manages to have the best average adaptation performance across domains without actually being the best on any single domain. As a consequence, GMM-DaDiL enjoys better stability, with respect distribution shift, than previous methods.

4.3 Lighter, Better, Faster Domain Adaptation

Our first experiment illustrates why GMM-DaDiL is **lighter** than previous barycenter-based algorithms, such as DaDiL. In this context, a lighter model needs less parameters to achieve a certain domain adaptation performance. We rank GMM-OT models by the number of components K, and empirical models by the number of samples n in their support. Note that these parameters regulate the complexity of these algorithms. We use the adaptation task $(Cl, Pr, Rw) \rightarrow Ar$ from Office-Home for our analysis. We show a comparison in Fig. 5 (a). From this figure, we see that GMM-DaDiL surpasses all other

Table 2. Classification accuracy of domain adaptation methods divided by benchmark. ⋆, †, ‡ and § denote results from [2,12,14,19], respectively.

Algorithm	Ar	Cl	Pr	Rw	Avg. ↑
ResNet101	72.90	62.20	83.70	85.00	75.95
M^{3}SDA	71.13	61.41	80.18	80.64	73.34
LtC-MSDA	74.52	60.56	85.52	83.63	76.05
KD3A	73.80	63.10	84.30	83.50	76.17
Co-MDA[‡]	74.40	64.00	85.30	83.90	76.90
WJDOT	74.28	63.80	83.78	84.52	76.59
WBT	75.72	63.80	84.23	84.63	77.09
DaDiL-E	**77.16**	64.95	85.47	84.97	78.14
DaDiL-R	75.92	64.83	85.36	**85.32**	77.86
GMM-WBT	75.31	64.26	**86.71**	85.21	77.87
GMM-DaDiL	**77.16**	**66.21**	86.15	**85.32**	**78.81**

(a) Office-Home.

Algorithm	A	D	W	Avg. ↑
ResNet50	67.50	95.00	96.83	86.40
M^{3}SDA	66.75	97.00	96.83	86.86
LtC-MSDA	66.82	100.00	97.12	87.98
KD3A	65.20	**100.0**	98.70	87.96
Co-MDA	64.80	99.83	98.70	87.83
WJDOT	67.77	97.32	95.32	86.80
WBT	67.94	98.21	97.66	87.93
DaDiL-E	70.55	**100.00**	98.83	89.79
DaDiL-R	70.90	**100.00**	98.83	**89.91**
GMM-WBT	70.13	99.11	96.49	88.54
GMM-DaDiL	**72.47**	**100.0**	**99.41**	**90.63**

(b) Office 31.

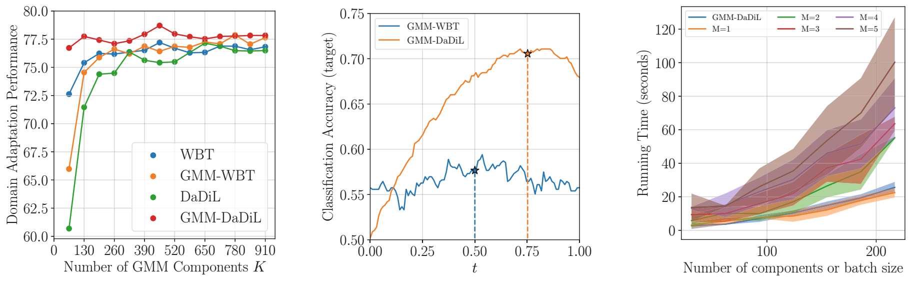
Algorithm	A	B	C	Avg. ↑
MLP[⋆]	70.90 ± 0.40	79.76 ± 0.11	72.26 ± 0.23	74.31
M3SDA	56.86 ± 7.31	69.81 ± 0.36	61.06 ± 6.35	62.57
LTC-MSDA[⋆]	82.21 ± 8.03	75.33 ± 5.91	81.04 ± 5.45	79.52
KD3A[§]	81.02 ± 2.92	78.04 ± 4.05	74.64 ± 5.65	77.90
Co-MDA	62.66 ± 0.96	55.78 ± 0.85	76.35 ± 0.79	64.93
WJDOT	99.96 ± 0.02	98.86 ± 0.55	**100.0 ± 0.00**	99.60
WBT[⋆]	99.28 ± 0.18	79.91 ± 0.04	97.71 ± 0.76	92.30
DaDiL-R[⋆]	99.86 ± 0.21	99.85 ± 0.08	**100.00 ± 0.00**	99.90
DaDiL-E[⋆]	93.71 ± 6.50	83.63 ± 4.98	99.97 ± 0.05	92.33
GMM-WBT	**100.00 ± 0.00**	**99.95 ± 0.07**	**100.00 ± 0.00**	**99.98**
GMM-DaDiL	**100.00 ± 0.00**	**99.95 ± 0.04**	**100.00 ± 0.00**	**99.98**

(c) CWRU.

Algorithm	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Avg. ↑
CNN[†]	80.82 ± 0.96	63.69 ± 1.71	87.47 ± 0.99	79.96 ± 1.07	74.44 ± 1.52	84.53 ± 1.12	78.48
M^{3}SDA[†]	81.17 ± 2.00	61.61 ± 2.71	79.99 ± 2.71	79.12 ± 2.41	75.16 ± 3.01	78.91 ± 3.24	75.99
KD3A[§]	72.52 ± 3.04	18.96 ± 4.54	81.02 ± 2.40	74.42 ± 1.60	67.18 ± 2.37	78.22 ± 2.14	65.38
Co-MDA	64.56 ± 0.62	35.99 ± 1.21	79.66 ± 1.36	72.06 ± 1.66	66.33 ± 0.97	78.91 ± 1.87	66.34
WJDOT	89.06 ± 1.34	75.60 ± 1.84	**89.99 ± 0.86**	89.38 ± 0.77	85.32 ± 1.29	87.43 ± 1.23	86.13
WBT[†]	**92.38 ± 0.66**	73.74 ± 1.07	88.89 ± 0.85	89.38 ± 1.26	85.53 ± 1.35	86.60 ± 1.63	86.09
DaDiL-R[‡]	91.97 ± 1.22	**77.15 ± 1.32**	85.41 ± 1.69	**89.39 ± 1.03**	84.49 ± 1.95	**88.44 ± 1.29**	86.14
DaDiL-E[‡]	90.45 ± 1.02	77.08 ± 1.21	86.79 ± 2.14	89.01 ± 1.35	84.04 ± 3.16	87.85 ± 1.06	85.87
GMM-WBT	92.23 ± 0.70	71.81 ± 1.78	84.72 ± 1.92	89.28 ± 1.55	**87.51 ± 1.73**	82.49 ± 1.81	84.67
GMM-DaDiL	91.72 ± 1.41	76.41 ± 1.89	89.68 ± 1.49	89.18 ± 1.17	86.05 ± 1.46	88.02 ± 1.12	**86.85**

(d) TEP.

methods over the entire range $K \in \{65, 130, \cdots, 910\}$. Especially, its empirical counterpart, DaDiL, needs a large number of samples for accurately represent



(a) Lighter. (b) Better. (c) Faster.

Fig. 5. Lighter, Better, Faster. In (a), we analyse the performance of interpolations $\mathcal{B}((\lambda_0, 1-\lambda_0); \mathcal{Q}_S)$, $\mathcal{Q}_S = \{Q_{S_1}, Q_{S_2}\}$ and $\mathcal{B}((\lambda_0, 1-\lambda_0); \mathcal{P})$ with learned $\mathcal{P} = \{P_1, P_2\}$ for GMM-WBT and GMM-DaDiL. In (b), we analyse the efficiency of barycenter-based methods under an increasing number of GMM components (number of samples for DaDiL and WBT). GMM-DaDiL has state-of-the-art performance even for the extreme case where $K = 65$. In (c), we compare the running time of GMM-DaDiL with that of DaDiL, as a function of number of components K and batch size n_b, respectively. This figure illustrates the speedup of GMM-DaDiL as the number of samples in DaDiL (and hence, $M = \lceil n/n_b \rceil$) increases. Circles represent the average over 5 independent runs, while the error bars show 2 times the standard deviation.

probability measures. Curiously, the performance of GMM-WBT and WBT are quite similar. Indeed, recent studies [21] show that Wasserstein barycenters are effective in compressing probability measures with respect the number of their samples. As a result, in this adaptation task, the GMM version of WBT has similar performance to the empirical version.

Our second experiment illustrates why GMM-Optimal Transport Domain Adaptation (OTDA) provides a **better** framework for MSDA. We use the adaptation $(D, W) \rightarrow A$ in the Office-31 benchmark as the basis of our experiment. Note that GMM-DaDiL reconstructs the target domain via a barycenter $\mathcal{B}(\lambda_T, \mathcal{P})$, where λ_T and $\mathcal{P}$ are learned parameters. We thus ablate the learning of λ_T and $\mathcal{P}$, i.e., we compare it to $\lambda_T = (\lambda_0, 1 - \lambda_0)$, $\lambda_0 \in [0, 1]$ and $\mathcal{Q}_S = \{Q_{S_\ell}\}_{\ell=1}^{N_S}$. This generates a series of measures parametrized by λ_0. To further match $\mathcal{B}(\lambda_T, \mathcal{Q}_S)$ with Q_T, we transport it to Q_T through Eq. 15. Note that this corresponds to performing GMM-WBT with a barycenter calculated with λ_T. The overall experiment is shown in Fig. 5 (b). While the performance of GMM-WBT remains approximately stable, that of GMM-DaDiL grows as we move closer to $\lambda_T^\star$ learned by dictionary learning (blue star). Overall, the interpolation space generated by atoms better captures the distributional shift occurring on the target domain.

Our third experiment shows that GMM-DaDiL is **faster** than DaDiL. We plot the running time of these methods on the Office 31 benchmark, for the $(D, W) \rightarrow A$ adaptation task. The variables that influence the complexity of GMM and empirical DaDiL are the number of components K and the batch size n_b, respectively. For GMM-DaDiL, we simply measure its running time for 5 independent runs of the algorithm (blue curve) for each $K \in \{31, 62, \cdots, 217\}$. For DaDiL, we set $n_b \in \{31, 62, \cdots, 217\}$, and set $n = M \times n_b$, where M is the number of mini-batches. We measure the performance over 5 independent runs as well. Other than these parameters, we fix $N_{iter} = 50$ and $C = 3$. As shown in Fig. 5 (c), the running time of GMM-DaDiL and DaDiL are essentially equivalent for $M = 1$. For $M > 1$, we have a speedup that is proportional to M.

In our fourth experiment, we use the $(B, C) \rightarrow A$ adaptation task of CWRU. We are interested in visualizing the evolution of atoms and reconstructions with respect DaDiL and GMM-DaDiL iterations. We visualize this evolution through UMAP [13], i.e., we concatenate the data from DaDiL's atoms, i.e., $\mathbf{x}_i^{(P_{c,it})}$, so that these are jointly embedded into $\mathbb{R}^2$. For GMM-DaDiL, we concatenate the mean parameters, i.e., $\mathbf{m}_i^{(P_{c,it})}$. We summarize our results in Fig. 6. Overall, as shown in Fig. 6 (a–d), GMM-DaDiL optimization is more stable than that of DaDiL, especially since we do not use mini-batches. This remark is also evidenced in the reconstructions in Fig. 6 (e–f).

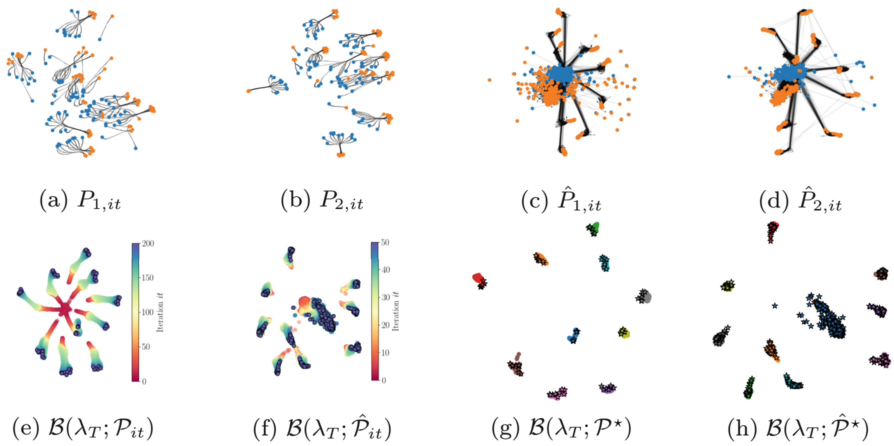


Fig. 6. From (a–d), we show the trajectory of atom distributions for GMM-DaDiL (a, b) and DaDiL (c, d). Blue and orange points represent the initializations and final values for atoms at convergence. In (e, f), we show the trajectory of barycentric reconstructions for the target domain for these two methods. In (g, h), we show the reconstructions alongside target domain data at convergence. (Color figure online)

5 Conclusion

In this work, we propose a novel framework for MSDA, using GMM-OT [7]. Especially, we propose a novel algorithm for calculating Wasserstein barycenters of GMMs (Algorithm 1). Based on this algorithm, we propose two new strategies for MSDA: GMM-WBT and GMM-DaDiL (Algorithm 2). The first method determines a labeled GMM on the target domain by transporting the barycenter of source domain GMMs towards the target. The second strategy uses dictionary learning to express each GMM in MSDA as the barycenter of learned GMMs. Overall, we propose methods that are **lighter, better, faster** than previous empirical OT methods in MSDA.

References

1. Agueh, M., Carlier, G.: Barycenters in the Wasserstein space. SIAM J. Math. Anal. **43**(2), 904–924 (2011)
2. Castellon, F.E., Fernandes Montesuma, E., Mboula, F.N., Mayoue, A., Souloumiac, A., Gouy-Pailler, C.: Federated dataset dictionary learning for multi-source domain adaptation. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 5610–5614 (2024)
3. Chen, Y., Georgiou, T.T., Tannenbaum, A.: Optimal transport for Gaussian mixture models. IEEE Access **7**, 6269–6278 (2018)
4. Courty, N., Flamary, R., Tuia, D., Rakotomamonjy, A.: Optimal transport for domain adaptation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(9), 1853–1865 (2016)
5. Crammer, K., Kearns, M., Wortman, J.: Learning from multiple sources. J. Mach. Learn. Res. **9**(8) (2008)
6. Cuturi, M., Doucet, A.: Fast computation of Wasserstein barycenters. In: International Conference on Machine Learning, pp. 685–693. PMLR (2014)
7. Delon, J., Desolneux, A.: A Wasserstein-type distance in the space of Gaussian mixture models. SIAM J. Imag. Sci. **13**(2), 936–970 (2020)
8. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc.: Ser. B (Methodol.) **39**(1), 1–22 (1977)
9. Feng, H., et al.: KD3A: unsupervised multi-source decentralized domain adaptation via knowledge distillation. In: ICML, pp. 3274–3283 (2021)
10. Gardner, P., Bull, L.A., Dervilis, N., Worden, K.: Domain-adapted Gaussian mixture models for population-based structural health monitoring. J. Civ. Struct. Heal. Monit. **12**(6), 1343–1353 (2022)
11. 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 (2016)
12. Liu, X., Xi, W., Li, W., Xu, D., Bai, G., Zhao, J.: Co-MDA: federated multi-source domain adaptation on black-box models. IEEE Trans. Circuits Syst. Video Technol. (2023)
13. McInnes, L., Healy, J., Melville, J.: UMAP: uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426 (2018)
14. Montesuma, E.F., Mboula, F., Souloumiac, A.: Multi-source domain adaptation through dataset dictionary learning in Wasserstein space. In: European Conference on Artificial Intelligence, pp. 1739–1745 (2023)
15. Montesuma, E.F., Mboula, F.M.N.: Wasserstein barycenter for multi-source domain adaptation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16785–16793 (2021)
16. Montesuma, E.F., Mboula, F.M.N.: Wasserstein barycenter transport for acoustic adaptation. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3405–3409 (2021)

17. Montesuma, E.F., Mboula, F.N., Souloumiac, A.: Recent advances in optimal transport for machine learning. arXiv preprint arXiv:2306.16156 (2023)
18. Montesuma, E.F., Mulas, M., Corona, F., Mboula, F.M.N.: Cross-domain fault diagnosis through optimal transport for a CSTR process. IFAC-PapersOnLine **55**(7), 946–951 (2022)
19. Montesuma, E.F., Mulas, M., Mboula, F.N., Corona, F., Souloumiac, A.: Multi-source domain adaptation for cross-domain fault diagnosis of chemical processes. arXiv preprint arXiv:2308.11247 (2023)
20. Montesuma, E.F., Ngolè Mboula, F.M., Souloumiac, A.: Optimal transport for domain adaptation through gaussian mixture models. arXiv preprint arXiv:2308.11247 (2024)
21. Montesuma, E.F., Ngolè Mboula, F., Souloumiac, A.: Multi-source domain adaptation meets dataset distillation through dataset dictionary learning. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 5620–5624 (2024)
22. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2009)
23. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: IEEE/CVF International Conference on Computer Vision, pp. 1406–1415 (2019)
24. Peyré, G., Cuturi, M., et al.: Computational optimal transport: with applications to data science. Found. Trends® Mach. Learn. **11**(5-6), 355–607 (2019)
25. Quinonero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.D.: Dataset Shift in Machine Learning. MIT Press, Cambridge (2008)
26. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 213–226. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_16
27. Schmitz, M.A., et al.: Wasserstein dictionary learning: optimal transport-based unsupervised nonlinear dictionary learning. SIAM J. Imag. Sci. **11**(1), 643–678 (2018)
28. Shen, J., Qu, Y., Zhang, W., Yu, Y.: Wasserstein distance guided representation learning for domain adaptation. In: Proceedings of the AAAI Conference on Artificial Intelligence (2018)
29. Takatsu, A.: Wasserstein geometry of Gaussian measures. Osaka J. Math. **48**(4), 1005–1026 (2011)
30. Turrisi, R., Flamary, R., Rakotomamonjy, A., et al.: Multi-source domain adaptation via weighted joint distributions optimal transport. In: The 38th Conference on Uncertainty in Artificial Intelligence (2022)
31. Vapnik, V.: Principles of risk minimization for learning theory. In: Advances in Neural Information Processing Systems, vol. 4 (1991)
32. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5018–5027 (2017)
33. Villani, C.: Optimal Transport: Old and New, vol. 338. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-71050-9
34. Wang, H., Xu, M., Ni, B., Zhang, W.: Learning to combine: knowledge aggregation for multi-source domain adaptation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12353, pp. 727–744. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58598-3_43

Subgraph Retrieval Enhanced by Graph-Text Alignment for Commonsense Question Answering

Boci Peng[1], Yongchao Liu[2], Xiaohe Bo[3], Sheng Tian[2], Baokun Wang[2], Chuntao Hong[2], and Yan Zhang[1](✉)

[1] School of Intelligence Science and Technology, Peking University, Beijing, China
bcpeng@stu.pku.edu.cn, zhyzhy001@pku.edu.cn
[2] Ant Group, Hangzhou, China
{yongchao.ly,tiansheng.ts,yike.wbk,chuntao.hct}@antgroup.com
[3] School of Artificial Intelligence, Beijing Normal University, Beijing, China
xiaohe@mail.bnu.edu.cn

Abstract. Commonsense question answering is a crucial task that requires machines to employ reasoning according to commonsense. Previous studies predominantly employ an extracting-and-modeling paradigm to harness the information in KG, which first extracts relevant subgraphs based on pre-defined rules and then proceeds to design various strategies aiming to improve the representations and fusion of the extracted structural knowledge. Despite their effectiveness, there are still two challenges. On one hand, subgraphs extracted by rule-based methods may have the potential to overlook critical nodes and result in uncontrollable subgraph size. On the other hand, the misalignment between graph and text modalities undermines the effectiveness of knowledge fusion, ultimately impacting the task performance. To deal with the problems above, we propose a novel framework: **S**ubgraph R**E**trieval Enhanced by Gra**P**h-**T**ext **A**lignment, named **SEPTA**. Firstly, we transform the knowledge graph into a database of subgraph vectors and propose a BFS-style subgraph sampling strategy to avoid information loss, leveraging the analogy between BFS and the message-passing mechanism. In addition, we propose a bidirectional contrastive learning approach for graph-text alignment, which effectively enhances both subgraph retrieval and knowledge fusion. Finally, all the retrieved information is combined for reasoning in the prediction module. Extensive experiments on five datasets demonstrate the effectiveness and robustness of our framework.

Keywords: Commonsense Question Answering · Pre-trained Language Models · Graph Neural Networks

1 Introduction

Commonsense question answering (CSQA) is a critical task in natural language understanding, which requires systems to acquire different types of commonsense knowledge and possess multi-hop reasoning ability [19,22,27]. Though massive pre-trained models have achieved impressive performance on this task, it is difficult to learn commonsense knowledge solely from the pre-training text corpus, as the commonsense knowledge is evident to humans and rarely expressed explicitly in natural language. Compared with unstructured text, structured data like knowledge graphs is much more efficient in representing commonsense [26]. The incorporation of external knowledge aids PLMs in comprehending question-answer (Q-A) pairs, while the entity relations enhance the model's reasoning capabilities. Therefore, various commonsense knowledge graphs (CSKGs) (e.g., ConceptNet [25]) have been adopted in previous studies.

Existing KG-augmented models for CSQA primarily adhere to a extracting-and-modeling paradigm [26,28,29,32,35,36]. First, the knowledge subgraphs or paths related to a given question are extracted by string matching or semantic similarity, which indicate the relations between concepts or imply the process of multi-hop reasoning. Subsequently, diverse strategies emerge for the efficient representation and fusion of the extracted structural knowledge. One research path [8,12] involves elaborately crafting graph neural networks for better modeling the extracted subgraphs, whereas another [26,34] explores the efficient incorporation of knowledge from KG into language models by enhancing the interactions between PLMs and GNNs.

Despite their success, these approaches still have several limitations. First, the subgraph's quality suffers when retrieved through a simple string or semantic matching, posing limitations for subsequent operations. To obtain sufficient relevant knowledge, the number of nodes will expand dramatically with the increase of hop count, inevitably raising the burden of the model. Despite its ample size, certain crucial nodes might remain elusive, since some entities are not learned during the pre-training. Besides, the edges linked to the peripheral nodes within the subgraph are pruned, causing the message-passing mechanism of GNN to be blocked and impairing the attainment of effective representations, consequently undermining valuable information. Second, the misalignment between graph and text encoders presents a challenge for PLMs to internalize the knowledge contained in the acquired subgraph, especially in scenarios with limited data, leading to a reduced task performance [35]. Though Dragon [33] proposes a pre-training method to align GNNs and PLMs, it requires additional corpus, and the text-to-graph style to construct semantically equivalent graph-text pairs is challenging. The necessity for substantial computational resources poses another hurdle, prompting the search for a more efficient alignment method.

In this paper, we propose a novel framework: **S**ubgraph R**E**trieval Enhanced by Gra**P**h-**T**ext **A**lignment (**SEPTA**), for CSQA. To mitigate the shortcomings of the subgraph extraction process, we establish a database of subgraph vectors derived from the knowledge graph. Consequently, the challenge shifts from retrieving a pertinent subgraph to obtaining relevant subgraph vectors. A BFS-style sampling method is employed to obtain the connected graph for each node and the embedding of the subgraph is subsequently stored in the database. Drawing on the parallels between BFS and the message-passing mechanism of

GNNs, the central node's representation learned from the subgraph could be closely aligned with that derived from the entire graph, with almost no information loss. Besides, to further improve the retrieval accuracy and facilitate knowledge fusion during the prediction, we consider aligning the semantic space of the graph and text encoders, proposing an effective approach for graph-text alignment. A novel graph-to-text method is proposed to construct high-quality semantically equivalent training pairs, with no requirement of external corpus and easy to train. Finally, all the information retrieved is combined by a simple attention mechanism to facilitate the model in commonsense reasoning.

Our contributions can be summarized as follows:

– We propose a novel and effective framework SEPTA, where we convert the knowledge graph into a subgraph vector database and retrieve relevant subgraphs to facilitate commonsense reasoning.
– We design a bidirectional contrastive learning method to align the semantic space of the graph and text encoders, with a graph-to-text method to construct high-quality graph-text pairs, which facilitates subgraph retrieval and knowledge fusion.
– We propose a BFS-style subgraph sampling strategy for subgraph construction. Drawing on the parallel between BFS and the message-passing mechanism, our method can preserve complete neighbor information for each node.
– We conduct extensive experiments on five datasets. Our proposed approach achieves better results than the state-of-the-art approaches and has promising performance in weakly supervised settings.

2 Related Work

2.1 Commonsense Question Answering

Commonsense question answering aims to evaluate the reasoning ability of models based on commonsense knowledge [7], e.g., physical commonsense [2]. To incorporate external knowledge and enhance reasoning ability, some works introduce commonsense knowledge graphs (CSKGs, e.g. ConceptNet [25]). Generally, these methods [8,12,26,28,29,32–37] extract relevant knowledge subgraphs through entity linking and adopt graph neural networks to learn knowledge representations. Among them, a category of research focuses on designing more efficient knowledge encoders. For example, SAFE [12] proposes a 2-layer MLP to improve the efficiency of graph encoding. HamQA [8] considers learning hierarchical structures in KGs with hyperbolic geometry. Another research line tries to enhance the interactions between PLMs and GNNs. For instance, QA-GNN [34] adds a QA context node to the retrieved subgraphs and incorporates relevant information from other entities. Unlike previous works, we convert the knowledge graph into a subgraph database and transform the task to a subgraph vector retrieval problem, thus bypassing the challenges inherent in the extracting-and-modeling paradigm.

2.2 Graph-Text Alignment

Aligning the embedding spaces of text encoders and graph encoders is an effective way to take the strengths of two modalities [18]. Previous alignment methods can be roughly classified into two groups, i.e. the symmetric method and the asymmetric method, based on their training objectives. The symmetric alignments enhance each modality equally, most of which adopt a two-tower style and utilize contrastive learning techniques [3,5]. However, the asymmetric methods aim to take advantage of the capabilities of GNNs to reinforce PLMs. The predominant approaches can be categorized into two types, with the first type trying to enhance PLMs by inserting graph encoders into transformers [13,38] and the second type directly using GNNs as teacher models to generate soft labels for the PLMs [39]. In this paper, we leverage PLM as a teacher model and distill its knowledge into GNN, enabling us to retrieve related subgraphs in a manner akin to retrieving relevant text. Although DRAGON [33] also proposes a pre-training method to align PLMs and GNNs, our approach does not require additional corpus and demands lower computational costs.

3 Task Formulation

We study the multiple-choice CSQA [19,27], which can be formulated as: given a natural language question q and a set of answer candidates $C = \{c_1, c_2, \ldots, c_n\}$, the aim is to identify the optimal choice $c^* \in C$. Consistent with previous works [15], the CSQA problem is addressed in a *knowledge-aware* setting, that is, we can utilize external commonsense knowledge graphs (CSKGs) to facilitate model prediction. A CSKG can be formally described as a multi-relational graph $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E}, \boldsymbol{X})$, where $\mathcal{V}$ is the set of concept nodes (e.g., *Sun* and *Holiday*), $\mathcal{R}$ is the set of relation types (e.g., *HasProperty* and *AtLocation*), $\mathcal{E} \in \mathcal{V} \times \mathcal{E} \times \mathcal{V}$ is the link set of the knowledge graph (or fact triplets, e.g. *(House, MadeOf, Wood)*), and $\boldsymbol{X} \in \mathbb{R}^{|\mathcal{V}| \times d}$ denotes pre-trained embedddings of all concept nodes. Generally, the task can be treated as a score prediction task for each Q-A pair.

4 Methods

In this section, we will introduce the design of our SEPTA. Departing from previous extracting-and-modeling approaches, we reframe the task as a subgraph vector retrieval problem and propose a graph-text alignment method to improve the retrieval accuracy and facilitate knowledge fusion for prediction. For ease of exposition, we first introduce the graph-text alignment process in Sect. 4.1. Then with the aligned encoder, the subgraph vector database is constructed and retrieved, which will be presented in Sect. 4.2. Finally, how to combine all the structural information retrieved for answer prediction is discussed in Sect. 4.3. Figure 1 shows the overview of our SEPTA.

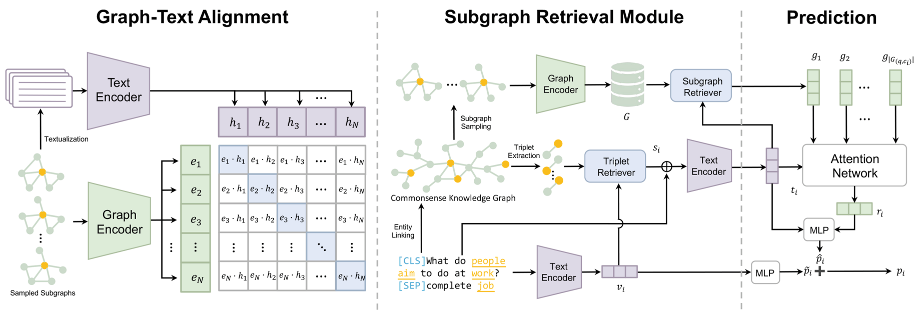

Graph-Text Alignment
Subgraph Retrieval Module
Prediction
Text Encoder
Textualization
Graph Encoder
Sampled Subgraphs
h_1 h_2 h_3 ⋯ h_N
e_1 e_2 e_3 e_N
$e_1 \cdot h_1$ $e_1 \cdot h_2$ $e_1 \cdot h_3$ ⋯ $e_1 \cdot h_N$
$e_2 \cdot h_1$ $e_2 \cdot h_2$ $e_2 \cdot h_3$ ⋯ $e_2 \cdot h_N$
$e_3 \cdot h_1$ $e_3 \cdot h_2$ $e_3 \cdot h_3$ ⋯ $e_3 \cdot h_N$
$e_N \cdot h_1$ $e_N \cdot h_2$ $e_N \cdot h_3$ ⋯ $e_N \cdot h_N$
Graph Encoder
G
Subgraph Retriever
Subgraph Sampling
Triplet Extraction
Triplet Retriever
s_i
Text Encoder
Commonsense Knowledge Graph
Entity Linking
[CLS]What do people aim to do at work?
[SEP]complete job
Text Encoder
v_i
g_1 g_2 $g_{|G_{(q,c_i)}|}$
Attention Network
t_i
r_i
MLP
$\hat{p}_i$
MLP
$\tilde{p}_i$
p_i

Fig. 1. The overview of our proposed SEPTA. First, a bidirectional contrastive method is proposed to align the semantic space of graph and text encoders. With the encoders aligned, we then transform the knowledge graph into a subgraph vector database and introduce a query enhancement strategy for better subgraph retrieval. Finally, all the information retrieved is combined by a simple attention mechanism to bolster the reasoning ability of PLMs for CSQA.

4.1 Graph-Text Alignment

To coordinate the embedding spaces of graph and text encoders and fully harness the respective strengths of text and KG, we propose an alignment process before downstream tasks. In our method, we initially address the challenge of generating training graph-text pairs with equivalent semantics and subsequently employ a bidirectional contrastive learning method to train the encoders of both modalities. The alignment process plays a pivotal role, as for one thing, it decides the efficacy of retrieving question-related subgraphs from the vector base, and for another, it determines the successful integration of graph information with the question context during the prediction phase.

Construction of Graph-Text Pairs. The construction of high-quality semantically equivalent graph-text pairs is crucial for the alignment process, yet not that straightforward. Previous methods mostly adopt a text-to-graph approach to construct training pairs, where the goal is to discover a graph structure that corresponds to the semantics of a provided text segment. However, utilizing existing transformation tools e.g. dependency graph could not well accommodate the downstream subgraph retrieval, while rule-based methods to extract text-related subgraphs from CSKGs are challenging. It is also notably time-consuming and laborious to construct through manual annotation. Therefore, in this paper, we propose a graph-to-text approach and consider constructing synonymous text descriptions of the subgraphs.

Specifically, we propose a BFS-style sampling strategy for subgraph construction, which initiates from the central node and proceeds to sample neighbors layer by layer. During the process, to address the challenge of an excessive number of neighbors, we set p as the probability for immediate neighbor selection. In addition, since it is sufficient to describe local neighborhoods for determining structural equivalence [10], we set a parameter d to constrain the depth of the sampled subgraphs. By restricting the search to nearby nodes, our sampling method achieves this characterization and obtains a microscopic view of the neighborhood of every node. Furthermore, a parameter n is established to regulate the size of the subgraphs. Once the number of sampled nodes reaches n, the layer-wise sampling is halted. Since nodes in the sampled neighbors tend to repeat many times, our method could reduce the variance in characterizing the distribution of 1-hop nodes with respect to the source node. The process can be formulated as:

$$\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i, \boldsymbol{A}_i, \boldsymbol{X}_i) = \mathrm{BFS}(v_i, p, d, n), \tag{1}$$

where $\mathcal{G}_i$ is the connected graph obtained, $\mathcal{V}_i$, $\mathcal{E}_i$ are sets of concept nodes and relation links in $\mathcal{G}_i$, $\boldsymbol{A}_i$ represents the adjacent matrix, $\boldsymbol{X}_i$ denotes embeddings of concept nodes, and v_i is the central concept node.

After that, it is necessary to textualize the subgraphs to construct synonymous text descriptions. The first step is to convert all relation links into triplet descriptions, which are later combined to compose the final description. Specifically, to transform relation links into sentences, we first map each relation type to a relation template and then concatenate the head concept, relation template, and tail concept as the description of each fact triplet. The textualization process can be denoted as:

$$s_i = \bigoplus_{e_j \in \mathcal{E}_i} \mathrm{TEXT}(e_j), \tag{2}$$

where s_i is the description of the graph $\mathcal{G}_i$ and $\oplus$ denotes the concatenation of sentences. Therefore, the training set can be denoted as $\{(\mathcal{G}_i, s_i)\}_{i=1}^{n}$. The overview of the construction process is shown in Fig. 2.

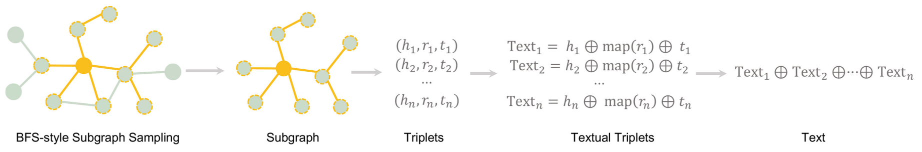

(h_1, r_1, t_1)
(h_2, r_2, t_2)
$\cdots$
(h_n, r_n, t_n)
$\mathrm{Text}_1 = h_1 \oplus \mathrm{map}(r_1) \oplus t_1$
$\mathrm{Text}_2 = h_2 \oplus \mathrm{map}(r_2) \oplus t_2$
$\cdots$
$\mathrm{Text}_n = h_n \oplus \mathrm{map}(r_n) \oplus t_n$
$\mathrm{Text}_1 \oplus \mathrm{Text}_2 \oplus \cdots \oplus \mathrm{Text}_n$
BFS-style Subgraph Sampling
Subgraph
Triplets
Textual Triplets
Text

Fig. 2. The overview of the construction of graph-text pairs.

Graph-Text Contrastive Learning. The graph-text alignment procedure is presented in the left part of Fig. 1. First, GNN and PLM are utilized to encode the knowledge subgraphs and natural language descriptions to obtain the corresponding representation, respectively, which can be formulated as:

$$\begin{aligned} \tilde{\boldsymbol{e}}_{\boldsymbol{i}} &= \mathrm{Pool}_G(\mathrm{GNN}(\mathcal{G}_i)), \\ \tilde{\boldsymbol{h}}_{\boldsymbol{i}} &= \mathrm{Pool}_T(\mathrm{PLM}(s_i)), \end{aligned} \tag{3}$$

where $\tilde{\boldsymbol{e}}_i$ is the average of all nodes' embeddings and $\tilde{\boldsymbol{h}}_i$ is the representation of the [CLS] token. To project $\tilde{\boldsymbol{e}}_i$ and $\tilde{\boldsymbol{h}}_i$ into the same semantic space, two linear projection layers are designed as follows:

$$\begin{aligned} \boldsymbol{e}_i &= \boldsymbol{W}_G \tilde{\boldsymbol{e}}_i + \boldsymbol{b}_G, \\ \boldsymbol{h}_i &= \boldsymbol{W}_T \tilde{\boldsymbol{h}}_i + \boldsymbol{b}_T. \end{aligned} \tag{4}$$

where $\boldsymbol{W}_G$, $\boldsymbol{W}_T$ and $\boldsymbol{b}_G$, $\boldsymbol{b}_T$ are the transform matrices and biases of the linear projection layers.

We then employ InfoNCE with in-batch negative sampling to align the representations of two modalities bidirectionally. The graph-to-text contrastive loss can be formulated as:

$$\mathcal{L}_{\mathrm{G2T}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(sim(\boldsymbol{e}_i, \boldsymbol{h}_i)/\tau)}{\sum_{j=1}^{N} \exp(sim(\boldsymbol{e}_i, \boldsymbol{h}_j)/\tau)}, \tag{5}$$

where τ is a temperature coefficient and N is the number of instances in a batch. Besides, function $sim(\cdot,\cdot)$ measures the similarity between two representations, which can be calculated by:

$$sim(\boldsymbol{e}_i, \boldsymbol{h}_i) = \frac{\boldsymbol{e}_i^T \boldsymbol{h}_i}{\|\boldsymbol{e}_i\| \cdot \|\boldsymbol{h}_i\|}. \tag{6}$$

Similarly, we also design a text-to-graph contrastive loss for uniformly aligning into the same semantic space, which is shown as:

$$\mathcal{L}_{\mathrm{T2G}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(sim(\boldsymbol{h}_i, \boldsymbol{e}_i)/\tau)}{\sum_{j=1}^{N} \exp(sim(\boldsymbol{h}_i, \boldsymbol{e}_j)/\tau)}. \tag{7}$$

The final contrastive loss $\mathcal{L}_{GT}$ is defined as the average of $\mathcal{L}_{G2T}$ and $\mathcal{L}_{T2G}$:

$$\mathcal{L}_{GT} = \frac{1}{2}(\mathcal{L}_{G2T} + \mathcal{L}_{T2G}). \tag{8}$$

Remarks. (1) To avoid the loss of inherent knowledge caused by over-fitting of the PLM, only the GNN and the linear projection layers are trainable during actual implementation. From another perspective, we essentially distill the semantic information from the PLM into the GNN, enabling the graph representations encoded by the GNN to encompass both structural and textual information. (2) As our ultimate goal is to obtain subgraph representations, we employ graph-level contrastive learning to align subgraph embeddings with text embeddings, yielding promising results. We also attempt other granular alignment signals, such as aligning entity node representations with text representations or applying the Masked Language Model (MLM) to text based on subgraph representations. However, with our current computational resources, these methods are unable to converge effectively. Through our contrastive learning, the model is able to rapidly converge and achieve promising performance.

4.2 Subgraph Retrieval Module

In this section, we initially present the establishment of the subgraph vector database. After that, to better accommodate the alignment process, we propose the query enhancement. Finally, we will outline the subgraph retrieval procedure.

Database Construction. Previous methodologies primarily adopt an extracting-and-modeling paradigm for knowledge subgraph retrieval. However, as the cornerstone of subsequent work, the retrieval of high-quality subgraphs proves to be challenging. Consequently, we suggest transforming the knowledge graph into a subgraph vector database, thereby transitioning the focus towards retrieving pertinent subgraphs. Leveraging the analogy between BFS and the message-passing mechanism, we adopt a BFS-style subgraph sampling strategy to construct subgraphs, instead of DFS or Random Walk. On the one hand, each subgraph contains complete neighbor information for at least one node, and on the other hand, each node appears in at least one subgraph. Therefore, the information supplied for our retrieval is complete, and each subgraph vector holds fine-grained knowledge regarding the central node. Specifically, we first apply the same method as in Sect. 4.1 to produce the graph embedding $\boldsymbol{e}_i$ and the text embedding $\boldsymbol{h}_i$. Then we add them up as the subgraph vector $\boldsymbol{g}_i \in \mathbb{R}^d$:

$$\boldsymbol{g}_i = \frac{1}{2}(\frac{\|\boldsymbol{h}_i\|}{\|\boldsymbol{e}_i\|}\boldsymbol{e}_i + \boldsymbol{h}_i), \tag{9}$$

where the regularization coefficient $\frac{\|\boldsymbol{h}_i\|}{\|\boldsymbol{e}_i\|}$ maintains the consistency between the norm of the subgraph vectors and the norm of the text representations, preventing the prediction from relying predominantly on the features with larger norms. Finally, a subgraph vector database $\boldsymbol{G} = \{\boldsymbol{g}_i\}_{i=1}^{|G|}$ is constructed with all subgraph vectors. Note that we only need to generate subgraph vectors once before performing downstream tasks, which saves computational resources.

Query Enhancement. Given a problem, we need to find the relevant subgraph vectors. An intuitive method is to apply the embedding of the question-answer pair as a query. However, there is a certain difference between such textual query and the pre-trained corpus of the aligned encoder, as the latter is constructed through triplet concatenation, which makes it difficult to ensure the quality of text encoding and reduces the accuracy of the retrieval process. Therefore, we propose to enhance the query by retrieving question-related triplets in the knowledge graph and concatenating them after the Q-A pairs. Specifically, given a pair of question-answers (q, c_i), we first apply entity linking to find all entities $E_q = \{e_q^{(1)}, e_q^{(2)}, \dots, e_q^{(n_q)}\}$, $E_{c_i} = \{e_{c_i}^{(1)}, e_{c_i}^{(2)}, \dots, e_{c_i}^{(n_{c_i})}\}$ appearing in question q and choice c, respectively. Then, we find all triplets in the CSKG containing the entities in E_q and E_{c_i}, which can be formulated as $T = \{(e^*, r, e), (e, r, e^*) | e \in E_q \cup E_{c_i}\}$. All fact triplets in T are serialized to natural language sentences and a pre-trained dense retriever is adopted to find the most relevant ones. We concatenate the fact triplets retrieved together, along with the questions and options: $s_i = q \oplus c_i \oplus \text{text}_1 \oplus \text{text}_2 \oplus \cdots \oplus \text{text}_K$. The aligned PLM is then utilized to encode s_i to $\boldsymbol{t_i} = \text{PLM}(s_i) \in \mathbb{R}^d$.

Subgraph Retrieval. After that, we employ the embedding of Q-A pairs concatenated with factual triples to retrieve the relevant subgraph vectors from the subgraph vector database. As the embedding space of two modalities has been aligned, the cosine similarity of $\boldsymbol{t}_i$ with each subgraph vector $\boldsymbol{g}_i$ in $\boldsymbol{G}$ is competent for the retrieval. We recall the top k subgraph vectors with the highest similarities, which is denoted as $\boldsymbol{G}_{q,c_i} \in \mathbb{R}^{k\times d}$.

4.3 Prediction

We combine all the knowledge retrieved to make the final predictions. We first integrate the retrieved subgraph vectors through multi-head attention with $\boldsymbol{t}_i$ as the query, which can be formulated as:

$$\begin{aligned}\boldsymbol{\alpha}_i^{(h)} &= \frac{(\boldsymbol{t}_i\boldsymbol{W}_Q^{(h)})(\boldsymbol{G}_{q,c_i}\boldsymbol{W}_K^{(h)})^T}{\sqrt{d}},\\ \boldsymbol{r}_i^{(h)} &= \text{Softmax}(\boldsymbol{\alpha}_i^{(h)})(\boldsymbol{G}_{q,c_i}\boldsymbol{W}_V^{(h)}),\\ \boldsymbol{r}_i &= \text{Concat}(\boldsymbol{r}_i^{(1)}, \boldsymbol{r}_i^{(2)}, \ldots, \boldsymbol{r}_i^{(H)})\boldsymbol{W}_O,\end{aligned} \tag{10}$$

where $\boldsymbol{W}_Q^{(h)}, \boldsymbol{W}_K^{(h)}, \boldsymbol{W}_V^{(h)} \in \mathbb{R}^{d\times d}$ are projection matrices under head h.

Subsequently, $\boldsymbol{r}_i$ and $\boldsymbol{t}_i$ are added and fed into a linear layer to predict the score of option c_i:

$$\hat{p}_i = \boldsymbol{W_1}^T(\boldsymbol{t}_i + \boldsymbol{r}_i) + b_1. \tag{11}$$

Since some questions are expected to be answered based solely on the question context, we also encode the Q-A pair (q, c_i) to infer directly:

$$\begin{aligned}\boldsymbol{v_i} &= \text{PLM}(q, c_i),\\ \tilde{p}_i &= \boldsymbol{W_2}^T\boldsymbol{v}_i + b_2.\end{aligned} \tag{12}$$

The two scores are weighted and summed to yield the final score:

$$p_i = \lambda\hat{p}_i + (1-\lambda)\tilde{p}_i, \tag{13}$$

where λ is the hyper-parameter for the balance.

During the training phase, we employ the softmax function to normalize the score for each choice and optimize the model by cross-entropy loss. For inference, we determine the prediction by selecting the choice with the highest score.

Table 1. Statistics of the datasets. '-' denotes the unavailable dataset split.

Task	Train	Dev	Test
CommonsenseQA official split	9,741	1,221	1,140
CommonsenseQA in-house split	8,500	1,221	1,241
OpenBookQA	4,957	500	500
SocialIQA	33,410	1,954	–
PIQA	16,113	1,838	–
RiddleSenseQA	3,510	1,021	–

5 Experiments

5.1 Datasets

We conduct experiments to evaluate our method on five CSQA datasets, which are shown in Table 1:

- **CommonsenseQA** [27] is a 5-way multiple-choice QA dataset, which is created based on ConceptNet [25]. Due to the dual split of CommonsenseQA: the official split [27] and the in-house (IH) split [15], we report the results for both settings. For the official split, the ground truth of the test set is not publicly available, so we submit our model's predictions to the official leaderboard[1] to evaluate the test accuracy.
- **OpenBookQA** [19] is a 4-choice dataset about elementary science questions to evaluate the science commonsense knowledge. We also submit the predictions of the test set to the official leaderboard[2].
- **SocialIQA** [22] is a 3-choice dataset to evaluate the understanding of commonsense social knowledge. Due to the unavailability of the test set, consistent with prior works [24], we report the accuracy of the development set.
- **PIQA** [2] is a 2-choice QA dataset regarding physical commonsense. Since the test set is hidden, evaluations are conducted on the development set.
- **RiddleSenseQA** [16] is a 5-choice QA dataset about commonsense riddles. Because the test set is not released, we only report the validation accuracy.

5.2 Baselines

We compare with the mainstream RoBERTa-Large + GNN methods, including RN [21], RGCN [23], GconAttn [31], MHGRN [9], QA-GNN [34], DGRN [37], GreaseLM [36], JointLK [26], GSC [29], SAFE [12], DRAGON [33], HamQA [8], and DHLK [32]. Among them, DRAGON introduces BookCorpus to joint-train GNN and PLM, and DHLK additionally retrieves paraphrases of key entities in WordNet and Wiktionary.

[1] https://www.tau-nlp.sites.tau.ac.il/csqa-leaderboard.
[2] https://leaderboard.allenai.org/open_book_qa/submissions/public.

5.3 Implementation Details

According to the previous works, we use RoBERTa-Large [17] as the text encoder and use GraphGPS [20] for the graph encoder. We also test AristoRoBERTa [6] for OpenBookQA. We use ConceptNet [25], including 799,273 nodes and 2,487,003 edges, as the commonsense knowledge graph. For each node, we treat it as the center and employ BFS to obtain the subgraph, which is then translated into the corresponding natural language description.

In the graph-text alignment phase, we randomly sample 64,000 graph-text pairs to train, and sample 16,000 pairs to evaluate two encoders. We fix the learning rate to 1e-3, the number of GNN layers to 2, and the dimensions of all embeddings to 1024. During the fine-tuning stage, we set the number of fact triplets to 10, tune the number of retrieved subgraph vectors k in $\{10, 30, 50, 70, 100\}$, the batch size in $\{4, 8, 16\}$, the balance coefficient λ from 0.1 to 1.0, and the learning rate in {2e-5, 1e-5, 5e-6, 2e-6, 1e-6}. The parameters of the model are optimized by RAdam. We train 30 epochs until the performance does not improve on the development sets for 3 consecutive epochs. We use the default parameter settings as their original implementations for the baseline methods. We conduct all experiments on NVIDIA A100-40GB GPUs.

5.4 Main Results

Following previous works [11,12,29,34], we compare our method with different baselines on CommonsenseQA and OpenBookQA as main results, which are shown in Table 2. The best and runner-up results in each column are highlighted in bold and underlined, respectively.

From the results, we can observe: (1) Our method can contribute performance gains to LMs, which improves 6.54% and 6.09% on IHdev and IHtest of CommonsenseQA compared to fine-tuned RoBERTa. (2) SEPTA outperforms all baselines without additional corpus on both datasets. For example, compared to the GSC method, our method improves by 2.00% and 0.70% on OpenBookQA using RoBERTa and AristoRoBERTa, respectively. (3) Compared to baselines incorporating additional corpus, our method also achieves comparable performance. Specifically, we surpass DHLK on both datasets and DRAGON on OpenBookQA and slightly lag behind DRAGON on CommonsenseQA. It should be noted that the DRAGON undergoes MLM training on the BookCorpus dataset and requires training on 8$\times$A100 GPUs for a week [32,33]. By eliminating the MLM, our SEPTA model demonstrates a definitive enhancement.

In Table 3, we evaluate SEPTA on the official CommonsenseQA and OpenBookQA leaderboards (as of March 22, 2024). Our method achieves results surpassing all baselines based on the same PLM and exhibits comparative performance compared with methods with larger-scale parameters (e.g., UnifiedQA).

To comprehensively evaluate the efficiency of SEPTA, we extend our comparative analysis to other commonsense reasoning datasets originating from diverse domains or tasks. As shown in Table 4, our SEPTA consistently achieves superior performance. This observation underscores the overall effectiveness of SEPTA in

Table 2. Evaluation on CommonsenseQA (in-house split) and OpenBookQA. We use RoBERTa-Large as the text encoder in CommonsenseQA, and use RoBERTA-Large and AristoRoBERTa in OpenBookQA. Methods with AristoRoBERTa use the textual evidence by [6] as an additional input to the QA context. The baselines incorporating extra corpus are marked with *.

Methods	CommonsenseQA		OpenBookQA	
	IHdev-Acc (%)	IHtest-Acc (%)	RoBERTa-Large (%)	AristoRoBERTa (%)
Fine-tuned LMs	73.07 (±0.45)	68.69 (±0.56)	64.80 (±2.37)	78.40 (±1.64)
+ RN	74.57 (±0.91)	69.08 (±0.21)	65.20 (±1.18)	75.35 (±1.39)
+ RGCN	72.69 (±0.19)	68.41 (±0.66)	62.45 (±1.57)	74.60 (±2.53)
+ GconAttn	72.61 (±0.39)	68.59 (±0.96)	64.75 (±1.48)	71.80 (±1.21)
+ MHGRN	74.45 (±0.10)	71.11 (±0.81)	66.85 (±1.19)	80.60
+ QA-GNN	76.54 (±0.21)	73.41 (±0.92)	67.80 (±2.75)	82.77 (±1.56)
+ DGRN	78.20	74.00	69.60	84.10
+ GreaseLM	78.50 (±0.50)	74.20 (±0.40)	68.80 (±1.75)	84.80
+ JointLK	77.88 (±0.25)	74.43 (±0.83)	70.34 (±0.75)	84.92 (±1.07)
+ GSC	79.11 (±0.22)	74.48 (±0.41)	70.33 (±0.81)	86.67 (±0.46)
+ SAFE	76.93 (±0.37)	74.03 (±0.43)	69.20	<u>87.13</u>
+ HamQA	76.88	73.91	71.12	84.59
+ DRAGON*	-	**76.00**	72.00	-
+ DRAGON (w/o MLM)*	-	73.80	66.40	-
+ DHLK*	<u>79.39</u> (±0.24)	74.68 (±0.26)	<u>72.20</u> (±0.40)	86.00 (±0.79)
+ SEPTA (**Ours**)	**79.61** (±0.17)	<u>74.78</u> (±0.23)	**72.33** (±0.35)	**87.37** (±0.51)

Table 3. Performance comparison on CommonsenseQA (left) and OpenBookQA (right) official leaderboard.

Methods	Test-Acc (%)	Methods	Test-Acc (%)
RoBERTa [17]	72.1	Careful Selection [1]	72.0
RoBERTa+FreeLB	72.2	AristoRoBERTa [6]	77.8
RoBERTa+HyKAS	73.2	KF+SIR	80.0
RoBERTa+KE	73.3	AristoRoBERTa+PG [30]	80.2
RoBERTa+KEDGN	74.4	AristoRoBERTa+MHGRN [9]	80.6
RoBERTa+MHGRN [9]	75.4	AristoRoBERTa+QA-GNN [34]	82.8
RoBERTa+QA-GNN [34]	76.1	AristoRoBERTa+GreaseLM [36]	84.8
RoBERTa+GSC [29]	76.2	AristoRoBERTa+GSC [29]	87.4
Albert	73.5	AristoRoBERTa+MVP-Tuning [11]	87.6
ALBERT+Path Generator [30]	75.6	ALBERT + KB	81.0
ALBERT+HGN [9]	77.3	T5	83.2
UnifiedQA (11B) [14]	**79.1**	UnifiedQA (11B) [14]	87.2
RoBERTa+SEPTA (**Ours**)	76.6	AristoRoBERTa+SEPTA (**Ours**)	**87.8**

addressing various commonsense reasoning datasets or tasks, demonstrating a unified methodology.

Table 4. Performance comparison on SocialIQA, PIQA, and RiddleSenseQA.

Methods	SocialIQA	PIQA	RiddleSenseQA
RoBERTa-Large	78.25	77.53	60.72
+ GconAttn	78.86	78.24	61.77
+ RN	78.45	76.88	62.17
+ MHGRN	78.11	77.15	63.27
+ QA-GNN	78.10	78.24	63.39
+ GreaseLM	77.89	78.02	63.88
+ GSC	78.61	78.40	64.07
+ SAFE	78.86	79.43	63.78
+ SEPTA (**Ours**)	**79.21**	**80.85**	**67.62**

Table 5. Ablation study on CommonsenseQA (IHtest) and OpenBookQA datasets. The values in parentheses denote the extent of performance decline.

Ablation	CommonsenseQA	OpenBookQA
SEPTA	74.78	72.33
w/o alignment	69.83 (−4.95)	67.20 (−5.13)
w/o subgraph	72.34 (−2.44)	70.23 (−2.10)
w/o triplets	71.25 (−3.53)	69.67 (−2.66)
$\lambda = 1.0$	74.13 (−0.65)	70.47 (−1.86)

5.5 Ablation Study

We conduct an ablation study on CommonsenseQA and OpenBookQA to explore the effectiveness of each component of SEPTA. We remove the alignment process (w/o alignment), retrieved subgraph vectors (w/o subgraph), fact triplets (w/o triplets), and scores predicted based on Q-A pairs (i.e. set $\lambda = 1.0$), respectively.

As shown in Table 5, four components are all crucial for SEPTA, and removing any part will result in a decrease in performance. Specifically, the performance drops the most significantly when we remove the graph-text alignment. This is because if the representations of graphs and texts are not semantically aligned, then during the knowledge retrieval stage, the retrieved subgraph vectors may be irrelevant and situated in different latent spaces from the textual information. Moreover, removing either fact triplets or subgraph vectors will affect the performance. On one hand, they represent different aspects of information, with the former providing more specific knowledge and the latter describing more comprehensive relationships between entities. On the other hand, fact triplets also play an auxiliary role in retrieving relevant subgraph vectors. Furthermore, only using knowledge-enhanced representations for predictions (i.e. $\lambda = 1.0$) cannot achieve optimal results. This is because some questions do not require additional knowledge, or relevant information cannot be found in CSKGs, which may instead become interference.

Table 6. Performance with different proportions of training data.

Methods	CommonsenseQA						OpenBookQA					
	5%	10%	20%	50%	80%	100%	5%	10%	20%	50%	80%	100%
RoBERTa-large	29.66	42.84	58.47	66.13	68.47	68.69	37.00	39.4	41.47	53.07	57.93	64.8
+ RGCN	24.41	43.75	59.44	66.07	68.33	68.41	38.67	37.53	43.67	56.33	63.73	62.45
+ GconAttn	21.92	49.83	60.09	66.93	69.14	68.59	38.60	36.13	43.93	50.87	57.87	64.75
+ RN	23.77	34.09	59.90	65.62	67.37	69.08	33.73	35.93	41.40	49.47	59.00	65.20
+ MHGRN	29.01	32.02	50.23	68.09	70.83	71.11	38.00	36.47	39.73	55.73	55.00	66.85
+ QA-GNN	32.95	37.77	50.15	69.33	70.99	73.41	33.53	35.07	42.40	54.53	52.47	67.80
+ GreaseLM	22.80	56.16	63.09	70.56	73.41	74.20	39.00	39.60	42.20	57.87	65.13	68.80
+ GSC	31.02	35.07	65.83	70.94	73.82	74.48	29.60	41.80	42.40	58.03	65.97	70.33
+ SAFE	36.45	56.51	65.16	70.72	73.22	74.03	38.80	41.20	44.93	58.33	65.60	69.20
+ SEPTA(**Ours**)	**50.69**	**62.37**	**68.09**	**71.80**	**74.05**	**74.78**	**45.63**	**54.80**	**58.10**	**66.57**	**68.30**	**72.33**

5.6 Low-Resource Setting

To evaluate the robustness of SEPTA, we conduct extensive experiments in low-resource settings, with different proportions of training data, including 5%, 10%, 20%, 50%, and 80%, in CommonsenseQA (IHtest) and OpenBookQA.

From the results in Table 6, we can observe that our SEPTA achieves promising performance in all settings, and it exhibits a trend where the performance improvement relative to other baselines is more significant with fewer training data. This is because we align text representations with graph representations before fine-tuning on downstream tasks, enabling retrieved subgraph vectors to integrate well with text representations even in low-resource settings. In contrast, other baselines make it hard to project subgraph representations and text representations into the same semantic space when training data is limited, resulting in structure embeddings becoming a noise that interferes with PLM reasoning.

5.7 Evaluation with other GNNs

To demonstrate the generality of SEPTA, We employ GraphGPS [20], FILM-GNN [4], and RGCN [23] as the graph encoders, respectively. Table 7 illustrates the results on CommonsenseQA and OpenBookQA. From the results, we can observe that different graph encoders achieve competitive results on both datasets, with their performances being relatively close (within a difference of around 0.5%), which demonstrates the effectiveness and robustness of SEPTA.

Table 7. Effect of different graph encoders.

GNN	CommonsenseQA	OpenBookQA
GraphGPS	74.78	72.33
FILM-GNN	74.67	72.17
RGCN	74.51	71.87

5.8 Hyper-parameter Analysis

We further conduct in-depth analyses to investigate the impact of hyper-parameters. With other parameters fixed, we compare the effect of the number of retrieved subgraph vectors k, the maximum number of nodes n in each subgraph, and the balance coefficient λ. The results on CommonsenseQA (IHtest) and OpenBookQA datasets are reported in Fig. 3.

Effect of Subgraph Vectors Number k From the results, we observe that the accuracy initially ascends with the increase in the number of retrieved subgraph vectors, achieving its peak before subsequently declining. This is because fewer subgraph vectors may lose crucial commonsense, while an excess of subgraph vectors could introduce irrelevant information.

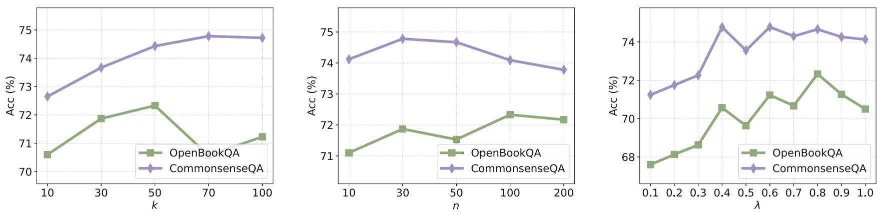
Effect of Maximum Number of Nodes n Based on the results, the performance of the model initially increases with the increment of n, reaching a peak, then decreases. It might be attributed to the fact that when n is relatively small, subgraphs are unable to fully encompass the neighbor information of the central nodes, leading to the inability to acquire sufficient relevant knowledge during the subgraph retrieval phase. Conversely, when n is excessively large, each subgraph may contain a significant amount of information irrelevant to the central node, resulting in overall information redundancy.

Effect of the Balance Coefficient λ λ controls the proportion of inference based on the retrieved knowledge. When λ is small, the model primarily relies on its own knowledge for inference, which may lead to a lack of relevant information for some questions. However, when λ is large, the model heavily depends on retrieved knowledge to derive answers, although many questions need to be resolved according to the question context. Therefore, in terms of results, the accuracy generally increases initially with the increase in λ and then decreases.

6 Ethical Considerations and Limitations

6.1 Ethical Considerations

Our work proposes a novel and effective framework to combine PLMs and external knowledge graphs for commonsense question answering. However, potential issues may arise from the utilization of PLMs and CSKGs. On one hand, PLMs tend to encapsulate certain biases present in the pre-training data. On the other



(a) Effect of k. (b) Effect of n. (c) Effect of λ.

Fig. 3. Hyper-parameter analysis.

hand, CSKGs may harbor biased concepts stemming from human annotations. To alleviate these biases, the implementation of appropriate screening rules offers a promising approach, e.g., filter biased concepts during the subgraph extraction process from the CSKG. Although a comprehensive analysis of such biases is not included in our work, it is imperative to implement supplementary measures before deploying the system in real-world scenarios.

6.2 Limitations

We propose a subgraph retrieval enhanced by a graph-text alignment framework named SEPTA for commonsense question answering. However, there are still limitations that demand resolution. Firstly, the corresponding text generated by rules from knowledge subgraphs still exhibits disparities from natural language. One possible solution is to reorganize the text using LLMs, but the cost is prohibitively high. Therefore, acquiring large-scale, high-quality graph-text pairs remains an ongoing challenge. Secondly, the number of retrieved subgraph vectors is required to tune according to the accuracy of development sets, which is time-consuming. Designing a module to automatically select the number may be a solution worth exploring. Thirdly, due to considerations of fairness in comparison and limited computational resources, we do not employ other PLMs, especially LLMs, as text encoders, which will be considered in our future work.

7 Conclusion

We propose an effective framework: subgraph retrieval enhanced by graph-text alignment, named SEPTA, for commonsense question answering. In our method, we reframe the task as a subgraph vector retrieval problem and introduce a graph-text alignment method to enhance retrieval accuracy and facilitate knowledge fusion for prediction. Subsequently, all the structural information retrieved is then combined by a simple attention mechanism to bolster the reasoning capabilities of PLMs. Extensive experiments on five benchmarks demonstrate the effectiveness of SEPTA.

In the future, our work will focus on the following aspects. First, we will explore more efficacious pre-training tasks for semantic alignment. Second, if there are sufficient computational resources, we intend to apply our approach to larger language models. Third, we will try SEPTA on relevant tasks, e.g., node classification and link predictions on text-attributed graphs.

Acknowledgments. This work is supported by Ant Group through Ant Research Intern Program.

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

References

1. Banerjee, P., Pal, K.K., Mitra, A., Baral, C.: Careful selection of knowledge to solve open book question answering. In: ACL (2019)
2. Bisk, Y., Zellers, R., Bras, R.L., Gao, J., Choi, Y.: PIQA: reasoning about physical commonsense in natural language. In: AAAI (2020)
3. Brannon, W., et al.: Congrat: Self-supervised contrastive pretraining for joint graph and text embeddings. CoRR **abs/2305.14321** (2023)
4. Brockschmidt, M.: Gnn-film: Graph neural networks with feature-wise linear modulation. In: ICML (2020)
5. Chandra, S., Mishra, P., Yannakoudakis, H., Nimishakavi, M., Saeidi, M., Shutova, E.: Graph-based modeling of online communities for fake news detection. CoRR **abs/2008.06274** (2020)
6. Clark, P., et al.: From 'f' to 'a' on the N.Y. regents science exams: an overview of the aristo project. AI Mag. **41**(4), 39–53 (2020)
7. Davis, E., Marcus, G.: Commonsense reasoning and commonsense knowledge in artificial intelligence. Commun. ACM **58**(9), 92–103 (2015)
8. Dong, J., Zhang, Q., Huang, X., Duan, K., Tan, Q., Jiang, Z.: Hierarchy-aware multi-hop question answering over knowledge graphs. In: WWW (2023)
9. Feng, Y., Chen, X., Lin, B.Y., Wang, P., Yan, J., Ren, X.: Scalable multi-hop relational reasoning for knowledge-aware question answering. In: EMNLP (2020)
10. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: SIGKDD (2016)
11. Huang, Y., et al.: Mvp-tuning: multi-view knowledge retrieval with prompt tuning for commonsense reasoning. In: ACL (2023)
12. Jiang, J., Zhou, K., Wen, J., Zhao, W.X.: Great truths are always simple: a rather simple knowledge encoder for enhancing the commonsense reasoning capacity of pre-trained models. In: NAACL (2022)
13. Jin, B., et al.: Patton: language model pretraining on text-rich networks. In: ACL (2023)
14. Khashabi, D., et al.: Unifiedqa: crossing format boundaries with a single QA system. In: EMNLP (2020)
15. Lin, B.Y., Chen, X., Chen, J., Ren, X.: Kagnet: knowledge-aware graph networks for commonsense reasoning. In: EMNLP (2019)
16. Lin, B.Y., Wu, Z., Yang, Y., Lee, D., Ren, X.: Riddlesense: reasoning about riddle questions featuring linguistic creativity and commonsense knowledge. In: ACL (2021)
17. Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach. CoRR **abs/1907.11692** (2019)

18. Mao, Q., Liu, Z., Liu, C., Li, Z., Sun, J.: Advancing graph representation learning with large language models: a comprehensive survey of techniques. CoRR **abs/2402.05952** (2024)
19. Mihaylov, T., Clark, P., Khot, T., Sabharwal, A.: Can a suit of armor conduct electricity? EMNLP, A new dataset for open book question answering. In (2018)
20. Rampásek, L., Galkin, M., Dwivedi, V.P., Luu, A.T., Wolf, G., Beaini, D.: Recipe for a general, powerful, scalable graph transformer. In: NeurIPS (2022)
21. Santoro, A., et al.: A simple neural network module for relational reasoning. In: NeurIPS (2017)
22. Sap, M., Rashkin, H., Chen, D., Bras, R.L., Choi, Y.: Social iqa: commonsense reasoning about social interactions. In: EMNLP (2019)
23. Schlichtkrull, M.S., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: ESWC (2018)
24. Shwartz, V., West, P., Bras, R.L., Bhagavatula, C., Choi, Y.: Unsupervised commonsense question answering with self-talk. In: EMNLP (2020)
25. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge. In: AAAI (2017)
26. Sun, Y., Shi, Q., Qi, L., Zhang, Y.: Jointlk: Joint reasoning with language models and knowledge graphs for commonsense question answering. In: NAACL (2022)
27. Talmor, A., Herzig, J., Lourie, N., Berant, J.: Commonsenseqa: a question answering challenge targeting commonsense knowledge. In: NAACL (2019)
28. Taunk, D., Khanna, L., Kandru, S.V.P.K., Varma, V., Sharma, C., Tapaswi, M.: Grapeqa: graph augmentation and pruning to enhance question-answering. In: WWW (2023)
29. Wang, K., Zhang, Y., Yang, D., Song, L., Qin, T.: GNN is a counter? revisiting GNN for question answering. In: ICLR (2022)
30. Wang, P., Peng, N., Ilievski, F., Szekely, P.A., Ren, X.: Connecting the dots: a knowledgeable path generator for commonsense question answering. In: EMNLP (2020)
31. Wang, X., et al.: Improving natural language inference using external knowledge in the science questions domain. In: AAAI (2019)
32. Wang, Y., Zhang, H., Liang, J., Li, R.: Dynamic heterogeneous-graph reasoning with language models and knowledge representation learning for commonsense question answering. In: ACL (2023)
33. Yasunaga, M., et al.: Deep bidirectional language-knowledge graph pretraining. In: NeurIPS (2022)
34. Yasunaga, M., Ren, H., Bosselut, A., Liang, P., Leskovec, J.: QA-GNN: reasoning with language models and knowledge graphs for question answering. In: NAACL (2021)
35. Ye, Q., Cao, B., Chen, N., Xu, W., Zou, Y.: Fits: fine-grained two-stage training for knowledge-aware question answering. In: AAAI (2023)
36. Zhang, X., et al.: Greaselm: graph reasoning enhanced language models. In: ICLR (2022)
37. Zheng, C., Kordjamshidi, P.: Dynamic relevance graph network for knowledge-aware question answering. In: COLING (2022)
38. Zhu, Y., Wang, Y., Shi, H., Tang, S.: Efficient tuning and inference for large language models on textual graphs. CoRR **abs/2401.15569** (2024)
39. Zou, T., Yu, L., Huang, Y., Sun, L., Du, B.: Pretraining language models with text-attributed heterogeneous graphs. In: EMNLP (2023)

HetCAN: A Heterogeneous Graph Cascade Attention Network with Dual-Level Awareness

Zeyuan Zhao[1], Qingqing Ge[1], Anfeng Cheng[2], Yiding Liu[2], Xiang Li[1](✉), and Shuaiqiang Wang[2]

[1] School of Data Science and Engineering, East China Normal University, Shanghai, China
{zeyuanzhao,qingqingge}@stu.ecnu.edu.cn, xiangli@dase.ecnu.edu.cn
[2] Baidu Inc., Beijing, China

Abstract. Heterogeneous graph neural networks (HGNNs) have recently shown impressive capability in modeling heterogeneous graphs that are ubiquitous in real-world applications. Most existing methods for heterogeneous graphs mainly learn node embeddings by stacking multiple convolutional or attentional layers, which can be considered as capturing the high-order information from node-level aspect. However, different types of nodes in heterogeneous graphs have diverse features, it is also necessary to capture interactions among node features, namely the high-order information from feature-level aspect. In addition, most methods first align node features by mapping them into one same low-dimensional space, while they may lose some type information of nodes in this way. To address these problems, in this paper, we propose a novel **Het**erogeneous graph **C**ascade **A**ttention **N**etwork (HetCAN) composed of multiple cascade blocks. Each cascade block includes two components, the type-aware encoder and the dimension-aware encoder. Specifically, the type-aware encoder compensates for the loss of node type information and aims to make full use of graph heterogeneity. The dimension-aware encoder is able to learn the feature-level high-order information by capturing the interactions among node features. With the assistance of these components, HetCAN can comprehensively encode information of node features, graph heterogeneity and graph structure in node embeddings. Extensive experiments demonstrate the superiority of HetCAN over advanced competitors and also exhibit its efficiency and robustness.

Keywords: Heterogeneous Information Network · Graph Neural Network · Graph Representation Learning

1 Introduction

Heterogeneous information networks (HINs) [17,19] typically include multiple types of nodes and edges, implying the rich semantic information. These come

A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14946, pp. 57–73, 2024.
https://doi.org/10.1007/978-3-031-70365-2_4

together with a lot of real-world data, such as social networks [7,24], citation networks [1,12] and recommendation systems [2,32]. The complicated heterogeneity and rich semantic information within HINs bring great challenges on heterogeneous graph tasks, such as node classification, link prediction and graph classification. Recently, the representation learning for heterogeneous graphs [28] receives a surge of research attention, which presents a great opportunity for analyzing HINs.

To capture both heterogeneity and structural information, heterogeneous graph neural networks (HGNNs) have been proposed and widely used to model HINs in recent years. Existing HGNNs can broadly be categorized into metapath-based models and metapath-free models. Generally, metapath-based approaches capture heterogeneity by using the predefined metapaths [5,26,29,31], while they have to redefine appropriate metapaths to adapt to various heterogeneous graphs. To get rid of the dependency on metapaths, metapath-free approaches encode graph heterogeneity by designing additional tailored modules [11,13,14]. With the capability of encoding both graph structure and heterogeneity, existing approaches give rise to the performance in a variety of downstream tasks on HINs. Their success demonstrates that leveraging the heterogeneity of HINs can significantly boost the model's performance.

Since nodes in different types may have diverse attributes, most existing metapath-free methods first align node features by projecting them into a shared low-dimensional space [13,14]. For example, as illustrated in Fig. 1, the input feature vectors of papers, authors and venues are first mapped into low-dimensional embeddings with same dimension. Although the low-dimensional node embeddings can preserve original feature information and topological information [28], node type information is not retained. This further leads to the loss of heterogeneity information in subsequent neighborhood aggregation operation. Under this condition, when aggregating information from a node's neighbors, most existing methods can only identify which neighbors are in the same type, but they fail to know the exact types of these neighbors. Based on the above analysis, the first challenge is to design an encoder that can seamlessly integrate the information of graph heterogeneity, including both node types and edge types, with node features and graph structure.

Additionally, most existing approaches only consider the interactions between nodes while neglecting the latent interactions among different node features [14,26,30]. Specifically, each convolutional layer can be considered as the one-order interaction between a node and its neighbors. By stacking multiple convolutional layers, the high-order information from multi-hop neighboring nodes is captured. However, the feature-level high-order information is also useful for label prediction. For example, in a co-authorship network, attributes of paper nodes include keywords, and our target is to predict their research topics. For a paper, suppose we only consider one keyword like *graph neural networks (GNNs)*, it is difficult to predict whether the paper's label is Information Retrieval (IR) or Artificial Intelligence (AI), as both IR and AI have sub-topics related to GNNs. If we consider three keywords *graph neural networks*, *personalized search* and

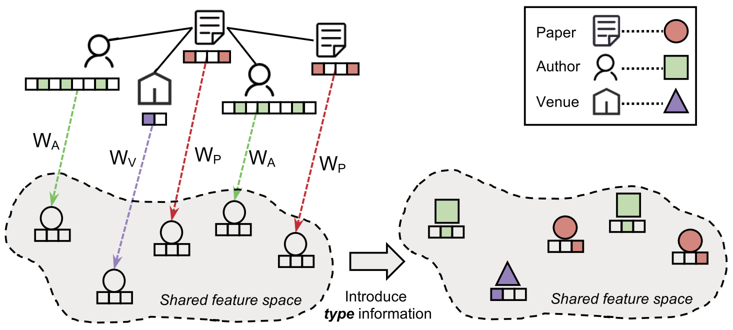


Fig. 1. An illustration of the feature processing for a toy citation network. W_P, W_A and W_V are type-specific transformation matrices w.r.t. node types.

query recommendation simultaneously, it is easier to classify the paper as IR rather than AI. This indicates that considering such feature-level interactions can boost the model's capability. Therefore, the second challenge is to design an encoder that can capture latent interactions among node features and leverage feature-level high-order information to enhance node embeddings.

To address the challenges, in this paper, we propose a novel **Het**erogeneous graph **C**ascade **A**ttention **N**etwork (HetCAN). **For the first challenge**, we put forward a type-aware encoder composed of multiple type-aware layers, in which learnable type embeddings are explicitly introduced for both nodes and edges. The key idea of introducing node type embeddings is to supplement the loss of type information for node embeddings in the low-dimensional space. To this end, we first propose to fuse node feature embeddings with node type embeddings and then obtain the fused node embeddings. After that, we use fused node embeddings and edge type embeddings to perform attention-based weighted aggregation to learn the type-aware node embeddings. Owing to type information of both nodes and edges, we can capture heterogeneity, node attributes and graph structure simultaneously at the type-aware encoder.

For the second challenge, inspired by Transformer's outstanding capability of modeling interactions among tokens in the input sequences [22,30], we propose a dimension-aware encoder to enhance hidden embeddings by learning the feature-level high-order information. To highlight the importance of node types, we also introduce node type embeddings as type encoding which is similar to the positional encoding in Transformer. This is intended to distinguish different feature interactions corresponding to node types. Regarding each dimension (or multiple dimensions) of node embedding as a token, we can construct an input sequence for each node. We then apply multi-head self-attention to these input sequences, allowing each dimension to attend to other dimensions and thereby learn their latent interactions. The outputs from the dimension-aware encoder are concatenated with those from the type-aware encoder to create the final node representations., i.e., the outputs of one-layer cascade block.

Overall, the proposed HetCAN can be regarded as a cascade model with dual-level awareness, i.e., **node-level and feature-level** awareness. On the one hand, each type-aware layer utilizes type embeddings of both nodes and edges, allowing nodes to be aware of their neighborhood's type information as well as feature information. On the other hand, at each dimension-aware layer, we employ multi-head self-attention on the sequences expanded from hidden embeddings and make each dimension to be aware of others, thereby learning the high-order information behind latent feature interactions. Finally, the main contributions of this work are summarized as follows:

– We present a type-aware encoder to make up for heterogeneous information and capture the node-level high-order information, as well as a dimension-aware encoder for learning the feature-level high-order information.
– We propose a metapath-free model HetCAN built upon the above encoders, which allows us to encode graph heterogeneity in a learnable way and obtain more expressive node representations in an end-to-end manner.
– We conduct extensive experiments to demonstrate the effectiveness and efficiency of the proposed HetCAN.

2 Related Work

Heterogeneous Network Embedding. A large number of graph embedding approaches have been proposed in recent years [6,15,21], which aim to map nodes or substructures into a low-dimensional space in which the connectivity of the graph can be preserved. Meanwhile, as most of real-world networks are usually composed of various types of nodes and relationships [28], researches on heterogeneous network embeddings (HNEs) [17,25] have also received significant attention. The approaches of HNEs can broadly be categorized into random walk methods [3,4] and first/second-order proximity methods [18,20].

Heterogeneous Graph Neural Networks. To capture the rich semantic information contained in heterogeneous graphs, a series of heterogeneous graph neural networks have been proposed in recent years [13,27]. According to the way to utilize the graph heterogeneity, HGNNs are divided into two categories, i.e., metapath-based HGNNs [5,16,26,29] and metapath-free HGNNs [11,13,14]. Typically, metapath-based approaches aggregate messages from type-specific neighboring nodes to generate semantic vectors, then fuse the semantic vectors of different metapaths to output the final node representations. Their dependencies on the predefined metapaths make them challenging to apply to complex real-world networks. Metapath-free models update node representations by directly employing message passing mechanism, with additional tailored modules to model the heterogeneity, which makes them free from the selection of metapaths. But tailored modules tend to make the encoding of heterogeneity separate from node features [14], which fails to capture the relations between them. We aim to encode both graph heterogeneity and feature information in a unified embedding for all nodes in HINs, allowing us to learn more expressive node representations and then improve the model's performance on downstream tasks.

3 Preliminaries

3.1 Heterogeneous Information Network

A heterogeneous information network (HIN) can be defined as $G = \{\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R}\}$, where $\mathcal{V}$ is the set of nodes as well as $\mathcal{E}$ is the set of edges. In HIN, each node $v \in \mathcal{V}$ has a type $\phi(v)$ and each edge $e \in \mathcal{E}$ has a type $\psi(e)$. The sets of node types and edge types are denoted by $\mathcal{A}$ and $\mathcal{R}$, where $\mathcal{A} = \{\phi(v) : \forall v \in \mathcal{V}\}$ and $\mathcal{R} = \{\psi(e) : \forall e \in \mathcal{E}\}$, respectively. And $\phi : \mathcal{V} \rightarrow \mathcal{A}$ is the node type mapping function, $\psi : \mathcal{E} \rightarrow \mathcal{R}$ is the edge type mapping function. Typically, a heterogeneous graph is with $|\mathcal{A}| + |\mathcal{R}| > 2$. Note that, when $|\mathcal{A}| = |\mathcal{R}| = 1$, the graph degenerates into a homogeneous graph.

3.2 Graph Neural Networks

GNNs [12,23] and HGNNs [13,16,26] commonly rely on the key operation of aggregating neighborhood information in a layer-wise manner, namely the node-level aggregation. In this manner, messages can be recursively passed and transformed from neighboring nodes to the target node. In the l-th layer, the representation of node v can be calculated by

$$\mathbf{h}_v^l = \textsc{Aggr}(\mathbf{h}_v^{l-1}, \{\mathbf{h}_u^{l-1} : u \in \mathcal{N}_v\}; \theta_g^l). \tag{1}$$

where $\mathcal{N}_v$ is the neighboring nodes set of node v (or type-specific neighboring nodes for HGNNs), and $\textsc{Aggr}(\cdot; \theta_g^l)$ denotes the neighborhood aggregation function parameterized by θ_g^l in the l-th layer. There are different neighborhood aggregation functions, e.g., mean-pooling aggregation in GCN [12] and attention-based aggregation in GAT [23]. Since GAT can distinguish different importance of neighboring nodes, we adopt it as the backbone of the proposed type-aware encoder, which will be discussed in next section.

3.3 Transformer-Style Architecture

In the following part, we introduce transformer encoder briefly. The transformer encoder [22] is composed of one or multiple transformer blocks, where each transformer block mainly contains a multi-head self-attention (MHSA) module and a feed-forward network (FFN). In natural language processing, the MHSA module, the critical component, aims to receive the semantic correlations among input tokens. Regarding each node feature as a token, it can also be generalized to learn the interactions among node features.

Suppose we have an input $\mathbf{H} \in \mathbb{R}^{n \times d}$, where n is the length (or number) of input tokens and d is the hidden dimension. The MHSA firstly projects $\mathbf{H}$ to $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ by three linear transformations as

$$\mathbf{Q} = \mathbf{H}\mathbf{W}_{\mathrm{q}}, \mathbf{K} = \mathbf{H}\mathbf{W}_{\mathrm{k}}, \mathbf{V} = \mathbf{H}\mathbf{W}_{\mathrm{v}}, \tag{2}$$

where $\mathbf{W}_\text{q}, \mathbf{W}_\text{k} \in \mathbb{R}^{d \times d_\text{k}}$ and $\mathbf{W}_\text{v} \in \mathbb{R}^{d \times d_\text{v}}$. Then we calculate the output of MHSA by the scaled dot-product attention mechanism as

$$\text{MHSA}(\mathbf{H}) = \textsc{Softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_\text{k}}})\mathbf{V}, \tag{3}$$

where $\sqrt{d_\text{k}}$ is the scaling factor. For simplicity, we use a single-head self-attention module for the description. Thereafter, the MHSA module is followed by the FFN module which contains two layers of Layer Normalization (LayerNorm) and the residual connection [8]. Then we can obtain the output of l-th Transformer block as

$$\begin{aligned} \mathbf{H}^l &= \textsc{LayerNorm}(\text{FFN}(\widetilde{\mathbf{H}}^l) + \widetilde{\mathbf{H}}^l) \\ \widetilde{\mathbf{H}}^l &= \textsc{LayerNorm}(\text{MHSA}(\mathbf{H}^{l-1}) + \mathbf{H}^{l-1}). \end{aligned} \tag{4}$$

By stacking L Transformer blocks, we can obtain the final output representation $\mathbf{H}^L \in \mathbb{R}^{n \times d}$, which can be used as the input of downstream tasks, such as node classification and link prediction.

4 The Proposed Model

4.1 Overall Architecture

The overall framework of HetCAN is illustrated in Fig. 2. Given a heterogeneous graph HG, we first adopt type-specific linear transformations to project nodes with different feature spaces into a shared feature space. Then, the aligned embeddings are employed as initial node feature matrix and are fed into the type-aware encoder, where each node can simultaneously perceive heterogeneity information, feature information and structural information within its neighborhood. After multiple type-aware layers, hidden node embeddings are passed to the dimension-aware encoder, where latent feature interactions will be modeled through multi-head self-attention mechanism. Afterward, we concatenate the outputs from the type-aware encoder and the dimension-aware encoder to construct the updated node embeddings, which are also referred to the outputs of each cascade block. HetCAN typically includes N cascade blocks. Finally, we perform downstream tasks based on the normalized final node representations. In the following parts, we will illustrate the details of the type-aware encoder and the dimension-aware encoder, respectively.

4.2 Type-Aware Encoder

In the type-aware encoder, we first introduce learnable type embeddings for both nodes and edges, and integrate feature embeddings and type embeddings as a whole. Formally, we first initialize a node type matrix denoted by $\mathbf{M} \in \mathbb{R}^{|\mathcal{A}| \times d_t}$, where $|\mathcal{A}|$ is the number of node types. For each node v_i, its node type embedding $\mathbf{t}_i \in \mathbb{R}^{d_t}$ is derived by $\mathbf{t}_i = \mathbf{M}[\phi(v_i), :]$. Then we can obtain type

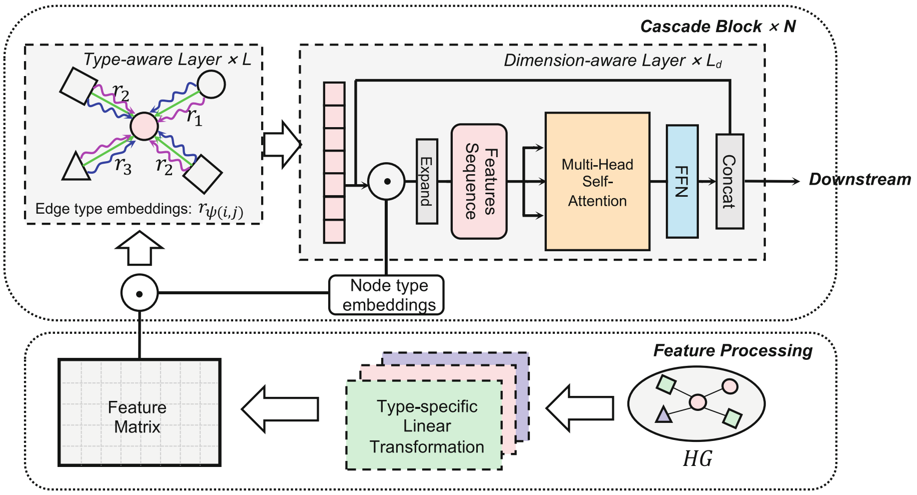


Fig. 2. The overall framework of HetCAN. Each cascade block consists of L type-aware layers and L_d dimension-aware layers.

embeddings of all nodes represented as $\mathbf{T} \in \mathbb{R}^{n \times d_t}$. As node features of different types on HINs usually exist in different feature spaces, we project them into a shared feature space before passing them to the type-aware encoder. Formally, the feature processing is denoted as

$$\mathbf{h}_i = \mathbf{W}_{\phi(v_i)}\mathbf{x}_i + \mathbf{b}_{\phi(v_i)}, \tag{5}$$

where $\mathbf{W}_{\phi(v_i)} \in \mathbb{R}^{d \times d_x}$ is the learnable parameter matrix corresponding to node type $\phi(v_i)$ and $\mathbf{b}_{\phi(v_i)} \in \mathbb{R}^d$ is an optional bias term. Then we can obtain node feature embeddings denoted by $\mathbf{H} \in \mathbb{R}^{n \times d}$. After that, to comprehensively supplement node type information, we present a combination function to integrate node feature embeddings with node type embeddings as

$$\widetilde{\mathbf{H}} = \textsc{Combine}(\mathbf{H}, \mathbf{T}), \tag{6}$$

where $\textsc{Combine}$ $(\cdot)$ can be any operator function, such as learnable functions or non-parametric functions. In practice, we simply implement it with Hadamard product which is an element-wise operation. Based on [13], we then extend attention mechanism with integrated node embeddings that contain the node type information. In this way, each type-aware layer calculates the attention coefficient between node v_i and node v_j as follows (layer marker (l) is omitted for simplicity)

$$\alpha_{ij} = \frac{\exp\left(\sigma\left(\mathbf{a}^T[\mathbf{W}\tilde{\mathbf{h}}_i||\mathbf{W}\tilde{\mathbf{h}}_j||\mathbf{W}_\mathbf{r}\mathbf{r}_{\psi(v_i,v_j)}]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\sigma\left(\mathbf{a}^T[\mathbf{W}\tilde{\mathbf{h}}_i||\mathbf{W}\tilde{\mathbf{h}}_k||\mathbf{W}_\mathbf{r}\mathbf{r}_{\psi(i,k)}]\right)\right)}, \tag{7}$$

where $\mathbf{r}_{\psi(v_i,v_j)} \in \mathbb{R}^{d_r}$ is learnable edge type embedding w.r.t. the type of edge between node v_i and node v_j, $\mathbf{W}$ and $\mathbf{W}_r$ are learnable matrices, and σ is the LeakyReLU activation function. To stabilize training process and improve the performance, inspired by [8,9], we employ residual connection on attention coefficients as

$$\hat{\alpha}_{ij}^{(l)} = (1-\beta)\alpha_{ij}^{(l)} + \beta\hat{\alpha}_{ij}^{(l-1)}, \tag{8}$$

where $\beta \in [0,1]$ denotes attention residual weight. Once obtained, the normalized attention coefficients are used to update the hidden node embedding $\mathbf{h}_i'$ for each node $v_i \in \mathcal{V}$ as

$$\mathbf{h}_i' = \sigma\left(\sum_{j\in\mathcal{N}_i} \hat{\alpha}_{ij}\mathbf{W}\tilde{\mathbf{h}}_j\right). \tag{9}$$

To enhance model's capacity and stabilize the learning process, we implement a multi-head attention mechanism by averaging

$$\mathbf{h}_i' = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i} \hat{\alpha}_{ij}^{k}\mathbf{W}^{k}\tilde{\mathbf{h}}_j\right), \tag{10}$$

where K is the number of heads. Overall, with the type-aware encoder, hidden node embeddings can seamlessly fuse the information of graph heterogeneity, node feature and graph structure and have more powerful expressive capabilities.

4.3 Dimension-Aware Encoder

The success of Transformer has demonstrated its outstanding capability of learning interactions among the tokens in a sequence. Motivated by this, we propose a dimension-aware encoder with transformer architecture for capturing the feature-level high-order information, which can further enhance the expressive capability of node embeddings.

After acquiring hidden embeddings $\mathbf{H}' \in \mathbb{R}^{n\times d}$ from L type-aware layers, the dimension-aware encoder constructs the input sequence for each node to adapt to transformer architecture. Specifically, for each node $v \in \mathcal{V}$, we expand its hidden embedding $\mathbf{h}_v' \in \mathbb{R}^{d}$ to a sequence $\hat{\mathbf{h}}_v' \in \mathbb{R}^{d\times 1}$, treating each dimension (or multiple dimensions) as a token represented by a one-dimensional (or multiple dimensional) vector. Then we perform multi-head self-attention on each input sequence to learn interactions among the tokens within it.

Besides, to learn distinct feature interaction patterns for different node types, inspired by the positional encoding in Transformer, we introduce node type embeddings $\mathbf{T} \in \mathbb{R}^{n\times d_t}$ as type encoding and combine them with the hidden node embeddings $\mathbf{H}'$ before performing attention mechanism. We denote this step as

$$\begin{aligned} \hat{\mathbf{H}} &= \textsc{Combine}(\mathbf{H}', \mathbf{T}) \\ \hat{\mathbf{H}}' &= \textsc{Expand}(\hat{\mathbf{H}}) \end{aligned} \tag{11}$$

where $\hat{\mathbf{H}}' \in \mathbb{R}^{n \times d \times 1}$ denotes the constructed sequences of all nodes, illustrated in upper right of Fig. 2. Similar to the type-aware encoder, the COMBINE $(\cdot)$ is also implemented with Hadamard product by simply setting $d_t = d$, so that the shape of $\hat{\mathbf{H}}$ remains consistent with $\mathbf{H}'$. With type encoding, the dimension-aware encoder can distinguish various node types and learn unique interaction patterns for them. Then, we perform multi-head self-attention on the input sequences $\hat{\mathbf{H}}'$ and obtain the outputs of each dimension-aware layer as follows

$$\overline{\mathbf{H}} = \textsc{Mhsa}(\hat{\mathbf{H}}'), \tag{12}$$

where $\textsc{Mhsa}(\cdot)$ denotes multi-head self-attention (refer to Sect. 3.3) and $\overline{\mathbf{H}} \in \mathbb{R}^{n \times d}$ is the node representations containing rich feature-level information. Finally, we concatenate the outputs of dimension-aware encoder $\overline{\mathbf{H}}$ and the type-aware encoder $\mathbf{H}'$ to construct the final node representations as

$$\mathbf{H}_{\mathrm{f}} = \mathbf{H}' \parallel \overline{\mathbf{H}}, \tag{13}$$

where $\mathbf{H}_{\mathrm{f}} \in \mathbb{R}^{n \times 2d}$ is the output of one-layer cascade block. For simplicity, we only illustrate one layer of cascade block. After one or multiple cascade blocks, we can obtain the enhanced node representations with more expressive capabilities and use them for various downstream tasks.

4.4 Time Complexity Analysis

In this subsection, we give the time complexity analysis of the proposed components in HetCAN. Let $|\mathcal{V}|$ and $|\mathcal{E}|$ are the number of nodes and edges. d is the dimension of both node feature embeddings and node type embeddings, and d_r is the dimension of edge embeddings. For each type-aware layer, the time complexity of a single attention head can be expressed as $O(|\mathcal{V}| \times d^2 + |\mathcal{E}| \times {d_r}^2 + |\mathcal{E}| \times (2d + d_r))$. For each dimension-aware layer, the time complexity of a single attention head can be expressed as $O(|\mathcal{V}| \times d^2)$. Thus, overall time complexity of HetCAN is linear to both the number of nodes $|\mathcal{V}|$ and the number of edges $|\mathcal{E}|$. The efficiency studies of our model are shown in Fig. 4.

5 Experiments

We evaluate HetCAN by conducting extensive experiments on node classification and link prediction, and compare various competitive approaches, including plain homogeneous GNNs, metapath-based HGNNs and metapath-free HGNNs. In addition, to further investigate the superiority of our model, we comprehensively conduct three studies including an ablation study, an efficiency study and a parameter study. The source code and datasets are available at https://github.com/zzyzeyuan/HetCAN.

5.1 Experimental Setups

Datasets. For node classification, we test our model on five public datasets. Specifically, DBLP, IMDB, ACM and Freebase are from [13], and OGB-MAG is from [10]. For link prediction, we test our model on three public datasets from [13]. Heterogeneous Graph Benchmark standardizes the process pipeline for fair comparison, so we follow their pipelines to conduct experiments. For datasets without node features, we assign them one-hot or all-one vector features to denote their existence. The statistics of all datasets are summarized in Table 1.

Table 1. Statistics of all datasets.

Datasets	#Nodes	#Node Types	#Edges	#Edge Types	Target	#Classes
DBLP	26,128	4	239,566	6	author	4
IMDB	21,420	4	86,642	6	movie	5
ACM	10,942	4	547,872	8	paper	3
Freebase	180,098	8	1,057,688	36	book	7
OGB-MAG	1,939,743	4	21,111,007	4	paper	349
Amazon	10,099	1	148,659	2	product-product	–
LastFM	20,612	3	141,521	3	user-artist	–
PubMed	63,109	4	244,986	10	disease-disease	–

Baselines. To comprehensively evaluate our proposed model against the state-of-the-art methods, we select a collection of baselines, including *basic models* (GCN [12], GAT [23], Transformer [22]), *metapath-based models* (RGCN [16], HAN [26], HetGNN [31], MAGNN [5], SeHGNN [29]) and *metapath-free models* HGT [11], Simple-HGN [13], HINormer [14]). Specifically, as all baselines do not utilize extra label embeddings, we report the results of SeHGNN without the utilization of extra label embeddings.

Settings. Regarding datasets from HGB, we follow the split proportion of 24:6:70 for the training, validation and test sets, respectively. Regarding OGB-MAG dataset, we propose to train on papers published until 2017, validate on those published in 2018, and test on those published since 2019. We evaluate classification performance of all baselines with Micro-F1 and Macro-F1 for HGB datasets and accuracy for OGB-MAG dataset. Following HGB, we use ROC-AUC (area under the ROC curve) and MRR (mean reciprocal rank) metrics to evaluate link prediction performance. Since our experimental setup is consistent with HGB and OGB, we directly borrow the results reported in HGB and OGB leaderboard for comparison. For those results that are not available in HGB or OGB, we conduct experiments based on original experimental setups.

Table 2. Experiment results on four HGB datasets. The best result is **bolded** and the runner-up is underlined. The error bar ($\pm$) denotes the standard deviation of the results over five runs. "–" denotes the models run out of memory.

Methods	DBLP		IMDB		ACM		Freebase	
	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
GCN	91.47 ± 0.34	90.84 ± 0.32	64.82 ± 0.64	57.88 ± 1.18	92.12 ± 0.23	92.17 ± 0.24	60.23 ± 0.92	27.84 ± 3.13
GAT	93.39 ± 0.30	93.83 ± 0.27	64.86 ± 0.43	58.94 ± 1.35	92.19 ± 0.93	92.26 ± 0.94	65.26 ± 0.80	40.74 ± 2.58
Transformer	93.99 ± 0.11	93.48 ± 0.12	66.29 ± 0.69	62.79 ± 0.65	92.32 ± 0.36	92.55 ± 0.34	65.74 ± 0.78	47.06 ± 2.42
HGT	93.49 ± 0.25	93.01 ± 0.23	67.20 ± 0.57	63.00 ± 1.19	91.00 ± 0.76	91.12 ± 0.76	60.51 ± 1.16	29.28 ± 2.52
Simple-HGN	94.46 ± 0.22	94.01 ± 0.24	67.36 ± 0.57	63.53 ± 1.36	93.35 ± 0.45	93.42 ± 0.44	66.29 ± 0.45	47.72 ± 1.48
HINormer	94.94 ± 0.21	94.57 ± 0.23	67.83 ± 0.34	64.65 ± 0.53	92.13 ± 0.32	92.20 ± 0.34	66.08 ± 0.74	49.37 ± 2.08
RGCN	92.07 ± 0.50	91.52 ± 0.50	62.05 ± 0.15	58.85 ± 0.26	91.41 ± 0.75	91.55 ± 0.74	58.33 ± 1.57	46.78 ± 0.77
HAN	92.05 ± 0.62	91.67 ± 0.49	64.63 ± 0.58	57.74 ± 0.96	90.79 ± 0.43	90.89 ± 0.43	54.77 ± 1.40	21.31 ± 1.68
MAGNN	93.76 ± 0.45	93.28 ± 0.51	64.67 ± 1.67	56.49 ± 3.20	90.88 ± 0.64	90.77 ± 0.65	–	–
SeHGNN	95.24 ± 0.13	94.86 ± 0.14	68.21 ± 0.32	66.63 ± 0.34	93.87 ± 0.50	93.95 ± 0.48	63.41 ± 0.47	50.71 ± 0.44
HetCAN	$\mathbf{95.78 \pm 0.28}$	$\mathbf{95.45 \pm 0.23}$	$\mathbf{69.50 \pm 0.34}$	$\mathbf{67.23 \pm 0.28}$	$\mathbf{94.35 \pm 0.35}$	$\mathbf{94.47 \pm 0.36}$	$\mathbf{66.79 \pm 0.52}$	$\mathbf{51.48 \pm 0.63}$

5.2 Node Classification

Tables 2 and 3 summarize experimental results on node classification over five runs. From the tables, we observe that:

(1) The plain models, i.e., GCN, GAT and Transformer, perform well on all datasets when using proper inputs from HGB, indicating that preprocessing for input node features has great impact on model performance.

(2) Compared to the vanilla models mentioned earlier, SeHGNN and HINormer demonstrate superior performance, with SeHGNN being the best among metapath-based models and HINormer excelling among metapath-free models. By using the predefined metapaths, SeHGNN exploits semantic information to boost model performance. HINormer samples a fixed-length sequence for each node and designs an additional heterogeneous relation encoder, which enlarges the receptive field for each node and also models the heterogeneity.

(3) HetCAN achieves superior results across all HGB datasets, demonstrating its ability to generalize to datasets with varying degrees of heterogeneity. We attribute the generalization ability of our model to the Cascade Block, which allows us to simultaneously learn node-level and feature-level information. This enables the node representations to have more powerful expressive capabilities, thereby boosting both node classification and link prediction tasks.

(4) Based on the results from the large-scale dataset OGB-MAG (see Table 3), we can observe that HetCAN outperforms all metapath-free competitors. This indicate that our method has further narrowed the gap between metapath-free models and metapath-based models on large-scale dataset. In addition, we further conduct efficiency studies on three datasets shown in Fig. 4, which demonstrates that our method is faster than SeHGNN, the winner of OGB-MAG dataset. Particularly, shown in Fig. 4(c), the convergence speed of our model is much faster than SeHGNN in the scenario with a large number of edge types (39 edge types in Freebase). This is because metapath-based methods require aggregating information through metapaths, and this inherent property results in slower training and convergence speeds.

Table 3. Experiment results on the large-scale dataset OGB-MAG. $*$ denotes metapath-free models. The best result is **bolded** and the runner-up are underlined.

Methods	Validation accuracy	Test accuracy
RGCN	48.35 ± 0.36	47.37 ± 0.48
HGT*	49.89 ± 0.47	49.27 ± 0.61
NARS	51.85 ± 0.08	50.88 ± 0.12
SAGN*	52.25 ± 0.30	51.17 ± 0.32
GAMLP*	53.23 ± 0.23	51.63 ± 0.22
LEGNN*	54.43 ± 0.09	52.76 ± 0.14
SeHGNN	55.95 ± 0.11	$\mathbf{53.99 \pm 0.18}$
HetCAN*	54.76 ± 0.18	$\underline{53.79 \pm 0.17}$

5.3 Link Prediction

Table 4 summarizes the results on the downstream link prediction task over five runs. Based on this table, we observe that:
(1) Our method HetCAN consistently outperforms all advanced methods over both ROC-AUC and MRR metrics. Particularly, we achieve significant improvements on the Amazon and LastFM datasets. This indicates that our method can learn more expressive node representations with such a cascade structure, while also giving rise to the model's performance on the link prediction task.
(2) Compared to Simple-HGN, the runner-up on link prediction, our method achieves better performance. Our method introduces both the node-level and the feature-level high-order information through the cascade block, while Simple-HGN only uses learnable type embeddings to compensate for graph heterogeneity and ignores the feature high-order interactions.

5.4 Model Analysis

Ablation Studies. To validate effectiveness of the proposed components, we conduct ablation studies on four datasets by comparing with two variants of HetCAN: (1) we remove node type embeddings and replace it with all-one vectors, which is denoted by *w/o Type-encoder*; (2) we remove the dimension-aware encoder, which is denoted by *w/o Dim-encoder*. We report the results of ablation studies in Fig. 3 and give the following observations.

Table 4. Experiment results on link prediction. The best result is **bolded** and the runner-up is underlined. The error bar (±) denotes the standard deviation of the results over five runs. "–" denotes the results are not available due to lack of metapaths on those datasets.

Methods	Amazon		LastFM		PubMed	
	ROC-AUC	MRR	ROC-AUC	MRR	ROC-AUC	MRR
GCN	92.84 ± 0.34	97.05 ± 0.12	59.17 ± 0.31	79.38 ± 0.65	80.48 ± 0.81	90.99 ± 0.56
GAT	91.65 ± 0.80	96.58 ± 0.26	58.56 ± 0.66	77.04 ± 2.11	78.05 ± 1.77	90.02 ± 0.52
RGCN	86.34 ± 0.28	93.92 ± 0.16	57.21 ± 0.09	77.68 ± 0.17	78.29 ± 0.18	90.26 ± 0.24
GATNE	77.39 ± 0.50	92.04 ± 0.36	66.87 ± 0.16	85.93 ± 0.63	63.39 ± 0.65	80.05 ± 0.22
HetGNN	77.74 ± 0.24	91.79 ± 0.03	62.09 ± 0.01	83.56 ± 0.14	73.63 ± 0.01	84.00 ± 0.04
HGT	88.26 ± 2.06	93.87 ± 0.65	54.99 ± 0.28	74.96 ± 1.46	80.12 ± 0.93	90.85 ± 0.33
Simple-HGN	93.40 ± 0.62	96.94 ± 0.29	67.59 ± 0.23	90.81 ± 0.32	83.39 ± 0.39	92.07 ± 0.26
HetCAN	**95.60±0.32**	**98.14±0.28**	**67.92±0.17**	**91.78±0.40**	**83.94±0.30**	**92.48±0.24**

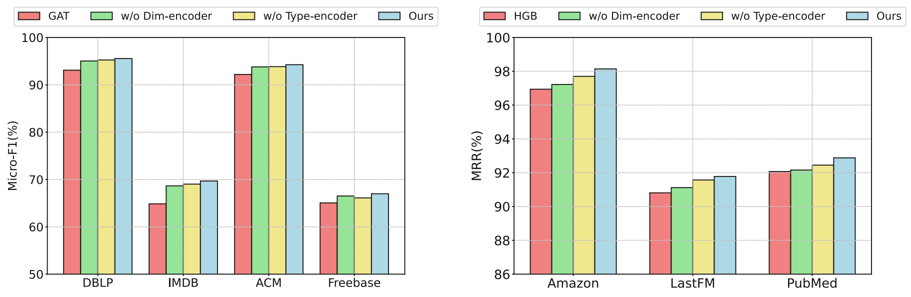


Fig. 3. Ablation studies.

Firstly, without the type-aware encoder, HetCAN fails to consider node type when performing attention mechanism, resulting in degradation of classification performance. Compared to other datasets, the absence of node type embeddings has more prominent impacts on Freebase that has more node types, indicating that introducing learnable node type embeddings explicitly can make up for type information and benefit the model's performance. From another perspective, improvements on some datasets are not as significant as that on Freebase, thus how to further exploit the underlying semantic relations between nodes is still a promising direction and we will investigate it in future work.

Secondly, without dimension-aware encoder, HetCAN fails to capture latent feature interactions, resulting in a significant reduction of performance. We also notice that the absence of dimension-aware encoder has a more significant impact on Macro-F1 scores than Micro-F1. Especially on Freebase, the Macro-F1 is reduced by 2.88% and has a larger standard deviation, which indicates that dimension-aware encoder can benefit the robustness of our model.

Efficiency Studies. To assess the efficiency of HetCAN, we compared the training times of several advanced methods in the same experimental environment, using the hyper-parameters corresponding to optimal performance for each

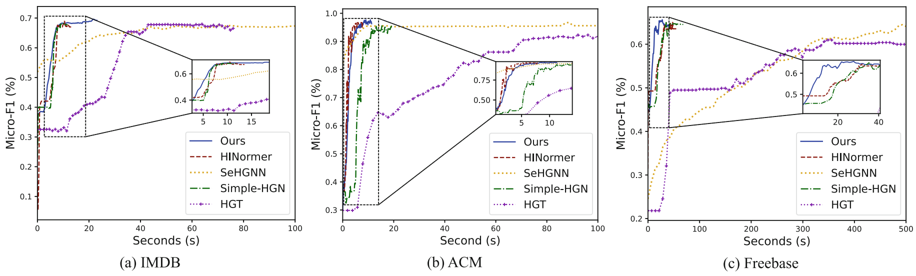


Fig. 4. Efficiency study: x-axis shows the training time and y-axis is the Micro-F1 score on the validation set.

method. The results are illustrated in Fig. 4. Specifically, on IMDB, HetCAN converges in around 10 s, while SeHGNN and HGT take more than 30 s. This indicates that our model is as efficient as other metapath-free methods and significantly faster than SeHGNN and HGT. On Freebase, HetCAN achieves the optimal performance around 20 s, while HINormer and Simple-HGN approach their optimal state around 40 s. This also demonstrates the efficiency and robustness of HetCAN on information networks with a greater variety of node types and edges. Surprisingly, we found that SeHGNN takes approximately 500 s, 20 times of our model, to converge to the optimal state. This demonstrates the superiority and flexibility of being free from the predefined metapaths.

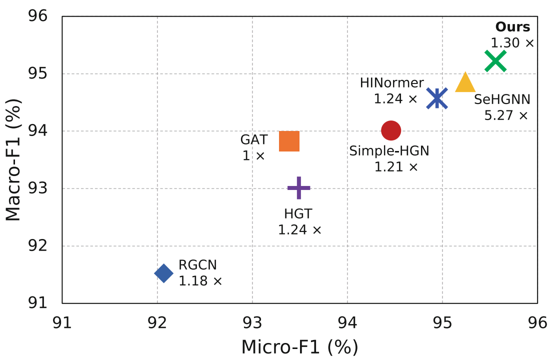


Fig. 5. Parameters comparison. The numbers below the model names represent the ratio of the total number of parameters relative to GAT. For example, "1.24" below HGT means its total parameters are 1.24 times that of GAT.

Parameter Studies. We experiment on DBLP (Fig. 5) to statistically compare HetCAN's total parameter count with that of other competitors. We use the hyper-parameters corresponding to the optimal performance of these models. We observe that SeHGNN achieves its peak performance with a large hidden size (512), which leads to a slower convergence speed. In contrast, HetCAN achieves state-of-the-art performance by introducing an affordable number of parameters, ensuring both efficiency and effectiveness.

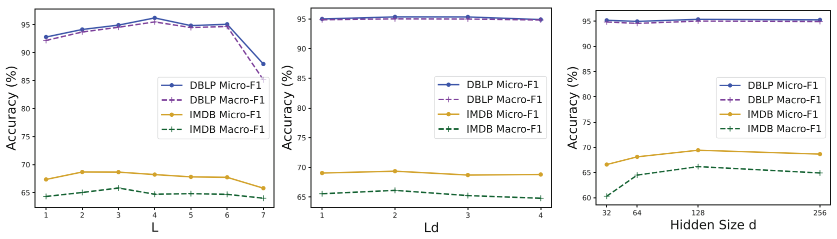


Fig. 6. Hyper-parameters sensitivity studies.

We also examine the sensitivity of hyperparameters, including the number of type-aware layers (L), dimension-aware layers (L_d), and hidden size. The results are depicted in Fig. 6. We consistently observe strong performance across a wide range of L_d values on both DBLP and IMDB datasets. The impact of hidden size (d) is more significant on IMDB than on DBLP. Increasing L initially improves performance gradually, but further increments eventually lead to a decline, indicating potential harm from overly deep layers.

6 Conclusion

In this paper, we investigate the problem of exploiting graph heterogeneity and the high-order feature information. To achieve our goals, we propose HetCAN composed of multiple cascade blocks, where each block comprises multiple type-aware layers and dimension-aware layers. The type-aware encoder seamlessly integrates node types with node features to comprehensively leverage graph heterogeneity. The dimension-aware encoder pays attention to latent interactions among node features, utilizing the high-order information inherent in such interactions through a transformer architecture. Extensive experiments and studies demonstrate the superiority, efficiency and robustness of the proposed HetCAN.

Acknowledgments. This work is supported by National Natural Science Foundation of China No. 62202172 and Shanghai Science and Technology Committee General Program No. 22ZR1419900.

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

References

1. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. In: NeurIPS (2016)
2. Berg, R.V.D., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263 (2017)
3. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: scalable representation learning for heterogeneous networks. In: SIGKDD, pp. 135–144 (2017)

4. Fu, T.Y., Lee, W.C., Lei, Z.: HIN2Vec: explore meta-paths in heterogeneous information networks for representation learning. In: CIKM, pp. 1797–1806 (2017)
5. Fu, X., Zhang, J., Meng, Z., King, I.: MAGNN: metapath aggregated graph neural network for heterogeneous graph embedding. In: The Web Conference (2020)
6. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: SIGKDD, pp. 855–864 (2016)
7. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS, vol. 30 (2017)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
9. He, R., Ravula, A., Kanagal, B., Ainslie, J.: RealFormer: transformer likes residual attention. arXiv preprint arXiv:2012.11747 (2020)
10. Hu, W., et al.: Open graph benchmark: datasets for machine learning on graphs. In: NeurIPS (2020)
11. Hu, Z., Dong, Y., Wang, K., Sun, Y.: Heterogeneous graph transformer. In: The Web Conference, pp. 2704–2710 (2020)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)
13. Lv, Q., et al.: Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks. In: SIGKDD, pp. 1150–1160 (2021)
14. Mao, Q., Liu, Z., Liu, C., Sun, J.: HINormer: representation learning on heterogeneous information networks with graph transformer. In: TheWebConf (2023)
15. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: SIGKDD, pp. 701–710 (2014)
16. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, 3–7 June 2018, Proceedings 15, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
17. Shi, C., Li, Y., Zhang, J., Sun, Y., Philip, S.Y.: A survey of heterogeneous information network analysis. IEEE TKDE **29**(1), 17–37 (2016)
18. Shi, Y., Zhu, Q., Guo, F., Zhang, C., Han, J.: Easing embedding learning by comprehensive transcription of heterogeneous information networks. In: KDD (2018)
19. Sun, Y., Han, J.: Mining heterogeneous information networks: a structural analysis approach. ACM SIGKDD Explor. Newsl. **14**(2), 20–28 (2013)
20. Tang, J., Qu, M., Mei, Q.: PTE: predictive text embedding through large-scale heterogeneous text networks. In: SIGKDD, pp. 1165–1174 (2015)
21. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: LINE: large-scale information network embedding. In: The Web Conference, pp. 1067–1077 (2015)
22. Vaswani, A., et al.: Attention is all you need. In: NeurIPS (2017)
23. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
24. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: KDD (2016)
25. Wang, X., Bo, D., Shi, C., Fan, S., Ye, Y., Philip, S.Y.: A survey on heterogeneous graph embedding: methods, techniques, applications and sources. IEEE Trans. Big Data **9**(2), 415–436 (2022)
26. Wang, X., et al.: Heterogeneous graph attention network. In: The Web Conference, pp. 2022–2032 (2019)
27. Wang, X., Liu, N., Han, H., Shi, C.: Self-supervised heterogeneous graph neural network with co-contrastive learning. In: SIGKDD, pp. 1726–1736 (2021)

28. Yang, C., Xiao, Y., Zhang, Y., Sun, Y., Han, J.: Heterogeneous network representation learning: a unified framework with survey and benchmark. In: TKDE (2020)
29. Yang, X., Yan, M., Pan, S., Ye, X., Fan, D.: Simple and efficient heterogeneous graph neural network. In: AAAI, vol. 37, pp. 10816–10824 (2023)
30. Ying, C., et al.: Do transformers really perform badly for graph representation? In: NeurIPS (2021)
31. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. In: SIGKDD, pp. 793–803 (2019)
32. Zhang, J., Shi, X., Zhao, S., King, I.: STAR-GCN: stacked and reconstructed graph convolutional networks for recommender systems. In: IJCAI, pp. 4264–4270 (2019)

Interpetable Target-Feature Aggregation for Multi-task Learning Based on Bias-Variance Analysis

Paolo Bonetti(✉), Alberto Maria Metelli, and Marcello Restelli

Diparimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy
{paolo.bonetti,albertomaria.metelli,marcello.restelli}@polimi.it

Abstract. Multi-task learning (MTL) is a powerful machine learning paradigm designed to leverage shared knowledge across tasks to improve generalization and performance. Previous works have proposed approaches to MTL that can be divided into feature learning, focused on the identification of a common feature representation, and task clustering, where similar tasks are grouped together. In this paper, we propose an MTL approach at the intersection between task clustering and feature transformation based on a two-phase iterative aggregation of targets and features. First, we propose a bias-variance analysis for regression models with additive Gaussian noise, where we provide a general expression of the asymptotic bias and variance of a task, considering a linear regression trained on aggregated input features and an aggregated target. Then, we exploit this analysis to provide a two-phase MTL algorithm (NonLinCTFA). Firstly, this method partitions the tasks into clusters and aggregates each obtained group of targets with their mean. Then, for each aggregated task, it aggregates subsets of features with their mean in a dimensionality reduction fashion. In both phases, a key aspect is to preserve the interpretability of the reduced targets and features through the aggregation with the mean, which is further motivated by applications to Earth science. Finally, we validate the algorithms on synthetic data, showing the effect of different parameters and real-world datasets, exploring the validity of the proposed methodology on classical datasets, recent baselines, and Earth science applications.

Keywords: Multi-Task Learning · Variable Aggregation · Bias-Variance Analysis

1 Introduction

Machine Learning [5, ML] approaches usually consider an individual learning problem, eventually decomposing complex problems into independent tasks.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_5.

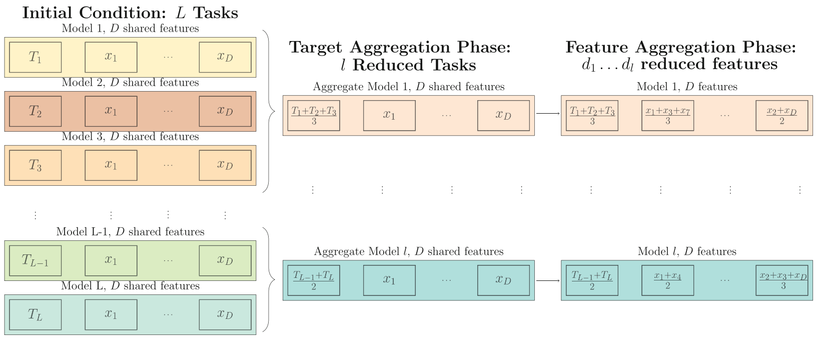
Inspired by the possibility to exploit their interconnections, *Multi-task learning* [10, MTL] methods are designed to simultaneously learn multiple related tasks, leveraging shared knowledge to improve performance and generalization. In recent years, MTL has gained significant attention across various domains, including natural language processing [12], computer vision [3], and healthcare [37]. Additionally, MTL proved to be efficient in various real-world applications such as gene expression [26], cardiac ECG analysis [9], and quantitative stock portfolio optimization [25].

Focusing on supervised MTL, *task-clustering* methods are designed to aggregate different tasks into clusters, exploiting their relationships to learn groups of tasks with the same model. On the other hand, *feature-based* MTL approaches are focused on the identification of a subset of relevant common input features (feature selection), or the extraction of a combination of relevant original inputs (feature transformation). In the feature transformation context, a *dimensionality reduction* approach may be considered to extract a set of relevant features common to all the tasks, reducing the dimension of the feature space.

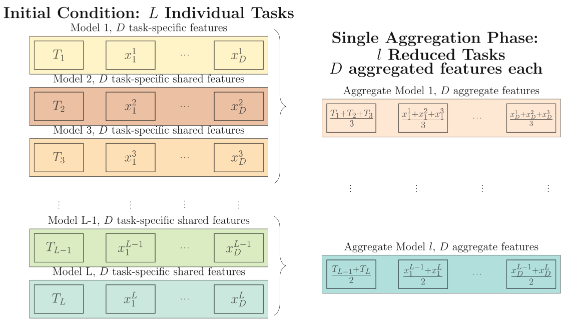
In this paper, we propose an MTL approach (NonLinCTFA) at the intersection between task clustering and feature transformation with dimensionality reduction. Firstly, we introduce a specific task clustering, based on partitioning the targets, aggregating each obtained group of targets with their mean. Then, for each aggregated task (the mean of the targets of a cluster), we aggregate subsets of features with their mean. This way, we first learn a single model for each group of tasks. Then, we aggregate subsets of features with their mean in each aggregated task, producing a set of reduced features in a dimensionality reduction fashion. In both cases, we provide theoretical guarantees on the improvement (or not worsening) of the mean squared error on each of the original tasks. A schematic view of this methodology can be found in Fig. 1a. Additionally, Fig. 1b shows a variant that will be discussed in the next sections.

The choice to consider the mean as an aggregation function is to preserve interpretability, following the definition of interpretability as the property of an ML pipeline to be understood by domain experts, without explanations by ML experts (see [20] for a broader discussion). Indeed, we aggregate targets and features, reducing the dimension and the variance of the models and preserving the meaning of each aggregation, which is a mean of variables. On the other hand, the aggregations induce an increase in bias that will be controlled in the analysis.

A motivational example is related to Earth science, where we may be interested in the prediction of a target variable at different (correlated) locations, given a set of meteorological features measured in each of them (see Fig. 2a). In this context, the proposed algorithm aggregates highly correlated targets and learns an individual model for their mean, resulting in a reduced number of models, simplifying data representation (through further aggregating highly correlated features) and enhancing the performances without loss of interpretability, mitigating overfitting, and limiting the computational complexity (see Fig. 2b, further discussed in the experimental section). In this example, preserving the

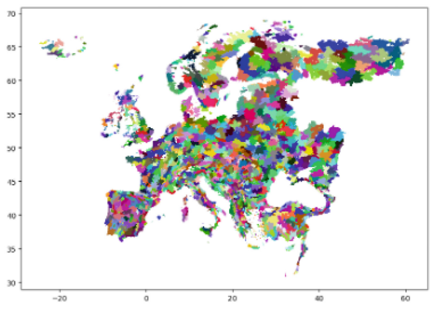

Initial Condition: L Tasks
Model 1, D shared features
Model 2, D shared features
Model 3, D shared features
Model L-1, D shared features
Model L, D shared features
Target Aggregation Phase: l Reduced Tasks
Aggregate Model 1, D shared features
Aggregate Model l, D shared features
Feature Aggregation Phase: $d_1 \dots d_l$ reduced features
Model 1, D features
Model l, D features

(a) Block scheme of the algorithm: partition of tasks, aggregation of each set of targets with the mean, replication of the procedure to the D shared features, for each reduced task.

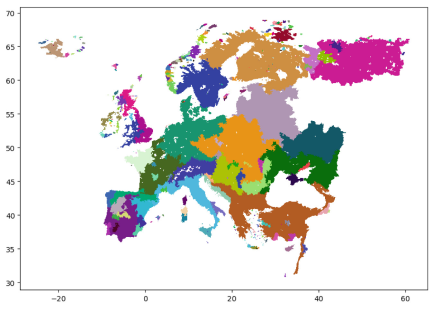

Initial Condition: L Individual Tasks
Model 1, D task-specific features
Model 2, D task-specific shared features
Model 3, D task-specific shared features
Model L-1, D task-specific shared features
Model L, D task-specific shared features
Single Aggregation Phase: l Reduced Tasks D aggregated features each
Aggregate Model 1, D aggregate features
Aggregate Model l, D aggregate features

(b) Block scheme of a variant of the algorithm with D features individually measured for each task. In the aggregation phase, both targets and features are averaged.

Fig. 1. Block scheme of main algorithm and a homogeneous case variant.



(a) Motivational example.



(b) Basins reduced by NonLinCTFA.

Fig. 2. Figure 2a represents some European hydrologic sub-basins, identified with different colors, where a target variable and meteorological features are available. Figure 2b shows the aggregated regions after the application of NonLinCTFA.

interpretability is essential for climatologists, filling the gap between advanced ML methods (considered as black-box algorithms) and their physical meaning.

Contributions. After introducing notation, problem formulation, and related works (Sect. 2), the contributions of this paper can be summarized as:

1) General theoretical analysis of the asymptotic bias and variance, estimating a target variable with a linear model, trained considering as target an average of targets and as features a reduced set of basis functions (Sect. 3).
2) Introduction of a novel MTL approach, with theoretical guarantees, aggregating targets and features with the mean for interpretability (Sect. 4).
3) Extension of recent dimensionality reduction methods [7,8], both to multiple targets and generalizing the theoretical results in the single-task case.
4) Validation of the proposed algorithm on synthetic data, benchmark MTL datasets and methods, and a multi-task dataset from Earth science (Sect. 5).

Technical proofs and additional results can be found in the Appendix, available in the supplementary material.

2 Preliminaries

Notation. Given L learning tasks $\{\mathcal{T}_i\}_{i=1}^{L}$, each task $\mathcal{T}_i$ is a supervised learning problem with training set $S_i = \{(\mathbf{x}^i, y_i)_j\}_{j=1}^{n_i}$, where n_i samples for the i-th task are available, $(\mathbf{x}^i)_j = (x_1^i, \dots, x_{D_i}^i)_j \in \mathbb{R}^{D_i}$ is the j-th training sample associated to task i composed of D_i features, and $(y_i)_j$ is the corresponding target. The random variable y_i represents the i-th target, x_k^i denotes the k-th feature of task $\mathcal{T}_i$, and the random vector $\mathbf{x^i}$ is the full set of features of the i-th task. The symbols σ_a^2, $cov(a,b)$, ρ_{ab} and $\hat{\sigma}_a^2$, $\hat{cov}(a,b)$, $\hat{\rho}_{ab}$ denote the variance of a random variable a, its covariance, and correlation with the random variable b and their estimators, respectively. Finally, $\mathbb{E}_a[f(a)]$ is the expected value of a function $f(\cdot)$ of the random variable a w.r.t. its distribution.

Remark 1. To simplify the notation, in this paper, we mainly consider a shared set of D features $\boldsymbol{x} := \{x_1, \dots, x_D\} := \{x_1^1, \dots, x_{D_1}^1, \dots, x_1^L, \dots, x_{D_L}^L\}$ and the same number of training samples n for each task. This simplification motivates the workflow of Fig. 1a. We will also discuss a variant with D task-specific features (workflow in Fig. 1b). Recalling the motivational example, this means that we will consider the full set of measurements of meteorological features as a shared set of D features. Then, we will discuss a variant with a set of meteorological features measured in each individual sub-region associated with its target. Additionally, this homogeneous-feature variant could also be applied in heterogeneous settings, aggregating corresponding features (e.g., temperature), without changing specific features associated to a sub-region (e.g., some snow-related features may be available only for a mountainous region).

Data Generation Process. We consider a general relationship between the full set of features and each target, with additive Gaussian noise. The noise variables $\{\epsilon_k\}_{k=1}^L$ can be correlated with each other, but independent from features.

In the theoretical analysis, we consider a partition $\mathcal{P}$ of the tasks. Therefore, each task $\mathcal{T}_k$ belongs to a set of the partition $\mathcal{P}_{\iota(k)}$. Without loss of generality, in the analysis we will focus on the i-th task $y_i = f_i(x_1 \dots x_D) + \epsilon_i,\ \epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$, and we will denote with $\mathcal{P}_\iota := \mathcal{P}_{\iota(i)}$ the set of $\mathcal{P}$ that contains $\mathcal{T}_i$, and with K_ι its cardinality. The aggregated (mean) target that contains y_i is therefore $\psi_\iota := \frac{1}{K_\iota}\sum_{k:y_k\in\mathcal{P}_\iota} y_k$. Finally, to simplify the computations, we assume the expected values of features and targets to be zero: $\mathbb{E}[x_k] = \mathbb{E}[y_i] = \mathbb{E}[f_i(x_1 \dots x_D)] = 0,\ \forall k \in \{1, \dots, D\},\ \forall i \in \{1, \dots, L\}$.

Loss Measure. As a natural performance measure for regression and in line with related works [7,8], in the theoretical analysis we evaluate the Mean Squared Error (MSE), focusing on its bias and variance components (bias-variance decomposition [16], see Eq. 1 of Appendix A).

Multi-task Learning via Aggregations. In the theoretical analysis of Sect. 3, we consider a task $y_i = f_i(x_1 \dots x_D) + \epsilon_i,\ \epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$, and we evaluate the effect (in terms of MSE) of learning a linear model trained on a target ψ_ι, the mean of a given subset of original targets which contains y_i. Additionally, d given transformations $\{\phi_1, \dots, \phi_d\}$ of the D original features are the inputs. They could be any zero-mean transformations (basis functions), but we focus on the means of d subsets of original features. Indeed, through Algorithm 1, we exploit the theoretical results to identify iteratively, and aggregate with the mean, convenient partitions of features and targets in polynomial time.

Remark 2 (Limitations). In the analysis, we focus on linear models to derive closed-form results. However, the NonLinCTFA algorithm can be applied with any supervised learning algorithm. In this case, its convenience becomes heuristic, and its effectiveness should be tested empirically. The second main limitation is the asymptoticity of the analysis, to identify expressions of variance and bias without considering confidence intervals of each statistical estimator.

2.1 Related Works: Dimensionality Reduction, Multi-task Learning

Dimensionality Reduction. Considering each individual task, the aggregation of its input features can be considered a dimensionality reduction method. These approaches map D features into a reduced dataset of dimension $d < D$, with transformation functions aimed to maximize specific performance measures. Linear, non-linear, supervised, and unsupervised methods exist. For broader discussions and algorithms on dimensionality reduction, we refer to [2,18,31].

In line with this work, some interpretable dimensionality reduction approaches have also been studied to increase the interpretability of classical

PCA [13], of kernel dimensionality reduction [33], or even of generative models [14]. The specific idea to aggregate groups of features with their mean resembles group regularization methods, such as Cluster LASSO [29] and OSCAR [6], although the method proposed in this paper aggregates features independently from the training of the supervised model. This is in line with some recent approaches [7,8], that we seek to generalize to a multi-task context and by considering the full set of features to identify the aggregations, rather than their bivariate analyses.

Multi-task Learning. The main algorithm of this paper is an MTL approach that identifies subsets of targets, learning a single model for their mean, that we claim to be convenient for the original individual tasks. A classical description of MTL is [10], and more recent algorithms are in [36]. Following their taxonomy, we briefly revise parameter-based and feature-based MTL approaches. A few instance-based approaches also exist, designed to share samples across tasks.

Parameter-based methods can be based on the assumption that similar tasks have similar model parameters, forcing the coefficient matrix to be low-rank (low-rank approaches, e.g., [35]); they can estimate and exploit task relationships such as covariances (task-relation approaches, e.g., [39]); they can share sets of parameters (decomposition approaches, e.g., [21]); or they can assume that different tasks form several clusters (task-clustering approaches, e.g., [24]). Some recent parameter-based MTL algorithms aim to identify groups of related tasks exploiting different metrics, such as the effect of the gradient of the loss associated with one task on the loss of another task [15], or summary statistics [19]. The approach of this paper can be considered a task-clustering method since we identify disjoint groups of tasks, averaging the corresponding targets.

Feature-based methods are based on the assumption that tasks share a common feature representation. Some approaches learn this feature representation by maximizing the information to each task (feature transformation). Among them, deep learning-based approaches are the most commonly used (e.g., [34]). Other approaches select subsets of original features, maximizing their relevance to the targets (feature selection). They usually optimize a loss function that both penalizes the selection of different features across tasks and minimizes the number of important features (e.g., [38]). The proposed algorithm combines the task-clustering phase with a feature transformation phase, where for each cluster of tasks, we identify and aggregate subsets of features.

3 Bias-Variance Analysis: Theoretical Results

In this section, we present the main theoretical results, deriving the asymptotic bias and variance of aggregated tasks. We consider a generic task $\mathcal{T}_i$:

$$y_i = f_i(x_1 \dots x_D) + \epsilon_i, \ \ \epsilon_i \sim \mathcal{N}(0, \sigma_i^2), \tag{1}$$

and the set of targets $\mathcal{P}_\iota$, that contains y_i. In this setting, we estimate the target y_i with a linear model, trained with the mean of the elements of $\mathcal{P}_\iota$ as target, and d inputs $\{\phi_1, \dots, \phi_d\}$. Recalling that $\psi_\iota := \frac{1}{K_\iota} \sum_{k: y_k \in \mathcal{P}_\iota} y_k$, we estimate:

$$\hat{y}_i = \hat{\psi}_\iota = \hat{w}_1^\iota \phi_1 + \cdots + \hat{w}_d^\iota \phi_d, \tag{2}$$

with $\hat{w}_j^\iota$ least squares estimates. In particular $\{\phi_1, \ldots, \phi_d\} = \{x_1, \ldots, x_D\}$ is the case with original features, and $\mathcal{P}_\iota = \{y_i\}$ identifies the single-task case.

Variance. We firstly derive the variance of the linear model $\hat{\psi}_\iota$ (Eq. 2), for the i-th target y_i. Proofs and additional discussions are in Appendix A.

Theorem 1. *Let the relationship between the features and the target of a task $\mathcal{T}_i$ be defined as Eq. 1. Let also each estimator converge in probability to the quantity that it estimates. In the asymptotic case, let var_d^ι be the variance of a linear regression trained with the basis functions $\{\phi_1, \ldots, \phi_d\}$ as inputs and the mean of a cluster of targets ψ_ι, defined in Eq. 2, as output. It holds:*

$$var_d^\iota = \frac{\bar{\sigma}_\iota^2}{(n-1)} \cdot d, \tag{3}$$

where $\bar{\sigma}_\iota^2 := var(\frac{1}{K_\iota} \sum_{k: y_k \in \mathcal{P}_\iota} \epsilon_k)$ is the variance of the mean of noises of the targets in $\mathcal{P}_\iota$, n is the number of training samples, and d is the number of inputs.

Remark 3. The theorem follows the intuition that the asymptotic variance is proportional to the number of inputs d (i.e., the number of estimated coefficients) and the variance gets smaller as the number of samples increases, since the estimate is more accurate. Finally, the asymptotic variance increases with $\bar{\sigma}_\iota^2$, which is the distortion associated with the averaged target ψ_ι.

Remark 4. Some specific cases are particularly relevant.

1) When the inputs are the D original features, we get $var_D^\iota = \frac{\bar{\sigma}_\iota^2}{(n-1)} \cdot D$. For a single-task approach, the variance is $var_d^i = \frac{\sigma_i^2}{(n-1)} \cdot d$. Combining them, ($D$ features and a single-task), the variance is $var_D^i = \frac{\sigma_i^2}{(n-1)} \cdot D$.
2) When the variance of the noises is constant ($\forall i : \sigma_i^2 = \sigma^2$), we get: $\bar{\sigma}_\iota^2 = \frac{\sigma^2}{K_\iota}(1 + (K_\iota - 1)\bar{\rho}_\iota)$, with $\bar{\rho}_\iota$ average correlation among noises of the targets in $\mathcal{P}_\iota$. Therefore, if $\bar{\rho}_\iota = 1$, the task aggregation does not reduce the variance since the variance of the averaged noise is equal to the individual (σ^2). On the contrary, since $\bar{\rho}_\iota \geq \frac{1}{1-K_\iota}$ by non-negativity of variance, maximum variance gain corresponds to the minimum average correlation among noises $\bar{\rho}_\iota = \frac{1}{1-K_\iota}$. Finally, when the noises are independent ($\bar{\rho}_\iota = 0$), the aggregated noise variance is reduced by a factor K_ι ($\frac{1}{K_\iota}\sigma^2$). Intuitively, when the noises are less correlated, the different tasks are better exploited to refine the knowledge about each task.
3) In [7,8] a similar asymptotic result is provided in the single-task setting, with an asymptotic variance equal to $\frac{2\sigma_i^2}{(n-1)}$ in the bivariate case and $\frac{\sigma_i^2}{(n-1)}$ in the univariate one. Theorem 1 generalizes that result to d input features, revealing a cost of $\frac{\sigma^2}{n-1}$ for each feature considered. Therefore, by reducing the features from D to $d < D$, the asymptotic variance decrease is $\frac{\sigma^2}{n-1} \cdot (D - d)$.

Bias. We now focus on the bias of the linear model $\hat{\psi}_\iota$ (Eq. 2), w.r.t. the i-th target y_i. Proofs and more discussions can be found in Appendix B.

Theorem 2. *Let the relationship between the features and the target of a task $\mathcal{T}_i$ be defined as Eq. 1. Let also each estimator converge in probability to the quantity that it estimates. In the asymptotic case, let $bias_d^\iota$ be the bias of a linear regression trained with the basis functions $\{\phi_1, \ldots, \phi_d\}$ as inputs and the mean of a cluster of targets ψ_ι, defined in Eq. 2, as output. It is equal to:*

$$bias_d^\iota = \sigma_{f_i}^2 - \sigma_{\psi_\iota}^2 R_{d,\iota}^2 + 2(cov(\psi_\iota, f_i - \psi_\iota | \mathbf{\Phi}) - cov(\psi_\iota, f_i - \psi_\iota)), \tag{4}$$

with $\sigma_{f_i}^2$ variance of the function of inputs defining the i-th task, $\sigma_{\psi_\iota}^2$ variance of the aggregated target ψ_ι, $R_{d,\iota}^2$ squared coefficient of multiple correlation between the d inputs and the aggregated target ψ_ι, and $cov(\psi_\iota, f_i - \psi_\iota | \mathbf{\Phi})$ partial covariance between the two random variables $\psi_\iota, f_i - \psi_\iota$, given the inputs $\mathbf{\Phi}$.

Corollary 1. *Considering the single-task setting, i.e., assuming $\psi_\iota = y_i$, the asymptotic bias of Theorem 2 reduces to:*

$$bias_d^i = \sigma_{f_i}^2 (1 - R_{d,i}^2). \tag{5}$$

Remark 5. Intuitively, Corollary 1 shows that, in single-task problems, the bias is proportional to the information, measured through the coefficient of multiple correlation, that the inputs share with the target. Additionally, the bias is proportional to the variance of the function f_i that regulates the i-th data generation process. More generally, Theorem 2 shows that estimating the i-th target with a linear model trained with the aggregated target ψ_ι, the bias is still proportional to the variance of the original i-th target, which is an irreducible cost. Then, the bias decreases proportionally to the coefficient of multiple correlation between aggregated target and features, representing the skill of the linear model to predict the aggregated target, weighted by the variance of the aggregated target itself, which accounts for the simplification introduced by aggregating. Finally, the third term of Eq. 4 is the difference of the covariance, with and without conditioning on the inputs, between the aggregated target and the gap between the i-th task and the aggregated target itself. This way, if the features reduce the information shared between ψ_ι and its gap with f_i, the bias reduces since we are exploiting the features to improve the learning of the i-th task.

Remark 6. Some specific cases can also be derived from the results on the bias.

1) Considering the D original features, the results of Theorem 2 and Corollary 1 hold, evaluating the quantities of the expressions with them (e.g., $R_{D,\iota}^2$).
2) In [7,8] a similar (bivariate) asymptotic analysis is provided in the single-task setting. Corollary 1 extends those findings to a general case with d inputs. Therefore, if we perform a single-task dimensionality reduction, aggregating d sets of features with their mean, the asymptotic bias variation is $\sigma_{f_i}^2(R_{D,i}^2 - R_{d,i}^2)$.

Theoretical Bounds for Aggregations. We conclude this Section by showing a condition for a convenient aggregation of two tasks and another for features in single-task problems. Both results can be deduced from Theorems 1 and 2, and they will be exploited in the two phases of the main algorithm, respectively.

Corollary 2. *Considering two tasks $\mathcal{T}_i, \mathcal{T}_j$ regulated by Eq. 1, training an individual linear model with the mean of the two targets as output is profitable w.r.t. the individual single-task models, in terms of MSE, if and only if:*

$$\begin{cases} \frac{\sigma_i^2}{(n-1)} \cdot D + \sigma_{\psi_\iota}^2 R_{D,\iota}^2 \geq \frac{\sigma_\iota^2}{(n-1)} \cdot D + \frac{1}{2}[\sigma_{f_i}^2 R_{D,i}^2 + \sigma_{f_j}^2 R_{D,j}^2] \\ \frac{\sigma_j^2}{(n-1)} \cdot D + \sigma_{\psi_\iota}^2 R_{D,\iota}^2 \geq \frac{\sigma_\iota^2}{(n-1)} \cdot D + \frac{1}{2}[\sigma_{f_i}^2 R_{D,i}^2 + \sigma_{f_j}^2 R_{D,j}^2]. \end{cases} \tag{6}$$

Proof. We compute variance and bias from Theorems 1 and 2, substituting $\psi_\iota = y_i$, $\psi_\iota = y_j$, and $\psi_\iota = \frac{y_i+y_j}{2}$. Then, we impose that the sum of variance and bias of the aggregated case is not worse than the individual one for both models.

Following the intuition, Eq. 6 shows the convenience of aggregating two targets if the variance of the noise is reduced or when the predictive capability of the aggregated model is better than the average of the single-task ones.

Corollary 3. *Considering a task $\mathcal{T}_i$, the aggregation of D features into $d < D$ aggregated ones is profitable, in terms of MSE of a linear model, if and only if:*

$$\frac{\sigma_i^2}{(n-1)} \cdot (D-d) \geq \sigma_{f_i}^2 (R_{D,i}^2 - R_{d,i}^2). \tag{7}$$

Proof. The result follows comparing the single-task bias and variance that are particular cases of the general results of Theorems 1 and 2.

Following the intuition, Eq. 7 shows the convenience in terms of variance to reduce the number of features or in terms of bias when the predictive capability of the linear regression is better when the features are aggregated.

4 Multi-task Learning via Aggregations: Algorithms

In this section, we present the NonLinCTFA algorithm, assuming L tasks and D shared features. The algorithm is depicted in Fig. 1a, while a multi-input variant, discussed in Remark 1, is depicted in Fig. 1b. Algorithm 1 reports the pseudocode of NonLinCTFA, exploiting the loop of Algorithm 1 (in Appendix D) to iteratively aggregate targets firstly, and features subsequently.

Phase I: Task-Aggregation. Firstly, the algorithm iteratively adds targets to a set, forming a partition until no aggregation is convenient. At any iteration, Eq. 6 is exploited to test the convenience of adding a target variable y_j into a set of the partition, initialized as a singleton of a random target. Additionally, a hyperparameter ϵ_1 regulates the propensity of the algorithm to aggregate.

Equation 6 regulates the convenience of an aggregation of two targets. Therefore, the algorithm starts with L individual tasks, and it identifies two tasks that benefit from the aggregation (e.g., $\mathcal{T}_1, \mathcal{T}_2$), moving to $L-1$ linear models. Then, the algorithm further aggregates a third task (e.g., $\mathcal{T}_3$) with them, moving to $L-2$ linear models, if the new aggregation ($\frac{y_1+y_2+y_3}{3}$ in the example) is convenient w.r.t. the aggregate two-task model ($\frac{y_1+y_2}{2}$) and the univariate one (y_3). The new aggregation is therefore convenient w.r.t. the original individual tasks, and it is further convenient w.r.t. the previous two-task aggregation.

Similar to forward feature selection, we add a target in the current set, if convenient, without inspecting all the possible combinations, which would be combinatorial. This way, we do not identify the *optimal* partition of targets, but a convenient one w.r.t. the single-tasks, with a quadratic number of comparisons in the number of tasks ($\mathcal{O}(L^2)$) in the worst case, i.e., with no aggregations.

Additionally, the aggregation depends on the ordering of targets. For this reason, we randomize it to avoid systemic biases. A possible variation could be to introduce a heuristic (e.g., the correlation between couples of targets) to rank them and test for the aggregations based on this ranking.

Phase II: Cluster-Level Feature-Aggregation. In the second phase, NonLinCTFA identifies groups of features to aggregate with their mean for each of the l reduced tasks. This way, for each reduced task ι, we identify d_ι reduced features, reducing the dimension and improving the performance.

Specifically, we exploit Eq. 7 to iteratively identify couples of features that are convenient to average, following the same iterative procedure of the target aggregation phase. This is quadratic w.r.t. the number of comparisons for each aggregated task ($\mathcal{O}(l \cdot D^2)$). Additionally, since the terms $\sigma_i^2, \sigma_{f_i}^2, n$ are constant across different comparisons, we include them in the hyperparameter ϵ_2, which regulates the propensity of the algorithm to aggregate features.

Remark 7. Algorithm 1 outputs a set of reduced tasks and the associated sets of reduced features. We do not include a final regression to decouple the algorithm from the regression model, which can be run independently, with theoretical guarantees in linear regression. In this sense, we propose a *filter* method.

Remark 8. Recalling Remark 1, we may want to consider a multi-input setting, with D features for each task individually $\{x_1^i, \dots, x_D^i\}_{i=1}^L$. In our example, they can be D meteorological features measured at each location and associated with its specific target. A variant of Algorithm 1 (Fig. 1b) compares, at each iteration, the model trained with the features associated with each individual target and the model for the aggregated task, trained on averaged features. This way, in a single phase, we aggregate couples of targets and pairwise couples of features. In our example, we would compare individual measurements of temperature and precipitation for single-task problems and the means of temperatures and precipitations as the two inputs of the averaged model. This variant can be extended to a heterogeneous case where, for example, the first task has a snow-related feature not available for the other. In this case, we consider it when

Algorithm 1 . NonLinCTFA:Non-Linear Correlated Target-Feature Aggregation

```
Require: features $\boldsymbol{x} = \{x_1 \dots x_D\}$; targets $\boldsymbol{y} = \{y_1 \dots y_L\}$; $n$ samples, tolerances $\epsilon_1, \epsilon_2$
Ensure: reduced tasks $\{\psi_1, \dots, \psi_l\}$, reduced features $\{\phi_1, \dots, \phi_{d_\iota}\}_{\iota=1}^{l}$
  function COMPUTE_THRESHOLD_FEATURES($\boldsymbol{x}_{curr}, y, z_\mathcal{P}, z_j, \epsilon$)    ▷ From Eq. 7
      $R_{sep} \leftarrow$ R2score($\boldsymbol{x}_{curr}, y$)
      $R_{aggr} \leftarrow$ R2score(($\boldsymbol{x}_{curr} \setminus \{z_\mathcal{P}, z_j\}) \cup \{mean(z_\mathcal{P}, z_j)\}, y$)
  return $R_{sep} - R_{aggr} \leq \epsilon$
  end function
  function COMPUTE_THRESHOLD_TARGETS($\boldsymbol{x}, y_\mathcal{P}, y_j, \epsilon$)    ▷ From Eq. 6
      $y_{ag} \leftarrow mean(y_\mathcal{P}, y_j)$
      $R_\mathcal{P}, \sigma^2_\mathcal{P}, \sigma^2_{f_\mathcal{P}} \leftarrow$ R2score($\boldsymbol{x}, y_\mathcal{P}$), var_res($\boldsymbol{x}, y_\mathcal{P}$), var($y_\mathcal{P}$) − var_res($\boldsymbol{x}, y_\mathcal{P}$)
      $R_j, \sigma^2_j, \sigma^2_{f_j} \leftarrow$ R2score($\boldsymbol{x}, y_j$), var_res($\boldsymbol{x}, y_j$), var($y_j$) − var_res($\boldsymbol{x}, y_j$)
      $R_{ag}, \sigma^2_{ag}, \sigma^2_{f_{ag}} \leftarrow$ R2score($\boldsymbol{x}, y_{ag}$), var_res($\boldsymbol{x}, y_{ag}$), var($y_{ag}$) − var_res($\boldsymbol{x}, y_{ag}$)
      $threshold_1 = \frac{D}{(n-1)}(\sigma^2_{ag} - \sigma^2_\mathcal{P}) + \frac{1}{2}(R_\mathcal{P}\sigma^2_{f_\mathcal{P}} + R_j\sigma^2_{f_j}) - R_{ag}\sigma^2_{f_{ag}}$
      $threshold_2 = \frac{D}{(n-1)}(\sigma^2_{ag} - \sigma^2_j) + \frac{1}{2}(R_\mathcal{P}\sigma^2_{f_\mathcal{P}} + R_j\sigma^2_{f_j}) - R_{ag}\sigma^2_{f_{ag}}$
  return ($threshold_1 \leq \epsilon$) AND ($threshold_2 \leq \epsilon$)
  end function

  function NONLINCTFA($Input$)    ▷ Main function
      PHASE I: task aggregations
      $\{\psi_1, \dots, \psi_l\} \leftarrow$ AGGREGATION($\boldsymbol{z} = \boldsymbol{y}, phase = 1, \epsilon = \epsilon_1, \boldsymbol{x} = \boldsymbol{x}, y = None$)
      PHASE II: feature aggregation for each task
      for each $\iota \in \{1, \dots, l\}$ do
          $\{\phi_1, \dots, \phi_{d_\iota}\} \leftarrow$ AGGREGATION($\boldsymbol{z} = \boldsymbol{x}, phase = 2, \epsilon = \epsilon_2, \boldsymbol{x} = None, y = \psi_\iota$)
      end for
  return $\{\psi_1, \dots, \psi_l\}, \{\phi_1, \dots, \phi_{d_\iota}\}_{\iota=1}^{l}$
  end function
```

the first task appears, keeping unchanged the aggregation of targets and other features.

5 Experimental Validation

In this section we show synthetic experiments, validating the proposed algorithm w.r.t. single-task regressions, showing its behavior by varying parameters, and with an ablation study of its two phases. Then, applications to real-world data show the competitiveness of the method w.r.t. single task and benchmark MTL approaches[1]. In particular, we firstly show the competitiveness of the proposed algorithm on standard benchmark datasets and in comparison with classical MTL approaches. Then, we further validate the proposed algorithm on a large challenging dataset related to molecules, in comparison with some state-of-the-art approaches, both re-running implementations from recently developed libraries and considering results from some related works, where this dataset has

[1] Code can be found at: https://github.com/PaoloBonettiPolimi/NonLinCTFA.

been considered to test graph neural network-based MTL approaches. Finally, we show an application on meteorological data, which are the main applicative interest of this work.

5.1 Synthetic Experiments and Ablation Study

We start with synthetic experiments, validating the NonLinCTFA against single tasks. Specifically, we consider $L = 10$ tasks, $D = 100$ features shared across all tasks, $n = 250$ training samples (same number for testing), the standard deviation of (independent) noises $\sigma = 10$, and hyperparameters $\epsilon_1 = 0, \epsilon_2 = 0.0001$. Each target is obtained as a linear combination of all the features, with additive Gaussian noise, randomly sampling each coefficient from a uniform distribution in the interval $[-1, -0.5]$ or $[0.5, 1]$, obtaining two groups of tasks similar among themselves. We perform linear regression on individual tasks and on aggregated tasks after the first phase of the algorithm, or fully applying NonLinCTFA. We repeat the experiment 10 times to produce confidence intervals, and we consider as metrics the MSE and the coefficient of determination (R^2), both in terms of absolute values and of percentage increase w.r.t. single-task. We obtain a single-task average R^2 score of 0.48 ± 0.02, which increases to $0.64 \pm 0.01 (+33.45 \pm 3.31\%)$ considering the aggregated tasks and $0.67 \pm 0.01 (+39.82 \pm 3.42\%)$ adding the feature aggregation. Similarly, the MSE is equal to 9.34 ± 0.04, reducing to $6.54 \pm 0.03 (-29.44 \pm 1.45\%)$ and $5.98 \pm 0.06 (-35.36 \pm 1.27\%)$. The $L = 10$ tasks become $l = 2.5 \pm 0.6$, and the $D = 100$ features reduce to $d = 3.43 \pm 1.76$. These results empirically validate the improvement provided by the proposed algorithm w.r.t. the single-task counterparts in linear regression. Additionally, they show a significant benefit with task-aggregation, with a subsequent feature-aggregation phase refining the performances, with the added value of simplifying models.

We further tested the proposed approach, varying one parameter at a time, and fixing the others. Figure 3 reports the results, in terms of MSE and of percentage increase of MSE w.r.t. the single-task, showing the improvement with task-aggregation and then the combination with feature aggregation. The first column of the figure shows that, with a small number of samples, both phases provide significant improvement, which is mitigated by a large number of samples. In the second column, when the number of features increases, the feature aggregation phase is more relevant. Similarly, the third column shows that the task-aggregation phase is more relevant when the number of tasks increases, assuming a constant number of features. Finally, the fourth column shows that the problem becomes more difficult when the noise of the targets increases, with the task aggregations that become more relevant. Figure 1 in Appendix E also shows the behavior of the MSE when the hyperparameters ϵ_1, ϵ_2 are varied. In particular, ϵ_1 represents the propensity to aggregate tasks, spanning from the single-task case to a single aggregated task, with a value equal to 0 that balances the aggregation. Similarly, ϵ_2 regulates the propensity to aggregate features, spanning from preserving all original inputs to a single aggregation.

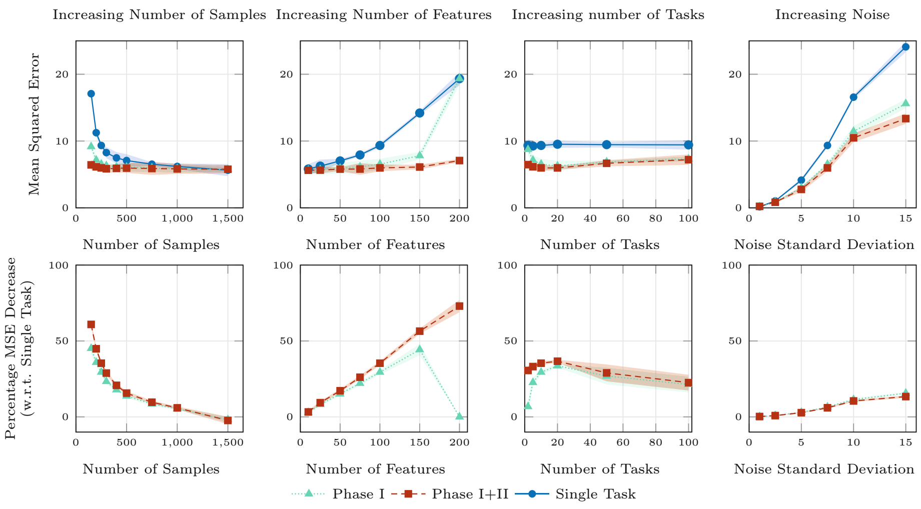


Fig. 3. Test MSE (first row) and corresponding percentage decrease w.r.t. single-task (second row), varying one parameter at a time, only aggregating targets (Phase I) or adding the feature-aggregation phase (Phase I+II).

5.2 Real World Datasets

As a first investigation on real data, we consider two classical datasets, comparing single-task regression performance, applying NonLinCTFA, and with some standard MTL approaches. In particular, the SARCOS dataset [32] is composed of 7 regression tasks, 21 shared features, and $n = 48933$ samples. We consider $n = 1000$ and the full set of $n = 48933$ samples to investigate the effect of the number of samples. Then, the *School* dataset [4] consists of 15362 samples, exam records, distributed across 139 tasks (schools), with 27 features. This is an example where different measures of the same features are associated to their tasks, allowing to apply the variant of NonLinCTFA discussed in Remark 8. In this case, we consider 27 tasks and 27 features, as well as the full set of 139 tasks.

We apply NonLinCTFA, combined with linear regression (LR), support vector regression (SVR), and multi-layer perceptron (MLP), in comparison with the corresponding single-task models. Additionally, we consider a random prediction as a trivial baseline and Alternating Structural Optimization [1, ASO], its convex relaxation [11, cASO], and Convex Clustered MTL [17, CMTL], as benchmarks representing feature-based and clustered-based MTL methods (adapting the implementation of https://github.com/chcorbi/MultiTaskLearning). In line with the implementation of these baselines, we evaluate test performance with normalized root mean squared error (NRMSE), randomly selecting the 30% of samples, with five different seeds, to produce test confidence intervals.

The results of Table 1 show that NonLinCTFA outperforms single-task models and is competitive w.r.t. MTL baselines, with the advantage of reducing the number of models and parameters, still preserving the interpretability.

Table 1. Experiments on SARCOS and *School* datasets, considering 70% of data for training and 30% for testing, randomizing over 5 seeds for confidence intervals. NRMSE is the performance measure (lower is better, best result in bold).

	SARCOS_1000samples	SARCOS_full	School_27tasks	School_full
# samples n	1000	48933	15362	15362
# tasks L	7	7	27	193
Reduced # tasks (**ours**)	6.0 ± 0.0	6.0 ± 0.0	6.0 ± 0.0	23.2 ± 2.7
# features D	21	21	27 (for each task)	27 (for each task)
Reduced # features (**ours**)	7.1 ± 0.3	6.9 ± 0.4	27 (for each aggregation)	27 (for each aggregation)
NRMSE random	0.363 ± 0.015	0.235 ± 0.011	0.634 ± 0.123	0.641 ± 0.294
NRMSE single-task LR	0.085 ± 0.001	0.069 ± 0.002	0.183 ± 0.002	0.165 ± 0.003
NRMSE single-task SVR	0.142 ± 0.010	0.092 ± 0.008	0.181 ± 0.009	0.162 ± 0.005
NRMSE single-task MLP	0.069 ± 0.002	0.045 ± 0.003	0.182 ± 0.003	0.174 ± 0.008
NRMSE ASO	0.075 ± 0.002	0.049 ± 0.001	0.172 ± 0.003	$\mathbf{0.152 \pm 0.006}$
NRMSE cASO	0.068 ± 0.001	0.048 ± 0.001	0.173 ± 0.002	0.154 ± 0.006
NRMSE CMTL	0.111 ± 0.001	0.067 ± 0.002	0.843 ± 0.316	1.187 ± 0.562
NRMSE NonLinCTFA + LR (**ours**)	0.054 ± 0.003	0.035 ± 0.002	0.162 ± 0.005	0.159 ± 0.002
NRMSE NonLinCTFA + SVR (**ours**)	0.154 ± 0.021	0.115 ± 0.024	$\mathbf{0.159 \pm 0.003}$	0.155 ± 0.004
NRMSE NonLinCTFA + MLP (**ours**)	$\mathbf{0.049 \pm 0.009}$	$\mathbf{0.031 \pm 0.001}$	0.160 ± 0.004	0.167 ± 0.003

In a second real-world application, we consider the QM9 dataset [28], a challenging quantum chemistry dataset with $L = 19$ tasks (properties of molecules), with 139000 graph inputs (molecules structures). We averaged node features, position, and edge attributes, obtaining $D = 19$ features. Following the experimental setup of [27], we retrieved the dataset from PyTorch Geometric, randomly selecting 10000 samples for testing and the others for training, normalizing each task and repeating the experiments three times to produce confidence intervals.

Table 2. Experiments on QM9 dataset. 10000 random samples are used for testing ($\sim$ 138000 for training), 3 different seeds for confidence intervals. MSE is the performance measure (lower is better). Training time is also reported.

QM9 Test Results	# reduced tasks	# reduced features	MSE	Time (minutes)
Single-task LR	19	20	0.969 ± 0.049	~ 1
Single-task MLP	19	20	0.518 ± 0.032	~ 11
Single-task baseline of [30]	19	13	0.533 ± 0.041	NA
HPS GNN + RLW	19	11+graph	0.619 ± 0.254	$\sim 1 \times 300$epochs
Best GNN of [30]	19	11+graph	0.216 ± 0.009	NA
NonLinCTFA + LR (**ours**)	12 ± 1.2	10.05 ± 2.96	0.955 ± 0.038	~ 2
NonLinCTFA + MLP (**ours**)	12 ± 1.2	10.05 ± 2.96	0.469 ± 0.024	~ 14

Table 2 shows the performance, in terms of test MSE, of the single-task linear regression (LR) and multi-layer perceptron (MLP), together with the MSE associated to the same models, applying NonLinCTFA. Additionally, we exploited the implementation of LibMTL [23] of some state-of-the-art MTL methods and its integration with the QM9 dataset for further benchmarking. In particular, the architecture of the library compatible with graph neural networks is the hard parameter sharing (HPS GNN) that we trained with all the 14 weighting strategies implemented in the library (we refer to it for details), up to 300 epochs, identifying the Random Loss Weighting strategy [22, RLW] as best performing. For further comparison with the literature, in the table, we also show the MSE of the baseline and the best-performing graph neural network (GNN) proposed in [30]. Together with the MSE, Table 2 shows the computational time. We can conclude that the NonLinCTFA provides significant aggregations, improving single task performances, especially combined with the MLP. Additionally, it is competitive w.r.t. GNN-based approaches, without outperforming all of them, given its much simpler tabular methodology, as also highlighted by the computational time.

A final experimental setup shows an application of the NonLinCTFA Algorithm to meteorological data, as depicted in Fig. 2, where $L = 29934$ European hydrological basins are considered as tasks, each with a satellite signal as target, and $D = 16$ meteorological measurements and climate indices as inputs. In this context, we apply the variant of the algorithm described in Remark 8, with linear regression (given the small amount of $n = 102$ monthly measurements and the necessity to preserve the interpretability of the entire workflow).

Table 3. Experiments on climate dataset with NonLinCTFA, varying the hyperparameter ϵ. The number of aggregations and MSE are obtained cross-validating.

NonLinCTFA	$\epsilon = 1$ (single-task)	$\epsilon = 0.5$	$\epsilon = 0.1$	$\epsilon = 0.05$	$\epsilon = 0.01$
# reduced tasks	29934	28024	2354	1345	969
MSE	1.058 ± 0.224	1.045 ± 0.221	0.758 ± 0.148	0.755 ± 0.143	0.753 ± 0.143
	$\epsilon = 0$	$\epsilon = -0.01$	$\epsilon = -0.05$	$\epsilon = -0.1$	$\epsilon = -1$
# reduced tasks	944	844	680	252	1
MSE	0.750 ± 0.140	0.746 ± 0.142	0.756 ± 0.146	0.909 ± 0.189	1.132 ± 0.193

Table 3 shows confidence intervals, in terms of MSE, associated with different values of the hyperparameter ϵ. As expected, the MSE on the original tasks reduces when reducing the value of ϵ since the aggregations of tasks are convenient. However, when the hyperparameter is too small, the algorithm aggregates too many tasks, becoming detrimental to the understanding of the behavior of the original tasks, until the limit case of a single aggregation of all the tasks. The average number of reduced tasks is also reported in the table to confirm this behavior.

6 Conclusions and Future Developments

In this paper, we introduced a two-phase MTL approach (NonLinCTFA) that aggregates sets of targets and features with their mean, motivated by meteorological applications, and aimed to improve the final regression performance, preserving interpretability. We provided a bias-variance analysis, considering linear regression, and we empirically validated the approach with synthetic and real-world datasets, showing promising results, also outside the context of linear regression. A future development can be an extension of the analysis to general ML models. Additionally, an applicative work with meteorological data is under development.

Acknowledgments. This work has been supported by the CLINT research project funded by the H2020 Programme of the European Union under Grant Agreement No 101003876. This work has been funded by the European Union - Next Generation EU within the project NRPP M4C2, Investment 1.,3 DD. 341 - 15 march 2022 - FAIR - Future Artificial Intelligence Research - Spoke 4 - PE00000013 - D53C22002380006.ci.

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

References

1. Ando, R.K., Zhang, T., Bartlett, P.: A framework for learning predictive structures from multiple tasks and unlabeled data. J. Mach. Learn. Res. **6**(11), 1817–1853 (2005)
2. Ayesha, S., Hanif, M.K., Talib, R.: Overview and comparative study of dimensionality reduction techniques for high dimensional data. Inf. Fusion **59**, 44–58 (2020)
3. Bachmann, R., Mizrahi, D., Atanov, A., Zamir, A.: MultiMAE: multi-modal multitask masked autoencoders. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. ECCV 2022. LNCS, vol. 13697, pp. 348–367. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19836-6_20
4. Bakker, B., Heskes, T.: Task clustering and gating for Bayesian multitask learning (2003)
5. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, Heidelberg (2006)
6. Bondell, H.D., Reich, B.J.: Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with OSCAR. Biometrics **64**(1), 115–123 (2008)
7. Bonetti, P., Metelli, A.M., Restelli, M.: Interpretable linear dimensionality reduction based on bias-variance analysis. arXiv preprint arXiv:2303.14734 (2023)
8. Bonetti, P., Metelli, A.M., Restelli, M.: Nonlinear feature aggregation: two algorithms driven by theory. arXiv preprint arXiv:2306.11143 (2023)
9. Boribalburephan, A., Treewaree, S., Tantisiriwat, N., Yindeengam, A., Achakulvisut, T., Krittayaphong, R.: Myocardial scar and left ventricular ejection fraction classification for electrocardiography image using multi-task deep learning. Sci. Rep. **14**(1), 7523 (2024)
10. Caruana, R.: Multitask learning. Mach. Learn. **28**, 41–75 (1997)
11. Chen, J., Tang, L., Liu, J., Ye, J.: A convex formulation for learning shared structures from multiple tasks. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 137–144 (2009)
12. Chen, Q., Zhuo, Z., Wang, W.: BERT for joint intent classification and slot filling. arXiv preprint arXiv:1902.10909 (2019)

13. Chipman, H.A., Gu, H.: Interpretable dimension reduction. J. Appl. Stat. **32**(9), 969–987 (2005)
14. Ding, J., Condon, A., Shah, S.P.: Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. Nat. Commun. **9**(1), 2002 (2018)
15. Fifty, C., Amid, E., Zhao, Z., Yu, T., Anil, R., Finn, C.: Efficiently identifying task groupings for multi-task learning. Adv. Neural. Inf. Process. Syst. **34**, 27503–27516 (2021)
16. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. SSS, Springer, New York (2009). https://doi.org/10.1007/978-0-387-84858-7
17. Jacob, L., Vert, J.P., Bach, F.: Clustered multi-task learning: a convex formulation. In: Advances in Neural Information Processing Systems, vol. 21 (2008)
18. Jia, W., Sun, M., Lian, J., Hou, S.: Feature dimensionality reduction: a review. Complex Intell. Syst. **8**, 2663–2693 (2022)
19. Knight, P., Duan, R.: Multi-task learning with summary statistics. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
20. Kovalerchuk, B., Ahmad, M.A., Teredesai, A.: Survey of explainable machine learning with visual and granular methods beyond quasi-explanations. In: Pedrycz, W., Chen, S.-M. (eds.) Interpretable Artificial Intelligence: A Perspective of Granular Computing. SCI, vol. 937, pp. 217–267. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-64949-4_8
21. Li, Y., Yan, H., Jin, R.: Multi-task learning with latent variation decomposition for multivariate responses in a manufacturing network. IEEE Trans. Autom. Sci. Eng. **20**, 285–295 (2023)
22. Lin, B., Ye, F., Zhang, Y., Tsang, I.W.: Reasonable effectiveness of random weighting: a litmus test for multi-task learning. arXiv preprint arXiv:2111.10603 (2021)
23. Lin, B., Zhang, Y.: LibMTL: a Python library for multi-task learning. arXiv:abs/2203.14338 (2022)
24. Liu, A., Su, Y., Nie, W., Kankanhalli, M.: Hierarchical clustering multi-task learning for joint human action grouping and recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**, 102–114 (2017)
25. Ma, Y., Mao, R., Lin, Q., Wu, P., Cambria, E.: Quantitative stock portfolio optimization by multi-task learning risk and return. Inf. Fusion **104**, 102165 (2024)
26. Mignone, P., Pio, G., Džeroski, S., Ceci, M.: Multi-task learning for the simultaneous reconstruction of the human and mouse gene regulatory networks. Sci. Rep. **10**(1), 22295 (2020)
27. Navon, A., et al.: Multi-task learning as a bargaining game. arXiv preprint arXiv:2202.01017 (2022)
28. Ramakrishnan, R., Dral, P.O., Rupp, M., Von Lilienfeld, O.A.: Quantum chemistry structures and properties of 134 kilo molecules. Sci. Data **1**(1), 1–7 (2014)
29. She, Y.: Sparse regression with exact clustering. Stanford University (2008)
30. Tong, A., et al.: Learnable filters for geometric scattering modules. arXiv:abs/2208.07458 (2022)
31. Van Der Maaten, L., Postma, E., Van den Herik, J.: Dimensionality reduction: a comparative review. J. Mach. Learn. Res. **10**, 66–71 (2009)
32. Vijayakumar, S., D'souza, A., Schaal, S.: Incremental online learning in high dimensions. Neural Comput. **17**(12), 2602–2634 (2005)
33. Wu, C., Miller, J., Chang, Y., Sznaier, M., Dy, J.: Solving interpretable kernel dimensionality reduction. In: Advances in Neural Information Processing Systems, vol. 32 (2019)

34. Ye, H., Xu, D.: TaskPrompter: spatial-channel multi-task prompting for dense scene understanding. In: ICLR (2022)
35. Zhang, Y., Zhang, Y., Wang, W.: Learning linear and nonlinear low-rank structure in multi-task learning. IEEE Trans. Knowl. Data Eng. **35**, 8157–8170 (2023)
36. Zhang, Y., Yang, Q.: A survey on multi-task learning. IEEE Trans. Knowl. Data Eng. **34**(12), 5586–5609 (2021)
37. Zhao, Y., Wang, X., Che, T., Bao, G., Li, S.: Multi-task deep learning for medical image computing and analysis: a review. Comput. Biol. Med. **153**, 106496 (2023)
38. Zhong, Y., Xu, W., Gao, X.: Heterogeneous multi-task feature learning with mixed $l_{2,1}$ regularization. Mach. Learn. **113**, 891–932 (2023)
39. Zhou, M., Yang, P.: Automatic temporal relation in multi-task learning. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2023)

The Simpler The Better: An Entropy-Based Importance Metric to Reduce Neural Networks' Depth

Victor Quétu(✉), Zhu Liao, and Enzo Tartaglione

LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France
{victor.quetu,zhu.liao,enzo.tartaglione}@telecom-paris.fr

Abstract. While deep neural networks are highly effective at solving complex tasks, large pre-trained models are commonly employed even to solve consistently simpler downstream tasks, which do not necessarily require a large model's complexity. Motivated by the awareness of the ever-growing AI environmental impact, we propose an efficiency strategy that leverages prior knowledge transferred by large models. Simple but effective, we propose a method relying on an **E**ntropy-b**AS**ed **I**mportance m**E**t**R**ic (**EASIER**) to reduce the depth of over-parametrized deep neural networks, which alleviates their computational burden. We assess the effectiveness of our method on traditional image classification setups. Our code is available at https://github.com/VGCQ/EASIER.

Keywords: Compression · Efficiency · Deep Learning

1 Introduction

Deep Neural Networks (DNNs) have drastically changed the field of computer vision. They have been crucial in obtaining state-of-the-art results in several important computer vision domains, such as semantic segmentation [8], classification [26], and object detection [47]. Beyond traditional computer vision tasks, DNNs have also impacted other fields by exhibiting unbridled potentials in natural language processing [45], and multi-modal tasks [41]. DNNs' use is growing significantly in our lives and appears to be perennial.

Despite DNNs have demonstrated scalability in terms of model and dataset size [21], they hinder high computational demands. Indeed, neoteric architectures are made up of millions, or even billions, of parameters, resulting in billions, or even trillions, of FLoating-point OPerations (FLOPs) for a single inference [17]. Hence, these large models require enormous resources both in terms of pure hardware capacity and energy consumption, for training and deployment, which raises issues for real-time and on-device applications and also has an environmental impact. For instance, GPT-3 [6], made of 175B parameters, emits around

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_6.

200tCO_2eq for its training and its operational carbon footprint reached around 550tCO_2eq [14].

The development of compression techniques, which constitute an essential means of remedying the resource-hungry nature of DNNs, has marked the research landscape over the past decade. It is well-known that the complexity of the model is intrinsically linked to the generalizability of DNNs [21], and since pre-trained architectures that can be used in downstream tasks tend to be over-parameterized, compression with no (or only slight) performance degradation is in principle possible [43]. To design a more efficient architecture, a set of methods has been proposed, ranging from parameter pruning [18] to the reduction of numerical precision [37]. Nonetheless, few approaches are capable of lessening the number of layers in a DNN. Indeed, removing single parameters or whole filters offers very few if any, practical benefits when it comes to using the model on recent computing resources, such as GPU. Thanks to the intrinsic parallel computation nature of GPUs or TPUs, the limitation on layer size, whether larger or smaller, comes mainly from memory caching and core availability.

In most cases, this parallelization capability avoids the need to reduce layer size, suggesting that another approach needs to be explored to address this problem. Indeed, reducing the critical path that computations must traverse [2] would help to relieve the DNN's computation demand, which can be achieved by strategically removing layers. Despite that existing approaches, like knowledge distillation [22], implicitly tackle this issue, the absence of performance degradation cannot be guaranteed, since a shallow target model is imposed. This motivates the exploration of designing a method for neural networks' depth reduction while preserving optimal performance.

In this work, we present our method EASIER, which iteratively tries to reduce the depth of deep neural networks. More precisely, EASIER identifies the average state of a given rectifier-activated neuron for the trained task. Given the definition of rectifier activation functions, EASIER can find the probability that this neuron uses one of the two regions, and hence can calculate an entropy-based metric per layer. Such a metric is then used to drive the linearization of layers toward neural network depth reduction. We summarize, here below, our key messages and contributions.

– We highlight how we can potentially reduce the depth of a neural network with a marginal impact on the performance by characterizing layer degeneration (Sect. 3.1).
– We propose EASIER, a method relying on an entropy-based importance metric that pinpoints rectifier-activated layers that can be linearized (Sect. 3.3) (Fig. 1).
– We test EASIER across multiple architectures and datasets for traditional image classification setups (Sect. 4), demonstrating that layer withdrawal can be achieved with little or no performance loss when over-parameterized networks are employed. Notably, we show the potential savings in terms of FLOPs and inference time on six different hardwares, highlighting the benefits of our method (Sect. 4.3).

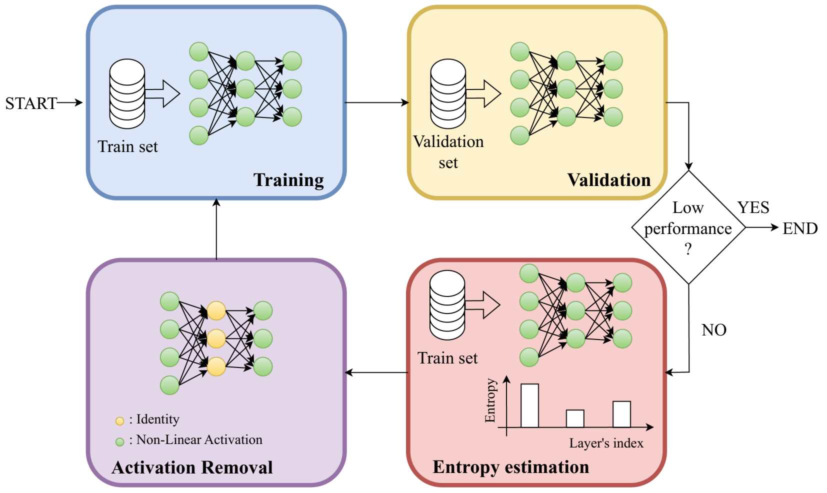


Fig. 1. Overview of EASIER. We iteratively train, evaluate, and estimate the entropy on the training set and linearize the lowest-entropy layer of the neural network, until the performance drops.

2 Related Works

Neural Architecture Search. Popular deep neural network architectures have mostly been designed by hand, among which we can cite VGG [40], ResNet [19], MobileNet [23] or Swin transformer [31]. Despite leading to remarkable performance on a variety of tasks, the design of novel architectures is time-consuming and can be prone to errors. Neural Architecture Search (NAS) was the answer to both these problems. Divided into subgroups such as evolutionary methods [34], methods based on reinforcement learning [49], and differentiable methods [30], NAS is finding the contemporary top-performing architectures [3]. While the firsts are based on efficient heuristic search methods based on evolution to capture global solutions of complex optimization problems [38], the second relies on goal-oriented optimization methods driven by an impact response or signal [1]. Differentiable methods learn architectural paths that enable the removal of entire layers and sometimes add width to the previous ones to balance [46]. By disentangling training and searching to reduce the cost, a popular approach proposed a large once-for-all network [7] supporting diverse architectural designs. The idea was to select a sub-network within the aforementioned model without the need for additional training.

Nonetheless, despite reducing the model size across diverse dimensions, like depth, width, kernel size, and resolution, NAS approaches, including also this work, generally need expensive computational resources to span several search space dimensions and train a super-network from scratch. In this paper, our sole focus lies on depth as the exclusive search dimension, by leveraging a pre-trained model, making easier convergence and reducing the overall training time.

Neural Network Pruning. Neural network pruning, whose goal is to shrink a large network to a smaller one while maintaining performance by removing irrelevant weights, filters, or other structures from neural networks, has gained significant attention in neoteric works since it allows a possible model performance enhancement and an over-fitting reduction. On the one hand, *structured* pruning focuses on removing entire neurons, filters, or channels [20,44]. On the other hand, *unstructured* pruning algorithms discard weights without explicitly taking the neural network's structure into account [18,43]. The main categories of unstructured pruning methods are magnitude-based pruning [18,32,48] and gradient-based pruning [28,43]. While the first eponymous approach takes the weights' magnitude as an importance score to prune parameters, the latter uses the gradient magnitude (or its higher-order derivatives) to rank them. The effectiveness of these techniques was compared by [4] and, in general, magnitude-based methods are more accurate than gradient-based. Moreover, they are a good trade-off between complexity and competitiveness. Indeed, [15] exposed that simple magnitude pruning approaches reach similar or better results than complex methods. From a computational perspective, in a general-purpose hardware configuration, larger benefits in terms of both memory and computation are produced by structured pruning compared to unstructured pruning, even though the reached sparsity can be significantly lower [5].

However, a recent work [29] proposed an unstructured Entropy-Guided Pruning (EGP) algorithm, that succeeds in reducing the depth of deep neural networks by prioritizing pruning connections in low-entropy layers, leading to their entire removal while preserving performance. Our method differs from the latter since EASIER considers a third state to calculate the entropy (Sect. 3) and unlike EGP, our method does not involve pruning. Although effective, EGP only allows a small number of layers to be removed. Indeed, after the removal of multiple layers, the accuracy drops dramatically. This will be verified by comparing this method with EASIER, in Sect. 4.

Activation Withdrawal. Private inference has led to an upsurge in works on removing non-linear activations. Indeed, a high latency penalty is incurred when computing on encrypted data, which is mainly due to non-linear activations such as ReLU. Methods such as DeepReduce [24] and SNL [9] have been developed to reduce private inference latency. While DeepReduce includes both optimizations for ReLU dropping and knowledge distillation training to maximize the performance, the latter proposes a gradient-based algorithm that selectively linearizes ReLUs while maintaining prediction accuracy. However, although SNL significantly reduces the number of ReLU units in the neural network, it never removes activation from an entire layer, but only from units such as pixels or channels. In contrast, our method focuses on removing activation functions at a layer level, in order to reduce the depth of deep neural networks. Moreover, DeepReduce [24] is based on a criticality metric requiring five optimized networks per optimization iteration, resulting in the exploration of $5 \times (D - 1)$ network architectures, for a network with D stages, which is not very efficient at training time. On the other hand, our method does not require leveraging the knowledge of a teacher model to boost performance, saving computation at training time. Although left for future work, we believe our work can also be effective in accelerating private inference.

In traditional classification setups, [13] introduced Layer Folding, a technique that determines whether non-linear activations can be withdrawn, enabling the folding of adjacent linear layers into one. More specifically, PReLU activations with a trainable slope, replace ReLU-activated layers. The almost linear PReLUs are eliminated post-training, enabling the layer to be folded with its successive one. Furthermore, a comparable channel-wise method enabling a notable reduction in non-linear units in the neural network while preserving performance was put forward by [2]. While the latter does not aim at reducing neural network depth, Layer Folding was originally proposed only for ReLU-activated networks. Designed for any rectifier, we will compare our method EASIER with Layer Folding and demonstrate its effectiveness in Sect. 4.

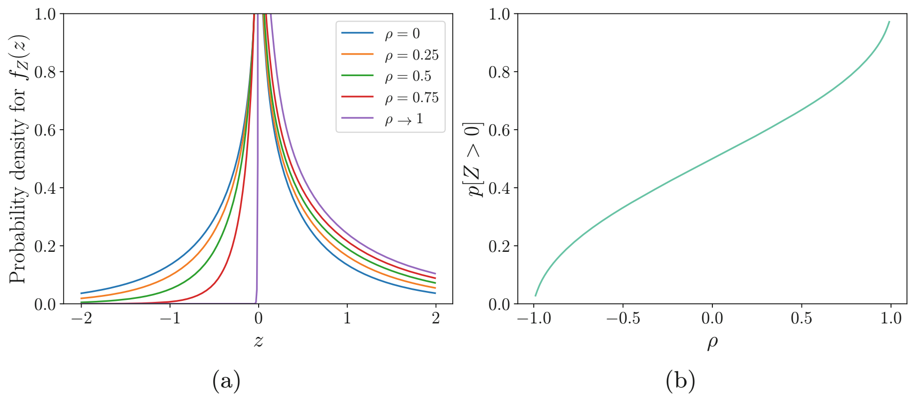


Fig. 2. Distribution of the product between $X \sim \mathcal{N}(0,1)$ and $W \sim \mathcal{N}(0,1)$ for different values of ρ (a), and $p[Z > 0]$ for different ρ (b).

3 Method

In this section, we first highlight how we can potentially reduce the depth of a neural network with a marginal impact on performance. Based on this observation, we then derive an entropy formulation for rectifier activations, which will be at the heart of our EASIER method.

3.1 How Layers Can Degenerate

Let us define the input $\boldsymbol{x}$ for a given neuron is a sequence of random variables $X \sim \mathcal{N}(\mu_X, \sigma_X^2)$. Similarly, we can assume the N parameters populating such neuron, for a large N limit, follow as well a Gaussian distribution, and we model it as $W \sim \mathcal{N}(\mu_W, \sigma_W^2)$. Under the assumption of $\mu_X = \mu_W = 0$ (for narration purposes, it is possible to derive a more general result according to [12]), we can obtain the distribution for the pre-activation z (resulting from the product of the weights and the input, modeled through the random variable Z), according to the result obtained by [11,12,39], follows the probability density function

$$f_Z(z) = \frac{1}{\pi\sigma_X\sigma_W\sqrt{1-\rho^2}} \exp\left[\frac{\rho z}{\sigma_X\sigma_W(1-\rho^2)}\right] K_0\left[\frac{|z|}{\sigma_X\sigma_W(1-\rho^2)}\right], \quad (1)$$

where K_n is the n-th order modified Bessel function of the second kind and ρ is the correlation coefficient between X and W. A visual representation of its distribution is pictured in Fig. 2a. We can clearly observe the large impact of ρ, steering how the values will effectively be distributed. Now, let us assume the activation function of such a neuron is a rectifier function, and we are interested in observing what is the probability of the post-activation output being in the linear region: we are interested in measuring $p[Z > 0] = 1 - F_Z(0)$, where $F_Z(x) = p[Z < x]$ is the cumulative distribution function (CDF) for the density $f_Z(z)$. A visual representation of how these values are distributed for different values of ρ is depicted in Fig. 2b.

The behavior of neurons, particularly when employing rectifiers like ReLU, is tightly linked to the learning process, and W becomes more and more (anti-) correlated with X. At $\rho \to 1$, neurons operate linearly, leading to a layer's degeneration (as the current layer becomes a linear combination with the next layer). Conversely, at $\rho \to -1$, neurons become effectively "OFF", leading to insignificance in their contribution. In both cases, there's a *layer degeneration* that we aim to detect to reduce the neural network's depth with a marginal impact on the performance. In the next section, we will draft a metric to estimate how close a layer is to degenerating.

3.2 Entropy for Rectifier Activations

To monitor the output $y_{l,i}^{\boldsymbol{x}}$ of the i-th neuron from a given input $\boldsymbol{x}$ of the dataset $\mathcal{D}$, we define ψ_l as the rectifier of the l-th layer, populated by N_L neurons. Hence, by assuming that $z_{l,i}^{\boldsymbol{x}}$ is the output of the i-th neuron inside the l-th layer, we obtain:

$$y_{l,i}^{\boldsymbol{x}} = \psi_l(z_{l,i}^{\boldsymbol{x}}), \quad (2)$$

Three possible "states" for the neuron can be identified from (2):

$$s_{l,i}^{\boldsymbol{x}} = \begin{cases} +1 \text{ if } y_{l,i}^{\boldsymbol{x}} > 0 \\ -1 \text{ if } y_{l,i}^{\boldsymbol{x}} < 0 \\ 0 \quad \text{if } y_{l,i}^{\boldsymbol{x}} = 0. \end{cases} \quad (3)$$

More precisely, for the output of the i-th neuron, by simply applying the sign function to $z_{l,i}^{\boldsymbol{x}}$, we get $s_{l,i}^{\boldsymbol{x}} = \text{sign}(z_{l,i}^{\boldsymbol{x}})$ and can hence easily pinpoint in which of these states we are. Candidly, the neuron is in the *ON state* when $s_{l,i}^{\boldsymbol{x}} = +1$, as this generally corresponds to the linear region, as opposed to the *OFF state* when $s_{l,i}^{\boldsymbol{x}} = -1$ (considering that $\lim_{x\to-\infty}\psi(x) = 0$).[1] Since it could belong to either the ON or OFF state, the third state $s_{l,i}^{\boldsymbol{x}} = 0$ is a special case, which will not be considered in the following derivation.

The probability (in the frequentist sense) of the i-th neuron belonging to either the ON or the OFF state can be calculated from the average over a batch of outputs for this neuron. More precisely, we define the ON state probability as:

$$p(s_{l,i}{=}{+}1) = \begin{cases} \dfrac{1}{S_{l,i}} \displaystyle\sum_{j=1}^{|\mathcal{D}|} s_{l,i}^{\boldsymbol{x}_j}\Theta(s_{l,i}^{\boldsymbol{x}_j}) & \text{if } S_{l,i} \neq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where

$$S_{l,i} = \sum_{j=1}^{|\mathcal{D}|} s_{l,i}^{\boldsymbol{x}_j}\,\mathrm{sign}(s_{l,i}^{\boldsymbol{x}_j}) \tag{5}$$

is the frequency of the ON and the OFF states encountered, $|\mathcal{D}|$ is the number of input samples, and Θ is the Heaviside function.[2] As explained above, the third state is excluded from this count, as it can be associated with either the ON or OFF state. We can therefore infer that since we are just concerned with the ON or OFF states, when $S_{l,i} \neq 0$, $p(s_{l,i}{=}{-}1) = 1 - p(s_{l,i}{=}{+}1)$. We define as an estimator for *neuron's degeneration* the entropy of the i-th neuron in the l-th layer, calculated as:

$$\mathcal{H}_{l,i} = -\sum_{s_{l,i}=\pm 1} p(s_{l,i}) \log_2\left[p(s_{l,i})\right] \tag{6}$$

Given the definition in (6), $\mathcal{H}_{l,i} = 0$ can be verified in two cases:

- $s_{l,i} = -1\ \forall j$. In this case, $z_{l,i} \leq 0\ \forall j$. The output of the i-th neuron is always 0 when for example employing a ReLU.
- $s_{l,i} = +1\ \forall j$. In this case, $z_{l,i} \geq 0\ \forall j$. As it belongs to the linear region, the output of the i-th neuron is equal to its input (or very close as in GeLU). Therefore, since there is no non-linearity between them anymore, this neuron can in principle be absorbed by the following layer.

Please note that the case $z_{l,i} = 0\ \forall j$, can be associated with both cases, as mentioned previously, and is therefore not taken into account in the previous case disjunction.

[1] Few exceptions to this exist, like LeakyReLU. In those occurrences, even though the activation will not converge to zero, we still choose to call it OFF state as, given the same input's magnitude, the magnitude of the output is lower.

[2] Please be aware that additional sum and average over the entire feature map generated per input are required for convolutional layers.

As an estimator for *layer's degeneration* we can employ the average entropy: for the l-th layer counting N_l neuron it is

$$\widehat{\mathcal{H}}_l = \frac{1}{N_l} \sum_i \mathcal{H}_{l,i}. \tag{7}$$

We would like to have $\widehat{\mathcal{H}}_l = 0$ since we target deep neural networks' depth reduction by eliminating layers with almost zero entropy. In the next section, we will present the whole framework that allows us to practically reduce the network's depth based on the layer degeneration estimator.

Algorithm 1. Our proposed method EASIER.

```
1: function EASIER($\boldsymbol{w}^{\text{INIT}}$, $\mathcal{D}$, $\delta$)
2:     $\boldsymbol{w} \leftarrow$Train($\boldsymbol{w}^{\text{init}}$, $\mathcal{D}_{\text{train}}$)
3:     dense_acc $\leftarrow$Evaluate($\boldsymbol{w}$, $\mathcal{D}_{\text{val}}$)
4:     current_acc $\leftarrow$ dense_acc
5:     while (dense_acc - current_acc) $>\delta$ do
6:         $\widehat{\mathcal{H}} = [\widehat{\mathcal{H}}_1, \widehat{\mathcal{H}}_2, ..., \widehat{\mathcal{H}}_L]$        ▷ Entropy calculation on $\mathcal{D}_{\text{train}}$
7:         $l \leftarrow$ argmin($\widehat{\mathcal{H}}$)        ▷ Finding the lowest-entropy layer
8:         $\psi_l$ = Identity()        ▷ Replacement of the rectifier with an Identity
9:         $\boldsymbol{w} \leftarrow$ Train($\boldsymbol{w}$, $\mathcal{D}_{\text{train}}$)        ▷ Finetune
10:        current_acc $\leftarrow$ Evaluate($\boldsymbol{w}$, $\mathcal{D}_{\text{val}}$)
11:    end while
12:    return $\boldsymbol{w}$
13: end function
```

3.3 EASIER

Depicted in Algorithm 1, we present here our method to remove the lowest-entropy layers. Indeed, the lowest-entropy layer is the one likely to make the least use of the different regions, or states, of the rectifier. Therefore, the need for a rectifier is reduced: the rectifier can be linearized entirely. In this regard, we first train the neural network, represented by its weights at initialization $\boldsymbol{w}^{\text{init}}$, on the training set $\mathcal{D}_{\text{train}}$ (line 2) and evaluate it on the validation set $\mathcal{D}_{\text{val}}$ (line 3). As defined in (7), we then calculate the entropy $\widehat{\mathcal{H}}$ on the training set $\mathcal{D}_{\text{train}}$ for all the L rectifier-activated layers, (therefore, the output layer is excluded) (line 6). We then find the lowest-entropy layer (line 7) and replace its activation with a linear one, i.e., the Identity function (line 8). Evidently, after this step, this layer is not considered anymore. To recover the potential performance loss, the model is then finetuned using the same policy (line 9) and re-evaluated on the validation set $\mathcal{D}_{\text{val}}$ (line 10). The final model is obtained once the performance on the validation set drops below the threshold δ.

4 Experiments

In this section, we empirically evaluate the effectiveness of our proposed approach, across multiple architectures and datasets for traditional image classification setups. We compare our results with EGP [29], an entropy-guided unstructured pruning technique, as well as the Layer Folding method [13].

Table 1. Test performance (top-1) and the number of removed layers (Rem.) for all the considered setups. Dense refers to the original trained model without layer deletion. The best results between LF, EGP, and EASIER are in **bold**.

Dataset	Approach	ResNet-18		Swin-T		MobileNetv2		VGG-16	
		top-1	Rem.	top-1	Rem.	top-1	Rem.	top-1	Rem.
CIFAR-10	Dense	92,47	0/17	91,66	0/12	93,65	0/35	93,50	0/15
	LF	90,65	1/17	85,73	2/12	89,24	9/35	86,46	3/15
	EGP	92,00	3/17	86,04	6/12	92,22	6/35	10,00	1/15
	EASIER	**92,10**	**8/17**	**91,41**	**7/12**	**93,16**	**12/35**	**93,61**	**8/15**
Tiny ImageNet 200	Dense	41,26	0/17	75,78	0/12	46,54	0/35	63,94	0/15
	LF	37,86	4/17	50,54	1/12	25,88	12/35	31,44	6/15
	EGP	39,82	4/17	67,38	3/12	47,52	6/35	—	—
	EASIER	**40,42**	4/17	**68,46**	3/12	**48,80**	**28/35**	**57,60**	**7/15**
PACS	Dense	79,70	0/17	97,30	0/12	95,50	0/35	95,40	0/15
	LF	82,90	3/17	87,70	2/12	79,70	1/35	93,60	3/15
	EGP	81,60	3/17	93,50	4/12	17,70	3/35	—	—
	EASIER	**84,30**	**13/17**	**94,30**	4/12	**94,20**	**8/35**	**95,50**	**4/15**
VLCS	Dense	68,13	0/17	83,04	0/12	81,36	0/35	82,76	0/15
	LF	66,91	5/17	70,92	1/12	68,87	2/35	**80,24**	6/15
	EGP	70,18	4/17	78,47	6/12	45,85	2/35	—	—
	EASIER	**70,27**	**14/17**	**79,12**	6/12	**78,56**	**4/35**	78,84	6/15
Flowers-102	Dense	88,88	0/17	92,70	0/12	88,50	0/35	86,47	0/15
	LF	77,57	5/17	63,07	4/12	2,86	5/35	87,90	3/15
	EGP	82,06	3/17	87,40	3/12	0,34	2/35	—	—
	EASIER	**83,43**	**6/17**	**88,89**	**5/12**	**88,37**	**10/35**	**88,32**	3/15
DTD	Dense	60,53	0/17	67,50	0/12	64,41	0/35	64,20	0/15
	LF	59,99	2/17	37,98	4/12	4,89	5/35	63,56	3/15
	EGP	59,10	2/17	60,21	5/12	2,13	2/35	—	—
	EASIER	**62,02**	**3/17**	**62,23**	5/12	**63,83**	**6/35**	**63,62**	**4/15**
Aircraft	Dense	73,36	0/17	76,39	0/12	73,36	0/35	75,85	0/15
	LF	67,60	2/17	44,76	4/12	4,98	4/35	**70,48**	6/15
	EGP	69,04	2/17	73,27	5/12	0,99	2/35	—	—
	EASIER	**70,33**	2/17	**74,44**	**7/12**	**72,55**	4/35	69,70	6/15

4.1 Experimental Setup

We cover a variety of setups by evaluating our method on four popular models: ResNet-18, MobileNet-V2, Swin-T and VGG-16, trained on seven datasets: CIFAR-10 [25], Tiny-ImageNet [27], PACS and VLCS from DomainBed [16], as well as Flowers-102 [35], DTD [10], and Aircraft [33]. All the hyperparameters, augmentation strategies, and learning policies are provided in Appendix, mainly following [29] and [36]. For ResNet-18, MobileNetv2, and VGG-16 all the ReLU-activated layers are taken into account. For Swin-T, all the GELU-activated layers are considered. Moreover, the threshold δ is established for each dataset and architecture pair to enable a fair comparison with the existing LF and EGP approaches in terms of top-1 performance with a comparable number of removed layers.[3]

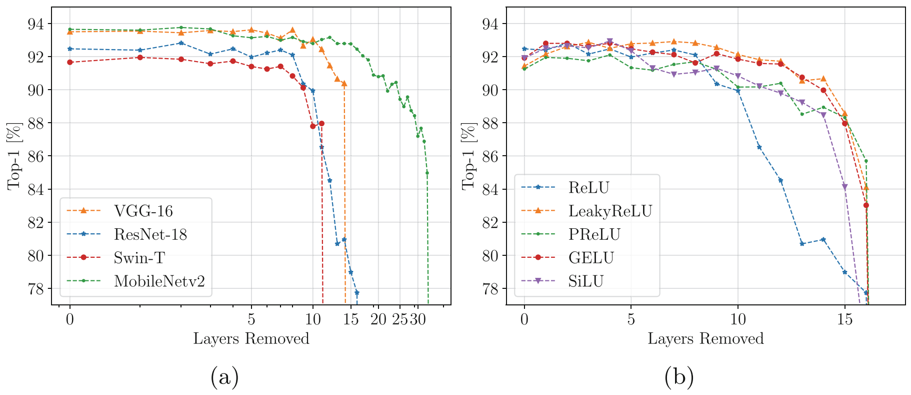


Fig. 3. (a) EASIER applied on ResNet-18, VGG-16, Swin-T and MobileNetv2 networks on CIFAR-10. For each model, we gradually remove non-linear layers. (b) EASIER applied on ResNet-18 on CIFAR-10 with different rectifiers: ReLU, LeakyReLU, PReLU, GELU, and SiLU. Our method is not bound to a specific one and is effective with the most popular.

4.2 Results

A First Overview. We first test our method on a widely known dataset: CIFAR-10. Figure 3a shows the test performance (Top-1) versus the number of removed layers for all the considered models on CIFAR-10, achieved with our method EASIER. Interestingly, all the models exhibit a similar depth-accuracy trend, regardless of their initial depth. Indeed, they first all preserve their original performance, until it drops significantly once ten or so layers have been removed.

Table 1 shows the test performance (top-1) as well as the number of removed layers (Rem.) for all the considered setups. For each combination of dataset and architecture, the performances obtained for each iteration are shown in the tables in the Appendix.

[3] The code and the Appendix are available at https://github.com/VGCQ/EASIER.

Concurrent Method Failure in Some Setups. First, we highlight that the results for EGP on the VGG-16 architecture are not reported apart from CIFAR-10. Indeed, the EGP technique suffers from the layer collapse phenomenon [42]: by forcing a layer to have a zero-entropy, it could force it to be always in the OFF region of its activation, hence preventing the signal from passing through this layer, and therefore leading to a complete failure of the algorithm. This is what is happening on CIFAR-10, where a whole layer is pruned. Since EGP is not working with the VGG architecture on this dataset, we choose not to run the experiments for VGG on other datasets to save computations. Nonetheless, this is not the case with other architectures like ResNet-18, Swin-T, and MobileNetv2, which all have skip connections, leaving another alternative for the signal to pass from the input to the output, in the case the full layer is pruned. However, we also report a problem with transfer learning tasks (Flowers-102, DTD, and Aircraft) for the MobileNetv2 architecture. Indeed, from the first iteration, the EGP's pruning mechanism focuses on the last single layer before the classifier head, leading to its complete removal and hence observing the same layer collapse phenomenon given the absence of a skip/residual connection at this point.

Moreover, even if the results are reported in the table, we underline the failure of Layer Folding for MobileNetv2 in transfer learning setups (Flowers-102, DTD, and Aircraft). The employed auxiliary loss that encourages activations to become linear appears to have a strong effect on the final loss function. The hyperparameter balancing this regularization plays a critical role: a high value prioritizes depth reduction at a cost of performance degradation whereas a small value leads to high performance but with no layers removed. For the mentioned transfer learning task, a trade-off allowing a comparison with EASIER has not been found. This is illustrated by the results obtained on Flowers-102: even with half as many layers removed, LF achieves mediocre performance.

Comparison with Existing Approaches. On most of the considered setups, we can observe the superiority of our method. Indeed, EASIER consistently produces models with better performance for the same number of layers removed, as observed on all the models trained on Tiny-ImageNet-200. For example, while all the methods are able to remove four layers for ResNet-18 on Tiny-ImageNet-200, EASIER achieves respectively 0,6% and 2,56% higher performance than EGP and LF. Moreover, on some setups, EASIER even achieves better performance than the other competitors with more layers removed. This is the case, for example, for all the models trained on CIFAR-10. For instance, for MobileNetv2 on CIFAR-10, EASIER can remove six more layers with 0,94% top-1 gain compared to EGP, which obtains the second-best performance in this setup.

Nevertheless, we highlight the superiority of Layer Folding in two setups: VGG-16 trained on VLCS and Aircraft, in which the models produced by LF achieve better performance for the same number of layers removed, with performance improvements of 1,4% and 0,78% respectively, compared to EASIER.

Comparison with the Original Model. Although on most setups (such as CIFAR-10) it succeeds in compressing models while maintaining performance similar to the original model, EASIER (but also competing methods) is not capable of compressing models without degrading performance. This is the case, for example, with Swin-T on Tiny-ImageNet-200, which displays a 7% loss compared to the original model. The question of a trade-off between performance and compressibility may therefore arise depending on the model's intended use. Nevertheless, apart from VGG-16 on VLCS and Aircraft, our method produces compressed models with the closest performance to the original model compared with existing methods.

4.3 Ablation Study

In this section, we first perform a study over the used rectifier, showing that our method is not bound to a specific one and is effective with any. Figure 3b shows the test performance of ResNet-18 on CIFAR-10, for different rectifiers versus the number of linearized layers. Our method removes at least 8 layers with a performance improvement for GELU, LeakyReLU, and PReLU and with a marginal performance loss for ReLU and SiLU. We hypothesize that it is due to the presence of more signal in backpropagation for GELU, LeakyReLU, and PReLU. Moreover, to find out whether it was necessary (to maintain good performance) to train the network starting from its previous iteration weights (before a layer linearization), a randomly initialized ResNet-18 with the 8 layers selected by EASIER linearized, was re-trained on CIFAR-10 using the same learning policy. The model achieves a top-1 score of 91,56%, down 0,54% on the performance achieved with EASIER. Despite being costly at training time, we concluded that to maximize the performance of the compressed model, it was important to keep training the model from its previous iteration weights.

Table 2. ResNet-18 on CIFAR-10.

Method	Top-1	Rem.
Dense	92,60	0/17
EASIER 2×	92,26	8/17
EASIER 4×	92,45	8/17
EASIER 8×	91,82	8/17

Furthermore, to clear the way for the design of a one-shot approach, we conduct some experiments directly removing several layers at a time, for example by iteratively linearizing the 2, 4, or 8 layers with the lowest entropy. These approaches are denoted respectively EASIER 2×, EASIER 4×, and EASIER 8×. The results for a ResNet-18 trained on CIFAR-10 are presented in Table 2. For fairness, we report the test performance (Top-1) for an equivalent number of layers removed (Rem.). Hence, for EASIER 2× (respectively 4× and 8×), four (respectively two and one) iterations were necessary to obtain these results.

Despite removing the same number of layers, we observe that EASIER 2× and EASIER 4× yield similar results with a slight drop in performance compared to the original model, while EASIER 8× leads to worse performance. With a performance loss of less than one percent compared to the original model, EASIER 8× raises hope for the design of a one-shot approach, which would be more efficient at training time.

Finally, Table 3 showcases the potential savings in terms of inference time and FLOPs for ResNet-18 on CIFAR-10 on six different devices, including CPUs and GPUs spanning from traditional GPU to embedded devices. In general, the fewer layers the network has, the shorter the inference time and the smaller the number of FLOPs. However, we also observe that blindly removing layers is not sufficient to reduce computation. Indeed, a layer removal can result in an increase in MFLOPs, as observed here at the fifth iteration, which is mainly due to the fusion of two convolutional layers, that can result in a layer having a greater size. For instance, to keep the same input/output ratio, two convolutional layers having a kernel size of 3 will fuse in a convolutional layer having a kernel size of 5. Moreover, looking at inference times, every device shows a different trend. While larger devices, like RTX A4500, show a monotonically decreasing inference time, for smaller devices, like P2000 or Jetson Orin, this is not always the case. We also note the same problem on CPUs, like Raspberry Pi 4, where caching is the major problem when dealing with larger kernels.

4.4 Limitations and Future Work

Despite being a successful approach to alleviating deep neural networks' depth, EASIER also presents some limits, which we discuss below.

Training Efficiency. The iterative nature of our method inevitably leads to a longer training time and more intensive computations to achieve the compressed models, compared for instance, to the Layer Folding approach. However, the increased computational cost of training can be offset by the benefits of using these models for inference. Indeed, since a neural network is going to be used multiple times for inference, it is also important to lessen its computational burden related to this use. As opposed to unstructured pruning which offers very few, if any, practical benefits when it comes to deploying the model in a resource-constrained system, our method reduces the critical path forward propagation undergoes, making it useful for processing on parallel systems like GPUs or TPUs, as the computational demands at inference time are reduced.

Nonetheless, even though the method has been thought out iteratively, there is hope for the design of a one-shot approach, which would be more efficient at training time, as shown by the results discussed in the previous ablation study. Another way to address this problem can be to include the entropy in the minimized objective function. However, this approach is not immediately feasible as it is a non-differentiable metric. Therefore, the exploration of differentiable proxies for the layer's entropy is left as future work.

Table 3. Inference time [ms] and MFLOPs of ResNet-18 on CIFAR-10.

Rem.	MFLOPs	Inference on CPU [ms]		Inference on GPU [ms]			
		Xeon E5-2640	Raspi 4	Jetson Orin	P2000	RTX 2080	A4500
0/17	725,47	13,50	135	8,52	4,45	4,43	3,32
1/17	258,24	9,33	111	8,31	4,53	4,43	3,27
2/17	243,46	9,69	106	7,83	4,28	4,21	3,10
3/17	231,79	9,43	139	7,38	4,02	3,93	2,96
4/17	197,85	10,10	117	6,91	3,79	3,68	2,78
5/17	159,05	11,30	144	6,44	3,60	3,46	2,60
6/17	159,99	8,39	225	6,13	4,11	3,18	1,79
7/17	152,36	9,18	144	6,06	4,16	3,10	1,71
8/17	149,84	9,14	149	6,14	3,67	3,21	1,55

Performance Degradation. It is difficult to compress existing parameter-efficient architectures that are not overfitting, and EASIER cannot decrease the depth of an already underfitting architecture without compromising performance, like for example Swin-T on Tiny-ImageNet-200.

Nevertheless, EASIER was able to demonstrate its superiority over existing methods on all the setups considered. Indeed, for the same number of removed layers, EASIER achieves the best performance or can compress more than existing approaches while maintaining performance. We therefore believe that EASIER is a serious candidate to be considered to achieve this kind of goal.

5 Conclusion

In this work, we have presented EASIER, an entropy-based method for layer withdrawal in rectifier-activated deep neural networks. An entropy-based importance metric has been designed to select layers to remove from the network, aiming at depth reduction while preserving high performance in the considered tasks. The capability and effectiveness of reducing the number of layers in a model of EASIER have been demonstrated by experiments conducted on four popular architectures across seven datasets for image classification. Concerned by the ever-growing AI environmental impact, we hope this work can inspire future optimizations and new ways of thinking about network design.

Acknowledgments. This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement 101120237 (ELIAS). Also, this research was partially funded by Hi!PARIS Center on Data Analytics and Artificial Intelligence. This project was provided with computer and storage resources by GENCI at IDRIS thanks to the grant 2023-AD011013930R1 on the supercomputer Jean Zay's the V100 partition.

References

1. Jaafra, Y., Laurent, J.L., Deruyver, A., Naceur, M.S.: Reinforcement learning for neural architecture search: a review. Image Vis. Comput. **89**, 57–66 (2019)
2. Ali Mehmeti-Göpel, C.H., Disselhoff, J.: Nonlinear advantage: trained networks might not be as complex as you think. In: ICML. PMLR (2023)
3. Baymurzina, D., Golikov, E., Burtsev, M.: A review of neural architecture search. Neurocomputing **474**, 82–93 (2022)
4. Blalock, D., Gonzalez Ortiz, J.J., Frankle, J., Guttag, J.: What is the state of neural network pruning? In: MLSys (2020)
5. Bragagnolo, A., Tartaglione, E., Fiandrotti, A., Grangetto, M.: On the role of structured pruning for neural network compression. In: ICIP. IEEE (2021)
6. Brown, T., et al.: Language models are few-shot learners. In: NeurIPS (2020)
7. Cai, H., Gan, C., Wang, T., Zhang, Z., Han, S.: Once-for-all: train one network and specialize it for efficient deployment. In: ICLR (2019)
8. Castillo-Navarro, J., Le Saux, B., Boulch, A., Lefèvre, S.: On auxiliary losses for semi-supervised semantic segmentation. In: ECML PKDD (2020)
9. Cho, M., Joshi, A., Reagen, B., Garg, S., Hegde, C.: Selective network linearization for efficient private inference. In: ICML. PMLR (2022)
10. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: CVPR (2014)
11. Craig, C.C.: On the frequency function of xy. Ann. Math. Stat. **7**, 1–15 (1936)
12. Cui, G., Yu, X., Iommelli, S., Kong, L.: Exact distribution for the product of two correlated Gaussian random variables. IEEE Signal Process. Lett. **23**(11), 1662–1666 (2016)
13. Dror, A.B., Zehngut, N., Raviv, A., Artyomov, E., Vitek, R., Jevnisek, R.: Layer folding: neural network depth reduction using activation linearization. In: BMVC (2022)
14. Faiz, A., et al.: LLMCarbon: modeling the end-to-end carbon footprint of large language models. In: ICLR (2023)
15. Gale, T., Elsen, E., Hooker, S.: The state of sparsity in deep neural networks. arXiv preprint arXiv:1902.09574 (2019)
16. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. In: ICLR (2020)
17. Guo, J., et al.: CMT: convolutional neural networks meet vision transformers. In: CVPR (2022)
18. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: NeurIPS. Curran Associates, Inc. (2015)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
20. He, Y., Xiao, L.: Structured pruning for deep convolutional neural networks: a survey. IEEE Trans. Pattern Anal. Mach. Intell **46**(5), 2900–2919 (2023)
21. Hestness, J., et al.: Deep learning scaling is predictable, empirically. arXiv preprint arXiv:1712.00409 (2017)
22. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
23. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
24. Jha, N.K., Ghodsi, Z., Garg, S., Reagen, B.: DeepReDuce: ReLU reduction for fast private inference. In: ICML. PMLR (2021)

25. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
26. Kuang, W., Zhu, Q., Li, Z.: Multi-label image classification with multi-scale global-local semantic graph network. In: Koutra, D., Plant, C., Gomez Rodriguez, M., Baralis, E., Bonchi, F. (eds.) Machine Learning and Knowledge Discovery in Databases: Research Track, ECML PKDD 2023. LNCS, vol. 14171, pp. 53–69. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43418-1_4
27. Le, Y., Yang, X.: Tiny ImageNet visual recognition challenge. CS 231N **7**, 3 (2015)
28. Lee, N., Ajanthan, T., Torr, P.: SNIP: single-shot network pruning based on connection sensitivity. In: ICLR (2019)
29. Liao, Z., Quétu, V., Nguyen, V.T., Tartaglione, E.: Can unstructured pruning reduce the depth in deep neural networks? In: ICCV (2023)
30. Liu, H., Simonyan, K., Yang, Y.: DARTS: differentiable architecture search. In: ICLR (2018)
31. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: ICCV (2021)
32. Louizos, C., Welling, M., Kingma, D.P.: Learning sparse neural networks through l_0 regularization. In: ICLR (2018)
33. Maji, S., Kannala, J., Rahtu, E., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. Technical report (2013)
34. Mouret, J.B., Clune, J.: Illuminating search spaces by mapping elites. arXiv preprint arXiv:1504.04909 (2015)
35. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Indian Conference on Computer Vision, Graphics and Image Processing, December 2008
36. Quétu, V., Tartaglione, E.: DSD^2: can we dodge sparse double descent and compress the neural network worry-free? In: AAAI (2024)
37. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet classification using binary convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 525–542. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_32
38. Real, E., et al.: Large-scale evolution of image classifiers. In: ICML. PMLR (2017)
39. Seijas-Macías, A., Oliveira, A.: An approach to distribution of the product of two normal variables. Discussiones Mathematicae Probab. Stat. **32**, 87–99 (2012)
40. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
41. Sun, D., Wang, M., Li, A.: A multimodal deep neural network for human breast cancer prognosis prediction by integrating multi-dimensional data. IEEE/ACM Trans. Comput. Biol. Bioinform. (2019)
42. Tanaka, H., Kunin, D., Yamins, D.L., Ganguli, S.: Pruning neural networks without any data by iteratively conserving synaptic flow. In: NeurIPS (2020)
43. Tartaglione, E., Bragagnolo, A., Fiandrotti, A., Grangetto, M.: Loss-based sensitivity regularization: towards deep sparse neural networks. Neural Netw. **146**, 230–237 (2022)
44. Tartaglione, E., Bragagnolo, A., Odierna, F., Fiandrotti, A., Grangetto, M.: SeReNe: sensitivity-based regularization of neurons for structured sparsity in neural networks. IEEE Trans. Neural Netw. Learn. Syst. **33**(12), 7237–7250 (2021)
45. Touvron, H., et al.: LLaMA: open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
46. Wang, R., Cheng, M., Chen, X., Tang, X., Hsieh, C.J.: Rethinking architecture selection in differentiable NAS. In: ICLR (2020)

47. Yang, Y., Li, M., Meng, B., Huang, Z., Ren, J., Sun, D.: Rethinking the misalignment problem in dense object detection. In: Amini, M.R., Canu, S., Fischer, A., Guns, T., Kralj Novak, P., Tsoumakas, G. (eds.) Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2022. LNCS, vol. 13715, pp. 427–442. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-26409-2_26
48. Zhu, M.H., Gupta, S.: To prune, or not to prune: exploring the efficacy of pruning for model compression (2018)
49. Zoph, B., Le, Q.: Neural architecture search with reinforcement learning. In: ICLR (2016)

Towards Few-Shot Self-explaining Graph Neural Networks

Jingyu Peng[1], Qi Liu[1,2], Linan Yue[1], Zaixi Zhang[1], Kai Zhang[1(✉)], and Yunhao Sha[1]

[1] State Key Laboratory of Cognitive Intelligence, University of Science and Technology of China, Hefei, China
{jypeng28,lnyue,zaixi,percy}@mail.ustc.edu.cn, {qiliuql,kkzhang08}@ustc.edu.cn

[2] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China

Abstract. Recent advancements in Graph Neural Networks (GNNs) have spurred an upsurge of research dedicated to enhancing the explainability of GNNs, particularly in critical domains such as medicine. A promising approach is the self-explaining method, which outputs explanations along with predictions. However, existing self-explaining models require a large amount of training data, rendering them unavailable in few-shot scenarios. To address this challenge, in this paper, we propose a **M**eta-learned **S**elf-**E**xplaining GNN (MSE-GNN), a novel framework that generates explanations to support predictions in few-shot settings. MSE-GNN adopts a two-stage self-explaining structure, consisting of an *explainer* and a *predictor*. Specifically, the *explainer* first imitates the attention mechanism of humans to select the explanation subgraph, whereby attention is naturally paid to regions containing important characteristics. Subsequently, the *predictor* mimics the decision-making process, which makes predictions based on the generated explanation. Moreover, with a novel meta-training process and a designed mechanism that exploits task information, MSE-GNN can achieve remarkable performance on new few-shot tasks. Extensive experimental results on four datasets demonstrate that MSE-GNN can achieve superior performance on prediction tasks while generating high-quality explanations compared with existing methods. The code is publicly available at https://github.com/jypeng28/MSE-GNN.

Keywords: Explainability · Graph Neural Network · Meta Learning

1 Introduction

Due to the widespread presence of graph data in diverse domains [48,49], Graph Neural Networks (GNNs) [6,14,36] are attracting increasing attention from the research community. Leveraging the message passing paradigm, GNNs have exhibited remarkable efficacy across multiple scenes, including molecule property prediction [35], social network analysis [2,45], and recommender system

[4]. Despite these successes, a significant drawback of GNN models is their lack of explainability, making it unavailable for humans to understand the basis of predictions. This limitation undermines the complete trust in GNN predictions, consequently restricting their application in high-stake scenarios including medical [50] and finance [24] fields. Furthermore, the European Union has explicitly emphasized the necessity of explainability for trustworthy AI in [28] and any studies focusing on explainability have been conducted on interpretability in other fields [41,43]. Therefore, there is an immediate and pressing need for research into the explainability of GNNs.

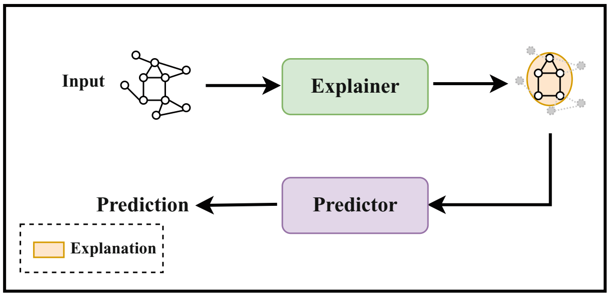


Fig. 1. Paradigm of "*explainer-predictor*" two-stage self-explaining models. The first part is composed of a *explainer* which selects an explanation subgraph for each input graph. The second part is a *predictor* which makes predictions based on the explanation subgraph. Given an input example from Synthetic dataset [39], *explainer* select as explanation, then *predictor* predicts $\hat{y} = house$ based on .

The field of GNN explainability has witnessed substantial scholarly attention [16,17,21,30]. Generally, research on the explainability of GNN can be divided into two main categories: post-hoc explanations and self-explaining methods [40]. Among them, the post-hoc explanation strives to elucidate the predictions made by a trained GNN model. Typically, this is achieved by leveraging another explanatory model to select a subset of input as the explanation for GNN prediction. Despite their utility, these post-hoc explainers often fall short of revealing the actual reasoning process of the model [25] and require optimization for each input graph, which is time-consuming. Therefore, in this paper, we focus on self-explaining methods.

The self-explaining method refers to intrinsically explainable GNN models that offer predictions and explanations concurrently, with the prediction being rooted in the explanation. One prevalent type of self-explaining model typically adopts a "*explainer-predictor*" two-stage paradigm, as illustrated in Fig. 1. This paradigm contains two stages, one is called the *explainer*, which generates an explanation for each input graph, and the other is the *predictor* making predictions based on the generated explanation [17,37].

Although the self-explaining methods in GNN are promising, they still suffer from heavily relying on extensive training data, which restricts their applicability in situations with limited data sizes. For instance, during new drug discovery processes, clinical trials are conducted to assess various drug attributes such as toxicity and side effects. Due to safety concerns, the number of participants in these trials is restricted, resulting in limited experimental data. In such few-shot scenarios, existing self-explaining models fail to achieve satisfactory performance, while existing few-shot learning methods are lack of explainability. Hence, there is a pressing need to design a self-explaining GNN for few-shot scenarios.

Drawing on the fundamental human intelligence traits of rapid learning and self-explainability [7,23,29], we develop **M**eta-learned **S**elf-**E**xplaining GNN (MSE-GNN) for few-shot scenarios:

I. During classification tasks, humans initially concentrate on regions that contain crucial features, and subsequently perform classification based on these features, adhering to a two-stage paradigm [23].
II. When learning new concepts, humans tend to seek representative instances or prototypes and compare new instances with these prototypes to categorize them [29].
III. Humans can learn meta-knowledge from a multitude of tasks, which enables them to achieve impressive performance on new tasks with limited data, which is called *"learn to learn"* [7].

By incorporating these attributes into our MSE-GNN, we aim to solve the explainability of GNNs in few-shot scenarios, and then enhance the performance of both explanation and prediction tasks.

Specifically, the MSE-GNN model follows the two-stage paradigm as depicted in Fig. 1, which naturally mimics the human's two-stage recognition process as mentioned in I. Among them, the *explainer*, which is composed of a GNN encoder and a MLP, predicts the probability of each node being selected as an explanation. Then, node representations encoded by another GNN encoder are separated into explanation and non-explanation based on the prediction of the *explainer*. Subsequently, the *predictor* mimics the decision-making process, which makes predictions based on the explanation with a MLP.

Furthermore, the MSE-GNN model incorporates a novel mechanism that exploits task information to help with selecting explanations and making predictions. Prototype, as stated in II, has been proven to be effective to generate representative representations for each category [31,46]. Therefore, in MSE-GNN, the concept of prototype is utilized in generating task information. The training framework of optimization-based meta-learning imitates the paradigm of "*learning to learn*" in III, where models can acquire meta-knowledge by learning from a vast array of tasks. One of the most popular and effective methods is MAML [7] (Model-Agnostic Meta-Learning). Therefore, we design a new meta-training framework based on MAML to train MSE-GNN.

We conduct extensive experiments on one synthetic dataset [39] and three real datasets of graph classification tasks [11,15], which show excellent performance on both prediction and explanation generated.

2 Problem Definition

In this section, we will elaborate on the problem definition of our research. Following [20], we form the few-shot graph classification problem as N-way K-shot graph classification. Given the dataset $\mathcal{G} = \{(G_1, y_1), (G_2, y_2), ..., (G_n, y_n)\}$, where G_i denotes a graph with a node set V_i and a edge set E_i. n_i denotes the number of nodes in the graph. The structure feature is represented by an adjacency matrix $A_i \in \mathbb{R}^{n_i \times n_i}$. The node attribute matrix is represented as $X_i \in \mathbb{R}^{n_i \times d}$, where d is the dimension of the node attribute.

Then, the dataset is splitted into $\{G^{train}, y^{train}\}$ and $\{G^{test}, y^{test}\}$ as training set and test set respectively according to label y. Where $y^{train} \bigcap y^{test} = \varnothing$. When training, a task $\mathcal{T}$ is sampled each time and each task contains support set $D^{train}_{sup} = (G^{train}_i, y^{train}_i)^s_{i=1}$ and query set $D^{train}_{que} = (G^{train}_i, y^{train}_i)^q_{i=1}$, where s and q stands for the size of support set and query set respectively. It is noteworthy that the same class space is shared in the same task.

In each task, our goal is to optimize our model on the support set D_{sup} and make predictions on the query set D_{que}. If a support set contains N classes and K data for each class, then we name the problem as N-way K-shot. When testing, we firstly finetune the learned model on support set $D^{test}_{sup} = (G^{test}_i, y^{test}_i)^s_{i=1}$ and then report the classification performance of finetuned model on $D^{test}_{que} = (G^{test}_i, y^{test}_i)^q_{i=1}$. Our goal of the few-shot graph classification problem is to develop a model that can obtain meta-knowledge across $\{G^{train}, y^{train}\}$ and predicts labels for graphs in the query set in test stage D^{test}_{que}.

In the explanation generation task, for each graph G_i, a node mask vector $m_i \in [0, 1]^{n_i \times 1}$ is the explanation subgraph selected, a higher value means that the corresponding node is more important for making prediction and vice versa. Although selecting edges for explanation is a viable approach, in this paper we focus on node selection due to its computational complexity.

3 The Proposed MSE-GNN

3.1 Architecture of MSE-GNN

In Fig. 2, we show the overall architecture of the MSE-GNN, which contains three components: an *explainer* g that outputs the explanation selected, a *predictor* p making the final prediction, and a graph encoder f.

Before we present the details of MSE-GNN, we first clarify several concepts. Specifically, existing works often combine self-explaining methods with the concept of rationale [17,37]. The rationale in graph data refers to the subsets of nodes or subsets of edges, which form subgraphs that determine the prediction. Hence, we posit that explanation and rationale are equivalent, as they share the same concept.

In MSE-GNN, the input graph is encoded by f and each node v is encoded into a node embedding $h_{(v)} \in \mathbb{R}^d$, where d stands for the dimension of hidden size. The encoder can be any kind of GNN, e.g. GCN [14], GIN [38], and

GraphSAGE [10]. The selector outputs a mask vector m for each graph as an explanation, which divides the graph into rationale (explanation) G_r and non-rationale G_n. Then the *predictor* makes predictions based on the graph embedding rationale subgraph. Meanwhile, augmented graphs that combine rationale and non-rationale from different graphs are fed into the *predictor* to ensure the robustness of the *predictor*. We categorize the parameters into fast parameters and slow parameters according to the timing of updating, which will be described in detail in Sect. 3.3.

Task Information. MSE-GNN generates task information for the *explainer* and the *predictor* to facilitate explanation generation and prediction within each task, which is composed of prototypes representing each class.

In each task, a support set is provided, which contains data from multiple classes. We aim to extract prototypes from these data that capture the characteristics of each class in the task, in order to help with task-specific selection of explanations and the classification task. Encoded by encoder f, each graph is represented by a matrix containing embedding of each node:

$$H_i = [..., h_{(v)}, ...]^T_{v \in V_i} = f(G_i) \in \mathbb{R}^{|V_i| \times d}. \tag{1}$$

To obtain representation for each graph h_i, the readout function, e.g. mean pooling is employed, to aggregate node embeddings. By leveraging the concept of prototype learning, we further fuse the graph representations of each class with another readout function. Thus, we can obtain a prototype embedding for each class:

$$TI_c = f_{readout}([..., f_{readout}(H_i), ...]_{y_i = c}) \in \mathbb{R}^d. \tag{2}$$

For an N-way K-shot classification problem, the task information is formed by concatenating prototypes of N classes. It is worth noting that, task information for each input graph of both D_{sup} and D_{que} is composed solely of graphs in D_{sup} to prevent label leakage.

Explainer. The *explainer* is responsible for choosing the explanation subgraph corresponding to each input graph. Specifically, given an input graph G_i, the *explainer* firstly uses another GNN encoder to map each node to another node embedding $h'_{(v)}$ for each node in V_i for selection. Then, a MLP is utilized to transform the node embeddings into a soft mask vector $m_i \in [0,1]^{n_i \times 1}$, with task information TI_c and node embedding $h'_{(v)}$ concatenated as input:

$$m_i = \sigma(MLP([..., [h'_{(v)}, TI], ...]^T_{v \in V_i})), \tag{3}$$

where σ denotes the sigmoid function. Hence, we can decompose the input graph G_i into a rationale subgraph and non-rationale subgraph according to m_i respectively:

$$G_i^r = \{A_i, X_i \odot m_i\} \qquad G_i^n = \{A_i, X_i \odot \overline{m_i}\}, \tag{4}$$

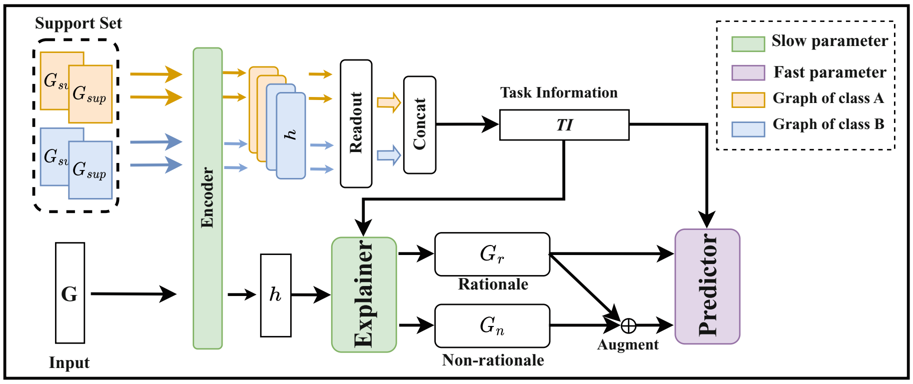


Fig. 2. Overall architecture of MSE-GNN. The model employs a "*explainer-predictor*" 2-stage self-explaining structure. The *explainer* selects explanation subgraphs for each input graph. The *predictor* mimics the decision-making process, which makes predictions solely based on the generated explanation.

where $\overline{m_i} = \mathbf{1} - m_i$. Meanwhile, given the node embedding $h_{(v)}$ from encoder f, we can obtain the graph embedding for G_i^r and G_i^n:

$$h_i^r = f_{readout}(H_i \odot m_i) \qquad h_i^n = f_{readout}(H_i \odot \overline{m_i}). \tag{5}$$

Predictor and Graph Augmentation. The *predictor* takes the graph embedding h as input and makes the final prediction $\hat{y} = p(h)$ with a MLP. Moreover, we enhance the robustness of the *predictor* through graph augmentation. Specifically, within the input graph, the rationale component represents the crucial part that determines the category, while the non-rationale component represents the noisy part. By combining the rationale and non-rationale from different graphs in the same task, additional data with noise are generated. Then we assign the label based on rationale. This approach allows us to increase the amount of noisy data, thereby improving the robustness of the *predictor*. We do the combination operation by adding subgraph embeddings:

$$h_{(i,j)} = h_i^r + h_j^n \qquad y_{(i,j)} = y_i, \tag{6}$$

where h_i^r denotes rationale from G_i and h_j^n means the non-rationale from G_j.

Therefore, in addition to task information TI, the *predictor* p receives the embeddings of both the rationale subgraphs $h_i^r$and the artificially augmented graphs $h_{(i,j)}$ for optimization, and the output are denoted as $\hat{y_i}$ and $y_{(\hat{i,j})}$ respectively.

3.2 Optimization Objective

The optimization objective of MSE-GNN is to achieve both high accuracy in predictions and generate precise explanations, which reveal the underlying reasons behind the predictions. Therefore, we design several types of loss functions and constraints. For the sake of simplicity, we consider a binary classification task without loss of generality.

Algorithm 1. Meta-training of MSE-GNN.

Input: Distribution over meta-training tasks: $p(\mathcal{T})$; Local learning rate: η_1; Global learning rate: η_2; Local update times: T.

Output: Meta-trained parameters for encoder and explanation selector: θ_f, θ_g, and initialization of parameters for *predictor* θ_p

```
1: Initialize $\theta = \{\theta_f, \theta_g, \theta_p\}$ randomly;
2: while not converged do
3:     Sample task $\mathcal{T}$ with support graphs $D_{sup}^{train}$ and query graphs $D_{que}^{train}$.
4:     Set fast adaptation parameters: $\theta'_p = \theta_p$
5:     for t = 0 → T do
6:         Evaluate $\nabla_{\theta_p}\mathcal{L}_{sup}(\theta_f, \theta_g, \theta'_p)$ by calculating loss via Eq. 10
7:         Update $\theta'_p : \theta'_p \leftarrow \theta'_p - \eta_1 \cdot \nabla_{\theta'_p}\mathcal{L}_{sup}(\theta_f, \theta_g, \theta'_p)$
8:     end for
9:     Evaluate $\nabla_{\theta}\mathcal{L}_{que}(\theta_f, \theta_g, \theta'_p)$ by calculating loss via Eq. 10
10:    Update $\theta : \theta \leftarrow \theta - \eta_2 \cdot \nabla_{\theta}\mathcal{L}_{que}(\theta_f, \theta_g, \theta'_p)$
11: end while
```

With the prediction of each rationale graph embedding $p(h_i)$ and corresponding ground-truth label y_i, the loss function is defined as:

$$L_i^r = y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i}). \tag{7}$$

For the artificially augmented graph, our aim is to minimize the prediction values for instances of the same category while maximizing the prediction values for instances of different categories. To achieve this, we employ a contrastive loss function. For example, for a 2-way K-shot classification task, we can obtain $4K^2$ augmented graphs, where each rationale graph is combined with other $2K - 1$ non-rationales, then the loss is computed as:

$$L_i^a = -\frac{1}{2k-1} \sum_{j=1}^{j=2K} 1_{i \neq j} \cdot 1_{y_i = y_j} \log \frac{\exp(\hat{y_i} \cdot \hat{y_j})/\tau}{\sum_{k=1}^{k=K} 1_{i \neq k} \exp(\hat{y_i} \cdot \hat{y_j})/\tau}, \tag{8}$$

where τ is a scalar temperature hyperparameter.

Besides, to address the deviation in the size of rationales, we introduce a penalty based on the number of rationale nodes, the following regularization term is utilized:

$$L^{reg} = |\frac{1_N^\top \cdot m_i}{n_i} - \gamma|, \tag{9}$$

where γ is manually set to control the rationale size. Finally, the total loss function can be formulated as:

$$L = \alpha_r \cdot L^r + \alpha_a \cdot L^a + \alpha_{reg} \cdot L^{reg}, \tag{10}$$

where α_r, α_a, and α_{reg} are hypermeters controlling the weight of each loss.

Table 1. Statistics of four datasets.

	Synthetic	MNIST-sp	Molsider	Moltox21
# Graphs	10,000	70,000	1,427	7,831
Avg # nodes	74.5	75.0	33.6	18.6
Avg # edges	237.8	777.0	70.7	38.6
# Train tasks/classes	5	5	19	7
# Validate tasks/classes	2	2	3	2
# Test tasks/classes	3	3	5	3

3.3 Meta Training

Inspired by the concept of *"learn to learn"* [7], we propose a new meta-training framework based on MAML to obtain meta knowledge from various tasks. We denote θ_f, θ_g, and θ_p as the parameters of encoder, explanation selector, and the *predictor*. Specifically, MSE-GNN is trained from two procedures. One is global update, which aims to learn the parameters of encoder θ_f, explanation generator θ_g, and initialization of the *predictor* θ_p from different tasks, the other is called local update, which performs fast adaption on new tasks and locally update only parameters of the *predictor* θ'_p within each task. According to the timing of updating, we categorize the parameters into fast parameters (θ_p) and slow parameters (θ_f and θ_g), as shown in Fig. 2.

The meta-training process is demonstrated in Algorithm 1. Firstly, we sample a task composed of support D^{train}_{sup} and query data D^{train}_{que} for each episode. Then adaption is operated by updating θ_p for T times on D^{train}_{sup}, where T is a hyperparameter controlling the number of local updates, which is shown in lines 5–8. With updated θ'_p, we utilize the loss on D^{train}_{que} to update θ_f, θ_g and θ_p.

It is important to highlight that, the *explainer* is trained from a variety of tasks and frozen when optimizing each task, which ensures the stability of the explanation selected across different tasks and prevents over-fitting. Therefore, θ_f and θ_g are only updated in the global update and fixed in the local update. While the *predictor* needs to learn the relationship between features and categories in different tasks based on the generated explanations. As a result, the θ_p is optimized in the local update to learn the association between features and categories. Hyperparameters of loss computation in line 6 and line 9 can be differently set according to the goal of local and global optimization.

4 Experiments

4.1 Datasets and Experimental Setup

Dataset. We conduct extensive experiments on four datasets to validate the performance of MSE-GNN: (i) **Synthetic**: Due to the lack of graph datasets with explanation ground-truth, following [39], we create a synthetic dataset for classification, which contains 10 classes and 500 samples for each class. Each graph is composed of two parts: the rationale part and the non-rationale part. The label of each graph is determined by the rationale part. Therefore, the ground-truth of the explanation subgraph is the rationale part of each graph. (ii) **MNIST-sp** [15]: MNIST-sp takes the MNIST images and transforms them into 70,000 superpixel graphs. Each graph consists of 75 nodes and is assigned one of 10 class labels. The subgraphs that represent the digits can be interpreted as ground truth explanations. (iii) **OGBG-Molsider** and **OGBG-Moltox21** [11]: These two datasets are molecule datasets from the graph property prediction task on Open Graph Benchmark (OGBG), they contain 27 and 12 binary labels for each graph, which transformed into 27 and 12 binary classification tasks respectively. The dataset statistics are available in Table 1.

Table 2. 2-way 5-shot Classification Performance with a standard deviation of baseline methods and MSE-GNN.

	Accuracy				AUC-ROC			
	Synthetic		MNIST-sp		OGBG-molsider		OGBG-moltox21	
	GIN	GraphSAGE	GIN	GraphSAGE	GIN	GraphSAGE	GIN	GraphSAGE
ProtoNet	$0.8284_{\pm 0.058}$	$0.8327_{\pm 0.027}$	$0.5736_{\pm 0.008}$	$0.6575_{\pm 0.034}$	$0.5540_{\pm 0.006}$	$0.5468_{\pm 0.006}$	$0.6614_{\pm 0.009}$	$0.6495_{\pm 0.008}$
MAML	$0.8259_{\pm 0.007}$	$0.6409_{\pm 0.327}$	$0.6283_{\pm 0.012}$	$0.6722_{\pm 0.009}$	$0.6219_{\pm 0.005}$	$0.6538_{\pm 0.016}$	$0.7217_{\pm 0.030}$	$0.6965_{\pm 0.014}$
ASMAML	$0.8911_{\pm 0.010}$	$0.7849_{\pm 0.014}$	$0.6526_{\pm 0.004}$	$0.6699_{\pm 0.023}$	$0.6288_{\pm 0.007}$	$\mathbf{0.6818}_{\pm \mathbf{0.008}}$	$0.7432_{\pm 0.030}$	$0.7181_{\pm 0.017}$
GREA_Raw	$0.6970_{\pm 0.005}$	$0.6970_{\pm 0.020}$	$0.6405_{\pm 0.009}$	$0.6667_{\pm 0.009}$	$0.5210_{\pm 0.009}$	$0.5180_{\pm 0.007}$	$0.5654_{\pm 0.015}$	$0.5479_{\pm 0.006}$
CAL_Raw	$0.7248_{\pm 0.006}$	$0.7488_{\pm 0.007}$	$0.6498_{\pm 0.006}$	$0.6670_{\pm 0.010}$	$0.5978_{\pm 0.044}$	$0.6230_{\pm 0.008}$	$0.6161_{\pm 0.064}$	$0.6814_{\pm 0.014}$
GREA_Meta	$0.8728_{\pm 0.013}$	$0.9180_{\pm 0.002}$	$0.6537_{\pm 0.009}$	$0.7430_{\pm 0.008}$	$0.6542_{\pm 0.005}$	$0.6303_{\pm 0.008}$	$0.7650_{\pm 0.004}$	$0.7582_{\pm 0.007}$
CAL_Meta	$0.8451_{\pm 0.021}$	$0.9096_{\pm 0.003}$	$\mathbf{0.6888}_{\pm \mathbf{0.007}}$	$\mathbf{0.7445}_{\pm \mathbf{0.019}}$	$0.6580_{\pm 0.012}$	$0.6553_{\pm 0.018}$	$0.7442_{\pm 0.012}$	$0.7652_{\pm 0.005}$
MSE-GNN	$\mathbf{0.9103}_{\pm \mathbf{0.004}}$	$\mathbf{0.9200}_{\pm \mathbf{0.004}}$	$0.6515_{\pm 0.008}$	$0.7309_{\pm 0.009}$	$\mathbf{0.6673}_{\pm \mathbf{0.007}}$	$0.6587_{\pm 0.002}$	$\mathbf{0.7735}_{\pm \mathbf{0.006}}$	$\mathbf{0.7728}_{\pm \mathbf{0.011}}$

Experimental Setup. To investigate whether generating explanations can help with the classification task, we chose three few-shot learning methods: ProtoNet [29], MAML [7], ASMAML [20]. To compare with existing self-explaining methods, we selected two state-of-the-art self-explaining models: GREA [17] and CAL [30] as baselines to compare the performance of classification and quality of generated explanations. Moreover, for fairness, we adapt meta-training to GREA [17] and CAL [30], enabling them to adapt to few-shot scenarios, which are denoted as GREA_Meta and CAL_Meta respectively.

We use GIN and GraphSAGE as GNN backbones for all methods. The performance of all models is evaluated on D_{que}^{test}. For the Synthetic and MNIST-sp with explanation ground-truth, we use Accuracy to evaluate the classification performance and AUC-ROC to evaluate the quality of the explanation selected. For the two molecule datasets, due to the absence of explanation ground-truth, we only evaluate the classification performance using Area under the ROC curve (AUC) following [17]. For meta-training, we utilize Adam optimizer for local and global updates and set local update times T to 5. Local learning rate η_1 is set to 0.001 and global learning rate η_1 is tuned over {1e-5, 1e-4, 1e-3}. γ in Eq. 9 is tuned over {0.1, 0.2, 0.3, 0.4, 0.5}, number of GNN layers is tuned over {2,3}. We select hyperparameters based on related works and grid searches. All our experiments are conducted with one Tesla V100 GPU.

Table 3. For the Synthetic and MNIST-sp with explanation ground-truth, AUC-ROC is utilized to evaluate the quality of the explanation selected.

		Synthetic	MNIST-sp
GIN	GREA_Raw	$0.4934_{\pm 0.006}$	$0.4789_{\pm 0.044}$
	CAL_Raw	$0.4741_{\pm 0.0250}$	$0.4395_{\pm 0.039}$
	GREA_Meta	$0.6745_{\pm 0.0265}$	$0.7855_{\pm 0.013}$
	CAL_Meta	$0.6201_{\pm 0.0550}$	$0.1707_{\pm 0.0243}$
	MSE-GNN	$\mathbf{0.7000_{\pm 0.006}}$	$\mathbf{0.8222_{\pm 0.030}}$
Graghsage	GREA_Raw	$0.4929_{\pm 0.023}$	$0.5496_{\pm 0.064}$
	CAL_Raw	$0.5080_{\pm 0.054}$	$0.4906_{\pm 0.116}$
	GREA_Meta	$0.7099_{\pm 0.014}$	$0.6513_{\pm 0.040}$
	CAL_Meta	$0.6858_{\pm 0.015}$	$0.6613_{\pm 0.229}$
	MSE-GNN	$\mathbf{0.7189_{\pm 0.012}}$	$\mathbf{0.7077_{\pm 0.038}}$

Performance on Synthetic Graphs and MNIST-sp. To explore whether MSE-GNN can achieve high performance on classification and generate high-quality explanation, we conduct 2-way 5-shot experiments on Synthetic and MNIST-sp datasets which contain ground-truth explanations for each graph. The experimental results are summarized in Table 2 and Table 3. We first compare meta-trained self-explaining baseline models (GREA_Meta, CAL_Meta) with themselves (GREA_Raw, CAL_Raw). We can observe that significant performance boosts are brought by meta-training on both classification and explanation, which indicates that meta-training can leverage the meta-knowledge learned across training tasks effectively on new tasks.

On Synthetic, MSE-GNN shows superiority to other baseline methods on the performance of classification and explanation quality. Compared to meta-trained self-explaining baselines, MSE-GNN performs better on both classification and explanation as MSE-GNN utilizes task information and effectively leverages the augmented graph through the introduction of supervised contrastive loss. Moreover, the inherent denoising capability of self-explaining models contributes to the superior classification performance of MSE-GNN compared to ProtoNet, MAML, and ASMAML.

Unexpectedly, CAL achieves the best classification performance on MNIST-sp, especially when using GIN as the backbone, surpassing MSE-GNN by over 5%. Meanwhile, the quality of explanations is significantly lower compared to GREA and MSE-GNN. By visualization in Fig. 3, which reveals the internal reasoning process of models, we can find that CAL generated explanations that were opposite to our expectations, indicating that CAL infers the digit based on the shape of the background. It is also easy to understand that the digital in a picture can be inferred from the background since the number part and the background part are complementary sets. Therefore, despite the generated explanations being contrary to our expectations, CAL's performance demonstrated that utilizing background information for digit prediction is more effective on MNIST-sp. The reason for CAL generating opposite explanations is that it lacks constraints on the size of the explanation. As a result, it tends to favor subgraphs that contain more useful information and overlook the size of the explanation subgraph. Furtherly comparing the visualization of explanations of MSE-GNN and GREA, we can find that explanations of MSE-GNN are more compact and focus more on the digital part, which is in line with the result in Table 3.

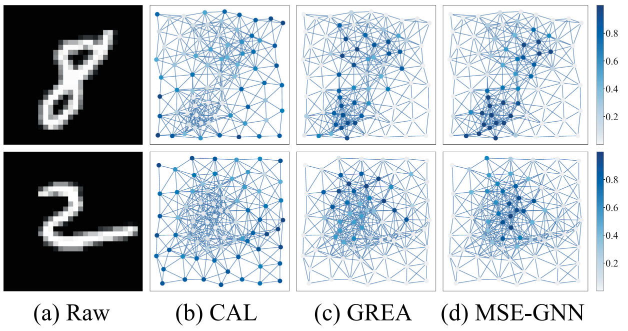
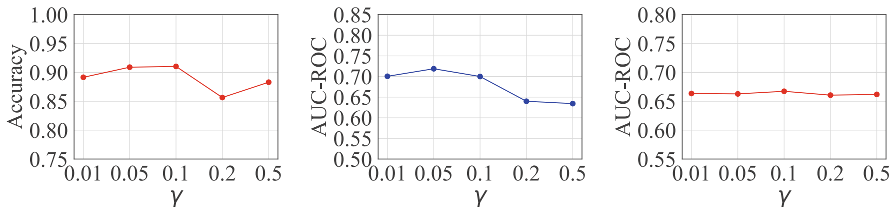


Fig. 3. Raw figure of MNIST-sp and visualization of explanations generated by CAL(a), GREA(b) and MSE-GNN(c). Darker nodes indicate higher importance scores.

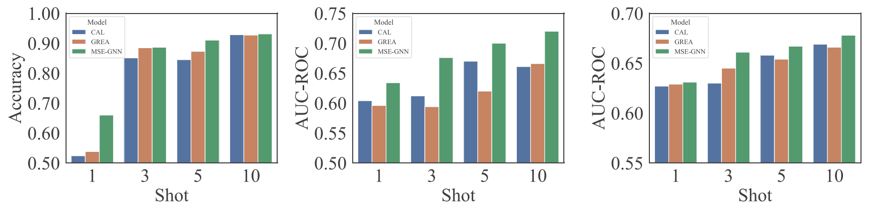


(a) Classification Performance on Synthetic (b) Quality of explanation on Synthetic (c) Classification Performance on OGBG-Molsider

Fig. 4. Classification Performance and quality of explanation selected on Synthetic and OGBG-Molsider with different γ.

Performance on OGBG. MSE-GNN achieves comparable classification performance on these two molecule datasets, demonstrating the effectiveness of its structure. Furthermore, we can observe that the self-explaining models with meta-training outperform all meta-learning models except on OGBG-molsider using GraphSAGE. This is because the process of generating explanations can potentially improve the classification task by eliminating irrelevant noise.

Performance with Different Size of Support Set. Intuitively, for a classification task, the size of the training set has a significant impact on the model's performance. Therefore, in the scenario of few-shot learning, we evaluate the performance of MSE-GNN and other self-explaining models under different support set sizes. Experimental results are shown in Fig. 5. First, comparing different methods, we observe that MSE-GNN consistently outperforms other baselines across different support set sizes, which further validates the performance of MSE-GNN on both classification and explaining. Next, comparing the performance of MSE-GNN across different support set sizes, we observe that as the support set size increases, both the classification accuracy and the quality of generated explanations improve. This also demonstrates the importance of training set size on model performance.



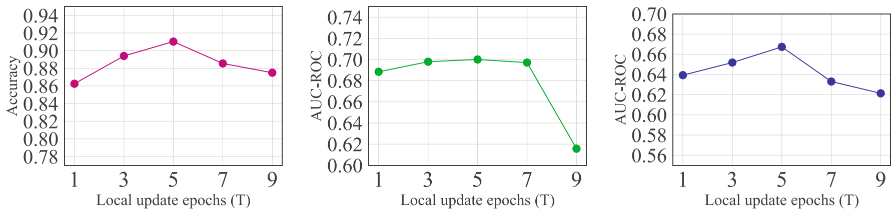
(a) Classification Performance on Synthetic (b) Quality of explanation on Synthetic (c) Classification Performance on OGBG-Molsider

Fig. 5. Classification Performance and quality of explanation selected on Synthetic and OGBG-Molsider with different size of support sets.

Ablation Study. Table 4 demonstrates the impact of contrastive loss and task information utilized in MSE-GNN on Synthetic with GIN. When applying Contrastive Loss (CL), both the classification accuracy and the quality of generated explanations of the model are improved. This indicates that introducing contrastive loss can enhance the model's performance and lead to better results in prediction and explanation tasks. On the other hand, when applying Task Information (TI), the model's performance is also improved across all datasets. This suggests that incorporating task information into the model can provide additional context and guidance, thereby enhancing the model's ability. Moreover, when both CL and TI are used together, the model excels significantly across all datasets, indicating that the combination of CL and TI can synergistically contribute to better performance on both classification and explanation tasks.

Table 4. Impact of contrastive loss and task information.

CL	TI	Synthetic		OGBG-molsider
		Classif.	Explain.	Classif.
		$0.8728_{\pm 0.013}$	$0.6745_{\pm 0.027}$	$0.6542_{\pm 0.005}$
✓		$0.8809_{\pm 0.037}$	$0.6860_{\pm 0.028}$	$0.6623_{\pm 0.011}$
	✓	$0.8800_{\pm 0.011}$	$0.6766_{\pm 0.014}$	$0.6616_{\pm 0.001}$
✓	✓	$\mathbf{0.9103_{\pm 0.004}}$	$\mathbf{0.7000_{\pm 0.006}}$	$\mathbf{0.6673_{\pm 0.007}}$



(a) Classification Performance on Synthetic (b) Quality of explanation on Synthetic (c) Classification Performance on OGBG-Molsider

Fig. 6. Classification Performance and quality of explanation selected on Synthetic and OGBG-Molsider with different T.

Sensitivity Analysis. In MSE-GNN, the parameter γ is crucial in controlling the size of the selected explanation. To examine the sensitivity of the model to different values of γ, we conduct a sensitivity analysis on the Synthetic and OGBG-Molsider datasets with GIN. As illustrated in Fig. 4, the results demonstrate that MSE-GNN achieves the best classification performance when γ is set to 0.1 on both datasets, while the explaining performance achieves best when γ equals 0.05 on Synthetic. We observe that as the value of γ deviates from these two optimal points, the classification performance or the quality of generated explanations decreases. We also notice that the impact of γ is less pronounced on the OGBG-Molsider dataset, indicating that the model is less sensitive to γ on OGBG-Molsider.

Furthermore, T, which stands for the number of local update epochs, affects both the effectiveness and efficiency of the MSE-GNN. We compared the performance of MSE-GNN with different local update epochs on the Synthetic and OGBG-Molsider datasets. The experimental results shown in Fig. 6 indicate that when T is set to 5, MSE-GNN achieves the best classification and explaining performance on both Synthetic and OGBG-molsider. A too-small (too-large) T may result in underfitting (overfitting) of the model for new tasks.

5 Related Works

Few-Shot Learning and Meta Learning on Graph Classification. Few-shot learning aims to learn a model with only a few samples. A promising kind of method is meta learning. Meta learning is also known as "learning to learn", which attempts to learn meta-knowledge from a variety of tasks. There two categories for meta-learning [44]: metric-based models [3,8,22,29,32] and optimization-based models [7,9,20,34,51]. The former focuses on computing the distance between query data and class prototypes [29]. The latter aims to learn an effective initialization of parameters, which enables rapid adaption [7]. [51] firstly applied meta learning framework to the node classification task. [20] utilize a step controller for the robustness and generalization of meta-learner. Notwithstanding the remarkable accuracy improvement achieved by these methods on few-shot learning tasks, their lack of explainability hinders their applicability in certain scenarios such as the medical and finance area.

Explainability in Graph Neural Network. With more attention paid to the applications of GNNs, the explainability of GNNs is more crucial. The explanation increases the models' transparency and enhances practitioners' trust in GNN models by enriching their understanding of why the decision is made by GNNs. Explainability of GNNs can be categorized into two classes [40,42]: post-hoc explanations and self-explainable GNNs. Post-hoc explanations attempt to give explanations for trained GNNs with additional *explainer* model [1,5,12,13,18,19,33,39]. However, these post-hoc explainers often fail to unveil the true reasoning process of the model due to the non-convexity and complexity of the underlying GNN models [25]. Self-explaining GNNs design specific GNN models which are interpretable intrinsically [1,17,21,30,37,50]. They output the prediction and corresponding explanation simultaneously. DIR [37] aims to extract causal rationales that remain consistent across various distributions while eliminating unstable spurious patterns. GREA [17] is another self-explainable model that introduces a new augmentation operation called environment replacement that automatically creates virtual data examples to improve rationale identification. Another category of self-explaining models leverages the concept of prototype learning [1,26,27,47,50]. ProtGNN [50] provides explanations by selecting subgraphs that are the most relevant to graph patterns for identifying graphs of each class. However, existing self-explainable GNNs overlook the scarcity of labeled graph data in many applications. Thus, it's important to build few-shot learning models with self-explainability.

6 Conclusion

In this paper, we proposed MSE-GNN to address the explainability of GNN in few-shot scenarios. To be specific, MSE-GNN adopted a "*explainer-predictor*" 2-stage self-explaining structure and a meta-training framework based on meta-learning, which improved performance in few-shot scenarios. MSE-GNN also introduced a mechanism to leverage task information to assist explanation generation and result prediction. Additionally, MSE-GNN employed graph augmentation to enhance model robustness. Extensive experimental results demonstrated that MSE-GNN achieves strong performance in classification tasks while selecting high-quality explanations in few-shot scenarios.

Acknowledgements. This research was partially supported by supported by Anhui Provincial Natural Science Foundation (No. 2308085QF229), the Fundamental Research Funds for the Central Universities (No. WK2150110034), Technology Innovation Community in Yangtze River Delta (No. 2023CSJZN0200), and the Fundamental Research Funds for the Central Universities.

References

1. Azzolin, S., Longa, A., Barbiero, P., Lio, P., Passerini, A.: Global explainability of GNNs via logic combination of learned concepts. In: The Eleventh International Conference on Learning Representations (2022)
2. Bian, T., et al.: Rumor detection on social media with bi-directional graph convolutional networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 549–556 (2020)
3. Chauhan, J., Nathani, D., Kaul, M.: Few-shot learning on graphs via super-classes based on graph spectral measures. In: International Conference on Learning Representations (2019)
4. Chen, L., Wu, L., Hong, R., Zhang, K., Wang, M.: Revisiting graph based collaborative filtering: a linear residual graph convolutional network approach. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 27–34 (2020)
5. Duval, A., Malliaros, F.D.: GraphSVX: Shapley value explanations for graph neural networks. In: Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., Lozano, J.A. (eds.) Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, 13–17 September 2021, Proceedings, Part II 21, pp. 302–318. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86520-7_19
6. Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking graph neural networks. J. Mach. Learn. Res. **24**, 1–48 (2023)
7. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning, pp. 1126–1135. PMLR (2017)
8. Gao, W., et al.: Leveraging transferable knowledge concept graph embedding for cold-start cognitive diagnosis. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 983–992 (2023)
9. Guo, Z., et al.: Few-shot graph learning for molecular property prediction. In: Proceedings of the Web Conference 2021, pp. 2559–2567 (2021)
10. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
11. Hu, W., et al.: Open graph benchmark: datasets for machine learning on graphs. Adv. Neural. Inf. Process. Syst. **33**, 22118–22133 (2020)

12. Huang, Q., Yamada, M., Tian, Y., Singh, D., Chang, Y.: GraphLIME: local interpretable model explanations for graph neural networks. IEEE Trans. Knowl. Data Eng. **35**(7), 6968–6972 (2022)
13. Kamal, A., Vincent, E., Plantevit, M., Robardet, C.: Improving the quality of rule-based GNN explanations. In: Koprinska, I., et al. (eds.) Joint European Conference on Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2022. CCIS, vol. 1752, pp. 467–482. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-23618-1_31
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2016)
15. Knyazev, B., Taylor, G.W., Amer, M.: Understanding attention and generalization in graph neural networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
16. Lin, W., Lan, H., Wang, H., Li, B.: OrphicX: a causality-inspired latent variable model for interpreting graph neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13729–13738 (2022)
17. Liu, G., Zhao, T., Xu, J., Luo, T., Jiang, M.: Graph rationalization with environment-based augmentations. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1069–1078 (2022)
18. Lucic, A., Ter Hoeve, M.A., Tolomei, G., De Rijke, M., Silvestri, F.: CF-GNNExplainer: counterfactual explanations for graph neural networks. In: International Conference on Artificial Intelligence and Statistics, pp. 4499–4511. PMLR (2022)
19. Luo, D., et al.: Parameterized explainer for graph neural network. Adv. Neural. Inf. Process. Syst. **33**, 19620–19631 (2020)
20. Ma, N., et al.: Adaptive-step graph meta-learner for few-shot graph classification. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 1055–1064 (2020)
21. Müller, P., Faber, L., Martinkus, K., Wattenhofer, R.: DT+GNN: a fully explainable graph neural network using decision trees. arXiv preprint arXiv:2205.13234 (2022)
22. Niu, G., et al.: Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 213–222 (2021)
23. Posner, M.I., Petersen, S.E.: The attention system of the human brain. Annu. Rev. Neurosci. **13**, 25–42 (1990)
24. Pourhabibi, T., Ong, K.L., Kam, B.H., Boo, Y.L.: Fraud detection: a systematic literature review of graph-based anomaly detection approaches. Decis. Support Syst. **133**, 113303 (2020)
25. Rudin, C.: Please stop explaining black box models for high stakes decisions. Stat **1050**, 26 (2018)
26. Seo, S., Kim, S., Park, C.: Interpretable prototype-based graph information bottleneck. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
27. Shin, Y.M., Kim, S.W., Yoon, E.B., Shin, W.Y.: Prototype-based explanations for graph neural networks (student abstract). In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 13047–13048 (2022)
28. Smuha, N.A.: The EU approach to ethics guidelines for trustworthy artificial intelligence. Comput. Law Rev. Int. **20**, 97–106 (2019)
29. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems, vol. 30 (2017)

30. Sui, Y., Wang, X., Wu, J., Lin, M., He, X., Chua, T.S.: Causal attention for interpretable and generalizable graph classification. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1696–1705 (2022)
31. Vuorio, R., Sun, S.H., Hu, H., Lim, J.J.: Multimodal model-agnostic meta-learning via task-aware modulation. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
32. Wang, S., Huang, X., Chen, C., Wu, L., Li, J.: Reform: error-aware few-shot knowledge graph completion. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 1979–1988 (2021)
33. Wang, X., Shen, H.W.: GNNInterpreter: a probabilistic generative model-level explanation for graph neural networks. In: The Eleventh International Conference on Learning Representations (2022)
34. Wang, Y., Abuduweili, A., Yao, Q., Dou, D.: Property-aware relation networks for few-shot molecular property prediction. Adv. Neural. Inf. Process. Syst. **34**, 17441–17454 (2021)
35. Wieder, O., et al.: A compact review of molecular property prediction with graph neural networks. Drug Discov. Today Technol. **37**, 1–12 (2020)
36. Wu, L., Cui, P., Pei, J., Zhao, L., Guo, X.: Graph neural networks: foundation, frontiers and applications. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 4840–4841 (2022)
37. Wu, Y., Wang, X., Zhang, A., He, X., Chua, T.S.: Discovering invariant rationales for graph neural networks. In: International Conference on Learning Representations (2021)
38. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: International Conference on Learning Representations (2018)
39. Ying, Z., Bourgeois, D., You, J., Zitnik, M., Leskovec, J.: GNNExplainer: generating explanations for graph neural networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
40. Yuan, H., Yu, H., Gui, S., Ji, S.: Explainability in graph neural networks: a taxonomic survey. IEEE Trans. Pattern Anal. Mach. Intell. **45**, 5782–5799 (2022)
41. Yue, L., Liu, Q., Du, Y., An, Y., Wang, L., Chen, E.: DARE: disentanglement-augmented rationale extraction. Adv. Neural. Inf. Process. Syst. **35**, 26603–26617 (2022)
42. Yue, L., Liu, Q., Liu, Y., Gao, W., Yao, F., Li, W.: Cooperative classification and rationalization for graph generalization. In: Proceedings of the ACM Web Conference, vol. 2024 (2024)
43. Yue, L., Liu, Q., Wang, L., An, Y., Du, Y., Huang, Z.: Interventional rationalization. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 11404–11418 (2023)
44. Zhang, C., et al.: Few-shot learning on graphs: a survey. In: The 31st International Joint Conference on Artificial Intelligence (IJCAI) (2022)
45. Zhang, K., et al.: EATN: an efficient adaptive transfer network for aspect-level sentiment analysis. IEEE Trans. Knowl. Data Eng. **35**(1), 377–389 (2021)
46. Zhang, K., Zhang, H., Liu, Q., Zhao, H., Zhu, H., Chen, E.: Interactive attention transfer network for cross-domain sentiment classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 5773–5780 (2019)
47. Zhang, K., et al.: Incorporating dynamic semantics into pre-trained language model for aspect-based sentiment analysis. arXiv preprint arXiv:2203.16369 (2022)
48. Zhang, Z., Hu, Q., Yu, Y., Gao, W., Liu, Q.: FedGT: federated node classification with scalable graph transformer. arXiv preprint arXiv:2401.15203 (2024)

49. Zhang, Z., Liu, Q., Hu, Q., Lee, C.K.: Hierarchical graph transformer with adaptive node sampling. Adv. Neural Inf. Process. Syst. **35**, 21171–21183 (2022)
50. Zhang, Z., Liu, Q., Wang, H., Lu, C., Lee, C.: ProtGNN: towards self-explaining graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 9127–9135 (2022)
51. Zhou, F., Cao, C., Zhang, K., Trajcevski, G., Zhong, T., Geng, J.: Meta-GNN: on few-shot node classification in graph meta-learning. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 2357–2360 (2019)

Uplift Modeling Under Limited Supervision

George Panagopoulos[1(✉)], Daniele Malitesta[2], Fragkiskos D. Malliaros[2], and Jun Pang[1]

[1] University of Luxembourg, 6 avenue de la Fonte, Esch-sur-Alzette, Luxembourg
{georgios.panagopoulos,jun.pang}@uni.lu
[2] Université Paris-Saclay, CentraleSupélec, Inria, 3 Rue Joliot Curie, Gif-sur-Yvette, France
{daniele.malitesta,fragkiskos.malliaros}@centralesupelec.fr

Abstract. Estimating causal effects in e-commerce tends to involve costly treatment assignments which can be impractical in large-scale settings. Leveraging machine learning to predict such treatment effects without actual intervention is a standard practice to diminish the risk. However, existing methods for treatment effect prediction tend to rely on training sets of substantial size, which are built from real experiments and are thus inherently risky to create. In this work we propose a graph neural network to diminish the required training set size, relying on graphs that are common in e-commerce data. Specifically, we view the problem as node regression with a restricted number of labeled instances, develop a two-model neural architecture akin to previous causal effect estimators, and test varying message-passing layers for encoding. Furthermore, as an extra step, we combine the model with an acquisition function to guide the creation of the training set in settings with extremely low experimental budget. The framework is flexible since each step can be used separately with other models or treatment policies. The experiments on real large-scale networks indicate a clear advantage of our methodology over the state of the art, which in many cases performs close to random, underlining the need for models that can generalize with limited supervision to reduce experimental risks.

Keywords: Graph Neural Networks · Causal Inference · Active Learning

1 Introduction

Statistical tests for causal inference are ubiquitous, from drug discovery [54] and psychology [62] to social studies and online platforms—being at the core

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_8.

of decision-making. A prevalent example is A/B testing [19], the de facto way to evaluate the potential of a system change before applying it in scenarios such as e-commerce. Typically, the population is split into two random groups, and the treatment is assigned to one of them ($T = 1$). The difference between their response variable Y (e.g., time spent on the system) quantifies the average treatment effect (ATE) of the randomized control trial (RCT) [44], i.e., $Y_T - Y_C = \mathbb{E}[Y|T = 1] - \mathbb{E}[Y|T = 0]$, where T and C refer to the treatment and control groups, respectively. In this setting, however, we risk causing churn on a massive scale if the proposed change is not effective [2]. It is thus more wise to test on a smaller scale and try to extrapolate our findings. In these cases, we can build a model that predicts the outcome without actually intervening, based on the samples' confounders, e.g., for a user, this could be the purchase history and demographics. The core hypothesis is that given the common assumption of unconfoundedness [10], samples with similar confounders will exhibit similar outcomes under the same treatment. Developing a model that can predict the effect of a treatment on unseen samples is referred to as *uplift modeling* (UM) [3, 42], mainly stemming from business context applications. The uplift refers to the prediction of the outcome's change due to the treatment, i.e., the ATE, for a set of samples [12]. The problem is akin to individual treatment effect (ITE) estimation, i.e., predicting the outcome of the same sample with and without treatment (counterfactual) and computing their difference.

In this work, our attention is directed towards a real-world scenario of a marketing campaign where promoting is costly/risky/time-consuming—a typical uplift modeling setting [12, 38]. In this context, we aim to rank the whole user base based on how they will respond to the campaign, i.e., how much more they will consume if they receive a coupon. The coupon resembles a treatment, and the difference in consumption is the effect. We assume there is a fixed budget for the number of users that can participate in the experiment and a predefined random balanced treatment allocation to diminish the risks and costs. The problem can then be broken down into two subproblems:

- Which users should participate in the experiment?
- How to use this experiment to predict the outcome of the rest of the dataset?

We approach the first problem with an active learning formulation [49], where we sequentially choose subsets of the samples to label in order to maximize the model's effectiveness until we reach our budget. The latter problem pertains to the model's capacity to generalize with limited supervision, which can be cast as a semi-supervised learning problem where graph neural networks (GNNs) are rather effective [30]. In general, GNNs are popular in social network applications and can be combined effectively with decision algorithms [39]. The majority of online systems can be represented as heterogeneous graphs, e.g., user-product purchases (e-commerce [60]) or follow relationships (social media [13]). Thus, we frame the problem as an active semi-supervised learning task on a bipartite graph and address it in alternating rounds.

Contributions. Our contributions can be summarized as follows:

1. We develop a novel modular framework, based on two steps, a GNN, and an active learning method, to address the need for limited supervision in UM, moving from the standard "70%–80%" train set rule [11,21] down to 5%–20%.
2. We formulate UM for networks and test it in an open, large-scale network with real experimental annotations. To the best of our knowledge, this is the first such attempt in the literature. Moreover, we focus on continuous outcomes, which are relatively understudied compared to binary UM [58] but equally prevalent in the real world.
3. We conduct experiments with models from the UM and ITE literature, including neural, tree-based, and graph-aware methods, and showcase that the proposed methodology surpasses the state-of-the-art substantially.

2 Related Work

One of the most prevalent applications of machine learning (ML) in causality is estimating the probability of a sample being assigned to a treatment, i.e., the propensity scores using the confounders, which can uncover design bias [32]. Models that predict the propensity score and the outcome can be used in tandem in the double ML framework [6], which is provably doubly robust (DR), in the sense that it is consistent if either the propensity or the outcome predictive model is consistent. DR has been utilized for heterogeneous causal effect estimation [29], which is similar to UM. Simpler models use treatment as an extra input (S-Learner). Two models that learn the output of treatment and control groups (T-Learner) are prevalent [41] and are the basis for the successful causal meta-learners (R/X-Learner) [31]. In addition, meta-learners have been utilized in the context of cost-effective uplift modeling [56].

Class transformation with regularized Logistic Regression [46] or XGBoost [52] is also effective for binary outcomes and under balanced treatment assignments, while random forests were some of the first models developed for causal effect estimation [59]. Causal estimator models that allow for instrumental variables have been developed based on deep neural networks [23]. UpliftTrees [48] have custom splitting criteria based on the outcome distribution in the treatment groups of the tree nodes.

The effect of the network has not been examined in the context of UM, but it has for ITE. One of the first works that proposed an adjustment to the causal effect estimation based on the network was Arbour et al. [1]. Veitch et al. [57] developed a neural method to identify hidden confounders in interconnected samples. They extend the doubly robustness for non-iid data and build a model that relies on random-walk-based node representations. The same team proposed Dragonnet [51], a neural architecture that is split into predicting the outcomes under each treatment and the propensity score.

Guo et al. [20] developed an ITE model with GNN encoding in the initial layers and split output layers for each treatment. Furthermore, it minimizes the distance between the representations' distribution under different treatments.

This stems from a seminal work on bounding the generalization error of ITE models by the sum of the given model's standard error and the distance between the treatment and control confounders' distributions [50]. Similar GNNs have been developed based on multi-task and adversarial learning [7,26], the geometric curvature of the network [14], or a network with multiple relations [33]. It should be noted that predicting the causal estimates under network confounding and the network interference [9,37] differs conceptually, although the models are similar. An effort to predict ITE considering both the network confounding and the interference was made using hypergraph neural networks [36]. We refer the reader to Appendix A, where we make a distinction between confounding and interference and justify why we follow the literature [7,14,57] on the assumption for the latter.

One important difference between UM and ITE is that the latter defines distinctly the counterfactual prediction and requires the respective ground truth for evaluation, which renders the use of simulated data imperative [37]. Hence, the aforementioned works on network-based ITE are evaluated with simulated experiments on observational data and hardcoded network confounding. Our work uses, for the first time, an open network with ground-truth experimental variables. Furthermore, we address a network with heterogeneous nodes, a setting prevalent in e-commerce. HINITE [33], which uses one type of node but multiple relations, is in a similar heterogeneous direction but not comparable.

3 Proposed Methodology

We consider the e-commerce scenario where data is commonly structured through bipartite, undirected graphs e.g. user-product. Given the effectiveness of GNNs in semi-supervised learning [22], we create a general framework that can be tested with several GNN layers. The model can be used as is, but we additionally define an active learning method to build the training set iteratively based on the model's uncertainty and the samples' properties. In the following, the scalars are represented by a lowercase letter, the vectors or sets with an upper case, and the matrices with an upper case bold letter.

3.1 Uplift Modeling with Graph Neural Networks (UMGNet)

Let $u \in U$ and $p \in P$ be a user and a product, respectively, having $|U| = n$ users and $|P| = m$ products in total. Then, let $\mathbf{R} \in \mathbb{R}^{n \times m}$ be the user-product interaction matrix, where $R_{up} = 1$ if there is a recorded interaction between user u and product p, 0 otherwise. Moreover, let each user u be represented by a tuple $Z_u = (\mathbf{X}_u, T_u, Y_u)$ where $\mathbf{X}_U \in \mathbb{R}^{n \times d}$ are the covariates representing features of dimensionality d from all n users, while $Y_U \in \mathbb{R}^n$ are the continuous trial outcomes, and $T_U \in \{0, 1\}^n$ is the treatment assignment.

Based on the Neyman-Rubin framework of potential outcomes [45] and using the *do* operator introduced by Pearl [40], we define the average treatment effect for the population as:

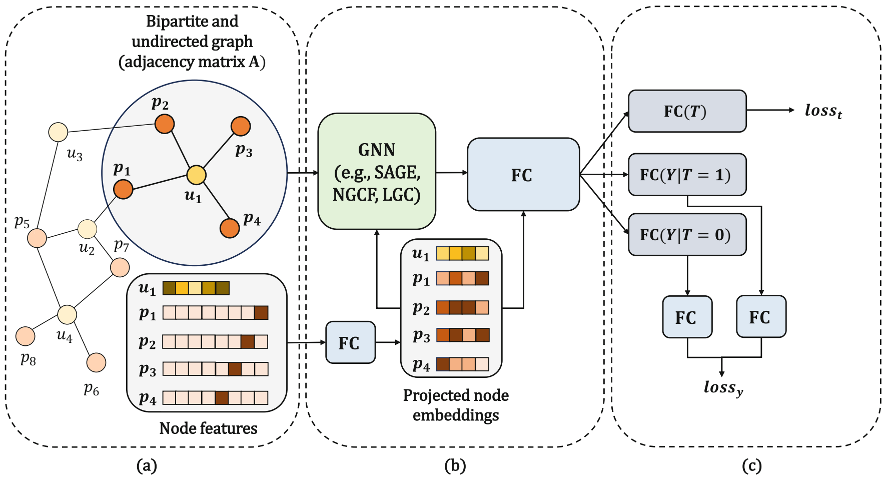


Fig. 1. Schematic representation of UMGNET. First **(a)**, the bipartite and undirected user-product graph, along with the node features, are injected into the framework. Second **(b)**, node features for users and products are projected to the same latent space through an FC layer and used as input to the GNN model; another FC layer takes the GNN's output and input to predict the regression outcome. Third **(c)**, outputs for $T = 1$ and $T = 0$ are considered separately and injected into two different FC layers to calculate $loss_y$; the general output of the regression FC is used to calculate $loss_t$.

$$ATE = \mathbb{E}[Y_U|do(T_U = 1)] - \mathbb{E}[Y_U|do(T_U = 0)]. \tag{1}$$

If we consider the law of total expectation and the unconfoundedness [10], i.e., that the confounders $\mathbf{X}$ block all possible ways that the outcome depends on treatment assignment, we can contend that the observed change is an unbiased causal effect from the treatment. Taking into consideration once again the user-product interaction matrix $\mathbf{R}$ as an extra confounder, we have:

$$ATE = \mathbb{E}[Y_U|\mathbf{X}_U, \mathbf{R}, T_U = 1] - \mathbb{E}[Y_U|\mathbf{X}_U, \mathbf{R}, T_U = 0]. \tag{2}$$

Training one model to predict $\mathbb{E}[Y_U|\mathbf{X}_U, \mathbf{R}, T_U]$, e.g., the S-LEARNER [31], has a specific disadvantage: we expect that Y and $\mathbf{X}$ differ between $T = 0$ and $T = 1$. To this end, the two-model methods have exhibited improved performance [31], especially in scenarios with imbalanced assignments. However, the two model approaches do not facilitate sharing the obvious knowledge overlaps. As mentioned in Sect. 2, this can be alleviated by developing a common neural architecture for the first layers [27], including treatment prediction [51] and random walk-based confounders [57]. That motivates us to adapt DRAGONNET [51] for bipartite graphs with GNN encoding.

We define the user-product bipartite and undirected graph as $G = (U, P, \mathbf{A})$, where we already introduced U and P, while $\mathbf{A} \in \mathbb{R}^{(n+m)\times(n+m)}$ is the adjacency matrix of G:

$$\mathbf{A} = \begin{pmatrix} 0 & \mathbf{R} \\ \mathbf{R}^\top & 0 \end{pmatrix}. \tag{3}$$

The users' features are represented as $\mathbf{X}_U \in \mathbb{R}^{n \times d}$, and the product features as $\mathbf{X}_P \in \mathbb{R}^{m \times m}$. First, we project users and products to the same latent space through a fully connected (FC) layer. The obtained projections are horizontally concatenated to get a common set of node embeddings $\mathbf{X} \in \mathbb{R}^{(n+m) \times w}$, as follows:

$$\mathbf{X} = \text{concat}\big([\text{ReLU}(\mathbf{X}_U \mathbf{W}_U), \text{ReLU}(\mathbf{X}_P \mathbf{W}_P)]\big), \tag{4}$$

where $\mathbf{W}_U \in \mathbb{R}^{d \times w}$ and $\mathbf{W}_P \in \mathbb{R}^{m \times w}$.

Subsequently, we learn graph features leveraging GNN layers. For this part, we have examined different architectures, including GraphSAGE [22], NGCF [60], and LGC [24], so we are going to keep it general in terms of formulation:

$$\mathbf{H}_1 = \text{GNN}(\mathbf{A}, \mathbf{X}). \tag{5}$$

The representations are then broken into two separate paths for $T = 1$ and $T = 0$ denoted as t and c, respectively, through another FC layer. We utilize a residual connection from $\mathbf{X}$, arriving for $T = 1$ in:

$$\mathbf{H}^t_{i+1} = \text{ReLU}\big(\big[\mathbf{X}[0:n], \mathbf{H}_1[0:n]\big]\mathbf{W}^t_i\big), \tag{6}$$

where we are slicing the matrices to keep the first n rows that correspond to the user representations, assuming the horizontal concatenation in Eq. (4). Two FC layers (with F as depth) predict the outcome under treatment t and control c:

$$\hat{Y}^c = \mathbf{H}^c_F \mathbf{W}^c_F, \qquad \hat{Y}^t = \mathbf{H}^t_F \mathbf{W}^t_F.$$

The loss takes into consideration only the factual outcomes, i.e., where the treatment vector $\mathbf{T}$ has 1 for the t output and 0 for the c:

$$loss_y = \frac{1}{n} \sum_{i=0}^{n} \big(T_i(\hat{Y}^t_i - Y_i)^2 + (1 - T_i)(\hat{Y}^c_i - Y_i)^2\big). \tag{7}$$

We refer to this generic GNN-based uplift modeling framework as UMGNet. A schematic representation of the model is shown in Fig. 1. As mentioned above, we have examined various instances of the model with different GNN architectures. Lastly, inspired by [51], we consider a variant denoted by UMGNet-Dr, in which we add an extra output layer that predicts the treatment. For this model, we add the following loss term to the one in Eq. (7):

$$loss_t = \frac{1}{n} \sum_{i=0}^{n} \text{CrossEntropy}\big(T, \text{Sigmoid}(\mathbf{H}^\top_F \mathbf{W}^\top_F)\big). \tag{8}$$

3.2 Active Learning for Uplift GNNs (UMGNET-AL)

Active learning relies on the structure of the data and the uncertainty of the model to build iteratively the train set in scenarios where data labeling is costly [49]. We can break it down into two parts: the first is the uncertainty estimation and the acquisition function over non-labeled samples, while the second is the active learning policy we use to gather new samples to label, i.e., to test in our case.

Uncertainty Estimation. Uncertainty estimation in graph learning is an active research topic [25,53]. The uncertainty can be distinguished based on its source: the model (epistemic) or the data (aleatoric). We will employ an unprincipled yet effective practice to quantify epistemic uncertainty by measuring the variance on the responses of an ensemble of models or simply performing dropout multiple times [18]. The dropout mask is random during inference, so the model will produce different outputs for each test sample, arriving at a Bayesian approximation [15] of the uncertainty.

Aleatoric uncertainty can be measured using the structure of the graph and the feature distance. Acquisition functions based on diverse criteria, including both types of uncertainties, have proven more effective in node-level tasks [17,63]. We adopt a similar approach and define $D_u, \forall u \in U$ to be the degree, Q_u is the model's uncertainty, and b is the budget for new training samples in each iteration. According to the literature [17,63], we define diversity based on feature clustering. Specifically, we cluster the samples with a k-means algorithm in a predefined number of clusters and store the assignments in $C \in \mathbb{Z}^n$. Then we calculate the distance $M_u \in \mathbb{R}^n$ between each sample and its cluster centroid and compute the budget for each cluster based on its relative size and b, in $C_b \in \mathbb{Z}^{|C|}$. We can cast the problem of choosing a batch to add to the training set as ranking based on U, D, and M, with the first constraint being on the batch size. The second constraint, which represents diversity, pertains to how many samples of a given cluster should be included in the train set in each round. Finally, we want to keep the train set balanced in terms of treatment and control samples, hence the third constraint:

$$\begin{aligned}
\text{maximize} \quad & O = \sum_{u=1}^{N} x_u (Q_u + D_u + M_u) \\
\text{subject to :} & \sum_{u=1}^{n} x_u \leq b \\
& \sum_{u=1}^{n} x_u C_u \leq C_b[C_u] \\
& \sum_{u=1}^{n} x_u T_u \leq \frac{b}{2} \\
& x_u \in \{0,1\}, \quad \text{for } u = 1, 2, \ldots, n.
\end{aligned}$$

Algorithm 1. UMGNet AL

```
Input: $O, D, M, b,$ UMGNet
  $S \leftarrow \{\text{argmax}_{s \subset U} O(D, M, T, b)\}$
  while $|S| \leq 5 * b$ do
      Train UMGNet($S$)
      $Q, \hat{Y} \leftarrow$ UMGNet($U \setminus S$)
      $S \leftarrow S \cup \{\text{argmax}_{s \subset U} O(Q, D, M, T, b)\}$
  end while
Output: $\hat{Y}$
```

The problem can be solved greedily in each iteration of batch selection, while D, C, and C_b are precomputed. In practice, we weigh each of the three criteria in the objective function with a coefficient in $[0, 1]$ based on the results of validation.

Acquisition Policy. Thompson sampling [47], and upper-confidence bandits [16] are some of the most popular policies to perform active learning [49]. The lack of knowledge at the initial iterations justifies the use of stochastic policies. However, greedy selection has been effective in batch active learning, and it is actually provably near-optimal in some cases [61]. Since our acquisition function does not rely solely on the model's output, which is less trustworthy due to the initial lack of samples, we are going to use a greedy policy and solve the ranking problem in every iteration. The whole procedure can be seen in Algorithm 1 for 5 iterations, assuming UMGNet includes by default the parameters $\mathbf{X}, T, Y, \mathbf{A}$ for clarity.

4 Experimental Evaluation

We report on the experimental results to empirically justify the soundness of our framework. First, we present the main datasets and how we built them. Second, we outline the benchmarking models adopted for this work. Finally, we discuss the obtained results and answer different questions with an ablation study. The code to reproduce the analysis is in GitHub[1].

4.1 Datasets

RetailHero. The *RetailHero* [43] dataset is comprised of two equal groups of anonymized users undergoing a marketing campaign, along with their product purchase history and some relevant features. The treatment group has received promotional SMS texts and the binary outcome corresponds to whether the user has made a purchase or not after the promotional SMS. We follow suit from the literature and the original competition and utilize features such as age, sex, coupon issue time, coupon redeem time, and delay between issue and redeem

[1] https://github.com/geopanag/UMGNet.

time. The products are represented by one-hot encoding. We filter the data for erroneous samples e.g., when delay time is negative, or users are less than 16 years old. We created a user-product graph based on the purchases performed before the time that the promotional coupon was redeemed from treated users. Note that issue and redeem timestamps are added by the organizers for untreated users as well, and they follow similar properties with the redeem time for treated users. Similar to a real-life experiment, purchases done after the intervention are not accessible in the initial form of the graph, except for training samples, i.e., users that have indeed taken part in the experiment in real life. We set two types of continuous outcomes Y:

- *RHC*: The difference between the average money spent before and after coupon redeem time. $ATE = 2.60, \bar{Y} = 266, \sigma(Y) = 521$.
- *RHP*: The average money spent after coupon redeem time. $ATE = 1.95, \bar{Y} = 423, \sigma(Y) = 387$.

The second outcome measures how much more prone treated users are to spend money compared to the control. The first is similar but takes into account the spending of each user before the treatment as a means of normalization.

MovieLens. We utilize the *Movielens25* and filter the users based on the minimum number of ratings. We extract the features of the movie nodes using a Universal Sentence Encoder-Lite [4] on the concatenation of the title, year, and genres and bring them down to 16 dimensions using principal component analysis. The adjacency is defined as in Eq. 3 from the movie-viewer bipartite network. We define the treatment as $T = 1$ if the movie's average rating is over the median of 3.15 or $T = 0$ otherwise, akin to [33,36]. Similar to the same literature, the regression outcome is simulated 5 times: $y_s = \text{ReLU}(w_s x + w_t t + e_s)$, where $w_s \in \mathbb{U}(10, 20)$ and $e_s \in\sim \mathcal{N}(10, 5^2)$ are random variables of the simulation (Table 1).

Table 1. Statistics of the bipartite datasets (directed).

Dataset	Nodes	E (before T)	E (after T)	$T = 0$	$T = 1$
RetailHero	(180,653+40,542)	2,522,096	12,021,243	90,097	90,556
MovieLens	(59,047+162,541)	9,369,966	15,630,129	29,518	29,529

4.2 Benchmark Models

To facilitate comparison with the state-of-the-art, we utilize a variety of methodologies from meta learners S, T, X, R [31] and doubly robust DR [29] to uplift trees UPLIFTTREE [42], and neural models CEVAE [34], DRAGONNET [51], and CFR [50]. We rely on the implementations from the CausalML package [5] for all, using XGBoost as the output and the build-in *elastic net* as the propensity model whenever required. We include NetDeconf [20] as the graph-aware benchmark from ITE literature but remove the Wasserstein distance cost because it fails to run on a GPU with 32GB. All methods are run with their suggested parameters.

4.3 Experiments

In contrast to the aforementioned studies in ITE [20,37], we can not measure the counterfactual error in the sample level because it does not exist. Moreover, since our task is not binary, we can not utilize the uplift curve [12]. Even if we could, our application focuses on the potential of the top ranked users in order to target them with the campaign, hence it is not sensible to focus on the estimation of the whole dataset. Instead, we rely on the realistic evaluation of our scenario that was also the success criterion for our main dataset[2]:
up@40/20: We take the top 40% and 20% of the test samples sorted based on their predicted uplifts, and measure the real ATE in this set [43].

Settings. To evaluate the models' capacity in semi-supervised generalization, we utilize 5 and 20-fold cross-validation where the test part is set for training and vice versa, i.e., we will have 20% and 5% training set size and 5 and 20 iterations, respectively. We run each method 5 times with random seeds from 0 to 4 and log the average and standard deviation of the respective metric in k-fold validation. For all datasets, we set the learning rate to 0.01 with a weight decay of $1e-4$. We used ReLU as an activation function. The dropout rate is set to 0.4, the number of epochs to 2000, and the hidden layers to (64, 64, 32). Finally, the number of clusters is set to 50 and the coefficients for the acquisition function are 0.2, 0.1, and 0.7, respectively.

In the active learning setting, denoted as UMGNet-AL, which utilizes the UMGNet-SAGE model with the proposed acquisition function, instead of 20 and 5-fold, we train an initial model in 1% and 4% of the dataset and increase the train set up to 5%/20% respectively with 5 batch queries using a greedy policy. The experiments are performed with an Nvidia V100 16 GB and 32 GB RAM. The results for *RetailHero* can be seen in Tables 2, 3. The best result is indicated in **boldface**, and the second-best is underlined.

Results and Discussion. It can be seen that the proposed methods outperform the benchmarks for both outcome variables. Some benchmarks even exhibit negative uplifts, which means the predicted sets have higher Y_c than Y_t. The regular *ATE* is the $\bar{Y}_t - \bar{Y}_c$ of the whole dataset, and it is what we expect to get by a random balanced subset of users without any ranking. If we consider it as a baseline, we see that in most cases, the benchmarks tend to be under the *ATE*. This is justified by the reduced supervision, which severely diminishes the

[2] https://ods.ai/competitions/x5-retailhero-uplift-modeling.

benchmarks' generalization, moving their results closer to a random sample. In contrast, the proposed methods achieve consistently and sometimes considerably higher uplift than the *ATE*, signifying their ability to generalize in this setting. We actually see that the difference between the proposed methods and the benchmarks is close to or sometimes larger than the actual *ATE*.

Table 2. Uplift metrics of the predicted sets in the *RHC*. Regular $ATE = 2.60$.

Model	20% training size		5% training size	
	up@40	up@20	up@40	up@20
S-XGB	-1.63 ± 1.85	-0.67 ± 3.6	1.13 ± 1.09	2.43 ± 1.79
T-XGB	0.58 ± 1.48	3.25 ± 2.72	-0.05 ± 0.32	1.33 ± 1.51
X-XGB	-2.55 ± 1.97	-2.62 ± 1.47	0.21 ± 1.02	1.19 ± 1.64
R-XGB	1.77 ± 0.97	3.70 ± 1.62	2.01 ± 0.53	5.2 ± 1.81
DR-XGB	-0.26 ± 1.04	1.18 ± 1.60	0.61 ± 1.00	2.65 ± 1.05
UpliftTree	$\underline{5.95 \pm 2.43}$	2.38 ± 7.00	$\underline{3.74 \pm 0.77}$	2.73 ± 1.83
CFR	0.76 ± 1.36	-2.09 ± 1.6	1.73 ± 0.8	1.41 ± 1.5
CEVAE	3.10 ± 0.59	3.23 ± 1.63	2.88 ± 0.75	3.34 ± 0.70
DragonNet	2.24 ± 0.14	3.08 ± 0.34	2.26 ± 0.04	3.01 ± 0.10
NetDeconf	1.89 ± 1.86	2.73 ± 1.39	1.08 ± 0.46	2.06 ± 0.62
UMGNet-SAGE	3.20 ± 0.25	$\mathbf{6.48 \pm 0.70}$	2.66 ± 0.75	$\underline{5.69 \pm 1.01}$
UMGNet-AL	$\mathbf{6.27 \pm 3.00}$	$\underline{4.64 \pm 3.60}$	$\mathbf{5.83 \pm 2.75}$	$\mathbf{6.83 \pm 3.77}$

To be more specific, UMGNet-SAGE is overall more effective in settings with 20% training size, and UMGNet-AL produces the strongest average uplift when the training size is limited to 5%, which is sensible due to active learning choosing the most informative training set. UpliftTree has the second best performance in *RHC* albeit with the highest standard deviation (7). Furthermore, an effective model should exhibit higher **up@20** than **up@40**, meaning the top 20% predictions should have higher real uplift than the top 40% if the predictive ranking is consistent. Our methods clearly follow this pattern in 15 out of 16 comparisons, in contrast to UpliftTree in *RHC*. The propensity-based methods, i.e., DR-XGB, DragonNet, R-XGB, are not as accurate because the treatment assignment in the dataset is balanced. Thus, the potential bias is minimized, and accordingly, the effect of the propensity score is diminished.

Throughout the experiments, the standard deviation grows with the average value, with exceptions such as UMGNet-SAGE in *RHC* and T-XGB in *RHP*. This makes sense given the range of values and their std. It should be noted that although both tasks are challenging, *RHC* has significantly greater std (521 to 387) with a lower average (266 to 423), and it is arguably more informative. *RHC* normalizes the effect of the treatment with the user's normal behavior, while *RHP* ranks based on absolute purchase average, which can be biased on the user's preferences.

Finally, the five realizations of the semi-simulated *MovieLens* dataset are used to compare UMGNet-SAGE with the best benchmarks from the first experiment. The average results are shown in Fig. 2, where it is visible that it consistently outperforms them. In this case, however, the benchmarks perform better compared to *RetailHero* if we consider the *ATE*, possibly due to the outcome values being smaller and the input features of the movies being considerably more informative.

Table 3. Uplift metrics of the predicted sets in the *RHP*. Regular $ATE = 1.95$.

Model	20% training size		5% training size	
	up@40	up@20	up@40	up@20
S-XGB	3.15 ± 2.07	4.54 ± 1.10	2.48 ± 0.93	3.56 ± 0.96
T-XGB	2.67 ± 0.26	$\underline{5.65 \pm 1.16}$	2.53 ± 0.39	4.65 ± 0.65
X-XGB	2.96 ± 1.01	5.33 ± 1.57	2.29 ± 0.35	4.04 ± 1.06
R-XGB	2.66 ± 1.10	4.58 ± 0.88	2.90 ± 0.32	4.11 ± 0.67
DR-XGB	3.23 ± 1.22	3.86 ± 1.38	2.77 ± 0.19	3.61 ± 1.00
UpliftTree	2.84 ± 0.75	5.01 ± 0.45	1.77 ± 0.70	3.26 ± 0.86
CFR	1 ± 1.31	0.25 ± 1.69	1.19 ± 0.55	1.39 ± 0.81
CEVAE	2.37 ± 1.67	3.26 ± 1.89	1.89 ± 0.54	2.10 ± 0.54
Dragonnet	0.21 ± 0.16	-0.01 ± 0.26	0.21 ± 0.01	-0.104 ± 0.01
NetDeconf	0.84 ± 1.28	2.00 ± 1.00	0.45 ± 0.63	0.770 ± 1.20
UMGNet-SAGE	$\mathbf{5.01 \pm 2.16}$	$\mathbf{7.28 \pm 4.34}$	$\underline{3.99 \pm 2.00}$	$\underline{5.07 \pm 3.61}$
UMGNet-AL	$\underline{4.11 \pm 2.59}$	4.69 ± 2.88	$\mathbf{5.89 \pm 2.48}$	$\mathbf{6.04 \pm 2.46}$

Ablation Study. We perform a number of extra experiments to answer certain questions of interest.

- *Does the GNN help?* Dragonnet [51] is one of the most popular neural architectures for *ITE*, though it has not been extensively utilized for uplift modeling or in semi-supervised settings. Although they are trained with different parameters, e.g., our architecture has fewer layers, UMGNet-Dr resembles a version of Dragonnet with bipartite SAGE encoding. In Table 4, we see that adding information from the network produces significantly better results. The lack of supervision is detrimental for a deep neural network like Dragonnet, as is prevalent in RHP.
- *Does the GNN layer impact the prediction?* We compare the results with GNNs like SAGE [22], NGCF [60], and LGC [24]. These GNN models are used to derive different instances of the UMGNet framework. The interested reader may refer to Appendix B for a detailed presentation of each GNN layer. Note that the layers also differ, i.e., SAGE has only 1 layer, but NGCF has

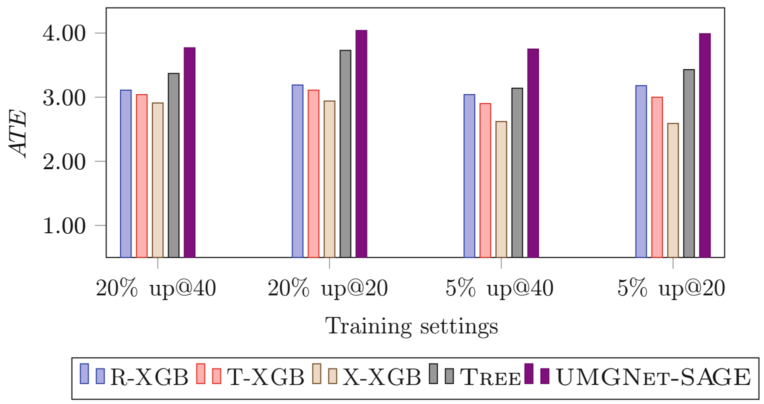


Fig. 2. Uplift of the predicted sets on the *MovieLens* dataset. Regular $ATE = 0.457$.

Table 4. The effect of GNN compared to vanilla DRAGONNET.

Dataset	*Model*	20% training size		5% training size	
		up@40	up@20	up@40	up@20
RHC	DRAGONNET	2.24 ± 0.14	3.08 ± 0.34	2.26 ± 0.04	3.01 ± 0.10
	UMGNET-DR	2.41 ± 1.27	5.20 ± 1.64	3.00 ± 0.48	5.70 ± 0.50
Improvement (%)		+7.6%	+68.8%	+32.7%	+89.4%
RHP	DRAGONNET	0.21 ± 0.16	-0.01 ± 0.26	0.21 ± 0.01	-0.10 ± 0.01
	UMGNET-DR	4.19 ± 1.33	6.15 ± 2.42	3.47 ± 0.82	4.33 ± 0.90
Improvement (%)		+1895.2%	+2700%	+1552.4%	+4430%

3. The results can be seen in Table 5, where it is clear that SAGE overall performs better, but NGCF is equally effective in *RHC*.

- *Is active learning helpful?* In Table 6 we are utilizing an e-greedy policy, that uses random batches with probability $\epsilon = 0.5$, noted as UMGNET-EG, as a baseline to clarify the need for the acquisition function. We see a clear improvement in each metric if we optimize the acquisition function in each step.

5 Conclusion

The creation of a large enough training set to use for uplift modeling can be a costly, time-consuming, or risky task. It is thus important to develop methodologies that select the right samples to intervene on and extrapolate efficiently on the rest of the dataset. We propose a two-step modular methodology that addresses these needs. The main problem is formulated as semi-supervised uplift modeling, and we solve it using bipartite graph neural networks. Additionally,

Table 5. The effect of different GNN layers in UMGNET.

Dataset	UMGNET	20% training size		5% training size	
		up@40	up@20	up@40	up@20
RHC	NGCF	$\mathbf{5.34 \pm 0.86}$	5.49 ± 1.15	$\mathbf{3.71 \pm 0.61}$	3.74 ± 0.97
	LGC	4.17 ± 1.55	3.96 ± 2.70	3.70 ± 0.73	3.72 ± 0.87
	SAGE	3.20 ± 0.25	$\mathbf{6.48 \pm 0.70}$	2.66 ± 0.75	$\mathbf{5.69 \pm 1.01}$
RHP	NGCF	3.53 ± 2.30	3.06 ± 2.51	1.97 ± 0.24	2.22 ± 0.23
	LGC	3.50 ± 2.00	2.89 ± 2.25	3.31 ± 0.52	3.30 ± 0.77
	SAGE	$\mathbf{5.01 \pm 2.16}$	$\mathbf{7.28 \pm 4.34}$	$\mathbf{3.99 \pm 2.00}$	$\mathbf{5.07 \pm 3.61}$

Table 6. The effect of different active learning policies in UMGNET.

Dataset	UMGNET	20% training size		5% training size	
		up@40	up@20	up@40	up@20
RHC	EG	6.01 ± 4.33	3.94 ± 3.18	4.12 ± 5.00	4.18 ± 3.00
	AL	6.27 ± 3.00	4.64 ± 3.60	5.83 ± 2.75	6.83 ± 3.77
Improvement (%)		+4.3%	+17.8%	+41.5%	+63.4%
RHP	EG	1.91 ± 2.40	3.59 ± 4.51	3.91 ± 3.20	5.66 ± 3.45
	AL	4.11 ± 2.59	4.69 ± 2.88	5.89 ± 2.48	6.04 ± 2.46
Improvement (%)		+115.2%	+30.6%	+50.6%	+6.7%

a batch active learning method is defined based on the model's uncertainty, structural importance, and feature diversity to build the training set.

We utilize, for the first time, a large-scale graph with ground-truth experimental information to test our hypothesis. The proposed methodology is compared to a breadth of benchmarks from both uplift modeling and individual treatment effect literature. Our results indicate a clear advantage of the proposed methodology compared to the benchmarks, which sometimes perform near random in semi-supervised settings. Moreover, active learning enhances the results as the supervision diminishes. It is important to note that each step can be utilized separately e.g., the acquisition function can be used with other models. This framework aspires to be an initial step towards addressing the realistic yet overlooked problem of uplift modeling under budget for experimental interventions.

Regarding future work, we plan to examine the theoretical aspects of the method. Specifically, we aim to understand better the tradeoff between the number of treated nodes and the generalization capability of the model. Moreover, as the main application revolves around social networks, it is vital to analyze the model's fairness in terms of treatment allocation or outcome prediction. Finally, we plan to experiment with other available semi-synthetic datasets of varying sizes to research the model's robustness and scalability.

Acknowledgements. Supported in part by ANR (French National Research Agency) under the JCJC project GraphIA (ANR-20-CE23-0009-01).

Ethics. This study only involved public datasets that are freely available for academic purposes. There are no obvious ethical considerations regarding negative impacts from the broader application of the method.

References

1. Arbour, D., Garant, D., Jensen, D.: Inferring network effects from observational data. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 715–724 (2016)
2. Bakshy, E., Eckles, D., Yan, R., Rosenn, I.: Social influence in social advertising: evidence from field experiments. In: Proceedings of the 13th ACM Conference on Electronic Commerce, pp. 146–161 (2012)
3. Betlei, A., Diemert, E., Amini, M.R.: Uplift modeling with generalization guarantees. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 55–65 (2021)
4. Cer, D., et al.: Universal sentence encoder (2018). arXiv preprint arXiv:1803.11175
5. Chen, H., Harinen, T., Lee, J.Y., Yung, M., Zhao, Z.: CausalML: Python package for causal machine learning (2020). arXiv preprint arXiv:2002.11631
6. Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W.: Double/Debiased/Neyman machine learning of treatment effects. Am. Econ. Rev. **107**(5), 261–265 (2017)
7. Chu, Z., Rathbun, S.L., Li, S.: Graph infomax adversarial learning for treatment effect estimation with networked observational data. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 176–184 (2021)
8. Cortez, M., Eichhorn, M., Yu, C.: Staggered rollout designs enable causal inference under interference without network knowledge. Adv. Neural. Inf. Process. Syst. **35**, 7437–7449 (2022)
9. Cristali, I., Veitch, V.: Using embeddings for causal estimation of peer influence in social networks. Adv. Neural. Inf. Process. Syst. **35**, 15616–15628 (2022)
10. Dawid, A.P.: Conditional independence in statistical theory. J. R. Stat. Soc. Ser. B Stat Methodol. **41**(1), 1–15 (1979)
11. Devriendt, F., Moldovan, D., Verbeke, W.: A literature survey and experimental evaluation of the state-of-the-art in uplift modeling: a stepping stone toward the development of prescriptive analytics. Big Data **6**(1), 13–41 (2018)
12. Diemert, E., Betlei, A., Renaudin, C., Amini, M.R.: A large scale benchmark for uplift modeling. In: Proceedings of the KDD Workshop on Artificial Intelligence for Computational Advertising (2018)
13. Fan, W., et al.: Graph neural networks for social recommendation. In: Proceeding of the 28th ACM Web Conference, pp. 417–426. ACM (2019)
14. Farzam, A., Tannenbaum, A., Sapiro, G.: Curvature and causal inference in network data. In: Causal Representation Learning Workshop at NeurIPS 2023 (2023)
15. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: representing model uncertainty in deep learning. In: International Conference on Machine Learning, pp. 1050–1059. PMLR (2016)

16. Garivier, A., Moulines, E.: On upper-confidence bound policies for switching bandit problems. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS (LNAI), vol. 6925, pp. 174–188. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24412-4_16
17. Gilhuber, S., Busch, J., Rotthues, D., Frey, C.M.M., Seidl, T.: DiffusAL: coupling active learning with graph diffusion for label-efficient node classification. In: Koutra, D., Plant, C., Gomez Rodriguez, M., Baralis, E., Bonchi, F. (eds.) Machine Learning and Knowledge Discovery in Databases: Research Track. ECML PKDD 2023. LNCS(), vol. 14169. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43412-9_5
18. Graff, D.E., Shakhnovich, E.I., Coley, C.W.: Accelerating high-throughput virtual screening through molecular pool-based active learning. Chem. Sci. **12**(22), 7866–7881 (2021)
19. Gui, H., Xu, Y., Bhasin, A., Han, J.: Network a/b testing: From sampling to estimation. In: Proceedings of the 24th International Conference on World Wide Web, pp. 399–409 (2015)
20. Guo, R., Li, J., Liu, H.: Learning individual causal effects from networked observational data. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 232–240 (2020)
21. Gutierrez, P., Gérardy, J.Y.: Causal inference and uplift modelling: a review of the literature. In: Proceedings of the 4th International Conference on Predictive Applications and APIs, pp. 1–13. PMLR (2017)
22. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
23. Hartford, J., Lewis, G., Leyton-Brown, K., Taddy, M.: Deep IV: a flexible approach for counterfactual prediction. In: International Conference on Machine Learning, pp. 1414–1423. PMLR (2017)
24. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: LightGCN: simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 639–648. ACM (2020)
25. Huang, K., Jin, Y., Candes, E., Leskovec, J.: Uncertainty quantification over graph with conformalized graph neural networks. In: Advances in Neural Information Processing Systems, vol. 36 (2023)
26. Jiang, S., Sun, Y.: Estimating causal effects on networked observational data via representation learning. In: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, pp. 852–861 (2022)
27. Johansson, F., Shalit, U., Sontag, D.: Learning representations for counterfactual inference. In: Proceedings of the 33rdh International Conference on Machine Learning, pp. 3020–3029. PMLR (2016)
28. Karrer, B., et al.: Network experimentation at scale. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 3106–3116 (2021)
29. Kennedy, E.H.: Towards optimal doubly robust estimation of heterogeneous causal effects. Electron. J. Stat. **17**(2), 3008–3049 (2023)
30. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks (2016). arXiv preprint arXiv:1609.02907
31. Künzel, S.R., Sekhon, J.S., Bickel, P.J., Yu, B.: Metalearners for estimating heterogeneous treatment effects using machine learning. Proc. Natl. Acad. Sci. **116**(10), 4156–4165 (2019)

32. Lee, B.K., Lessler, J., Stuart, E.A.: Improving propensity score weighting using machine learning. Stat. Med. **29**(3), 337–346 (2010)
33. Lin, X., Zhang, G., Lu, X., Bao, H., Takeuchi, K., Kashima, H.: Estimating treatment effects under heterogeneous interference. In: Koutra, D., Plant, C., Gomez Rodriguez, M., Baralis, E., Bonchi, F. (eds.) Machine Learning and Knowledge Discovery in Databases: Research Track. ECML PKDD 2023. LNCS(), vol. 14169. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43412-9_34
34. Louizos, C., Shalit, U., Mooij, J.M., Sontag, D., Zemel, R., Welling, M.: Causal effect inference with deep latent-variable models. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
35. Ma, J., Guo, R., Chen, C., Zhang, A., Li, J.: Deconfounding with networked observational data in a dynamic environment. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, pp. 166–174 (2021)
36. Ma, J., Wan, M., Yang, L., Li, J., Hecht, B., Teevan, J.: Learning causal effects on hypergraphs. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1202–1212 (2022)
37. Ma, Y., Tresp, V.: Causal inference under networked interference and intervention policy enhancement. In: International Conference on Artificial Intelligence and Statistics, pp. 3700–3708. PMLR (2021)
38. Olaya, D., Verbeke, W., Van Belle, J., Guerry, M.A.: To do or not to do: cost-sensitive causal decision-making. Eur. J. Oper. Res. **305**(2), 838–852 (2023)
39. Panagopoulos, G., Tziortziotis, N., Vazirgiannis, M., Malliaros, F.: Maximizing influence with graph neural networks. In: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining, pp. 237–244 (2023)
40. Pearl, J.: Causality. Cambridge University Press, Cambridge (2009)
41. Radcliffe, N.: Using control groups to target on predicted lift: building and assessing uplift model. Dir. Mark. Anal. J. 14–21 (2007)
42. Radcliffe, N.J., Surry, P.D.: Real-world uplift modelling with significance-based uplift trees. White Paper TR-2011-1, Stochastic Solutions, pp. 1–33 (2011)
43. Rafla, M., Voisine, N., Crémilleux, B.: Evaluation of Uplift Models with Non-Random Assignment Bias. In: Bouadi, T., Fromont, E., Hüllermeier, E. (eds.) IDA 2022. LNCS, vol. 13205, pp. 251–263. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-01333-1_20
44. Rubin, D.B.: Estimating causal effects of treatments in randomized and nonrandomized studies. J. Educ. Psychol. **66**(5), 688 (1974)
45. Rubin, D.B.: Causal inference using potential outcomes: design, modeling, decisions. J. Am. Stat. Assoc. **100**(469), 322–331 (2005)
46. Rudaś, K., Jaroszewicz, S.: Regularization for uplift regression. In: Koutra, D., Plant, C., Gomez Rodriguez, M., Baralis, E., Bonchi, F. (eds.) Machine Learning and Knowledge Discovery in Databases: Research Track. ECML PKDD 2023. LNCS(), vol. 14169. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43412-9_35
47. Russo, D.J., Van Roy, B., Kazerouni, A., Osband, I., Wen, Z., et al.: A tutorial on Thompson sampling. Found. Trends® Mach. Learn. **11**(1), 1–96 (2018)
48. Rzepakowski, P., Jaroszewicz, S.: Decision trees for uplift modeling with single and multiple treatments. Knowl. Inf. Syst. **32**, 303–327 (2012)
49. Settles, B.: Active learning literature survey (2009)
50. Shalit, U., Johansson, F.D., Sontag, D.: Estimating individual treatment effect: generalization bounds and algorithms. In: Proceedings of the 34th International Conference on Machine Learning, pp. 3076–3085. PMLR (2017)

51. Shi, C., Blei, D., Veitch, V.: Adapting neural networks for the estimation of treatment effects. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
52. Sołtys, M., Jaroszewicz, S.: Boosting algorithms for uplift modeling (2018). arXiv preprint arXiv:1807.07909
53. Stadler, M., Charpentier, B., Geisler, S., Zügner, D., Günnemann, S.: Graph posterior network: Bayesian predictive uncertainty for node classification. Adv. Neural. Inf. Process. Syst. **34**, 18033–18048 (2021)
54. Tye, H.: Application of statistical 'design of experiments' methods in drug discovery. Drug Discov. Today **9**(11), 485–491 (2004)
55. Ugander, J., Karrer, B., Backstrom, L., Kleinberg, J.: Graph cluster randomization: network exposure to multiple universes. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 329–337 (2013)
56. Vanderschueren, T., Verbeke, W., Moraes, F., Proença, H.M.: Metalearners for ranking treatment effects (2024). arXiv preprint arXiv:2405.02183
57. Veitch, V., Wang, Y., Blei, D.: Using embeddings to correct for unobserved confounding in networks. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
58. Verhelst, T., Petit, R., Verbeke, W., Bontempi, G.: Uplift vs. predictive modeling: a theoretical analysis (2023). arXiv preprint arXiv:2309.12036
59. Wager, S., Athey, S.: Estimation and inference of heterogeneous treatment effects using random forests. J. Am. Stat. Assoc. **113**(523), 1228–1242 (2018)
60. Wang, X., He, X., Wang, M., Feng, F., Chua, T.: Neural graph collaborative filtering. In: SIGIR, pp. 165–174. ACM (2019)
61. Wei, K., Iyer, R., Bilmes, J.: Submodularity in data subset selection and active learning. In: Proceedings of the 32nd International Conference on Machine Learning, pp. 1954–1963. PMLR (2015)
62. Wright, D.B.: Comparing groups in a before-after design: when t test and ANCOVA produce different results. Br. J. Educ. Psychol. **76**(3), 663–675 (2006)
63. Wu, Y., Xu, Y., Singh, A., Yang, Y., Dubrawski, A.: Active learning for graph neural networks via node feature propagation (2019). arXiv preprint arXiv:1910.07567

Self-supervised Spatial-Temporal Normality Learning for Time Series Anomaly Detection

Yutong Chen[1,2], Hongzuo Xu[3], Guansong Pang[4](✉), Hezhe Qiao[4], Yuan Zhou[3], and Mingsheng Shang[1,2](✉)

[1] Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China
{chenyutong,msshang}@cigit.ac.cn
[2] Chongqing School, University of Chinese Academy of Sciences, Chongqing 400714, China
[3] Intelligent Game and Decision Lab (IGDL), Beijing 100091, China
[4] Singapore Management University, Singapore 178902, Singapore
gspang@smu.edu.sg, hezheqiao.2022@phdcs.smu.edu.sg

Abstract. Time Series Anomaly Detection (TSAD) finds widespread applications across various domains such as financial markets, industrial production, and healthcare. Its primary objective is to learn the normal patterns of time series data, thereby identifying deviations in test samples. Most existing TSAD methods focus on modeling data from the temporal dimension, while ignoring the semantic information in the spatial dimension. To address this issue, we introduce a novel approach, called Spatial-Temporal Normality learning (**STEN**). STEN is composed of a sequence Order prediction-based Temporal Normality learning (OTN) module that captures the temporal correlations within sequences, and a Distance prediction-based Spatial Normality learning (DSN) module that learns the relative spatial relations between sequences in a feature space. By synthesizing these two modules, STEN learns expressive spatial-temporal representations for the normal patterns hidden in the time series data. Extensive experiments on five popular TSAD benchmarks show that STEN substantially outperforms state-of-the-art competing methods. Our code is available at https://github.com/mala-lab/STEN.

Keywords: Anomaly Detection · Time Series · Self-supervised Learning · Normality Learning

1 Introduction

Time series data are pervasive in many real-world application domains, including finance, industrial production, network traffic, and health monitoring [2,25,26].

Y. Chen and H. Xu—Equal contribution.

Within these time series data, there can exist exceptional data observations, a.k.a., anomalies, that deviate significantly from the majority of data, which often indicates an abnormal change of the data-generating mechanism [10]. These anomalies are of great interest to the analysts since accurately detecting them is significant in alarming faults in target systems and preventing potential losses.

Anomaly detection aims to learn data normality during training and identify those exceptional data during inference [23]. It is a non-trivial task to precisely learn the normality of time series data. The challenge is primarily due to the unsupervised nature of anomaly detection. Models like neural networks are actuated by supervisory signals to learn high-level patterns from given training data. However, it is prohibitively costly to collect large-scale labeled data for anomaly detection. Thus, learning from unlabeled data is generally more preferable yet it is difficult due to the lack of labeled training data. The challenge is aggravated by the inherent complexity of time series data. Time series data is characterized by multiple components from temporal and spatial perspectives, including the trending and seasonal changes along the time dimension and the spatial proximity relation of temporal sequences in the feature space.

Current time series anomaly detection (TSAD) methods typically leverage an encoder-decoder architecture and assess data abnormality according to reconstruction/prediction errors of testing data [7,12,26,30,33–35]. Despite their general effectiveness on various datasets, these methods often tend to overfit the training data and fail to distinguish anomalies from the normal sequences, since both types of sequences have similarly small reconstruction/prediction errors. Self-supervised learning has been emerging as a promising technique in anomaly detection due to its capability of deriving supervisory signals from the data itself. Existing self-supervised methods, e.g., via pretext tasks like association learning [34] and dual attention contrastive representation learning [35], successfully learn discriminative models to capture normal semantics of the temporal continuity into the representation space. However, apart from the data regularities reflected in the temporal dimension, spatial normality is also a desideratum of comprehensive normality learning in time series data.

In light of this limitation, this paper investigates an intriguing question: *Can we simultaneously learn spatial-temporal normality of time series data in the self-supervised paradigm?* As shown in Fig. 1, the input data sequences are the training data of the Epilepsy dataset [33], which illustrates the normal patterns. Time series continuously change along the time dimension, and this continuity can be fully leveraged in temporal normality learning. More specifically, after splitting the raw time sequences into several sub-sequences, the normality can be modeled by learning the sub-sequence order. During inference, giving a testing sequence with anomalies (Fig. 1(a)), the learning model ranks the randomly shuffled sub-sequences and finally yields a sequence that resembles the training data (Fig. 1(b)). This predicted order depicts a large difference compared to the original distribution for the anomalous sequence, whereas the difference is small for normal sequences.

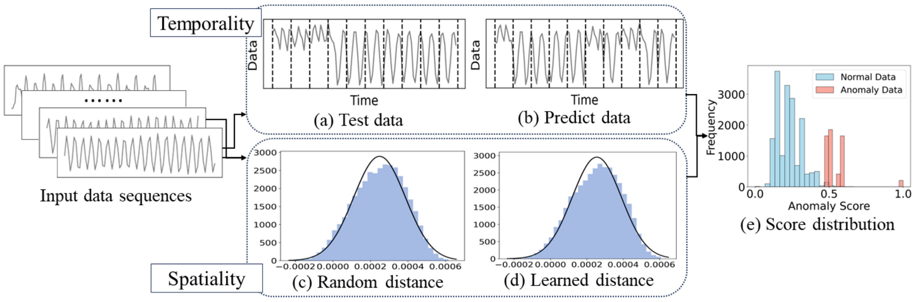


Fig. 1. Our key insights. (a) A sequence containing abnormal data is divided into equal-length sub-sequences by dashed lines, with anomalies presented in the first three sub-sequences. (b) Sub-sequences arranged according to the predicted order distribution, showing differences between the predicted order distribution and the original data due to the presence of the anomaly. (c) The distribution of distances between sequence pairs in a random projection space, which can well preserve the spatial information of the sequences within the feature space. (d) The distribution of distances between sequence pairs learned by a trainable network, effectively resembling the distance distribution. (e) The distribution of anomaly scores obtained by considering the prediction discrepancies in both temporal and spatial dimensions for normal and abnormal data. The results are based on the Epilepsy dataset [33].

In terms of spatial normality, we investigate the pairwise distance of time series data sequences in a projection space. Generally, it is challenging to obtain the spatial relation of data samples evolving along the time, especially under the unsupervised learning paradigm. Motivated by the effective spatial information preservation by random projection [4,16,29], we leverage the pairwise distance in a randomly projected feature space. Since normal sequences often locate in a dense region, as shown in Fig. 1(c), the distance distribution of a normal sequence typically follows a Gaussian-like distribution with the expected value approximating zero. On the contrary, that of anomalous sequences tends to have a markedly different distance-based spatial distribution (e.g., uniform-like distribution). Neural networks are capable of predicting those distance-based spatial information, as shown in Fig. 1(d). By learning to accurately predict these distances, the networks learn the spatial distribution of the sequence normality. During inference, testing anomalous sequences manifest significant deviation from the normal data distribution according to the anomaly score (see Fig. 1(e)).

Based on this insight, we propose a novel approach for TSAD, called Spatial-Temporal Normality learning (**STEN** for short), in which two pretext tasks are designed to construct a self-supervised learning pipeline for the joint spatial-temporal normality modeling. Specifically, we first devise an Order prediction-based Temporal Normality learning module (OTN). The inputting time sequences are split into several sub-sequences. Supervised by their genuine order, the neural network learns how to sort shuffled sub-sequences, during which the temporal continuity can be captured and represented. On the other hand, we further introduce a Distance prediction-based Spatial Normality learning module (DSN), which models the spatial proximity of sequences in the representation space.

The main contributions are summarized as follows.

– We introduce the concept of spatial-temporal normality of time series data and propose a novel anomaly detection approach STEN. STEN achieves spatial-temporal modeling of the data normality, contrasting to the existing methods that are focused on the temporal perspective only.
– To learn the temporal normality, we propose the OTN module. This module captures temporal associations and contextual information in time series by learning the distribution of sub-sequence orders. Different from many Transformer-based methods that calculate associations via heavy attention mechanisms, our method successfully models temporal normality by fully harnessing the unique continuity of time series data.
– To learn the spatial normality, we propose the DSN module. By wielding an informative random projection space, our approach further restrains spatial proximity in the representation space. Compared to mainstream reconstruction-based models that focus on individual normality, our approach investigates data affinity among the sequences, thereby capturing the normality beyond the temporal perspective.
– STEN outperforms state-of-the-art methods on five popular benchmark datasets for time series anomaly detection.

2 Related Work

Time Series Anomaly Detection. Time series anomaly detection is an old discipline, which has received increasing attention in recent years. Early traditional methods focus on statistical approaches such as moving averages, AutoRegressive Integrated Moving Average (ARIMA) models [5], and their multiple variants [20]. With the emergence of machine learning technology, techniques including classification [19], clustering [15], ensemble learning [32], and time series forecasting [14] are applied to time series anomaly detection. Besides, Tsfresh has inspired the window-to-feature approach, enhancing the efficiency of feature extraction in time series analysis [8]. ROCKET's focus on sub-sequence patterns through random convolutional kernels has inspired advancements in capturing local temporal patterns [11]. However, traditional methods are often constrained by the learning capability and the quantity of labeled data, making it challenging to achieve satisfactory performance.

With the burgeoning of deep learning techniques, many deep anomaly detection methods have been introduced in the literature. Owing to deep learning's powerful capability in modeling intricate data patterns and distributions, deep anomaly detection methods dramatically improve the detection performance over conventional methods [23]. Mainstream deep time-series anomaly detection methods are based on generative models, in which the learning models are trained to predict or reconstruct original raw time series data [10]. Prediction-based methods train models to forecast the value of the next timestamp, using the discrepancy between predicted and actual values to indicate the abnormal degree of the current timestamp. Reconstruction-based methods compare the error between reconstructed and actual values. These methods employ various neural network architectures such as internal memory, dilated convolutions, and graph structure learning, intending to capture the temporal characteristic of time series, yielding impressive results across numerous benchmarks [7,12,26,30]. Additionally, some studies have coupled them with adversarial training to amplify the discriminability of anomalies [3,22]. A recent work, named Anomaly Transformer [34], utilizes the Transformer to model the associations between sequences and their neighboring priors, identifying anomalies through association differences.

Self-supervised Learning on Time Series. Self-supervised learning generates supervision signals from the data itself. Via various proxy learning tasks, the learning models embed data patterns into the projection space, offering semantic-rich representations to downstream tasks. Self-supervised time series analysis can be categorized into generative-, contrastive-, and adversarial-based methods [38]. As introduced above, generative methods rely on prediction/reconstruction learning objectives to capture the characteristics of time series data. These methods often overfit the training data, and anomalies may have similarly small reconstruction errors. Adversarial-based techniques, noted in [21,39], might be hampered by complex and unstable training regimes.

Contrastive learning defines positive and negative pairs, and the learning process minimizes the distance between positive pairs and maximizes the distance between negative pairs in the feature space. This approach can encourage the model to learn meaningful and discriminative features. This concept has been applied in studies such as [9,27,36]. DCdetector [35] is one of the latest frameworks devised for anomaly detection tasks, which employs a dual-attention contrastive representation mechanism to differentiate between normal and abnormal samples effectively. However, this approach still overlooks the importance of spatial normality learning in comprehending the normal patterns of time series data.

3 STEN: Spatial-Temporal Normality Learning

3.1 Problem Statement

Let $\mathcal{X} = \langle \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N \rangle$ be a sequence of multivariate time series with N observations, with each observation $\mathbf{x}_t \in \mathbb{R}^D$ denotes the data values of D variants at a certain timestamp t, where $1 \leq t \leq N$, then unsupervised time series anomaly detection (TSAD) aims to learn an anomaly scoring function f without reliance on any labels of the training data $\mathcal{X}$. f is applied to measure the abnormality of observations in testing sequences $\mathcal{X}_{\text{test}}$. Higher anomaly scores indicate a higher likelihood to be anomalies. In STEN, we design a particular f that can effectively capture spatial-temporal normality in training data $\mathcal{X}$.

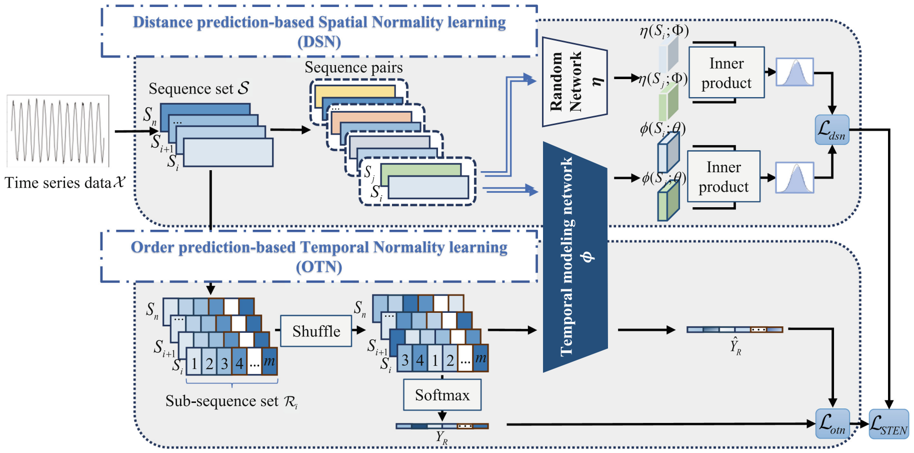

Distance prediction-based Spatial Normality learning (DSN)
Sequence set $\mathcal{S}$
Sequence pairs
Time series data $\mathcal{X}$
Random Network η
$\eta(S_i;\Phi)$
$\eta(S_j;\Phi)$
$\phi(S_i;\theta)$
$\phi(S_j;\theta)$
Inner product
Inner product
$\mathcal{L}_{dsn}$
Order prediction-based Temporal Normality learning (OTN)
Temporal modeling network ϕ
Shuffle
Softmax
Sub-sequence set $\mathcal{R}_i$
$\hat{Y}_R$
Y_R
$\mathcal{L}_{otn}$
$\mathcal{L}_{STEN}$

Fig. 2. Overview of STEN. STEN consists of two self-supervised components: DSN and OTN. In DSN, the distances between sequence pairs after being projected by a random network η form a compact distribution, from which we distill the spatial normality patterns using a MSE loss function $\mathcal{L}_{dsn}$. OTN captures temporal normality by predicting the order among sub-sequences after a random shuffling using a distribution similarity-based loss function, Jensen-Shannon divergence.

3.2 Overview of The Proposed Approach

This paper proposes a novel unsupervised TSAD method STEN, in which data normality is learned from both temporal and spatial perspectives. As shown in Fig. 2, STEN consists of an Order prediction-based Temporal Normality learning module (OTN) and a Distance prediction-based Spatial Normality learning module (DSN). OTN learns the temporal normality information of the time series by modeling the sequential distribution of sub-sequences. On the other hand, DSN is designed to utilize the distance prediction between the sequence pairs for spatial normality learning. Finally, the anomaly score which reflects both temporal and spatial normality can be used to distinguish normal and abnormal sequences. The details of OTN and DSN are discussed in the following two sections.

3.3 OTN: Order Prediction-Based Temporal Normality Learning

Time series datasets are essentially organized into specific sequences, where connectivity and sequential information of sub-sequences illuminate time-related patterns and contextual relationships within the data. The temporality of time series data reflects the dynamic changes of data in the time dimension. Analyzing temporality can help understand the dynamic changes and time correlation of data. Therefore, the OTN module is designed to extract temporal normality in time series data by predicting the temporal order between their sub-sequences.

To this end, OTN models the temporal patterns by predicting the primary temporal order of randomly shuffled sub-sequences. Specifically, a sliding window of length L and stride R are first used to divide $\mathcal{X}$ into a collection of n sequences $\mathcal{S} = \{S_1, S_{1+R}, ..., S_n\}$, where each $S_t = \langle \mathbf{x}_t, \mathbf{x}_{t+1}, ..., \mathbf{x}_{t+L-1} \rangle$. For each sequence $S \in \mathcal{S}$, we generate a fixed number of equal-length short sequences with a fixed length l and stride r. One sequence S_t will result in m sub-sequences $\mathcal{R}_t = \{R_1, R_2, ..., R_m\}$. The position orders of the sub-sequences are used as a set of self-supervised class labels to train the neural network ϕ. The Jensen-Shannon (JS) divergence, a popular method for measuring the difference between two different distributions, is then used to define our self-supervised loss $\mathcal{L}_{otn}$ in predicting the order distributions of these sub-sequences, which is formulated as follows:

$$\mathcal{L}_{\text{otn}} = \sum_{i=1}^{c} \hat{Y}_{R_i} \log \left(\frac{\hat{Y}_{R_i}}{\frac{1}{2}\left(\hat{Y}_{R_i} + Y_{R_i}\right)} \right) + \sum_{i=1}^{c} Y_{R_i} \log \left(\frac{Y_{R_i}}{\frac{1}{2}\left(\hat{Y}_{R_i} + Y_{R_i}\right)} \right), \quad (1)$$

where $\hat{Y}_R$ represents the softmax results of the temporal order prediction for the sub-sequence set $\mathcal{R}$ and Y_R is the ground truth order distribution. Minimizing this loss enables the learning of temporal normal patterns based on the local sub-sequence context.

3.4 DSN: Distance Prediction-Based Spatial Normality Learning

The spatial patterns of time series data we aim to capture are the spatial distribution of the data that changes over time in a feature space. It describes the spatial distribution patterns, trends, and changes of the data over a period of time, which cannot be captured by the temporal modeling. To complement the temporal normality learning, we propose the DSN module to model the spatial normality of the time series data in a feature space. Different from OTN that is designed to learn the temporal dynamics within the individual sequences, DSN is designed to model the spatial relation between the sequences, and thus, it is performed on the sequences set $\mathcal{S}$ rather than the sub-sequences.

One challenge here is the lack of supervision signals for learning the spatial patterns. Inspired by the solid theoretical and empirical results in [4,16,29], we propose to use the distance of the sequences in a feature space spanned by random projection as our supervision source for learning the spatial patterns among the sequences. It has been theoretically justified and empirically shown in these prior studies that although it is in a randomly projected feature space, the distance information in the original space can be well preserved. To utilize these random distances through deep neural networks, we design the self-supervised distance prediction method. More specifically, for each sequence S_i, we randomly select one sequence from the remaining sequences and construct a sequence pair (S_i, S_j). They are then respectively fed to a trainable network $\phi : \mathbb{R}^D \mapsto \mathbb{R}^M$ with parameters Θ and a random network $\eta : \mathbb{R}^D \mapsto \mathbb{R}^M$, which is a representation learner with parameters $\varPhi$. The parameters $\varPhi$ in the network η is initialized with random weights and frozen, which serves as the random project model. It is used to project the original sequences and obtain the distance-based relation via:

$$d_\eta(S_i, S_j) = \eta(S_i; \varPhi)^T \eta(S_j; \varPhi), \tag{2}$$

where $d_\eta(S_i, S_j)$ represents a spatial relation of the two sequences in the projected feature space. To learn these spatial relations, we train the network ϕ to predict/distill these distances via:

$$\mathcal{L}_{dsn} = \frac{1}{n} \sum_{i=1}^{n} (d_\phi(S_i, S_j) - d_\eta(S_i, S_j))^2, \tag{3}$$

where n is the number of sequence pairs and $d_\phi(S_i, S_j)$ is defined as follows:

$$d_\phi(S_i, S_j) = \phi(S_i; \varTheta)^T \phi(S_j; \varTheta). \tag{4}$$

The training of the network ϕ can be seen as distilling semantic information from the distance of sequence pairs, learning the relative spatial normality of time series data. A normal sequence typically include a dense set of normal sequences in their local neighborhood while being distant from the other normal sequences. Thus, its distance-based spatial relation to the other sequences forms a Gaussian-like distribution. By contrast, the anomalous sequences are assumed to be distant from most sequences, so its distribution of the distances to other sequences tends to be a uniform one. DSN is designed to learn such discriminative patterns by minimizing the loss $\mathcal{L}_{dsn}$.

3.5 Training and Inference

Training. The OTN and DSN are synthesized in our approach STEN to capture the spatial-temporal dependencies of the time series data, offering a comprehensive modeling of the normal patterns. Thus, the overall loss function in STEN is composed of the loss functions from the two above two self-supervised tasks:

$$\mathcal{L}_{STEN} = \mathcal{L}_{otn} + \alpha \mathcal{L}_{dsn}, \tag{5}$$

where α is a hyperparameter used to modulate the two modules, and the learnable parameters in network ϕ are jointly learned by the loss function $\mathcal{L}_{STEN}$.

Inference. In the OTN module, because of the continuity of the normal patterns, the order distribution reflects the overall order prediction of a sub-sequence collection $\mathcal{R}$, and the discrete order of a single sub-sequence R_i reflects the prediction of the current sub-sequence. Thus, the discrepancies in both types of predictions for normal data will be less than those for anomalies. Motivated by this, we use both of these differences in our anomaly scoring, which is defined as follows:

$$Score_{otn}(R_i) = \frac{\left|\hat{Y}_{R_i} - Y_{R_i}\right|}{\mathcal{L}_{otn_i}}, \tag{6}$$

where the numerator is the difference in a single subspace while the denominator summarizes the differences across all sub-sequences. Note that $\mathcal{L}_i$ has a different scale from $\mathcal{L}_{otn}$, so we perform an upsampling of this score in Eq. (6) by replicating it from a scalar to a vector of length of m.

The spatial normality of the sequences is captured in $\mathcal{L}_{dsn}$, i.e., for a normal sequence S_i, it is expected to have substantially smaller $\mathcal{L}_{dsn}$ than anomalous sequences when paired with other sequences S_j. To utilize spatial-temporal normality for TSAD, STEN defines an overall anomaly score as:

$$Score(R_i) = Score_{otn}(R_i) + \beta Score_{dsn}(R_i), \tag{7}$$

where β is a hyperparameter to control the importance of the spatial normality term $Score_{dsn}(R_i) = \mathcal{L}_{dsn}(R_i)$. Note that $\mathcal{L}_{dsn}$ measures the normality at the sequence level, and we assign the same anomaly score for the sub-subsequences within a sequence.

4 Experiments

4.1 Experimental Setup

Benchmark Datasets. Five publicly available multivariate time series datasets are used in our experiments, with the relevant statistics shown in Table 1. PSM (Pooled Server Metrics Dataset) [1] is a dataset of IT system monitoring signals from eBay server machines with 25 dimensions. Both MSL (Mars Science Laboratory Dataset) and SMAP (Soil Moisture Active Passive Dataset) [18] are public datasets collected by NASA with 27 and 55 entities respectively, which contain sensor and actuator data from the Mars Rover, as well as soil samples and telemetry information from a satellite. Epilepsy (Epilepsy seizure dataset) is an activity dataset collected from a triaxial accelerometer on the wrist of a human subject's hand, and we treat data during walking, running, and sawing as normal data and seizures as anomalies according to [33]. DSADS (Daily and Sports Activities Dataset) collects motion sensors from eight subjects, including 19 daily and physical activities. Following [33], we use intense activities as the anomaly class, with the data collected from other activities used as normal.

Competing Methods. STEN is compared with the following eight state-of-the-art (SotA) anomaly detectors specifically designed for time series data. These competing methods can be generally categorized into four types, i.e., (1) reconstruction-based models: MSCRED (MSC for short) [37], TranAD (Tran for short) [28], and AnomalyTransformer (AT for short) [34]; (2) forecasting-based models: GDN [12]; (3) one-class classification models: TcnED (TED for short) [13], COUTA (COU for short) [33], and NCAD (NCA for short) [6]; (4) contrastive learning-based models: DCdetector (DC for short) [35].

Table 1. Key dataset statistics.

	N_{train}	N_{test}	#Dimensions	#Entities	AnomalyRatio(%)
PSM	132,481	87,841	25	1	27.8
MSL	2,160	2,731	55	27	10.7
SMAP	2,556	8,071	25	55	13.1
Epilepsy	33,784	22,866	5	1	10.2
DSADS	85,500	57,000	47	1	6.3

Table 2. AUC-ROC, AUC-PR, and F_1 results on five TSAD datasets.

	Dataset	Ours	DC	AT	NCA	Tran	COU	TED	GDN	MSC
AUC-ROC	PSM	**0.998**	0.967	0.993	0.973	0.972	0.975	0.970	0.968	0.963
	MSL	**0.996**	0.961	0.979	0.983	0.986	0.989	0.983	0.976	0.899
	SMAP	**0.999**	0.991	0.993	0.970	0.921	0.919	0.926	0.978	0.821
	Epilepsy	**0.998**	0.886	0.996	0.984	0.935	0.951	0.933	0.990	0.832
	DSADS	**0.994**	0.983	0.962	0.985	0.984	0.992	0.985	0.974	0.905
AUC-PR	PSM	**0.995**	0.946	0.989	0.942	0.955	0.955	0.948	0.940	0.932
	MSL	**0.941**	0.915	0.933	0.868	0.905	0.886	0.897	0.849	0.483
	SMAP	**0.991**	0.961	0.971	0.884	0.753	0.758	0.717	0.869	0.644
	Epilepsy	**0.991**	0.837	0.980	0.943	0.790	0.760	0.773	0.967	0.565
	DSADS	**0.948**	0.876	0.926	0.880	0.917	0.947	0.913	0.785	0.659
BEST F_1	PSM	**0.986**	0.959	0.982	0.908	0.914	0.925	0.881	0.870	0.889
	MSL	**0.944**	0.932	0.942	0.809	0.888	0.867	0.890	0.852	0.605
	SMAP	**0.974**	0.963	0.967	0.845	0.699	0.701	0.711	0.781	0.668
	Epilepsy	**0.991**	0.875	0.987	0.904	0.803	0.793	0.777	0.918	0.640
	DSADS	**0.934**	0.908	0.932	0.875	0.846	0.901	0.844	0.825	0.657

Evaluation Metrics. The performance of STEN is measured according to a wide range of evaluation metrics. Following the mainstream studies in this research line [3,28,33], we adopt three popular metrics including the Area Under the Receiver Operating Characteristic Curve (AUC-ROC), the Area Under the Precision-Recall Curve (AUC-PR), and the best F_1 score. Note that these three metrics are calculated upon anomaly scores processed by the point-adjust strategy [31]. It is a commonly used strategy in time series anomaly detection [6,33–35], which adjusts the anomaly score of each anomaly segment to the highest score within this segment. To circumvent biases introduced by point adjustment, we further employ five recently proposed evaluation measures. Considering the temporal relationship/distance between ground truths and predictions, [17] calculates the affiliation precision and recall. We utilize the harmonic mean of these two measures, i.e., Affiliation F_1 (denoted by Aff-F_1). [24] introduces a novel metric called Range-AUC, which extends the AUC measurement to account for range-based anomalies. Additionally, the paper introduces volume under the ROC surface (VUS-ROC) and volume under the PR surface (VUS-PR) as new metrics for computing the volume under ROC and PR curves, respectively.

Implementation Details. We summarize the default implementation settings of our method STEN as follows. Both the number of generated sub-sequences m and the sub-sequence length l are set to 10 across all datasets. The temporal modeling network ϕ is composed of a single-layer GRU network with shared parameters across 10 units, and the dimension of the hidden state d_{model} is set to 256. To balance the influence of OTN and DSN, the hyperparameters α and β are set to 1 by default. For calculating the affiliation metric, an anomaly is defined as any timestamp whose anomaly score exceeds the $(100 - \delta)$ percentile, with δ setting to 0.6 by default. The weight parameters are optimized using Adam optimizer with a learning rate of 10^{-5} for a total of 5 epochs. For the PSM, SMAP, and DSADS datasets, the batch size is set to 256, while for the MSL and Epilepsy datasets, each mini-batch contains 64 training samples.

4.2 Main Results

To verify the detection performance of our method, we conduct experiments on five real-world TSAD datasets and compare it with eight SotA methods.

The results for the methods in AUC-ROC, AUC-PR, and best F_1 metrics are shown in Table 2. Our method STEN is the best performer on all five datasets across the three metrics, indicating the importance and effectiveness of joint modeling of spatial-temporal normality. On average, STEN obtains an AUC-ROC of 0.997, an AUC-PR of 0.973, and an F_1 of 0.966. Specifically, in terms of the AUC-PR metric, STEN achieves an average improvement ranging from approximately 1.3% to 31.6% over eight SotA methods, which highlights the effectiveness of our method in the precision and recall rates of detecting anomalies. Impressively, our method marks a significant improvement on the SMAP dataset, achieving an improvement of 2% to 34.7% in the AUC-PR metric. Compared to other methods, AT exhibits relatively better detection performance on these datasets, utilizing advanced Transformer structures and concepts like association differences for effective normality modeling. Nevertheless, our method STEN still consistently outperforms AT, e.g., by a large margin on some challenging datasets like MSL and DSADS. This is mainly due to the additional spatial normality modeling in STEN, besides the temporal normality learning.

It is important to note that recent studies have had vigorous discussions on how to fairly evaluate the performance of TSAD methods. Although the results in Table 2 have already demonstrated that our method outperforms the SotA methods under the three traditional metrics, to have a more comprehensive and fair evaluation, we also use several recently proposed evaluation metrics. The evaluation results of our model under the new metrics are shown in Table 3, with the best competing models NCA, DC, and AT as our baselines. It is evident that on the PSM, MSL and SMAP datasets, our method outperforms all others across almost all new metrics. For the remaining two datasets, we still achieve

Table 3. Aff-F_1, $R_{\text{AUC-ROC}}$, $R_{\text{AUC-PR}}$, VUS-ROC, and VUS-PR results.

Dataset	Method	Aff-F_1	$R_{AUC-ROC}$	R_{AUC-PR}	VUS-ROC	VUS-PR
PSM	NCA	0.394	0.862	0.826	0.868	0.828
	DC	0.583	0.817	0.839	0.805	0.826
	AT	0.507	0.952	**0.959**	0.883	0.905
	Ours	**0.636**	**0.973**	0.957	**0.966**	**0.945**
MSL	NCA	0.586	0.947	0.755	0.949	0.760
	DC	0.641	0.844	0.791	0.841	0.786
	AT	0.659	0.915	0.844	0.917	0.804
	Ours	**0.687**	**0.955**	**0.853**	**0.957**	**0.858**
SMAP	NCA	0.662	0.958	0.838	0.957	0.837
	DC	0.662	0.944	0.914	0.939	0.908
	AT	0.675	0.968	0.938	0.958	0.930
	Ours	**0.677**	**0.985**	**0.954**	**0.985**	**0.952**
Epilepsy	NCA	0.585	**0.965**	0.917	0.965	0.917
	DC	0.655	0.770	0.808	0.773	0.811
	AT	0.691	0.935	0.936	0.939	0.940
	Ours	**0.746**	0.964	**0.943**	**0.966**	**0.947**
DSADS	NCA	0.650	0.835	0.779	0.839	0.785
	DC	0.617	**0.942**	0.829	0.943	0.834
	AT	**0.682**	0.866	**0.860**	0.870	0.864
	Ours	0.657	0.940	0.856	**0.944**	**0.865**

superior results, with a minimal margin of less than 0.005 in all cases except Aff-F_1 on DSADS. These results re-affirm the improvement of our method over SotA methods.

4.3 Ablation Study

Significance of the OTN and DSN Modules. We then investigate the importance of the two components DSN and OTN. We remove each of them from our method STEN individually, resulting in two ablation variants: one with only DSN and another with only OTN. The results are presented in Table 4, which demonstrate that both of the DSN and OTN modules have some major contributions to the model's superior detection performance. This is particularly true for the OTN module, which significantly increases the model's performance in terms of the F_1 score by an average of approximately 16.9% over the five datasets.

OTN vs Error Prediction-Based Approach in Modeling Temporal Normality. Furthermore, to illustrate the effectiveness of OTN in capturing temporal patterns, we derive a variant of STEN that combines the DSN with a popular temporal pattern modeling module based on error prediction (EP) of the current timestamp. The experiment results reveal that the OTN module achieves improvements of approximately 7.3% in AUC-PR and 10.1% in the F_1 score, respectively. These results demonstrate that our OTN module is significantly more effective compared to the popular error prediction methods, which is mainly due to the fact that the OTN module allows the modeling of normal patterns in multiple diverse temporal contexts, contrasting to the single temporal context in the existing error prediction method.

Table 4. Ablation study results of STEN. EP represents the popular error prediction-based approach for temporal pattern modeling.

DSN	OTN	PSM		MSL		SMAP		Epilepsy		DSADS	
		AUC-PR	F_1	AUC-PR	F_1	AUC-PR	F_1	AUC-PR	F_1	AUC-PR	F_1
✓	✗	0.913	0.902	0.763	0.752	0.847	0.794	0.794	0.808	0.790	0.728
✗	✓	0.995	0.975	**0.946**	0.919	0.990	0.963	0.979	0.978	0.929	0.906
✓	✓	**0.995**	**0.986**	0.941	**0.944**	**0.991**	**0.974**	**0.991**	**0.991**	**0.948**	**0.934**
DSN + EP		0.984	0.952	0.930	0.897	0.921	0.856	0.957	0.939	0.707	0.681

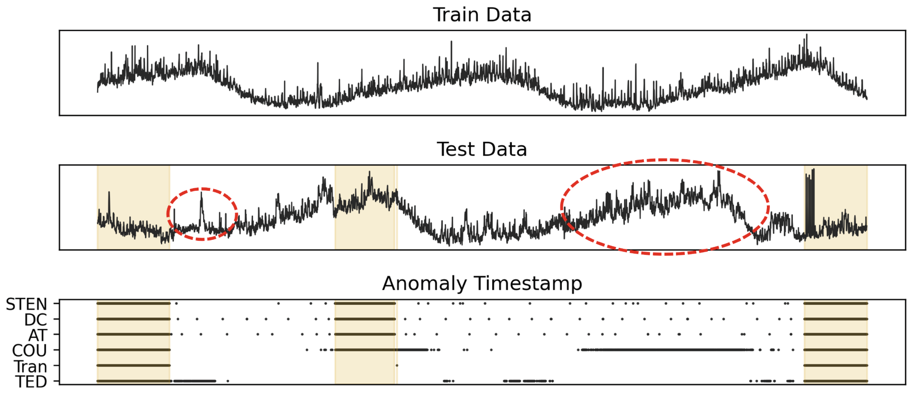


Fig. 3. (Top to Bottom) Training/testing segments of the PSM dataset, and the anomalous timestamps (marked by small dots) predicted by STEN and the five best-performing competing methods. The ground truth anomalous segments are highlighted in gold. The most prominent false-positive data segments are encircled in red dashed lines.

4.4 Qualitative Analysis

We visualize the results to further showcase the anomaly detection capabilities of STEN. Fig. 3 presents the data from the PSM dataset (feature id 22), along with the anomaly timestamps detected by STEN and its competing methods. This dataset exhibits complex temporal patterns, which may lead to false positives by many algorithms during detection. Compared to the competing methods, our method STEN can detect all three anomalies while at the same time having the least false positives. For example, COU and TED incorrectly label test data segments that are similar to the normal patterns in the training data as anomalies. Meanwhile, both Tran and TED miss an anomaly, i.e., the second anomalous segment. DC and AT illustrate comparable performance in detecting all three anomalies, but they still produce more false positives than our method. This advatange is due to the spatial normality modeling of STEN, which is often ignored in existing methods like DC and AT.

4.5 Sensitivity Analysis

We also conduct sensitivity analysis on four key hyperparameters. The F_1 score and AUC-PR results have a similar trend and we report the AUC-PR results in Fig. 4 due to page limitation. Figure 4(a) illustrates the sensitivity of the hyperparameter regulating the loss $\mathcal{L}_{dsn}$, i.e., α, in the model's overall loss function. This analysis aims to investigate the model's capability to capture the normal patterns of data when applying different weights to our two modules. The results indicate that the model's performance remains stable across multiple benchmark datasets within a relatively large value range. Figure 4(b) focuses on the weight β of $Score_{dsn}(R_i)$, examining the contribution of the two modules to the anomaly scoring. The results suggest that varying β does not significantly affect the model's detection performance, showcasing the model's robustness. We also experiment with different sub-sequence lengths l. The results are shown in Fig. 4(c), from which it is clear that our method achieves stable detection performance with sub-sequences of various lengths ranging from 10 to 100. Furthermore, the anomaly threshold δ serves as a hyperparameter that distinguishes anomalies from normal fluctuations in the data. Figure 4(d) illustrates the experimental results when δ varies between 0.5 and 1. It can be observed that our method performs stably w.r.t. changes in δ.

4.6 Time Efficiency

We compare training time of neural network-based models across various datasets to investigate their time efficiency. Due to the obviously high algorithmic complexity of GDN and MSCRED, we select six methods with comparable training duration for comparison, as shown in Table 5. Unlike previous methods, our model employs two modules (DSN and OTN) to train and combine sub-sequences of different length, potentially increasing the computational complexity. Although not showing the most efficient performance, our method illustrates significantly better detection accuracy than the competing methods.

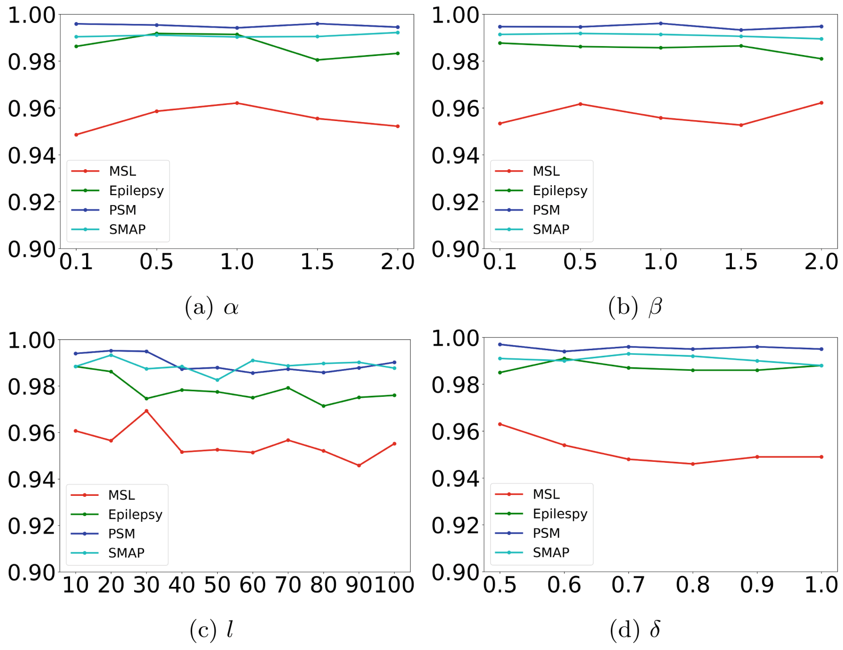


Fig. 4. AUC-PR results of STEN w.r.t different hyperparameters.

Table 5. Run times (in seconds) in multiple datasets. The best results are indicated in bold, while the worst results are underlined.

Methods	PSM	MSL	SMAP	Epilepsy	DSADS
TED	308.2	140.0	317.3	76.0	1646.3
Tran	144.5	70.3	152.8	35.2	755.1
NCA	711.3	330.3	746.8	179.0	421.5
COU	282.5	127.0	286.6	71.5	1440.2
DC	508.6	380.8	967.1	98.3	1536.5
AT	**11.7**	**11.1**	**12.6**	**10.8**	**15.4**
Ours	214.9	137.3	477.8	44.4	483.3

5 Conclusion

This article introduces a novel TSAD method named STEN. Unlike existing methods that solely focus on the temporal normality of time series data, STEN incorporates the spatial normality of the data as well, enabling a more comprehensive normality learning of the time series data. This enables STEN to achieve greater discriminative feature representations in distinguishing abnormal sequences from the normal ones. In STEN, we design two pretext tasks to extract spatial-temporal features of time series data. The order prediction-based temporal normality learning (OTN) models the temporal dependencies of the data by modeling the distribution of sub-sequence order, while the distance prediction-based spatial normality learning (DSN) learns the spatial relation of sequence pairs in the projected space in the form of distance prediction. Extensive experiments demonstrate that STEN exhibits superior TSAD performance, outperforming eight SotA algorithms on five benchmark datasets, with the effectiveness of each of the proposed two modules justified in our ablation study.

Acknowledgements. This work was supported in part by the Key Cooperation Project of the Chongqing Municipal Education Commission under Grants HZ2021017 and HZ2021008, in part by the National Natural Science Foundation of China under Grant 62102442.

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: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2485–2494 (2021)
2. Anandakrishnan, A., Kumar, S., Statnikov, A., Faruquie, T., Di, X.: Anomaly detection in finance: editors' introduction. Knowledge Discovery and Data Mining (2017)
3. Audibert, J., Michiardi, P., Guyard, F., Marti, S., Zuluaga, M.A.: USAD: unsupervised anomaly detection on multivariate time series. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2020)
4. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 245–250 (2001)
5. Box, G.E.P., Pierce, D.A.: Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. J. Am. Stat. Assoc. 1509 (1970)
6. Carmona, C.U., Aubet, F.X., Flunkert, V., Gasthaus, J.: Neural contextual anomaly detection for time series. arXiv preprint arXiv:2107.07702 (2021)
7. Chen, F., et al.: LARA: a light and anti-overfitting retraining approach for unsupervised time series anomaly detection. In: Proceedings of the ACM on Web Conference, pp. 4138–4149 (2024)
8. 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)
9. Sun, Y., Pang, G., Ye, G., Chen, T., Hu, X., Yin, H.: Unraveling the 'Anomaly' in time series anomaly detection: a self-supervised tri-domain solution. arXiv preprint arXiv:2311.11235 (2023)
10. Darban, Z.Z., Webb, G.I., Pan, S., Aggarwal, C.C., Salehi, M.: Deep learning for time series anomaly detection: a survey. arXiv preprint arXiv:2211.05244 (2022)

11. 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)
12. Deng, A., Hooi, B.: Graph neural network-based anomaly detection in multivariate time series. In: Proceedings of the AAAI conference on artificial intelligence, vol. 35, pp. 4027–4035 (2021)
13. Garg, A., Zhang, W., Samaran, J., Savitha, R., Foo, C.S.: An evaluation of anomaly detection and diagnosis in multivariate time series. IEEE Trans. Neural Networks Learn. Syst. 2508–2517 (2022)
14. Günnemann, N., Günnemann, S., Faloutsos, C.: Robust multivariate autoregression for anomaly detection in dynamic product ratings. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 361–372 (2014)
15. Hautamaki, V., Karkkainen, I., Franti, P.: Outlier detection using k-nearest neighbour graph. In: Proceedings of the 17th International Conference on Pattern Recognition, vol. 3, pp. 430–433. IEEE (2004)
16. Hegde, C., Wakin, M., Baraniuk, R.: Random projections for manifold learning. In: Advances in Neural Information Processing Systems, vol. 20 (2007)
17. Huet, A., Navarro, J.M., Rossi, D.: Local evaluation of time series anomaly detection algorithms. In: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 635–645 (2022)
18. Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T.: Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2018)
19. Karczmarek, P., Kiersztyn, A., Pedrycz, W., Al, E.: K-means-based isolation forest. Knowl.-Based Syst 105659 (2020)
20. Li, J., Di, S., Shen, Y., Chen, L.: FluxEV: a fast and effective unsupervised framework for time-series anomaly detection. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining (2021)
21. Li, X., Metsis, V., Wang, H., Ngu, A.H.H.: TTS-GAN: a transformer-based time-series generative adversarial network. In: Michalowski, M., Abidi, S.S.R., Abidi, S. (eds.) AIME 2022. LNCS, vol. 13263, pp. 133–143. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-09342-5_13
22. Liu, S., et al.: Time series anomaly detection with adversarial reconstruction networks. IEEE Trans. Knowl. Data Eng. **35**, 4293–4306 (2023)
23. Pang, G., Shen, C., Cao, L., Hengel, A.V.D.: Deep learning for anomaly detection: a review. ACM Comput. Surv. (CSUR) **54**(2), 1–38 (2021)
24. 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)
25. Pereira, J., Silveira, M.: Learning representations from healthcare time series data for unsupervised anomaly detection. In: 2019 IEEE International Conference on Big Data and Smart Computing (BigComp), pp. 1–7. IEEE (2019)
26. Ren, H., et al.: Time-series anomaly detection service at Microsoft. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2019)
27. Schneider, T., et al.: Detecting anomalies within time series using local neural transformations. arXiv preprint arXiv:2202.03944 (2022)
28. Tuli, S., Casale, G., Jennings, N.R.: TRANAD: deep transformer networks for anomaly detection in multivariate time series data. arXiv preprint arXiv:2201.07284 (2022)

29. Wang, H., Pang, G., Shen, C., Ma, C.: Unsupervised representation learning by predicting random distances. arXiv preprint arXiv:1912.12186 (2019)
30. Wu, W., et al.: Developing an unsupervised real-time anomaly detection scheme for time series with multi-seasonality. IEEE Trans. Knowl. Data Eng. **34**(9), 4147–4160 (2020)
31. Xu, H., et al.: Unsupervised anomaly detection via variational auto-encoder for seasonal KPIS in web applications. In: Proceedings of the Conference on World Wide Web (2018)
32. Xu, H., Pang, G., Wang, Y., Wang, Y.: Deep isolation forest for anomaly detection. IEEE Trans. Knowl. Data Eng. **35**(12), 12591–12604 (2024)
33. Xu, H., Wang, Y., Jian, S., Liao, Q., Wang, Y., Pang, G.: Calibrated one-class classification for unsupervised time series anomaly detection. IEEE Trans. Knowl. Data Eng. (2024)
34. Xu, J., Wu, H., Wang, J., Long, M.: Anomaly transformer: time series anomaly detection with association discrepancy. In: International Conference on Learning Representations (2022)
35. Yang, Y., Zhang, C., Zhou, T., Wen, Q., Sun, L.: DCdetector: dual attention contrastive representation learning for time series anomaly detection. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3033–3045 (2023)
36. Yue, Z., et al.: Ts2vec: towards universal representation of time series. In: Proceedings of the AAAI Conference on Artificial Intelligence (2022)
37. Zhang, C., et al.: A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 1409–1416 (2019)
38. Zhang, K., et al.: Self-supervised learning for time series analysis: taxonomy, progress, and prospects. arXiv preprint arXiv:2306.10125 (2023)
39. Zhou, B., Liu, S., Hooi, B., Cheng, X., Ye, J.: BeatGAN: anomalous rhythm detection using adversarially generated time series. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (2019)

Modeling Text-Label Alignment for Hierarchical Text Classification

Ashish Kumar and Durga Toshniwal(✉)

Indian Institute of Technology Roorkee, Roorkee, India
{ashish_k,durga.toshniwal}@cs.iitr.ac.in

Abstract. Hierarchical Text Classification (HTC) aims to categorize text data based on a structured label hierarchy, resulting in predicted labels forming a sub-hierarchy tree. The semantics of the text should align with the semantics of the labels in this sub-hierarchy. With the sub-hierarchy changing for each sample, the dynamic nature of text-label alignment poses challenges for existing methods, which typically process text and labels independently. To overcome this limitation, we propose a Text-Label Alignment (TLA) loss specifically designed to model the alignment between text and labels. We obtain a set of negative labels for a given text and its positive label set. By leveraging contrastive learning, the TLA loss pulls the text closer to its positive label and pushes it away from its negative label in the embedding space. This process aligns text representations with related labels while distancing them from unrelated ones. Building upon this framework, we introduce the Hierarchical Text-Label Alignment (HTLA) model, which leverages BERT as the text encoder and GPTrans as the graph encoder and integrates text-label embeddings to generate hierarchy-aware representations. Experimental results on benchmark datasets and comparison with existing baselines demonstrate the effectiveness of HTLA for HTC.

Keywords: Multi-Label Classification · NLP · Representation Learning

1 Introduction

In HTC, documents are assigned labels corresponding to nodes within a label hierarchy tree [27]. It has applications across diverse domains, such as scientific text categorization [1], bioinformatics [18], and online product labeling [20]. However, the imbalance in label frequency, coupled with the complex hierarchical structure, makes HTC a challenging task [16].

Recent approaches to HTC employ a two-encoder framework, where a text encoder processes the input text while a graph encoder captures the label hierarchy [3,7,15,21,27]. The hierarchy is predefined based only on parent-child relationships between labels, but there are aspects to the hierarchy beyond these static links. For instance, a text sample is associated with a subset of labels that can be considered a sub-hierarchy tree. In HTC, the semantics of the text

A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14946, pp. 163–179, 2024.
https://doi.org/10.1007/978-3-031-70365-2_10

should align with the semantics of the labels in this sub-hierarchy. Aligning the semantics of the text with the semantics of the associated labels ensures that the classification model comprehensively captures the meaning conveyed in the text and accurately assigns it to the appropriate categories within the label hierarchy. This text-label alignment is dynamic since the sub-hierarchy changes for each text sample. Furthermore, existing two-encoder frameworks overlook this alignment between them as they encode text and labels separately.

We propose a text-label alignment (TLA) loss to address this challenge. TLA is based on the principle of contrastive learning and is formulated along lines similar to the NT-Xent loss [4]. For TLA to be effective, it is essential to carefully construct a negative label set consisting of challenging labels that are semantically distant from the text within the hierarchical structure. A hard negative mining technique is employed to select labels that demonstrate high similarity to the text sample but are not included in the positive label set, thus serving as effective negative labels. Positive and negative pairs are formed by associating each text sample with labels from the corresponding positive and negative label sets. The TLA loss increases alignment for the positive pairs, pulling text samples and their positive labels closer in the embedding space. Simultaneously, it decreases the alignment for negative pairs, thus pushing the text and negative labels away from each other in the embedding space. By dynamically aligning text and labels to the sub-hierarchy associated with each sample, the TLA loss approach inherently adjusts to the hierarchy's depth. This adaptability simplifies implementation and ensures robust performance across datasets with varying levels of hierarchy. Furthermore, in HTC, certain labels may be more prevalent as they are assigned to several documents, while others are linked to relatively fewer documents. This variation in label frequencies can result in label imbalance, posing challenges for model training and performance. Since TLA involves explicitly modeling text-label alignment for each positive label, regardless of its frequency, it also helps mitigate the label imbalance issue.

Building on this, we introduce the Hierarchical Text-Label Alignment (HTLA) model, which utilizes text-label alignment for HTC. HTLA uses BERT as its text encoder and a custom implementation of GPTrans as its graph encoder. GPTrans [5] uses transformer blocks and outperforms state-of-the-art graph models on several graph learning tasks. Its ability to model the graph from multiple dimensions makes it easily customizable for the HTC task. Within this framework, the text and label features are combined through addition, yielding a composite representation. HTLA is jointly optimized using the binary cross entropy (BCE) and TLA loss. Including TLA loss contributes to performance enhancement across datasets with simple and complex hierarchies. It models the dynamic alignment between text and labels within the hierarchical structure, addressing a challenge inadequately tackled in existing two-encoder frameworks. We summarize the contribution of our work as follows:

- We propose using the Text-Label Alignment (TLA), a loss function designed to align text with its related labels in the hierarchy.

- We introduce HTLA, a model that utilizes BERT as the text encoder and GPTrans as the graph encoder, optimized with BCE and TLA loss functions.
- Experimental results across several datasets demonstrate the superiority of HTLA in improving classification performance.

2 Related Work

HTC's existing methods can be divided into local and global approaches based on how they utilize hierarchical information. Local approaches use multiple classifiers [9,11,25] to make independent predictions at each node of the hierarchy, considering the local context and relationships within that specific node and its neighborhood. Global approaches model the entire hierarchical structure with a single classifier to generate predictions. Early global approaches aimed to merge the hierarchical label space using meta-learning [23], recursive regularization [10], and reinforcement learning [16]. These methods primarily focused on refining decoders based on hierarchical paths. The typical approach in recent studies involves enhancing flat predictions by using a graph encoder to comprehensively model the entire label structure. In their study, Zhou et al. [27] developed a graph encoder that effectively integrates existing knowledge of the hierarchical label space to acquire representations of the labels. Building upon this research, several subsequent models have emerged to explore how the hierarchical structure interacts with the text. For instance, in [2], the authors performed a joint embedding of text and labels within the hyperbolic space. Similarly, Chen et al. [3] treated the problem as semantic matching, utilizing a shared space to learn representations of both text and labels. Deng et al. [7] introduced an information maximization module that enhances the interaction between text and labels while imposing constraints on label representation. Zhao et al. [26] presented a self-adaptive fusion strategy capable of extracting representations from text and labels. Wang et al. [21] utilized contrastive learning techniques to incorporate hierarchical information into the text encoder embedding directly. Ning et al. [17] utilizes a unidirectional message-passing mechanism to improve hierarchical label information and propose a generative model for HTC. Liu et al. [15] enhance label features by introducing density coefficients for label importance in the hierarchy tree and address label imbalance with a rebalanced loss. Existing methods have employed various intricate approaches to learn hierarchical relationships and merge text-label features. However they have not emphasized on learning text-label alignment within the hierarchy. HTLA explicitly models for this dynamic alignment, ensuring that the semantics of the text align with associated labels in each sample's sub-hierarchy. This simplifies merging text and label features, requiring only addition for obtaining the composite features.

3 Methodology

The overall architecture of HTLA is depicted in Fig. 1. This section details the components of our HTLA model, which includes the text encoder, graph encoder, generation of composite representation, and the loss functions used.

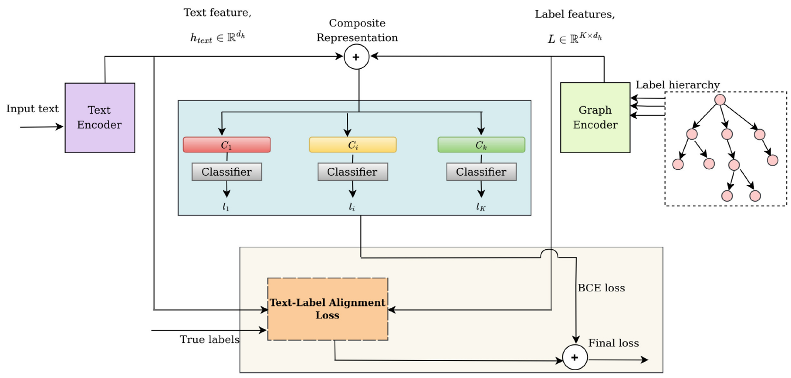


Fig. 1. Architecture of the Hierarchical Text-Label Alignment (HTLA) model. For a label i, its feature is combined with the text feature h_{text} through addition to produce the composite feature $C_i \in \mathbb{R}^{d_h}$ for each label. A shared classifier is then utilized for each C_i, and the corresponding logit l_i is selected from the output vector. The model is jointly optimized for BCE and TLA loss.

3.1 Text Encoder

We use BERT [8], a transformer-based model that generates highly contextualized text embeddings by leveraging bidirectional context and pre-trained knowledge, as our text encoder. The input text is padded with two special tokens to mark the start and end of the text, as $w = \{[CLS], w_1, w_2, \ldots, w_{n-2}, [SEP]\}$. This is then fed to the BERT encoder to produce token representations as:

$$H = \phi_{BERT}(w) \tag{1}$$

where $H \in \mathbb{R}^{n \times d_h}$ contains encoded representations for all n tokens. The token representation for [CLS] is chosen as the text feature for the entire sequence because it captures its contextual information, denoted as $h_{text} \in \mathbb{R}^{d_h}$.

3.2 Graph Encoder

GPTrans, a graph neural network, introduces the Graph Propagation Attention (GPA) mechanism into the Transformer architecture. Unlike existing Transformer-based models that often fuse node and edge information without explicit consideration, GPA in GPTrans dynamically propagates information among nodes and edges, offering a more comprehensive and nuanced understanding of the graph structure.

Our customised implementation of GPTrans consists of three main components: Feature Initialization, GPA, and *LabelEnhancer* module.

Feature Initialization. The node and edge features are initialized in this component. For each label node i, the node feature $g_i \in \mathbb{R}^{d_h}$ is initialized as:

$$g_i = embed_{node}(i) + embed_{name}(i) \tag{2}$$

- $embed_{node}(.)$ is a learnable embedding function that generates embedding of size d_h for each input node to capture essential node characteristics.
- $embed_{name}(.)$ function uses the BERT tokenizer to tokenize each label name, calculates the average of the token embeddings, and assigns it to the label. This process aids in extracting semantic information and summarizing distinctive characteristics associated with each label. The weights used for learning text embeddings with BERT are shared with $embed_{name}(.)$, ensuring informativeness in label features.

The edge feature $x_{ij} \in \mathbb{R}^{d_p}$ for each pair of nodes is initialized as:

$$x_{ij} = S_{f(i,j)} + E_{ij} \tag{3}$$

- $S_{f(i,j)}$ is the spatial encoding component, indexed by distance measure function $f(i, j)$, representing the distance between nodes i and j. It is a learnable embedding of size d_p.
- E_{ij} is the edge encoding component, accounting for edge weights along the unique path $(e_1, e_2, ..., e_D)$ connecting nodes i and j in the label hierarchy tree, where $D = f(i, j)$. The computation for E_{ij} involves averaging the edge weights along this path, expressed as $\frac{1}{D}\sum_{z=1}^{D} w_{e_z}$, where each $w_{e_z} \in \mathbb{R}^{d_p}$ represents the weight parameter for the corresponding edge e_z.

Finally, matrices $g \in \mathbb{R}^{K\times d_h}$ and $x \in \mathbb{R}^{K\times K\times d_p}$ are formed by stacking node and edge features, respectively, where K is the number of label nodes.

Graph Propagation Attention. This modified attention module explicitly defines the information flow between nodes and edges, allowing for the capture of both local and higher-order relationships within the label hierarchy. To simplify, we assume single-head self-attention in the following equations.

In the **node-to-node flow**, self-attention is improved by incorporating edge information. For this edge features x are transformed using $W_1 \in \mathbb{R}^{d_p\times n_{head}}$ which is then added to the attention map. The update node features $g' \in \mathbb{R}^{K\times d_h}$ are then computed by multiplying with value matrix V as:

$$x' = xW_1;\ A = \frac{(gW_Q)(gW_K)^T}{\sqrt{dim_h}} + x';\ g' = softmax(A)V \tag{4}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d_h\times d_h}$, $V = gW_V$, and $dim_h = d_h/n_{head}$ refers to the size of each head.

The **node-to-edge** flow updates the edge features based on attention patterns observed during node-to-node interactions. The attention scores $A \in \mathbb{R}^{K\times K\times n_{head}}$ are combined with their softmax values, creating a weighted sum, which is then transformed by the matrix $W_2 \in \mathbb{R}^{n_{head}\times d_p}$ as:

$$x' = (A + softmax(A))W_2 \tag{5}$$

In the **edge-to-node** flow, weights are computed based on edge features $x' \in \mathbb{R}^{K\times K\times d_p}$ calculated in previous step. Subsequently, a weighted sum of edge features is utilized to update node features, followed by linear transformations $W_3 \in \mathbb{R}^{d_p \times d_h}$ and $W_4 \in \mathbb{R}^{d_h \times d_h}$, as:

$$g'' = (sum(x'.softmax(x'), dim = 1))W_3;\ g''' = (g' + g'')W_4 \tag{6}$$

For more details on GPA please refer to the original paper [5].

LabelEnhancer. The label node features $g''' \in \mathbb{R}^{K \times d_h}$ generated by GPA serve as input to *LabelEnhancer*, a multi-layered neural network. It refines these node representations, producing the final label features $L \in \mathbb{R}^{K \times d_h}$ as:

$$L = LabelEnhancer(g''') \tag{7}$$

3.3 Generation of Composite Representation

To create a composite representation, we merge the text and label features by adding them together. In the label feature matrix $L \in \mathbb{R}^{K \times d_h}$, each f_i represents the feature vector for label i. We enhance the label feature f_i by incorporating the text feature $h_{\text{text}} \in \mathbb{R}^{d_h}$ from the corresponding sample. This results in a composite feature C_i that captures both the textual context and the specific characteristics of label i. Subsequently, this composite feature is fed into the classifier. The logit score l_i for label i is calculated as the i^{th} element of the resulting classifier output vector, and the predicted output for label i is obtained after applying *sigmoid*(.) on l_i. This process is formally defined in Eq. 8 below:

$$C_i = h_{text} + f_i; \quad l_i = (W_c^T C_i + b)_i; \quad \hat{y}_i = sigmoid(l_i) \tag{8}$$

where $W_c \in \mathbb{R}^{d_h \times K}$ and $b \in \mathbb{R}^K$ are weights and bias of the classifier. The parameters of the classifier (W_c and b) are shared across all labels, ensuring consistency in predictions.

3.4 Loss Functions

Text-Label Alignment Loss. In HTC, it is desired that the representation of a sample not only reflects its semantic content but also aligns closely with its positive labels while remaining distinct from negative labels in the embedding space. The challenge lies in identifying negative labels to establish the necessary contrasting relationship for alignment. We use hard negative mining to select a set of negative labels for each sample. Once both positive and negative labels are identified, we form pairs with the text samples and compute the TLA loss. This encourages closer alignment between text and its positive labels while maximizing dissimilarity with negative ones.

The TLA loss operates on a batch of text samples, denoted as M, each associated with a set of positive labels $P(i)$, where i represents the index of the text sample. For each sample, we obtain a set of negative labels with high similarity scores to the text sample, excluding those already identified as positive labels and denote it as $N(i)$. A positive pair is formed consisting of (h_{text_i}, f_p) where f_p denotes the label feature for label $p \in P(i)$ and h_{text_i} represents text feature of the i^{th} sample. Similarly, a negative pair is formed consisting of (h_{text_i}, f_n), where f_n denotes the label feature for label $n \in N(i)$. The TLA loss is then defined as:

$$Loss_{TLA} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{|P(i)|} \sum_{p \in P(i)} -\log \left(\frac{\exp(\text{sim}(h_{text_i}, f_p)/\tau)}{\sum_{s \in S(i)} \exp(\text{sim}(h_{text_i}, f_s)/\tau)} \right) \tag{9}$$

where sim(.) computes cosine similarity, $|P(i)|$ denotes cardinality of label set $P(i)$, $S(i) = N(i) \cup P(i)$, and $\tau \in \mathbb{R}^+$ controls temperature. Algorithm 1 outlines the steps to compute TLA loss for a batch of text samples.

Algorithm 1. Text-label alignment (TLA) loss

```
1: **Input:** Text features $Z(M \times d_h)$, Label features $L(K \times d_h)$, True labels $Y(M \times K)$, Temperature $\tau$
2: **Output:** TLA loss, $Loss_{TLA}$
3: $P \leftarrow \{\}$, $N \leftarrow \{\}$    ▷ *Initialize set for positive and negative labels*
4: $sim\_mat \leftarrow cos\_sim(Z, L^T)$    ▷ *Compute cosine similarity*
5: $P \leftarrow \{p_i \mid p_i = \{j \mid Y_{ij} = 1\}, \forall i \in \{1, 2, \ldots, M\}\}$    ▷ *Add indices of positive labels*
6: **for each** $i$ **from** 1 **to** $M$ **do**    ▷ *HardMining to get negative label set*
7:     $N[i] \leftarrow \{\}$
8:     $p\_labels \leftarrow P[i]$
9:     $neg\_sim \leftarrow sim\_mat[i]$
10:    **for each** $label$ **in** $p\_labels$ **do**
11:        $neg\_sim[label] \leftarrow -\infty$    ▷ *Set similarity to negative infinity for positive labels*
12:    **end for**
13:    $sorted\_indices \leftarrow argsort(neg\_sim, descending = True)$
14:    $hard\_negative\_labels \leftarrow \{sorted\_indices[k] \mid k \in [1, len(p\_labels)]\}$
15:    $N[i] \leftarrow N[i] \cup hard\_negative\_labels$
16: **end for**
17: $S \leftarrow \{\}$
18: **for each** $i$ **from** 1 **to** $M$ **do**    ▷ *Combine positive and negative label sets*
19:     $S[i] \leftarrow P[i] \cup N[i]$
20: **end for**
21: Compute $Loss_{TLA}$ using Equation 9
22: **return** $Loss_{TLA}$
```

Binary Cross Entropy Loss. While TLA enhances semantic alignment by aligning text with its labels, BCE complements this by emphasizing the correctness of label predictions, enabling the model to learn the distinctive features of each label independently. BCE loss for a batch of M samples is formulated as:

$$Loss_{BCE} = -\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{K}\Big(Y_{ij}\log(\hat{Y}_{ij}) + (1-Y_{ij})\log(1-\hat{Y}_{ij})\Big) \tag{10}$$

where $Y \in \mathbb{R}^{M\times K}$ represents the true label values and $\hat{Y} \in \mathbb{R}^{M\times K}$ represents the predicted label probabilities.

Final Loss. The final loss for the HTLA model is obtained by the sum of both BCE and TLA losses as:

$$Loss_{HTLA} = Loss_{BCE} + Loss_{TLA} \tag{11}$$

Table 1. Statistical details for the WOS, RCV1-V2, and NYT datasets. $|Level|$ indicates the number of hierarchy levels, $|L|$ is the total label count, and Mean-$|L|$ denotes the mean number of labels per sample

Dataset	$\|Level\|$	Train	Val	Test	$\|L\|$	Mean-$\|L\|$
WOS	2	30070	7518	9397	141	2.0
RCV1-V2	4	20833	2316	781265	103	3.3
NYT	8	23345	5834	7292	166	7.6

4 Experiments

4.1 Datasets and Evaluation Metrics

We conducted experiments and model evaluations using three datasets: WOS [13], RCV1-V2 [14], and NYT [19]. The WOS dataset contains abstracts from scientific papers, with their corresponding labels arranged in a single-path hierarchy. RCV1-V2 and NYT are news categorization datasets with multiple label paths in the hierarchy. Table 1 provides detailed statistics for each dataset. In line with previous HTC studies [3,7,15,21], we followed the label hierarchy taxonomy, data preprocessing steps and train-val-test splits outlined in [27]. We evaluated performance using the Micro-F1 and Macro-F1 scores, consistent with previous research [3,7,15,21,27].

4.2 Implementation Details

Our code is available at: https://github.com/havelhakimi/TLA. In our implementation, we use the *bert-base-uncased* model from the hugging face transformers library [22] as our BERT-based text encoder. We utilize a single layer of the GPTrans block, which includes a multi-headed attention mechanism with 12 attention heads (n_{head}). The edge feature size, d_p, is set to 30 for all datasets, determined through grid search on validation set. As for the node feature size,

d_h, we keep it identical to the text representation size of 768. The temperature hyperparameter τ for TLA is set to 0.07 for all datasets. During training, we use a batch size of 10 and opt for the Adam optimizer with a learning rate of 1e-5. Our model is implemented in PyTorch and trained end-to-end. We assess the model's performance on the validation set after each epoch and halt the training procedure if the Macro-F1 score does not show improvement for six consecutive epochs. The architectural details of the *LabelEnhancer* module are outlined in Table 2.

Table 2. Layer specification for the *LabelEnhancer* module

Layer	Input/Output Shape
Input	$K \times d_h$ (*label features* g''')
Linear	$K \times d_h / K \times 4d_h$
Activation (GELU)	$K \times 4d_h / K \times 4d_h$
Dropout	$K \times 4d_h / K \times 4d_h$
Linear	$K \times 4d_h / K \times d_h$
Dropout	$K \times d_h / K \times d_h$ (*intermediate label features* $\hat{g}$)
Residual Connection	$K \times d_h / K \times d_h$ ($g''' + \hat{g}$)
Layer Normalization	$K \times d_h / K \times d_h$ (*Final label features* L)

4.3 Experimental Results

Table 3 displays the results of HTLA and compares them with various baselines. For a detailed analysis and comparison, we also implemented fine-tuned BERT (*bert-base-uncased* from Hugging Face) and the BERT-GPTrans and HGCLR [21] alongside HTLA. While BERT employs a flat multi-label classification without considering hierarchy, BERT-GPTrans models hierarchy and is trained solely on the BCE loss. HGCLR uses contrastive learning to embed hierarchy information into BERT encoder. HGCLR, constructs positive samples for input text by masking unimportant tokens from the representation obtained through cross-attention between text and label features. The masking of tokens is determined by a threshold value, an additional hyperparameter that needs tuning for each dataset. This can inevitably introduce noise and overlook label correlations if the threshold is not appropriate. HTLA aligns text with its positive labels on a per-sample basis, ensuring that relationships between labels within the sub-hierarchy tree are implicitly captured. We conducted a one-sided paired t-test with significance level set at 0.05 to determine whether HTLA yield significantly improved outcomes. t-tests are recommended for assessing hypotheses related to average performance [6], and they remain robust even when normality assumptions are violated [12]. Across all datasets, the performance scores of HTLA show

Table 3. Comparison of results across three datasets. We report average score of 8 random runs for our implemented models(denoted with an asterisk (*)), with the second best results among our implemented models underlined. Results for other models were sourced from their respective papers.

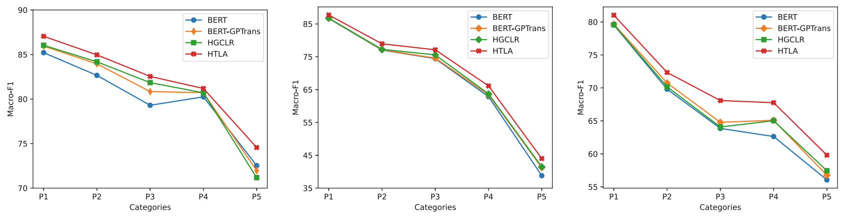
Model	WOS		RCV1-V2		NYT	
	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1
TextCNN [27]	82.00	76.18	79.37	59.54	70.11	56.84
TextRCNN [27]	83.55	76.99	81.57	59.25	70.83	56.18
HiLap-RL [16]	-	-	83.30	60.10	74.60	51.60
HiAGM [27]	85.82	80.28	83.96	63.35	74.97	60.83
HTCInfoMax [7]	85.58	80.05	83.51	62.71	-	-
HiMatch [3]	86.20	80.53	84.73	64.11	-	-
LSE-HiAGM [15]	86.01	80.01	83.86	64.57	75.01	61.29
BERT+HiAGM [21]	86.04	80.19	85.58	67.93	78.64	66.76
BERT+HTCInfoMax [21]	86.30	79.97	85.53	67.09	78.75	67.31
HiMatch-BERT [3]	86.70	81.06	86.33	68.66	-	-
HGCLR [21]	87.11	81.20	86.49	68.31	78.86	67.96
BERT*	85.85	79.93	86.14	67.10	78.65	66.31
BERT-GPTrans*	86.74	80.62	<u>86.28</u>	<u>68.19</u>	<u>78.89</u>	<u>67.34</u>
HGCLR*	<u>87.09</u>	<u>81.08</u>	86.27	68.09	78.53	67.20
HTLA*	**87.38**	**81.88**	**87.14**	**70.05**	**80.30**	**69.74**

a statistically significant improvement. Further details regarding the statistical tests can be found in Appendix A.

For the WOS, RCV1-V2 and NYT datasets, the HTLA shows a 0.8%, 1.9%, 2.4%, increase in the Macro-F1 (MaF1) compared to the second best. HTLA is more effective in enhancing text-label alignment for datasets with deeper hierarchies like RCV1-V2 and NYT, where multiple positive labels exist at each level. However, in WOS, characterized by a shallow two-level hierarchy and only one related label per level, the improvements are comparatively modest. Also, the improvements in Micro-F1(MiF1) are somewhat limited across all datasets, mainly due to its computation method. MiF1 aggregates the confusion matrix for each label, making it sensitive to predominant labels characterized by high frequencies. Conversely, MaF1 computes distinct F1 scores for each label and then averages them, assigning equal importance to all labels, irrespective of their occurrence frequency. The considerable increase in MaF1 suggests that our models effectively handle label imbalance and improve the classification of less common labels.

4.4 Analysis

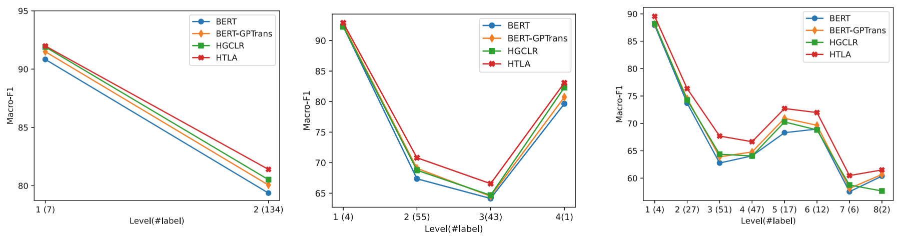
Performance Amid Label Imbalance. Evaluating a model's performance across different levels of label prevalence can provide insight into its efficacy under label imbalance. To assess model performance, we arrange the labels in descending order by the number of associated documents and divide them into five equally sized groups, denoted P1 to P5. Each group contains 20% of the labels, with P1 comprising the most prevalent labels and P5 the least. Figure 2 illustrates performance across these prevalence categories. HTLA outperforms other models, particularly for less prevalent labels in category P5, demonstrating its effectiveness in addressing label imbalance.



(a) WOS (b) RCV1-V2 (c) NYT

Fig. 2. Model performance across label prevalence categories

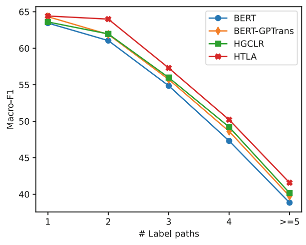
Performance Across Hierarchy Levels. Labels within hierarchies can span from general to highly specific categories. Models that excel at capturing broad patterns may struggle with finer distinctions, particularly at lower levels of the hierarchy. Figure 3 illustrates the model performance across hierarchy levels for datasets with shallow hierarchies (WOS) and those with deeper hierarchies (RCV1-V2 and NYT). In WOS, HTLA outperforms its counterparts, particularly for fine-grained labels at the second level. In RCV1-V2, characterized by numerous ambiguous labels at the second level and fine-grained labels at levels two and three, HTLA consistently outperforms other models. In NYT, which features the deepest hierarchy and an uneven distribution of labels across different levels, HTLA exhibits superior performance, especially at the deeper levels.



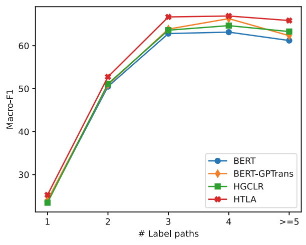
(a) WOS (b) RCV1-V2 (c) NYT

Fig. 3. Model performance across hierarchy levels

Performance Based on the Number of Label Paths. We conduct a performance analysis for datasets with multiple label paths by grouping samples based on the number of paths they traverse in the label hierarchy. Figure 4 illustrates the performance on samples for both the RCV1-V2 and NYT datasets. For both datasets, HTLA demonstrates a performance boost compared to other models as the number of label paths increases. These results indicate that HTLA excels in handling hierarchical structures with multiple label paths, making it a robust performer for datasets with intricate and complex hierarchies.



(a) RCV1-V2



(b) NYT

Fig. 4. Model performance across label paths

Ablation Study and Model Generalizability. Our model, HTLA, leverages TLA Loss and customized GPTrans, which consists of $embed_{node}(.)$ and $embed_{name}(.)$ functions to initialize features, along with a $LabelEnhancer(LE)$ module to refine label features. To assess each component's impact, we systematically removed them one at a time. The first part of Table 4 presents ablation results for HTLA. The results clearly indicate that the removal of these components leads to a decrease in performance, while HTLA, with all components intact, achieves the best performance among the compared models. Furthermore, to demonstrate model generalizability, we conducted experiments on two additional text datasets: AAPD [24] and BGC [1], using the same train-val-test splits as the original studies. Further details regarding these datasets are provided in Appendix B. The second part of Table 4 presents the results on these additional datasets, where the use of HTLA shows a performance boost compared to other models.

Table 4. Ablation results for HTLA (first part) and results on AAPD and BGC datasets (second part)

Model	WOS		RCV1-V2		NYT	
	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1
w/o TLA(BERT-GPTrans)	86.74	80.62	86.28	68.19	78.89	67.34
w/o $embed_{name}$	86.37	80.51	86.71	68.10	78.87	67.21
w/o $embed_{node}$	86.48	80.58	86.90	68.45	79.58	68.24
w/o *LE*	86.81	80.87	86.53	68.38	79.15	68.75
HTLA	**87.38**	**81.88**	**87.14**	**70.05**	**80.30**	**69.74**

Model	AAPD (2-level hierarchy)		BGC (4-level hierarchy)	
	MiF1	MaF1	MiF1	MaF1
BERT	57.65	80.90	63.21	79.77
BERT-GPTrans	58.17	81.17	64.28	80.48
HTLA	**62.37**	**81.95**	**66.05**	**81.05**

5 Conclusion

Existing methods face challenges in effectively aligning text-label semantics within the hierarchy. To address this, we propose TLA, a loss function explicitly modeling the alignment between text and its associated labels. Building upon this, we introduce HTLA model, employing a two-encoder architecture to merge text-label embeddings for enhanced representations in HTC. Our experiments show HTLA outperforms existing methods on benchmark datasets. We further analyze its performance amid label imbalance, across hierarchy levels, and based on the number of label paths to demonstrate effectiveness. Additionally, we validate HTLA's components and generalization capabilities. In future work, we aim to extend our approach to non-textual domains like images, biological data, and other multi-modal datasets.

Acknowledgements. This study was funded by the PMRF (Prime Minister's Research Fellow) program, run by the Ministry of Education, Government of India. We also acknowledge National Supercomputing Mission (NSM) for providing computing resources of 'PARAM Ganga' at IIT Roorkee, which is implemented by C-DAC and supported by the Ministry of Electronics and Information Technology (MeitY) and Department of Science and Technology (DST), Government of India.

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

A Details of Statistical Test

We evaluated the effectiveness of our implemented models by analyzing Micro-F1 (MiF1) and Macro-F1 (MaF1) scores, reporting average results from 8 runs.

Subsequently, we employed one-sided paired t-tests to assess the significance of performance variations among the models across the three datasets as detailed in Table 5. Except for the Micro-F1 score for the HTLA vs. HGCLR comparison in WOS, all p-values for comparisons are significantly below the threshold of 0.05, implying that the HTLA model demonstrates a statistically significant performance improvement.

Table 5. p-value for one-sided t-test

Model	WOS		RCV1-V2		NYT	
	MiF1	MaF1	MiF1	MaF1	MiF1	MaF1
HTLA vs HGCLR	0.23	**1.8e−4**	**3.7e−5**	**1.8e−4**	**1.3e−6**	**3.4e−7**
HTLA vs BERT-GPTrans	**2.4e−2**	**4.2e−4**	**1.5e−6**	**3.1e−5**	**5.1e−6**	**2.7e−7**
HTLA vs BERT	**5.1e−3**	**1.7e−4**	**6.2e−8**	**2.2e−6**	**4.5e−7**	**1.3e−8**

B Performance Analysis on Additional Datasets

We conducted experiments on two additional datasets, namely AAPD and BGC, to validate the generalization capabilities of the HTLA model. AAPD consists of abstracts of scientific papers from the arXiv.org[1] website, while BGC[2] contains book blurbs from the Penguin Random House website. Both datasets consist of multipath labels. Table 1 provides detailed statistics for the two datasets (Table 6).

Table 6. Statistical details for the AAPD and BGC. $|Level|$ indicates the number of hierarchy levels, $|L|$ is the total label count, and Mean-$|L|$ denotes the mean number of labels per sample

Dataset	$\|Level\|$	Train	Val	Test	$\|L\|$	Mean-$\|L\|$
AAPD	2	53840	1000	1000	61	4.09
BGC	4	58715	14785	18394	146	3.01

References

1. Aly, R., Remus, S., Biemann, C.: Hierarchical multi-label classification of text with capsule networks. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pp. 323–330. Association for Computational Linguistics, Florence, Italy, July 2019. https://doi.org/10.18653/v1/P19-2045. https://aclanthology.org/P19-2045

[1] https://arxiv.org/.

[2] https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/blurb-genre-collection.html.

2. Chen, B., Huang, X., Xiao, L., Cai, Z., Jing, L.: Hyperbolic interaction model for hierarchical multi-label classification. Proc. AAAI Conf. Artif. Intell. **34**(05), 7496–7503 (2020). https://doi.org/10.1609/aaai.v34i05.6247. https://ojs.aaai.org/index.php/AAAI/article/view/6247
3. Chen, H., Ma, Q., Lin, Z., Yan, J.: Hierarchy-aware label semantics matching network for hierarchical text classification. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 4370–4379. Association for Computational Linguistics, August 2021. https://doi.org/10.18653/v1/2021.acl-long.337. https://aclanthology.org/2021.acl-long.337
4. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: Proceedings of the 37th International Conference on Machine Learning, ICML 2020. JMLR.org (2020)
5. Chen, Z., et al.: Graph propagation transformer for graph representation learning. In: Elkind, E. (ed.) Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23, pp. 3559–3567. International Joint Conferences on Artificial Intelligence Organization (2023). https://doi.org/10.24963/ijcai.2023/396
6. Cunha, W., et al.: On the cost-effectiveness of neural and non-neural approaches and representations for text classification: a comprehensive comparative study. Inf. Process. Manag. **58**(3), 102481 (2021). https://doi.org/10.1016/j.ipm.2020.102481. https://www.sciencedirect.com/science/article/pii/S0306457320309705
7. Deng, Z., Peng, H., He, D., Li, J., Yu, P.: HTCInfoMax: a global model for hierarchical text classification via information maximization. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 3259–3265. Association for Computational Linguistics, June 2021. https://doi.org/10.18653/v1/2021.naacl-main.260. https://aclanthology.org/2021.naacl-main.260
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota, June 2019. https://doi.org/10.18653/v1/N19-1423. https://aclanthology.org/N19-1423
9. Dumais, S., Chen, H.: Hierarchical classification of web content. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2000, pp. 256–263. Association for Computing Machinery, New York, NY, USA (2000). https://doi.org/10.1145/345508.345593
10. Gopal, S., Yang, Y.: Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, pp. 257–265. Association for Computing Machinery, New York, NY, USA (2013). https://doi.org/10.1145/2487575.2487644
11. Huang, W., et al.: Hierarchical multi-label text classification: an attention-based recurrent network approach. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, pp. 1051–1060. Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3357384.3357885

12. Hull, D.: Using statistical testing in the evaluation of retrieval experiments. In: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1993, pp. 329–338. Association for Computing Machinery, New York, NY, USA (1993). https://doi.org/10.1145/160688.160758
13. Kowsari, K., Brown, D.E., Heidarysafa, M., Jafari Meimandi, K., Gerber, M.S., Barnes, L.E.: Hdltex: hierarchical deep learning for text classification. In: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 364–371 (2017). https://doi.org/10.1109/ICMLA.2017.0-134
14. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: RCV1: a new benchmark collection for text categorization research. J. Mach. Learn. Res. **5**, 361–397 (2004)
15. Liu, H., Huang, X., Liu, X.: Improve label embedding quality through global sensitive gat for hierarchical text classification. Expert Syst. Appl. **238**, 122267 (2024). https://doi.org/10.1016/j.eswa.2023.122267. https://www.sciencedirect.com/science/article/pii/S0957417423027690
16. Mao, Y., Tian, J., Han, J., Ren, X.: Hierarchical text classification with reinforced label assignment. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 445–455. Association for Computational Linguistics, Hong Kong, China, November 2019. https://doi.org/10.18653/v1/D19-1042. https://aclanthology.org/D19-1042
17. Ning, B., Zhao, D., Zhang, X., Wang, C., Song, S.: UMP-MG: a uni-directed message-passing multi-label generation model for hierarchical text classification. Data Sci. Eng. **8**, 1–12 (2023). https://doi.org/10.1007/s41019-023-00210-1
18. Peng, S., You, R., Wang, H., Zhai, C., Mamitsuka, H., Zhu, S.: DeepMeSH: deep semantic representation for improving large-scale MeSH indexing. Bioinformatics **32**(12), i70–i79 (2016). https://doi.org/10.1093/bioinformatics/btw294
19. Sandhaus, E.: The New York Times Annotated Corpus - Linguistic Data Consortium. The New York Times (2008). https://catalog.ldc.upenn.edu/LDC2008T19
20. Shen, J., Qiu, W., Meng, Y., Shang, J., Ren, X., Han, J.: TaxoClass: hierarchical multi-label text classification using only class names. In: Toutanova, K. (eds.) (eds.) Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4239–4249. Association for Computational Linguistics, June 2021. https://doi.org/10.18653/v1/2021.naacl-main.335. https://aclanthology.org/2021.naacl-main.335
21. Wang, Z., Wang, P., Huang, L., Sun, X., Wang, H.: Incorporating hierarchy into text encoder: a contrastive learning approach for hierarchical text classification. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 7109–7119. Association for Computational Linguistics, Dublin, Ireland, May 2022. https://doi.org/10.18653/v1/2022.acl-long.491. https://aclanthology.org/2022.acl-long.491
22. Wolf, T., et al.: Transformers: state-of-the-art natural language processing. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45. Association for Computational Linguistics, October 2020. https://doi.org/10.18653/v1/2020.emnlp-demos.6. https://aclanthology.org/2020.emnlp-demos.6

23. Wu, J., Xiong, W., Wang, W.Y.: Learning to learn and predict: a meta-learning approach for multi-label classification. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 4354–4364. Association for Computational Linguistics, Hong Kong, China, November 2019. https://doi.org/10.18653/v1/D19-1444. https://aclanthology.org/D19-1444
24. Yang, P., Sun, X., Li, W., Ma, S., Wu, W., Wang, H.: SGM: sequence generation model for multi-label classification. In: Bender, E.M., Derczynski, L., Isabelle, P. (eds.) Proceedings of the 27th International Conference on Computational Linguistics, pp. 3915–3926. Association for Computational Linguistics, Santa Fe, New Mexico, USA, August 2018. https://aclanthology.org/C18-1330
25. Zhao, F., Wu, Z., He, L., Dai, X.Y.: Label-correction capsule network for hierarchical text classification. IEEE/ACM Trans. Audio Speech Lang. Process. **31**, 2158–2168 (2023). https://doi.org/10.1109/TASLP.2023.3282099
26. Zhao, R., Wei, X., Ding, C., Chen, Y.: Hierarchical multi-label text classification: self-adaption semantic awareness network integrating text topic and label level information. In: Qiu, H., Zhang, C., Fei, Z., Qiu, M., Kung, S.-Y. (eds.) KSEM 2021. LNCS (LNAI), vol. 12816, pp. 406–418. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-82147-0_33
27. Zhou, J., et al.: Hierarchy-aware global model for hierarchical text classification. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1106–1117. Association for Computational Linguistics, July 2020. https://doi.org/10.18653/v1/2020.acl-main.104. https://aclanthology.org/2020.acl-main.104

Secure Aggregation Is Not Private Against Membership Inference Attacks

Khac-Hoang Ngo[1(✉)], Johan Östman[2], Giuseppe Durisi[1], and Alexandre Graell i Amat[1]

[1] Department of Electrical Engineering, Chalmers University of Technology, Gothenburg, Sweden
{ngok,durisi,alexandre.graell}@chalmers.se
[2] AI Sweden, Gothenburg, Sweden
johan.ostman@ai.se

Abstract. Secure aggregation (SecAgg) is a commonly-used privacy-enhancing mechanism in federated learning, affording the server access only to the aggregate of model updates while safeguarding the confidentiality of individual updates. Despite widespread claims regarding SecAgg's privacy-preserving capabilities, a formal analysis of its privacy is lacking, making such presumptions unjustified. In this paper, we delve into the privacy implications of SecAgg by treating it as a local differential privacy (LDP) mechanism for each local update. We design a simple attack wherein an adversarial server seeks to discern which update vector a client submitted, out of two possible ones, in a single training round of federated learning under SecAgg. By conducting privacy auditing, we assess the success probability of this attack and quantify the LDP guarantees provided by SecAgg. Our numerical results unveil that, contrary to prevailing claims, SecAgg offers weak privacy against membership inference attacks even in a single training round. Indeed, it is difficult to hide a local update by adding other independent local updates when the updates are of high dimension. Our findings underscore the imperative for additional privacy-enhancing mechanisms, such as noise injection, in federated learning.

Keywords: Federated learning · Secure aggregation · Differential privacy · Membership inference

1 Introduction

Federated learning (FL) [27] allows multiple clients to collaboratively train a machine learning model. In each training round, the clients share their local model updates with a central server, which then aggregates them to improve the global model. Although raw data is not shared in the clear, vanilla FL is prone to model-inversion attacks [18] and membership-inference attacks [29]. To mitigate such attacks, secure aggregation (SecAgg) [9] has been proposed, where the clients jointly mask their local model updates so that only the aggregate is revealed to the server.

Many papers explicitly or implicitly assume that SecAgg provides strong privacy against honest-but-curious servers in a single round [16,19,30,31]. However, a formal analysis of the privacy offered by SecAgg is lacking, making this presumption unjustified. SecAgg has been combined with differential privacy (DP) [14] to ensure that the server only sees the aggregate of the noisy local updates [2,22]. However, in these works, the privacy analysis does not account for SecAgg. It remains unclear how much privacy SecAgg by itself provides for individual updates.

Main Contributions. We address the question: *how much privacy does SecAgg by itself guarantee for the local updates?* Specifically, we formally analyze the privacy of SecAgg against membership inference attacks wherein the server aims to distinguish, from two potential update vectors, the one a client submitted in a single training round of FL with SecAgg. Our approach consists in treating SecAgg as a local differential privacy (LDP) mechanism for each update, where the sum of the other clients' updates plays the role of a source of uncontrolled noise. We then characterize the privacy parameters (ϵ, δ) for SecAgg to satisfy (ϵ, δ)-LDP via the following steps.

- We show that, under some practical assumptions, as the client population grows, the sum of the clients' updates converges to a Gaussian vector (Theorem 1). We analyze the optimal privacy guarantee of the Gaussian mechanism with correlated noise (Theorem 2).
- We evaluate the optimal LDP parameters of SecAgg in some special cases (Theorem 3 and Corollary 1) and verify that these parameters are close to that of a Gaussian mechanism, even for a small number of clients (Fig. 1).
- Exploiting the similarity of SecAgg and a Gaussian mechanism, we audit the privacy of SecAgg. Specifically, we design a simple membership inference attack wherein the server regards SecAgg as a Gaussian mechanism with correlated noise. We then evaluate the achievable false negative rate (FNR) and false positive rate (FPR) of this attack and use these values to compute a lower bound on the smallest ϵ for SecAgg to satisfy (ϵ, δ)-LDP.

We apply our privacy auditing procedure to federated averaging for a classification problem on the ADULT dataset [5] and the EMNIST Digits dataset [11]. We show that both the FNR and FPR can be small simultaneously, and the audited (ϵ, δ) are high. Our results reveal that SecAgg provides weak privacy even for a single training round. Indeed, it is difficult to hide a local update by adding other independent local updates when the updates are of high dimension. Therefore, SecAgg cannot be used as a sole privacy-enhancing mechanism in FL.

2 Related Work

Secure Aggregation. Based on cryptographic multi-party computation, SecAgg ensures that the central server sees only the aggregate of the clients' local updates, while individual updates are kept confidential. This is achieved by letting the clients jointly add randomly sampled masks to their updates via secret sharing, such that when the masked updates are aggregated, the masks cancel out [6,9]. With SecAgg, a client's update is obfuscated by many other clients' updates. However, the level of privacy provided by SecAgg lacks a formal analysis. In [16], this level was measured by the mutual information between a local update and the aggregated update. However, mutual information only measures the average privacy leakage and does not capture the threat to the most vulnerable data points. Furthermore, the bound provided in [16] is not explicit, i.e., not computable.

Differential Privacy. DP is a rigorous privacy measure that quantifies the ability of an adversary to guess which dataset, out of two neighboring ones, a model was trained on [14,15]. DP is typically achieved by adding noise to the model/gradients obtained from the dataset [1]. A variant of DP is LDP [13,24], where the noise is added to individual data points. When applied to achieve client-level privacy in FL, LDP lets the clients add noise to their updates before sending the updates to the server.

Privacy Attacks in FL with SecAgg. Model inversion attacks [18] and membership-inference attacks [29] have been shown to seriously jeopardize the integrity of FL. When SecAgg is employed, the server can perform disaggregation attacks to learn the individual data. A malicious server performs active attacks by suppressing the updates of non-target clients [8,17]. For an honest-but-curious server, existing passive attacks require that the server leverages the aggregated model across many rounds [25,26]. Differently from these works, we consider a *passive* attack based only on the observation in a *single round*.

3 Preliminaries

We denote random quantities with lowercase nonitalic letters, such as a scalar x and a vector $\mathbf{x}$. The only exception is the privacy-loss random variable (PLRV) L, which is in uppercase. Deterministic quantities are denoted with italic letters, such as a scalar x and a vector $\boldsymbol{x}$. We denote the multidimensional normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ by $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and its probability density function (PDF) evaluated at $\boldsymbol{x}$ by $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. We denote by $\Phi(x)$ the cummulative distribution function (CDF) of the standard normal distribution $\mathcal{N}(0,1)$, i.e., $\Phi(x) \triangleq \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-u^2/2} \mathrm{d}u$. We denote by $[m:n]$ the set of integers from m to n; $[n] \triangleq [1:n]$; $(\cdot)^+ \triangleq \max\{0, \cdot\}$. Furthermore, $\mathbb{1}\{\cdot\}$ denotes the indicator function, and $f(n) = o(g(n))$ means that $f(n)/g(n) \to 0$ as $n \to \infty$.

Let ϕ be a decision rule of a hypothesis test between $\{H\colon$ the underlying distribution is $P\}$ and $\{H'\colon$ the underlying distribution is $Q\}$. Specifically, ϕ returns 0 and 1 in favor of H and H', respectively. A false positive (resp. false negative) occurs when H (resp. H') is true but rejected. The FPR and FNR of the test are given by $\alpha_\phi \triangleq \mathbb{E}_P[\phi]$ and $\beta_\phi \triangleq 1 - \mathbb{E}_Q[\phi]$, respectively.

Definition 1 (Trade-off function). *The trade-off function $T_{P,Q}(\cdot)\colon [0,1] \to [0,1]$ is the map from the FPR to the corresponding minimum FNR of the test between P and Q, i.e., $T_{P,Q}(\alpha) \triangleq \inf_{\phi\colon\, \alpha_\phi \le \alpha} \beta_\phi$, $\alpha \in [0,1]$.*

We also write the trade-off function for the distributions of x and y as $T_{\mathrm{x},\mathrm{y}}(\cdot)$. We next state the definition of LDP.

Definition 2 (LDP [13,24]). *A mechanism M satisfies (ϵ,δ)-LDP if and only if, for every pair of data points $(\boldsymbol{x},\boldsymbol{x}')$ and for every measurable set $\mathcal{E}$, we have $\mathbb{P}[M(\boldsymbol{x}) \in \mathcal{E}] \le e^{\epsilon}\mathbb{P}[M(\boldsymbol{x}') \in \mathcal{E}] + \delta$.*

For a mechanism M, we define the optimal LDP curve $\delta_M(\epsilon)$ as the function that returns the smallest δ for which M satisfies (ϵ,δ)-LDP. We next define a variant of LDP that is built upon the trade-off function in a similar manner as f-DP [12, Def. 3].

Definition 3 (f-LDP). *For a function f, a mechanism M satisfies f-LDP if for every pair of data points $(\boldsymbol{x},\boldsymbol{x}')$, we have that $T_{M(\boldsymbol{x}),M(\boldsymbol{x}')}(\alpha) \ge f(\alpha), \forall \alpha \in [0,1]$.*

For a mechanism M, we define the optimal f-LDP curve $f_M(\cdot)$ as the upper envelope of all functions f such that M satisfies f-LDP. In the paper, we regard SecAgg as a LDP mechanism and provide bounds on both its optimal LDP curve and optimal f-LDP curve.

4 Privacy Analysis of Secure Aggregation

We consider a FL scenario with $n+1$ clients and a central server. The model update of client i can be represented as a vector $\mathbf{x}_i \in \mathbb{R}^d$. Under SecAgg, the server only learns the aggregate model update $\bar{\mathbf{x}} = \sum_{i=0}^{n} \mathbf{x}_i$, while the individual updates $\{\mathbf{x}_i\}_{i=0}^{n}$ remain confidential.

4.1 Threat Model

Server. The server is honest and follows the FL and SecAgg protocols. We assume that it observes the exact sum $\bar{\mathbf{x}}$. In practical SecAgg, the clients discretize their updates (using, e.g., randomized rounding) and the server obtains a modulo sum. These operations introduce perturbations that can improve privacy [34]. However, they do not capture the essence of SecAgg which is to use the updates of other clients to obfuscate an individual update. The rounding and modulo operations can be applied even in a setting without SecAgg. We ignore the perturbation caused by these operations to focus on the privacy obtainable by using the updates of other clients to obfuscate an update.

Clients. The clients are also honest. Client i computes the local model update $\mathbf{x}_i$ from the global model in the previous round and its local dataset. We also assume that each vector $\mathbf{x}_i$, $i \in [0:n]$, has correlated entries since these entries together describe a model, and that the vectors are mutually independent. The latter assumption holds in the first training round if the clients have independent local data. Furthermore, the independence assumption results in the best-case scenario for privacy as, if the vectors are dependent, the sum reveals more information about each vector. Therefore, the privacy level for the case of independent $\{\mathbf{x}_i\}_{i=0}^n$ acts as an upper bound on the privacy for the case of dependent vectors. So if privacy does not hold for independent data, it will also not hold when there is dependence.

Privacy Threat. The server is curious. It seeks to infer the membership of a targeted client, say client 0, from the aggregate model updates $\bar{\mathbf{x}}$. We consider a membership inference game [33] where: i) a challenger selects a pair of possible local updates $(\boldsymbol{x}_0, \boldsymbol{x}'_0)$ of client 0, one of which is used in the aggregation, and sends this pair to the server, ii) the server observes $\bar{\mathbf{x}}$ and guesses if $\boldsymbol{x}_0$ or $\boldsymbol{x}'_0$ was submitted by client 0. Note that this attack can be an entry point for the server to further infer the data of client 0. Our goal is to quantify the capability of SecAgg in mitigating this attack.

4.2 SecAgg as a Noiseless LDP Mechanism

Hereafter, we focus on client 0; the analysis for other clients follows similarly. Our key idea is to view SecAgg through the lens of noiseless DP [7], where the contribution of other clients can be seen as noise and no external (controlled) noise is added. More precisely, for client 0, SecAgg plays the role of the mechanism

$$M(\mathbf{x}_0) = \mathbf{x}_0 + \mathbf{y}, \tag{1}$$

where $\mathbf{y} = \sum_{i=1}^n \mathbf{x}_i$ is a source of uncontrolled noise.

The aforementioned membership inference game can be cast as follows: given $M(\mathbf{x}_0)$, the server guesses whether it came from $P_{M(\mathbf{x}_0)|\mathbf{x}_0=\boldsymbol{x}_0}$ or $P_{M(\mathbf{x}_0)|\mathbf{x}_0=\boldsymbol{x}'_0}$ for the worst-case pair $(\boldsymbol{x}_0, \boldsymbol{x}'_0)$. This game is closely related to the LDP framework. First, the tradeoff between the FPR and FNR of the server's guesses is captured by the f-LDP guarantee of M. Second, as M achieves a stronger (ϵ, δ)-LDP guarantee, the distributions $P_{M(\mathbf{x}_0)|\mathbf{x}_0=\boldsymbol{x}_0}$ and $P_{M(\mathbf{x}_0)|\mathbf{x}_0=\boldsymbol{x}'_0}$ become more similar, and the hypothesis test between them becomes harder. Therefore, we shall address the following question: how much LDP or f-LDP does SecAgg guarantee for client 0? Specifically, we shall establish bounds on the optimal LDP curve $\delta_M(\epsilon)$ and optimal f-LDP curve $f_M(\cdot)$ of the mechanism M.

4.3 Asymptotic Privacy Guarantee

Let us first focus on the large-n regime. The following asymptotic analysis will be used as inspiration for our privacy auditing procedure to establish a lower bound on the LDP curve in Sect. 4.5.

We assume that the ℓ_2 norm of the vectors $\{\mathbf{x}_i\}_{i=1}^n$ scales as $o(\sqrt{n})$, which holds if, e.g., d is fixed. In this case, $\mathbf{y}$ converges to a Gaussian vector when $n \to \infty$, as stated in the next theorem.

Theorem 1 (Asymptotic noise distribution). *Assume that* $\{\mathbf{x}_i\}_{i=1}^n$ *are independent,* $\|\mathbf{x}_i\|_2 = o(\sqrt{n})$ *for* $i \in [n]$, *and* $\frac{1}{n}\sum_{i=1}^n \mathrm{Cov}[\mathbf{x}_i] \to \boldsymbol{\Sigma}$ *as* $n \to \infty$. *Then* $\frac{1}{\sqrt{n}}\left(\sum_{i=1}^n \mathbf{x}_i - \mathbb{E}\left[\sum_{i=1}^n \mathbf{x}_i\right]\right)$ *converges in distribution to* $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ *as* $n \to \infty$.

Proof. Theorem 1 follows by applying the multivariate Lindeberg-Feller central limit Theorem [32, Prop. 2.27] to the triangular array $\left\{\frac{\mathbf{x}_i}{\sqrt{n}}\right\}_{n,i}$, upon verifying the Lindeberg condition $\lim_{n\to\infty} \sum_{i=1}^n \mathbb{E}\left[\frac{\|\mathbf{x}_i\|_2^2}{n} \mathbb{1}\left\{\frac{\|\mathbf{x}_i\|_2}{\sqrt{n}} > \varepsilon\right\}\right] = 0, \forall \varepsilon > 0$. Since $\|\mathbf{x}_i\|_2 = o(\sqrt{n})$, i.e., $\|\mathbf{x}_i\|_2/\sqrt{n} \to 0$ as $n \to \infty$, this condition indeed holds. $\square$

Theorem 1 implies that, when n is large, under the presented assumptions, the mechanism $\widetilde{M}(\boldsymbol{x}_0) = M(\boldsymbol{x}_0) - \mathbb{E}[\mathbf{y}]$ behaves like a Gaussian mechanism with noise distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\mathbf{y})$, where $\boldsymbol{\Sigma}_\mathbf{y}$ is the covariance matrix of $\mathbf{y}$. Furthermore, since the map from M to $\widetilde{M}$ is simply a shift by a fixed vector $\mathbb{E}[\mathbf{y}]$, i.e., it is a bijection, we have from the post-processing property[1] that the optimal LDP curve of M is the same as that of $\widetilde{M}$.

We now provide privacy guarantees for a Gaussian mechanism with correlated noise, to capture the correlation between the entries of the vectors $\mathbf{x}_i$. The next theorem, proved in Appendix A.1, is an extension of the optimal privacy curve of the uncorrelated Gaussian mechanism [4, Theorem 8].

Theorem 2 (Correlated Gaussian mechanism). *Consider the mechanism* $G(\mathbf{x}) = \mathbf{x} + \mathbf{y}$ *where* $\mathbf{x}$ *belongs to a set* $\mathcal{S}_d \subset \mathbb{R}^d$, *and* $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\mathbf{y})$. *The optimal LDP curve of* G *is*

$$\delta_G(\epsilon) = \Phi\Big(\frac{\Delta}{2} - \frac{\epsilon}{\Delta}\Big) - e^{\epsilon}\Phi\Big(-\frac{\Delta}{2} - \frac{\epsilon}{\Delta}\Big) \tag{2}$$

where $\Delta = \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{S}_d} \Delta_{\boldsymbol{x},\boldsymbol{x}'}$ *with*

$$\Delta_{\boldsymbol{x},\boldsymbol{x}'} \triangleq \sqrt{(\boldsymbol{x} - \boldsymbol{x}')^\intercal \boldsymbol{\Sigma}_\mathbf{y}^{-1} (\boldsymbol{x} - \boldsymbol{x}')}. \tag{3}$$

In Sect. 4.4, we shall verify the similarity between the privacy of SecAgg and that of the Gaussian mechanism G via numerical examples.

Parameter Δ is the maximum Euclidean distance between a pair of input vectors transformed by matrix $\boldsymbol{\Sigma}^{-1/2}$ (similar to the whitening transformation). It plays the same role as the ratio of the sensitivity and the noise standard deviation in the case of uncorrelated noise [4]. We remark that the privacy guarantee of G is weakened as Δ increases: for a given ϵ, $\delta_G(\epsilon)$ increases with Δ. To achieve small ϵ and δ, we need Δ to be small. The impact of Δ can also be

[1] If a mechanism M satisfies (ϵ, δ)-LDP, then so does $h \circ M$ for a mapping h that is independent of M. The proof of this result is similar to the proof of the post-processing property of DP [15, Prop. 2.1].

seen via the hypothesis test associated to the considered membership inference game. Consider an adversary that observes an output $\boldsymbol{z}$ of G and tests between $\{H : \boldsymbol{z}$ came from $P_{G(\mathbf{x})|\mathbf{x}=\boldsymbol{x}}\}$ and $\{H' : \boldsymbol{z}$ came from $P_{G(\mathbf{x})|\mathbf{x}=\boldsymbol{x}'}\}$. This is effectively a test between $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\Sigma}_{\mathbf{y}})$ and $\mathcal{N}(\boldsymbol{x}', \boldsymbol{\Sigma}_{\mathbf{y}})$. The trade-off function for this test is stated in the following proposition, which is proved in Appendix A.3.

Proposition 1. $T_{\mathcal{N}(\boldsymbol{x},\boldsymbol{\Sigma}_{\mathbf{y}}),\mathcal{N}(\boldsymbol{x}',\boldsymbol{\Sigma}_{\mathbf{y}})}(\alpha) = \Phi\big(\Phi^{-1}(1-\alpha) - \Delta_{\boldsymbol{x},\boldsymbol{x}'}\big)$, $\alpha \in [0,1]$.

The trade-off function decreases with $\Delta_{\boldsymbol{x},\boldsymbol{x}'}$. A large $\Delta_{\boldsymbol{x},\boldsymbol{x}'}$ facilitates the distinguishability of the pair $(\boldsymbol{x}, \boldsymbol{x}')$, and thus weakens the privacy guarantee. Furthermore, the worst-case pair $(\boldsymbol{x}, \boldsymbol{x}')$ that minimizes the trade-off function is given by the maximizer of $\Delta_{\boldsymbol{x},\boldsymbol{x}'}$. It follows that the optimal f-LDP curve of the Gaussian mechanism G is $f_G(\alpha) = \Phi\big(\Phi^{-1}(1-\alpha) - \Delta\big)$, $\alpha \in [0,1]$.

4.4 Upper Bounding $\boldsymbol{\delta_M(\epsilon)}$ via Dominating Pairs of Distributions

We now upper-bound the optimal LDP curve of M in (1) for finite n. We shall then consider the case in which the upper bound is tight and verify the convergence of SecAgg to a Gaussian mechanism.

We define the hockey-stick divergence with parameter α between two probability measures P and Q as $\mathsf{E}_\alpha(P\|Q) = \sup_{\mathcal{E}}(P(\mathcal{E}) - \alpha Q(\mathcal{E}))$. We also write the hockey-stick divergence between the distributions of x and y as $\mathsf{E}_\alpha(\mathrm{x}\|\mathrm{y})$. The condition for LDP in Definition 2 is equivalent to $\sup_{\boldsymbol{x}\neq\boldsymbol{x}'} \mathsf{E}_{e^\epsilon}(M(\boldsymbol{x})\|M(\boldsymbol{x}')) \le \delta$. Therefore, the optimal LDP curve of mechanism M is given by

$$\delta_M(\epsilon) = \sup_{\boldsymbol{x}\neq\boldsymbol{x}'} \mathsf{E}_{e^\epsilon}(M(\boldsymbol{x})\|M(\boldsymbol{x}'))\,. \tag{4}$$

A pair of measures (P, Q) is called a dominating pair of distributions for M if

$$\sup_{\boldsymbol{x}\neq\boldsymbol{x}'} \mathsf{E}_{e^\epsilon}(M(\boldsymbol{x})\|M(\boldsymbol{x}')) \le \mathsf{E}_{e^\epsilon}(P\|Q), \quad \forall \epsilon \ge 0\,. \tag{5}$$

If equality is achieved in (5) for every $\epsilon \ge 0$, then (P, Q) is said to be a tightly dominating pair of distributions for M. For each dominating pair (P, Q), we associate a privacy-loss random variable $\mathrm{L} \triangleq \ln \frac{\mathrm{d}P}{\mathrm{d}Q}(\mathbf{y})$ with $\mathbf{y} \sim P$, where $\frac{\mathrm{d}P}{\mathrm{d}Q}$ is the Radon-Nikodym derivative. We have that

$$\mathsf{E}_{e^\epsilon}(P\|Q) = \mathbb{E}\left[(1 - e^{\epsilon-\mathrm{L}})^+\right] \triangleq \delta_{\mathrm{L}}(\epsilon)\,. \tag{6}$$

It follows readily that $\delta_{\mathrm{L}}(\epsilon)$ is an upper bound on the optimal LDP curve $\delta_M(\epsilon)$.

Without a known distribution of $\mathbf{y}$, it is challenging to characterize a dominating pair of distributions for the mechanism M in (1). In the next theorem, proved in Appendix B, we make some assumptions on $P_{\mathbf{y}}$ to enable such characterization.

Theorem 3 (Dominating pair of distributions). *Let* $\mathbf{x}_0 = (\mathrm{x}_{01}, \mathrm{x}_{02}, \dots, \mathrm{x}_{0d})$ *and assume that* $\underline{r}_j \le \mathrm{x}_{0j} \le \overline{r}_j$, $j \in [d]$. *Assume further that* $\mathbf{y}$ *has independent entries, i.e.,* $P_{\mathbf{y}} = P_{\mathrm{y}_1} \times \dots \times P_{\mathrm{y}_d}$, *and that the marginal probabilities* $\{P_{\mathrm{y}_j}\}$

are log-concave and symmetric.[2] *Then, a dominating pair of distributions for the mechanism $M(\mathbf{x}_0)$ in (1) is given by $(P_{\underline{r}_1+\mathrm{y}_1}\times\cdots\times P_{\underline{r}_d+\mathrm{y}_d}, P_{\overline{r}_1+\mathrm{y}_1}\times\cdots\times P_{\overline{r}_d+\mathrm{y}_d})$.*

The family of log-concave distributions includes the typical noise distributions in DP, namely, the Gaussian and Laplace distributions, as well as many other common distributions, e.g., the exponential and uniform distributions [3]. If each vector $\mathbf{x}_i, i \in [n]$, has independent entries following a log-concave distribution, then so does the sum $\mathbf{y} = \sum_{i=1}^n \mathbf{x}_i$, because log-concavity is closed under convolutions [28]. Under the presented assumptions, Theorem 3 allows us to characterize an upper bound on the LDP curve of M as $\delta_{\mathrm{L}}(\epsilon)$ with $\mathrm{L} = \sum_{j=1}^d \ln \frac{P_{\underline{r}_j+\mathrm{y}_j}(\mathrm{z}_j)}{P_{\overline{r}_j+\mathrm{y}_j}(\mathrm{z}_j)}$ where $\mathrm{z}_j \sim P_{\underline{r}_j+\mathrm{y}_j}$, $j \in [d]$.

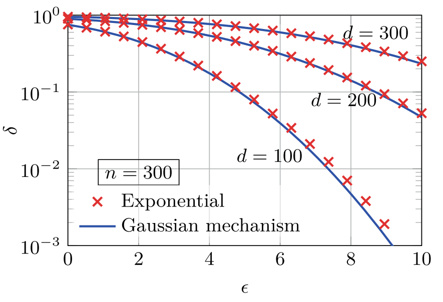
Corollary 1. *If the support of $\mathbf{x}_0$ contains $(\underline{r}_1,\dots,\underline{r}_d)$ and $(\overline{r}_1,\dots,\overline{r}_d)$, the dominating pair of distributions in Theorem 3 becomes tight, and the resulting upper bound $\delta_{\mathrm{L}}(\epsilon)$ is the optimal LDP curve.*

We now use Corollary 1 to evaluate the optimal LDP curve of mechanism M in (1) when each $\mathbf{x}_i$ has independent entries. We aim to verify the convergence of SecAgg to a Gaussian mechanism implied by Theorem 1 and understand how the LDP curve depends on the model size d. We consider two cases. In the first case, the entries follow the exponential distribution with parameter 1, and thus $\mathbf{y}$ has independent entries following the Gamma distribution with shape n and scale 1. For convenience, we further assume that $\mathbf{x}_0$ is truncated such that $0 \le \mathrm{x}_{0j} \le 4$, $j \in [d]$. In the second case, the entries are uniformly distributed in $[-1/2, 1/2]$, and thus $\mathbf{y}$ has independent entries following the shifted Irwin-Hall distribution with PDF $p_{\mathrm{y}_i}(y) = \frac{1}{(n-1)!}\sum_{k=0}^n (-1)^k \binom{n}{k} \left[(y+n/2-k)^+\right]^{n-1}$. Both cases satisfy the conditions of Corollary 1. We can therefore obtain the optimal LDP curves and depict them in Fig. 1. We also show the optimal LDP curve of the Gaussian mechanism G with the same noise covariance matrix $\boldsymbol{\Sigma}_{\mathbf{y}}$. We see that the optimal LDP curve of M is indeed close to that of G, even for a small value of n in the second case. Furthermore, although Theorem 1 assumes a fixed d and $n \to \infty$, Fig. 1 suggests that M behaves similarly to a Gaussian mechanism even for large d. Remarkably, for a given ϵ, the parameter δ increases rapidly with d, indicating that the privacy of SecAgg is weak for high-dimensional models.

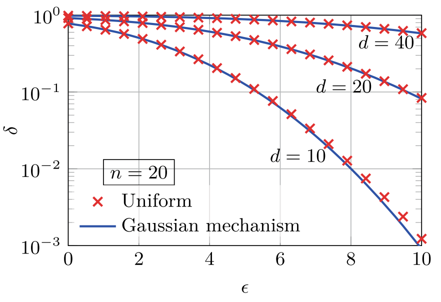
4.5 Lower Bounding $\delta_M(\epsilon)$ and Upper Bounding $f_M(\cdot)$ via Privacy Auditing

In practical FL, the updates typically have a distribution that does not satisfy the conditions of Theorem 3 and is not known in closed form. Therefore, we now establish a numerical routine to compute a lower bound on the optimal LDP curve and an upper bound on the optimal f-LDP curve of M. The proposed

[2] P_{y_j} is symmetric if there exists a y^* such that $P_{\mathrm{y}_j}(\mathcal{A}+y^*) = P_{\mathrm{y}_j}(-\mathcal{A}+y^*)$ for every subset $\mathcal{A}$ of the support of y_j. Here, $-\mathcal{A} \triangleq \{-y\colon y \in \mathcal{A}\}$.



(a) $\mathbf{x}_i$, $i \in [n]$, has independent entries exponentially distributed with parameter 1; $0 \leq \mathrm{x}_{0j} \leq 4$, $j \in [d]$. For the Gaussian mechanism, $\mathcal{S}_d = [0, 4]^d$ and $\boldsymbol{\Sigma}_{\mathbf{y}} = n\boldsymbol{I}_d$.



(b) $\mathbf{x}_i$, $i \in [n]$, has independent entries uniformly distributed in $[-1/2, 1/2]$; $-1/2 \leq \mathrm{x}_{0j} \leq 1/2$, $j \in [d]$. For the Gaussian mechanism, $\mathcal{S}_d = [-1/2, 1/2]^d$ and $\boldsymbol{\Sigma}_{\mathbf{y}} = \frac{n}{12}\boldsymbol{I}_d$.

Fig. 1. The optimal LDP curve of M in (1) where each $\mathbf{x}_i$ has independent entries, compared with the Gaussian mechanism G with the same noise covariance matrix.

numerical routine exploits the similarity of SecAgg and a Gaussian mechanism as discussed in Sects. 4.3 and 4.4. The bounds are based on the following result.

Proposition 2 (LDP via the trade-off function). *A mechanism M satisfies (ϵ, δ)-LDP if and only if for every $\alpha \in [0, 1]$,*

$$\epsilon \geq \ln \max \left\{ \frac{1 - \delta - \alpha}{\inf_{\boldsymbol{x} \neq \boldsymbol{x}'} T_{M(\boldsymbol{x}), M(\boldsymbol{x}')}(\alpha)}, \frac{1 - \delta - \inf_{\boldsymbol{x} \neq \boldsymbol{x}'} T_{M(\boldsymbol{x}), M(\boldsymbol{x}')}(\alpha)}{\alpha} \right\}. \quad (7)$$

The proof of Proposition 2 follows from similar arguments for DP in [23, Thm. 2.1]. This proposition implies that, if a pair (FPR, FNR) is achievable for some decision rule ϕ between the distributions of $M(\boldsymbol{x})$ and $M(\boldsymbol{x}')$ for some $(\boldsymbol{x}, \boldsymbol{x}')$, then the mechanism does not satisfy (ϵ, δ)-LDP for $\delta \in [0, 1]$ and

$$\epsilon < \ln \max \left\{ \frac{1 - \delta - \mathrm{FPR}}{\mathrm{FNR}}, \frac{1 - \delta - \mathrm{FNR}}{\mathrm{FPR}} \right\}. \quad (8)$$

This gives a lower bound on the optimal LDP curve of M. Furthermore, it follows readily from Definition 3 that, for an achievable pair (FPR, FNR) of the mentioned test, the mechanism does not satisfy f-LDP for any trade-off function f such that $f(\mathrm{FPR}) < \mathrm{FNR}$. Therefore, a collection of achievable pairs (FPR, FNR) constitutes an upper bound on the optimal f-LDP curve of M.

We shall use this result to perform *privacy auditing* [21,33]. Specifically, following the defined membership inference game (see privacy threat in Sect. 4.1), we conduct a hypothesis test between $\{H\colon \boldsymbol{z}$ came from $P_{M(\mathbf{x}_0)|\mathbf{x}_0 = \boldsymbol{x}_0}\}$ and $\{H'\colon \boldsymbol{z}$ came from $P_{M(\mathbf{x}_0)|\mathbf{x}_0 = \boldsymbol{x}_0'}\}$ for a given output $\boldsymbol{z}$ of M. That is, we select a pair $(\boldsymbol{x}_0, \boldsymbol{x}_0')$ for the challenger and a decision rule ϕ for the server. We then evaluate the achievable pair $(\mathrm{FPR}, \mathrm{FNR})$, and obtain therefrom a lower bound on the optimal LDP curve and an upper bound on the optimal f-LDP curve of M. To design the attack, we draw inspiration from the asymptotic analysis in Sect. 4.3 as follows.

We consider the likelihood-ratio test, i.e., the test rejects H if

$$\ln \frac{P_{M(\mathbf{x}_0)\,|\,\mathbf{x}_0}(\boldsymbol{z}\,|\,\boldsymbol{x}_0)}{P_{M(\mathbf{x}_0)\,|\,\mathbf{x}_0}(\boldsymbol{z}\,|\,\boldsymbol{x}_0')} = \ln \frac{P_{\mathbf{y}}(\boldsymbol{z}-\boldsymbol{x}_0)}{P_{\mathbf{y}}(\boldsymbol{z}-\boldsymbol{x}_0')} \le \theta \tag{9}$$

for a given threshold θ. We choose the input pair $(\boldsymbol{x}_0, \boldsymbol{x}_0')$ as the worst-case pair, i.e., $(\boldsymbol{x}_0, \boldsymbol{x}_0') = \arg\min_{\boldsymbol{x},\boldsymbol{x}'} T_{M(\boldsymbol{x}),M(\boldsymbol{x}')}(\alpha),\ \forall \alpha \in [0,1]$. However, the trade-off function is not known in closed-form in general, and thus finding the worst-case pair is challenging. Motivated by Theorem 1, we treat $\mathbf{y}$ as a Gaussian vector with the same mean $\boldsymbol{\mu}_{\mathbf{y}}$ and covariance $\boldsymbol{\Sigma}_{\mathbf{y}}$. We thus approximate the trade-off function $T_{M(\boldsymbol{x}),M(\boldsymbol{x}')}(\cdot)$ by $T_{\mathcal{N}(\boldsymbol{x},\boldsymbol{\Sigma}_{\mathbf{y}}),\mathcal{N}(\boldsymbol{x}',\boldsymbol{\Sigma}_{\mathbf{y}})}(\cdot)$, and choose $(\boldsymbol{x}_0, \boldsymbol{x}_0')$ as the minimizer of the latter. Using Proposition 1, we have that

$$(\boldsymbol{x}_0, \boldsymbol{x}_0') = \arg\max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}_0} \Delta_{\boldsymbol{x},\boldsymbol{x}'} \tag{10}$$

where $\mathcal{X}_0$ is the support of $\mathbf{x}_0$.

If the server does not know $P_{\mathbf{y}}(\boldsymbol{y})$ in closed form, we let it approximate $P_{\mathbf{y}}(\boldsymbol{y})$ as $\mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{y}})$. That is, the test rejects $\boldsymbol{x}_0$ if

$$\ln \frac{\mathcal{N}(\boldsymbol{z}-\boldsymbol{x}_0; \boldsymbol{\mu}_{\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{y}})}{\mathcal{N}(\boldsymbol{z}-\boldsymbol{x}_0'; \boldsymbol{\mu}_{\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{y}})} \le \theta\,. \tag{11}$$

Moreover, if the server does not know $\boldsymbol{\mu}_{\mathbf{y}}$ and $\boldsymbol{\Sigma}_{\mathbf{y}}$ but can generate samples from $P_{\mathbf{y}}$, we let it estimate $\boldsymbol{\mu}_{\mathbf{y}}$ and $\boldsymbol{\Sigma}_{\mathbf{y}}$ as the sample mean and sample covariance matrix, and use these estimates instead of the true values in (10) and (11).

We evaluate the FNR and FPR of the test via Monte-Carlo simulation. Specifically, we repeat the test N_{s} times and count the number of false negatives N_{FN} and the number false positives N_{FP}. We obtain a high-confidence upper bound on FNR using the Clopper-Pearson method [10] as $\overline{\mathrm{FNR}} = B(1-\gamma/2; N_{\mathrm{FN}}+1, N_{\mathrm{s}}-N_{\mathrm{FN}})$, where $B(x;a,b)$ is the quantile of the Beta distribution with shapes (a,b), and $1-\gamma$ is the confidence level. A high-confidence upper bound $\overline{\mathrm{FPR}}$ on FPR is obtained similarly. By varying the threshold θ, we obtain an empirical trade-off curve $\overline{\mathrm{FNR}}$ vs. $\overline{\mathrm{FPR}}$. This curve is an upper bound on the optimal f-LDP curve of SecAgg. For a given $\delta \in [0,1]$, we also compute a lower confidence bound on ϵ for SecAgg to satisfies (ϵ,δ)-LDP. Specifically, we use $\overline{\mathrm{FNR}}$ and $\overline{\mathrm{FPR}}$ in place of FNR and FPR in (8). Note that $\overline{\mathrm{FNR}}$ and $\overline{\mathrm{FPR}}$ are lower bounded by $B(1-\gamma/2; 1, N_{\mathrm{s}})$ even if $N_{\mathrm{FN}} = N_{\mathrm{FP}} = 0$. Therefore, the estimated ϵ is upper bounded by $\bar{\epsilon}_\delta \triangleq \ln \frac{1-\delta-B(1-\gamma/2;1,N_{\mathrm{s}})}{B(1-\gamma/2;1,N_{\mathrm{s}})}$. That is, it is impossible to audit an arbitrarily large ϵ with a finite number of trials.

5 Experiments and Discussion

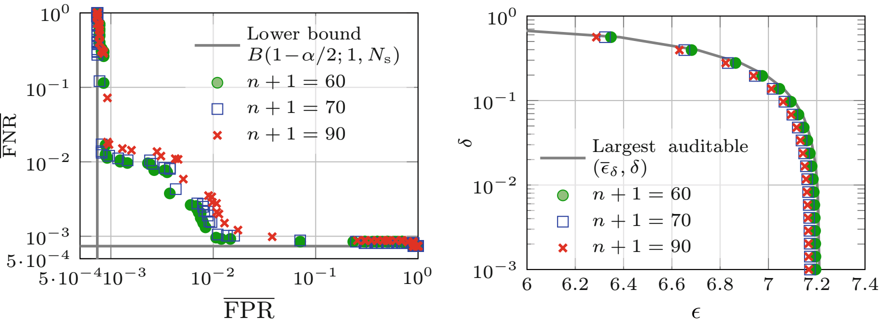
Experimental Setting. We consider federated averaging with $n_{\text{tot}} = 100$ clients out of which $n+1$ clients are randomly selected in each round. The experiments[3] are conducted for a classification problem on the ADULT dataset [5] and the EMNIST Digits dataset [11]. The ADULT dataset contains 30 162 entries with 104 features; the entries belong to two classes with 7508 positive labels and 22 654 negative labels. The EMNIST Digits dataset contains 280 000 images of size 28×28 of handwritten digits belonging to 10 balanced classes. We allocate the training samples between n_{tot} clients according to a latent Dirichlet allocation model with concentration parameter ω. Here, with $\omega \to \infty$, the training samples are distributed evenly and uniformly between the clients; with $\omega \to 0$, each client holds samples from only one class. We consider a single-layer neural network and use the cross-entropy loss and stochastic gradient descent with a learning rate of 0.01 and batch size of 64. The model size is $d = 210$ for ADULT and $d = 7850$ for EMNIST Digits.

We focus on the first round, containing one local epoch, of federated averaging and perform privacy auditing for a fixed initial model, which is known to the server. Note that performing an attack in the first round is the most challenging because, in later rounds, the server accumulates more observations. Let $\{\mathbf{x}_i\}_{i=0}^n$ be the local updates of the selected clients in the first round. The server does not know the distribution of $\{\mathbf{x}_i\}_{i=0}^n$ in closed form, but can sample from this distribution by simulating the learning scenario. Note that it is a common assumption in membership inference attacks that the adversary can sample from the population [33]. We let the server compute the sample mean $\hat{\boldsymbol{\mu}}_{\mathbf{x}}$ and sample covariance matrix $\widehat{\boldsymbol{\Sigma}}_{\mathbf{x}}$ from 25 000 samples of $\mathbf{x}_i$, then estimate the mean and covariance matrix of $\mathbf{y} = \sum_{i=1}^n \mathbf{x}_i$ as $\hat{\boldsymbol{\mu}}_{\mathbf{y}} = n\hat{\boldsymbol{\mu}}_{\mathbf{x}}$ and $\widehat{\boldsymbol{\Sigma}}_{\mathbf{y}} = n\widehat{\boldsymbol{\Sigma}}_{\mathbf{x}}$. The server then uses $\hat{\boldsymbol{\mu}}_{\mathbf{y}}$ and $\widehat{\boldsymbol{\Sigma}}_{\mathbf{y}}$ for privacy auditing, as described in Sect. 4.5. Following (10), we find the worst-case input pair $(\boldsymbol{x}_0, \boldsymbol{x}_0')$ by searching for the maximizer of $\Delta_{\boldsymbol{x}_0,\boldsymbol{x}_0'}$ among 5000 and 1000 samples of $\mathbf{x}_0$ for the ADULT and EMNIST Digits datasets, respectively. The Clopper-Pearson confidence level is $1-\gamma = 95\%$. For a given initial model, we consider $N_{\text{s}} = 5000$ trials with random data partition and batch selection. In the simulation results, we report the average of $\overline{\text{FPR}}$, $\overline{\text{FNR}}$, and the audited (ϵ, δ) over 10 and 5 initial models for the ADULT and EMNIST Digits datasets, respectively.

Homogeneous Data Partitioning. We first consider $\omega = \infty$. In Fig. 2(a), we show the trade-off between the estimated FNR and FPR for the ADULT dataset, achieved by varying the threshold θ in (11) for $n+1 \in \{60, 70, 90\}$ clients. Both the FNR and FPR can be as small as 0.005 simultaneously. Hence, the server can reliably distinguish the selected input pair, and the membership inference attack is successful. We note that a reference for the trade-off curve that represents different privacy levels is given in [12, Fig. 3]. There, the case with

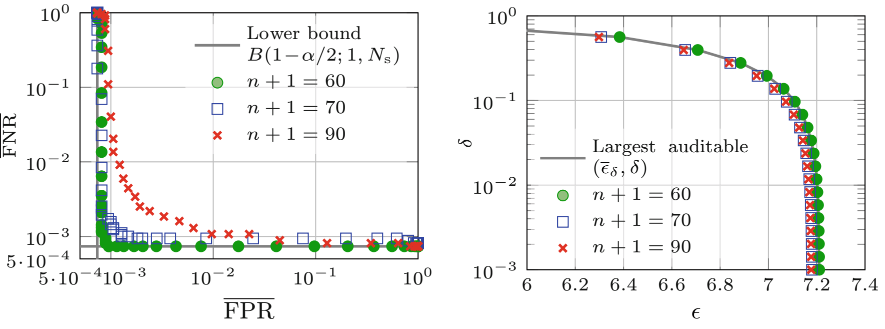
[3] The code is available at https://github.com/khachoang1412/SecAgg_not_private.

both FNR and FPR equal to 0.07 is already considered nonprivate. Comparing Fig. 2(a) with this reference, we conclude that SecAgg provides essentially no privacy for the ADULT dataset. Next, in Fig. 2(b), we show the average audited LDP curves for the ADULT dataset. We observe that the audited LDP curves are close to the largest auditable $(\bar{\epsilon}_\delta, \delta)$ with the considered N_s and γ. As ϵ increases, δ remains high until it drops due to the limit of the Clopper-Pearson method–even for $\epsilon = 7$, $\delta > 10^{-1}$. Furthermore, increasing n provides only a marginal privacy improvement. This shows that the privacy of SecAgg, viewed through the lens of LDP, is weak.



(a) FPR vs. FNR trade-off curve (b) LDP curve

Fig. 2. The audited FPR vs. FNR trade-off and LDP curve, averaged over 10 initial models, for SecAgg in federated learning on the ADULT dataset with homogeneous data partitioning. Here, $d = 210$.

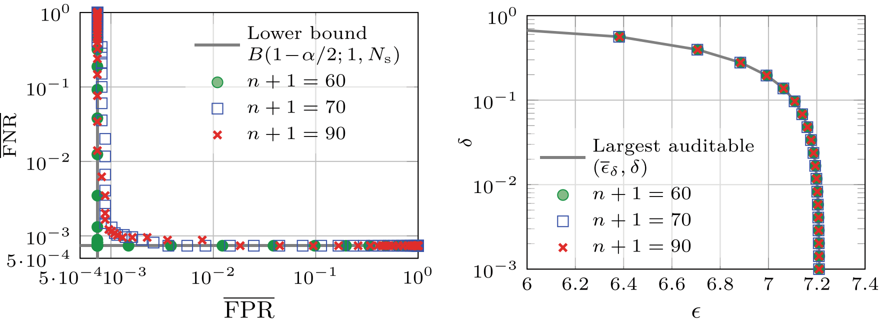


(a) FPR vs. FNR trade-off curve (b) LDP curve

Fig. 3. Audited FPR vs. FNR trade-off and LDP curves, averaged over 5 initial models, for federated learning with SecAgg on the EMNIST Digits dataset with homogeneous data partitioning. Here, $d = 7850$.

In Fig. 3, we show the FPR vs. FNR trade-off and the audited LDP curve for the EMNIST Digits dataset. Similar conclusions hold: SecAgg provides weak privacy. In this case, with a larger model size than the ADULT dataset, the adversary achieves even smaller FPR and FNR simultaneously.

Heterogeneous Data Partitioning. We next consider $\omega = 1$ and show the FPR vs. FNR trade-off and the audited LDP curve for the EMNIST Digits dataset in Fig. 4. In this case, the FPR and FNR are simultaneously reduced with respect to the homogeneous case, and the audited (ϵ, δ) coincide with the largest auditable values. This is because the worst-case pair $(\boldsymbol{x}_0, \boldsymbol{x}'_0)$ is better separated than in the homogeneous case and thus easier to distinguish.



(a) FPR vs. FNR trade-off curve (b) LDP curve

Fig. 4. Same as Fig. 3 but with heterogeneous data partitioning.

Discussion. We have seen that SecAgg is expected to perform like a correlated Gaussian mechanism (see Sect. 4.3). Why does SecAgg fail to prevent membership inference attacks, given that the Gaussian mechanism (with appropriate noise calibration) is known to be effective? We explain it as follows. We assume that the individual updates have entries with a bounded magnitude such that $\|\mathbf{x}_i\|_2 \leq r\sqrt{d}$, $i \in [0:n]$. For large n, we expect the mechanism M in (1) to have similar privacy guarantee as G in Theorem 2 with $\mathcal{S}_d = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq r\sqrt{d}\}$ and $\boldsymbol{\Sigma}_{\mathbf{y}}$ being the covariance matrix of $\sum_{i=1}^n \mathbf{x}_i$. In this case, $\Delta \geq 2\sqrt{d/n}$ (see Appendix A.2). A strong privacy guarantee requires Δ to be small, which implies that d/n must be small. This suggests that the privacy guarantee of SecAgg is weak if the vector dimension d is large compared to the number of clients n. This is, however, the case in most practical FL scenarios, as the model size is typically much larger than the number of clients. Note that reducing the ratio d/n by dividing the model into smaller chunks that are federated via SecAgg does not solve this issue, as the privacy loss composes over these chunks.

While with our results we have shown that SecAgg provides weak privacy for small models where d is in the order of 10^2–10^3, we remark that the privacy guarantee is expected to further deteriorate for larger models. This is supported by the rapid degradation of the privacy guarantee with d of the Gaussian mechanism (see Fig. 1) and the similarity of SecAgg and a Gaussian mechanism.

6 Conclusions

We analyzed the privacy of SecAgg through the lens of LDP. Via privacy auditing, we showed that membership inference attacks on the output of SecAgg succeed with high probability: adding independent local updates is not sufficient to hide a local update when the model is of high dimension. While this result may not be surprising, our work fills an important gap by providing a formal analysis of the privacy of SecAgg and challenges the prevailing claims of the privacy robustness of SecAgg. Hence, it underscores that additional privacy mechanisms, such as noise addition, are needed in federated learning.

Acknowledgments. This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 101022113. This work was also partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) and by the Swedish Research Council (VR) under grant 2020-03687. The authors would like to thank Onur Günlü and Balázs Pejó for fruitful discussions.

A Correlated Gaussian Mechanism

We present an analysis of the LDP guarantee of the Gaussian mechanism $G(\mathbf{x}) = \mathbf{x} + \mathbf{y}$, where $\mathbf{x}$ belongs to a subset $\mathcal{S}_d$ of $\mathbb{R}^d$, and $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{y}})$.

A.1 Optimal LDP Curve: Proof of Theorem 2

We extend [4, Thm. 8] to the case of correlated noise. First, for a mechanism M and a pair $\boldsymbol{x}, \boldsymbol{x}'$, we define the privacy loss function as $L_{M,\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{z}) \triangleq \ln \frac{P_{M(\boldsymbol{x})}(\boldsymbol{z})}{P_{M(\boldsymbol{x}')}(\boldsymbol{z})}$. The PLRV $\mathrm{L}_{M,\boldsymbol{x},\boldsymbol{x}'}$ is defined as the output of $L_{M,\boldsymbol{x},\boldsymbol{x}'}$ when the input follows $P_{M(\boldsymbol{x})}$. The PLRV can be used to express the optimal LDP curve as

$$\delta_M(\epsilon) = \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{S}_d} \left(\mathbb{P}\left[\mathrm{L}_{M,\boldsymbol{x},\boldsymbol{x}'} \geq \epsilon\right] - e^{\epsilon}\mathbb{P}\left[\mathrm{L}_{M,\boldsymbol{x}',\boldsymbol{x}} \leq -\epsilon\right] \right). \tag{12}$$

Equation (12) is obtained from a similar result for DP given in [4, Thm. 5], upon modifying the notion of neighboring datasets.

For the mechanism G, we have that $P_{G(\boldsymbol{x})}(\boldsymbol{z}) = \frac{\exp(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{x})^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\boldsymbol{z}-\boldsymbol{x}))}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}_{\mathbf{y}}|}}$. Therefore, the PLRV can be expressed as

$$\begin{aligned}\mathrm{L}_{G,\boldsymbol{x},\boldsymbol{x}'} &= \frac{1}{2}(\mathbf{z}-\boldsymbol{x}')^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{z}-\boldsymbol{x}') - \frac{1}{2}(\mathbf{z}-\boldsymbol{x})^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{z}-\boldsymbol{x}) &(13)\\ &= \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\boldsymbol{x}-\boldsymbol{x}') + (\boldsymbol{x}-\boldsymbol{x}')^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{z}-\boldsymbol{x}). &(14)\end{aligned}$$

With $\mathbf{z}$ identically distributed to $G(\boldsymbol{x})$, i.e., $\mathbf{z} \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{\Sigma}_{\mathbf{y}})$, we have that $\mathrm{L}_{G,\boldsymbol{x},\boldsymbol{x}'} \sim \mathcal{N}(\eta, 2\eta)$ with $\eta = \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\boldsymbol{x}-\boldsymbol{x}')$. It follows from [4, Lemma 7] that for $\mathrm{x} \sim \mathcal{N}(\eta, 2\eta)$, $\mathbb{P}\left[\mathrm{x} \geq \epsilon\right] - e^{\epsilon}\left[\mathrm{x} \leq -\epsilon\right]$ is monotonically increasing in

η. By applying this result with $\mathrm{x} = \mathrm{L}_{G,\boldsymbol{x},\boldsymbol{x}'}$, we obtain that the maximum in the right-hand side of (12) is achieved when η is maximized, i.e., when $\eta = \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{S}_d} \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^\intercal \boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\boldsymbol{x}-\boldsymbol{x}') = \Delta^2/2$. We then obtain the optimal LDP curve (2) after some simple computations.

A.2 The Case $\mathcal{S}_d = \{\boldsymbol{x} \in \mathbb{R}^d : ||\boldsymbol{x}||_2 \le r\sqrt{d}\}$

In this case, for every $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}_d$, we have that

$$(\boldsymbol{x}-\boldsymbol{x}')^\intercal \boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\boldsymbol{x}-\boldsymbol{x}') = \|\boldsymbol{x}-\boldsymbol{x}'\|_2^2 \cdot \frac{(\boldsymbol{x}-\boldsymbol{x}')^\intercal}{\|\boldsymbol{x}-\boldsymbol{x}'\|_2} \boldsymbol{\Sigma}_{\mathbf{y}}^{-1} \frac{\boldsymbol{x}-\boldsymbol{x}'}{\|\boldsymbol{x}-\boldsymbol{x}'\|_2} \tag{15}$$

$$\le (\|\boldsymbol{x}\|_2 + \|\boldsymbol{x}'\|_2)^2 \cdot \lambda_{\max}(\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}) \tag{16}$$

$$\le (2r\sqrt{d})^2 \lambda_{\min}^{-1}(\boldsymbol{\Sigma}_{\mathbf{y}}), \tag{17}$$

where $\lambda_{\max}(\boldsymbol{\Sigma}_{\mathbf{y}}^{-1})$ is the largest eigenvalue of $\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}$ and $\lambda_{\min}(\boldsymbol{\Sigma}_{\mathbf{y}})$ is the smallest eigenvalue of $\boldsymbol{\Sigma}_{\mathbf{y}}$. Here, (16) follows from the triangle inequality and the Rayleigh-Ritz theorem, and (17) holds because both $\|\boldsymbol{x}\|_2$ and $\|\boldsymbol{x}'\|_2$ are bounded by $r\sqrt{d}$. Equalities occur in (16) and (17) if $\boldsymbol{x} = -\boldsymbol{x}' = r\sqrt{d}\boldsymbol{v}_{\min}$ where $\boldsymbol{v}_{\min}$ is the eigenvector of $\boldsymbol{\Sigma}_{\mathbf{y}}$ corresponding to $\lambda_{\min}^{-1}(\boldsymbol{\Sigma}_{\mathbf{y}})$. Therefore, $\Delta = 2r\sqrt{d/\lambda_{\min}(\boldsymbol{\Sigma}_{\mathbf{y}})}$.

If we let $\boldsymbol{\Sigma}_{\mathbf{y}}$ equal the covariance matrix of the sum of n independent vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ in $\mathcal{S}_d$, it holds that $\lambda_{\min}(\boldsymbol{\Sigma}_{\mathbf{y}}) \le \frac{1}{d}\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathbf{y}}) = \frac{1}{d}\mathbb{E}\left[\mathrm{Tr}\left(\sum_{i=1}^n \mathbf{x}_i\mathbf{x}_i^\intercal\right)\right] = \frac{1}{d}\sum_{i=1}^n \mathbb{E}\left[\mathrm{Tr}(\mathbf{x}_i\mathbf{x}_i^\intercal)\right] = \frac{1}{d}\sum_{i=1}^n \mathbb{E}\left[\|\mathbf{x}_i\|_2^2\right] \le nr^2$. As a consequence, $\Delta \ge 2\sqrt{d/n}$.

A.3 Trade-Off Function: Proof of Proposition 1

The log-likelihood ratio (LLR) for the test between $\{H\colon \boldsymbol{z}$ is generated from $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\Sigma}_{\mathbf{y}})\}$ and $\{H'\colon \boldsymbol{z}$ is generated from $\mathcal{N}(\boldsymbol{x}', \boldsymbol{\Sigma}_{\mathbf{y}})\}$ is given by the privacy loss function $L_{G,\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{z})$. For a threshold θ, the FNR and FPR are given by

$$\mathrm{FNR}(\theta) = \mathbb{P}_{\mathbf{z}\sim\mathcal{N}(\boldsymbol{x}',\boldsymbol{\Sigma}_{\mathbf{y}})}\left[L_{G,\boldsymbol{x},\boldsymbol{x}'}(\mathbf{z}) \ge \theta\right] = \mathbb{P}\left[\mathrm{L}_{G,\boldsymbol{x}',\boldsymbol{x}} \le -\theta\right], \tag{18}$$

$$\mathrm{FPR}(\theta) = \mathbb{P}_{\mathbf{z}\sim\mathcal{N}(\boldsymbol{x},\boldsymbol{\Sigma}_{\mathbf{y}})}\left[L_{G,\boldsymbol{x},\boldsymbol{x}'}(\mathbf{z}) < \theta\right] = \mathbb{P}\left[\mathrm{L}_{G,\boldsymbol{x},\boldsymbol{x}'} < \theta\right]. \tag{19}$$

In the proof of Theorem 2, we have shown that $\mathrm{L}_{G,\boldsymbol{x},\boldsymbol{x}'}(\mathbf{z}) \sim \mathcal{N}(\eta, 2\eta)$ with $\eta = \frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^\intercal \boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\boldsymbol{x}-\boldsymbol{x}')$. Therefore, $\mathrm{FNR}(\theta) = \Phi(\frac{-\theta-\eta}{\sqrt{2\eta}})$ and $\mathrm{FPR}(\theta) = \Phi(\frac{\theta-\eta}{\sqrt{2\eta}})$. To achieve $\mathrm{FPR}(\theta) \le \alpha$, the threshold must satisfy $\theta \le \eta - \sqrt{2\eta}\Phi^{-1}(1-\alpha)$. Under this constraint, the minimum FNR is given by $\Phi(\Phi^{-1}(1-\alpha) - \sqrt{2\eta})$. This is by definition the optimal trade-off $T_{\mathcal{N}(\boldsymbol{x},\boldsymbol{\Sigma}),\mathcal{N}(\boldsymbol{x}',\boldsymbol{\Sigma})}(\alpha)$.

B LDP Analysis of the Mechanism (1) in a Special Case: Proof of Theorem 3

To prove Theorem 3, we shall use the following preliminary results.

Lemma 1. *For two pairs of distributions* (P_1, Q_1) *and* (P_2, Q_2)*, if* $T_{P_1,Q_1}(\alpha) \geq T_{P_2,Q_2}(\alpha)$ *for every* $\alpha \in [0,1]$*, then* $\mathsf{E}_{e^\epsilon}(P_1 \| Q_1) \leq \mathsf{E}_{e^\epsilon}(P_2 \| Q_2)$ *for every* $\epsilon > 0$.

Lemma 1 follows directly by expressing the hockey-stick divergence in terms of the trade-off function as follows.

Lemma 2. *Consider two distributions* P *and* Q *defined over* $\mathcal{X}$*. Define a random variable* $\mathrm{L}_P = \ln \frac{Q(\mathrm{x})}{P(\mathrm{x})}$, $\mathrm{x} \sim P$*, and denote its CDF by* $F_P(x) = \mathbb{P}[\mathrm{L}_P \leq x]$*. It holds that* $\mathsf{E}_{e^\epsilon}(P \| Q) = F_P(-\epsilon) - e^\epsilon T_{P,Q}(1 - F_P(-\epsilon))$.

Proof of Lemma 2. Define the random variable $\mathrm{L}_Q = \ln \frac{Q(\mathrm{x})}{P(\mathrm{x})}$, $\mathrm{x} \sim Q$, and denote its CDF by $F_Q(x)$. Observe that $1 - F_P(\theta)$ and $F_Q(\theta)$ are the FPR and FNR of the likelihood test between P and Q with threshold θ. It follows from Definition 1 and the Neyman-Pearson lemma [20, Thm. 8.6.1] that

$$T_{P,Q}(1 - F_P(\theta)) = F_Q(\theta). \tag{20}$$

We further have that

$$\begin{aligned}
\mathsf{E}_{e^\epsilon}(P \| Q) &= \int_{\mathcal{X}} \big(P(x) - e^\epsilon Q(x)\big)^+ \mathrm{d}x \\
&= \int_{\mathcal{X}} \big(P(x) - e^\epsilon Q(x)\big) \mathbb{1}\{P(x) \geq e^\epsilon Q(x)\} \mathrm{d}x \\
&= \int_{\mathcal{X}} P(x) \mathbb{1}\{P(x) \geq e^\epsilon Q(x)\} \mathrm{d}x - e^\epsilon \int_{\mathcal{X}} Q(x) \mathbb{1}\{P(x) \geq e^\epsilon Q(x)\} \mathrm{d}x \\
&= F_P(-\epsilon) - e^\epsilon F_Q(-\epsilon).
\end{aligned} \tag{21}$$

By substituting (20) with $\theta = -\epsilon$ into (21), we complete the proof. □

We are now ready to prove Theorem 3. For brevity, we omit the subscript 0 in $\mathbf{x}_0$ and x_{0j}, $j \in [d]$.

The Univariate Case. We first consider $d = 1$. Then M in (1) is written as

$$M(x) = x + \mathrm{y} \tag{22}$$

where $\underline{r} \leq x \leq \overline{r}$ and y follows a symmetric and log-concave distribution P_y. We next show that a dominating pair of distribution for this mechanism is

$(P_{\underline{r}+\mathrm{y}}, P_{\overline{r}+\mathrm{y}})$. By symmetry of P_y around y_0, we have that, for every $a, b \in \mathbb{R}$,

$$\begin{aligned}
\mathsf{E}_{e^\epsilon}(a+\mathrm{y}\|b+\mathrm{y}) &= \sup_{\mathcal{A}} P_{a+\mathrm{y}}(\mathcal{A}) - e^\epsilon P_{b+\mathrm{y}}(\mathcal{A}) && (23)\\
&= \sup_{\mathcal{A}} P_\mathrm{y}(\mathcal{A}-a) - e^\epsilon P_\mathrm{y}(\mathcal{A}-b) && (24)\\
&= \sup_{-\mathcal{A}+z_0} P_\mathrm{y}(-\mathcal{A}+z_0-a) - e^\epsilon P_\mathrm{y}(-\mathcal{A}+z_0-b) && (25)\\
&= \sup_{\mathcal{A}} P_\mathrm{y}(\mathcal{A}+a+z_0) - e^\epsilon P_\mathrm{y}(\mathcal{A}+b+z_0) && (26)\\
&= \sup_{\mathcal{A}'=\mathcal{A}+a+b+z_0} P_\mathrm{y}(\mathcal{A}'-b) - e^\epsilon P_\mathrm{y}(\mathcal{A}'-a) && (27)\\
&= \sup_{\mathcal{A}} P_\mathrm{y}(\mathcal{A}-b) - e^\epsilon P_\mathrm{y}(\mathcal{A}-a) && (28)\\
&= \mathsf{E}_{e^\epsilon}(b+\mathrm{y}\|a+\mathrm{y}). && (29)
\end{aligned}$$

Therefore, it suffices to prove that

$$\mathsf{E}_{e^\epsilon}(a+\mathrm{y}\|b+\mathrm{y}) \le \mathsf{E}_{e^\epsilon}(\underline{r}+\mathrm{y}\|\overline{r}+\mathrm{y}), \quad \forall \underline{r} \le a \le b \le \overline{r}. \tag{30}$$

To this end, we shall show that $\mathsf{E}_{e^\epsilon}(a+\mathrm{y}\|b+\mathrm{y})$ increases with $b-a$, and thus maximized when $(a,b) = \operatorname*{arg\,max}_{a',b'\in[\underline{r},\overline{r}],a'\le b'} (b'-a') = (\underline{r},\overline{r})$. In light of Lemma 1, it suffices to show that $T_{a+\mathrm{y},b+\mathrm{y}}(x)$ decreases with $b-a$ for all $x \in \mathbb{R}$. Indeed, this is true because $T_{a+\mathrm{y},b+\mathrm{y}}(\alpha) = F_\mathrm{y}(F_\mathrm{y}^{-1}(1-\alpha)-(b-a))$ for y following a log-concave distribution (this follows from [12, Prop. A.3]), and because the CDF $F_\mathrm{y}(\cdot)$ is an increasing function.

The Multivariate Case. We now address the general case with $d \ge 1$. Using the independence assumption, we can write the mechanism (1) as $M(\boldsymbol{x}) = (M_1(\boldsymbol{x}), M_2(\boldsymbol{x}), \dots, M_d(\boldsymbol{x}))$ where $M_j(\boldsymbol{x}) = \boldsymbol{e}_j^\top \boldsymbol{x} + \mathrm{y}_j$, with $\boldsymbol{e}_j$ being the jth d-dimensional canonical basis vector. Therefore, $M(\boldsymbol{x})$ is a (nonadaptive) composition of d mechanisms $\{M_j\}_{j\in[d]}$. Observe that each mechanism M_j has the form (22). Therefore, $(P_{\underline{r}_j+\mathrm{y}_j}, P_{\overline{r}_j+\mathrm{y}_j})$ is a dominating pair of distributions for M_j. The proof is completed by applying [35, Thm. 10], which states that if (P_j, Q_j) is a dominating pair of distributions for mechanism M_j, $j \in [d]$, then $(P_1 \times \cdots \times P_d, Q_1 \times \cdots \times Q_d)$ is a dominating pair of distributions for mechanism $M(\boldsymbol{x}) = (M_1(\boldsymbol{x}), \dots, M_d(\boldsymbol{x}))$.

References

1. Abadi, M., et al.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 308–318. ACM, New York (2016)
2. Agarwal, N., Suresh, A.T., Yu, F., Kumar, S., McMahan, H.B.: CpSGD: communication-efficient and differentially-private distributed SGD. In: Proceedings of the International Conference in Neural Information Processing Systems (NIPS), NIPS 2018, pp. 7575–7586 (2018)

3. Bagnoli, M., Bergstrom, T.: Log-concave probability and its applications. In: Aliprantis, C.D., Matzkin, R.L., McFadden, D.L., Moore, J.C., Yannelis, N.C. (eds.) Rationality and Equilibrium. Studies in Economic Theory, vol. 26, pp. 217–241. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-29578-X_11
4. Balle, B., Wang, Y.X.: Improving the Gaussian mechanism for differential privacy: analytical calibration and optimal denoising. In: Proceedings of the International Conference Machine Learning (ICML), pp. 394–403. PMLR (2018)
5. Becker, B., Kohavi, R.: Adult. UCI Machine Learning Repository (1996)
6. Bell, J.H., Bonawitz, K.A., Gascón, A., Lepoint, T., Raykova, M.: Secure single-server aggregation with (poly)logarithmic overhead. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS), pp. 1253–1269. ACM, New York (2020)
7. Bhaskar, R., Bhowmick, A., Goyal, V., Laxman, S., Thakurta, A.: Noiseless database privacy. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 215–232. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_12
8. Boenisch, F., Dziedzic, A., et al.: Reconstructing individual data points in federated learning hardened with differential privacy and secure aggregation. In: Proceedings of the European Symposium on Security and Privacy (EuroS&P), pp. 241–257 (2023)
9. Bonawitz, K., et al.: Practical secure aggregation for privacy-preserving machine learning. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, pp. 1175–1191 (2017)
10. Clopper, C.J., Pearson, E.S.: The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika **26**(4), 404–413 (1934)
11. Cohen, G., Afshar, S., Tapson, J., van Schaik, A.: EMNIST: extending MNIST to handwritten letters. In: Proceedings of International Joint Conference on Neural Networks (IJCNN), pp. 2921–2926 (2017)
12. Dong, J., Roth, A., Su, W.: Gaussian differential privacy. J. Roy. Stat. Soc. **84**(1), 3–37 (2021)
13. Duchi, J.C., Jordan, M.I., Wainwright, M.J.: Local privacy and statistical minimax rates. In: Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 429–438 (2013)
14. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_14
15. Dwork, C., Roth, A., et al.: The algorithmic foundations of differential privacy. Found. Trends® Theoret. Comput. Sci. **9**(3–4), 211–407 (2014)
16. Elkordy, A.R., Zhang, J., Ezzeldin, Y.H., Psounis, K., Avestimehr, S.: How much privacy does federated learning with secure aggregation guarantee? In: Proceedings of the Privacy Enhancing Technologies Symposium (PETS), pp. 510–526 (2023)
17. Fowl, L.H., Geiping, J., Czaja, W., Goldblum, M., Goldstein, T.: Robbing the fed: directly obtaining private data in federated learning with modified models. In: Proceedings of the International Conference on Learning Representations (ICLR) (2022)
18. Geiping, J., Bauermeister, H., Dröge, H., Moeller, M.: Inverting gradients - how easy is it to break privacy in federated learning? In: Proceedings of the International Conference on Neural Information Processing Systems (NeuRIPS), NeuRIPS 2020 (2020)
19. Hatamizadeh, A., et al.: Do gradient inversion attacks make federated learning unsafe? IEEE Trans. Med. Imaging **42**(7), 2044–2056 (2023)

20. Hogg, R.V., Tanis, E.A., Zimmerman, D.: Probability and Statistical Inference, 9th edn. Pearson, Upper Saddle River (2015)
21. Jagielski, M., Ullman, J., Oprea, A.: Auditing differentially private machine learning: how private is private SGD? In: Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 22205–22216 (2020)
22. Kairouz, P., Liu, Z., Steinke, T.: The distributed discrete Gaussian mechanism for federated learning with secure aggregation. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 5201–5212. PMLR (2021)
23. Kairouz, P., Oh, S., Viswanath, P.: The composition theorem for differential privacy. IEEE Trans. Inf. Theory **63**(6), 4037–4049 (2017)
24. Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.: What can we learn privately? SIAM J. Comput. **40**(3), 793–826 (2011)
25. Kerkouche, R., Ács, G., Fritz, M.: Client-specific property inference against secure aggregation in federated learning. In: Proceedings of the Workshop Privacy in the Electronic Society, WPES 2023, pp. 45–60. ACM, New York (2023)
26. Lam, M., Wei, G.Y., Brooks, D., Reddi, V.J., Mitzenmacher, M.: Gradient disaggregation: breaking privacy in federated learning by reconstructing the user participant matrix. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 5959–5968. PMLR (2021)
27. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.y.: Communication-efficient learning of deep networks from decentralized data. In: Singh, A., Zhu, J. (eds.) Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR, vol. 54, pp. 1273–1282. PMLR, 20–22 April 2017
28. Merkle, M.: Convolutions of logarithmically concave functions. Publikacije Elektrotehničkog fakulteta. Serija Matematika, pp. 113–117 (1998)
29. Nasr, M., Shokri, R., Houmansadr, A.: Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In: Proceedings of the IEEE Symposium on Security and Privacy (SP), pp. 739–753 (2019)
30. So, J., Ali, R.E., Güler, B., Jiao, J., Avestimehr, A.S.: Securing secure aggregation: mitigating multi-round privacy leakage in federated learning. In: Proceedings of the AAAI Conference on Artificial Intelligence & Conf. Innov. App. Art. Intel. & Symp. Edu. Adv. Art. Intel. AAAI 2023/IAAI 2023/EAAI 2023. AAAI Press (2023)
31. Ullah, E., Choquette-Choo, C.A., Kairouz, P., Oh, S.: Private federated learning with autotuned compression. In: Proceedings of the International Conference on Machine Learning (ICML), ICML 2023. JMLR.org (2023)
32. Van der Vaart, A.W.: Asymptotic Statistics, vol. 3. Cambridge University Press, Cambridge (2000)
33. Ye, J., Maddi, A., Murakonda, S.K., Bindschaedler, V., Shokri, R.: Enhanced membership inference attacks against machine learning models. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, pp. 3093–3106 (2022)
34. Youn, Y., Hu, Z., Ziani, J., Abernethy, J.: Randomized quantization is all you need for differential privacy in federated learning. In: ICML Workshop (2023)
35. Zhu, Y., Dong, J., Wang, Y.X.: Optimal accounting of differential privacy via characteristic function. In: Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS). PMLR, vol. 151, pp. 4782–4817. PMLR, 28–30 March 2022

Evaluating Negation with Multi-way Joins Accelerates Class Expression Learning

Nikolaos Karalis(✉), Alexander Bigerl, Caglar Demir, Liss Heidrich, and Axel-Cyrille Ngonga Ngomo

DICE Group, Department of Computer Science, Paderborn University, Paderborn, Germany
{nikolaos.karalis,alexander.bigerl,caglar.demir,liss.heidrich, axel.ngonga}@uni-paderborn.de

Abstract. Class expression learning based on refinement operators is a popular family of explainable machine learning approaches for RDF knowledge graphs with ontologies in description logics. However, most implementations of this paradigm fail to scale to the large knowledge graphs found on the Web. One common bottleneck of these implementations is the instance retrieval function. We address this drawback by introducing an algorithm inspired by worst-case optimal multi-way joins for the evaluation of SPARQL queries that correspond to $\mathcal{ALC}$ class expressions. The main characteristic of our algorithm is the inclusion of negation, which is prominent in SPARQL queries generated from $\mathcal{ALC}$ class expressions, in multi-way join plans. We evaluate the implementation of our approach on five benchmark datasets against four state-of-the-art graph storage solutions for RDF knowledge graphs. The results of our extensive evaluation show that our approach outperforms its competition across all datasets and that it is the only one able to scale to large datasets. With our approach, we enable learning algorithms to retrieve information from Web-scale knowledge graphs, hence making ante-hoc explainable machine learning easier to deploy on the Semantic Web.

Keywords: knowledge graphs · class expression learning · multi-way joins

1 Introduction

RDF knowledge graphs are now first-class citizens of the Web. Over 80 billion RDF assertions are found in the 2021 crawl of the Web Data Commons.[1] RDF data dumps on the Web cover a similar order of magnitude.[2] Learning on RDF data at this scale is hence crucial for the deployment of machine learning on the Web—the world's largest shared information source with over 5 billion users. The large proportion of the human population impacted by machine learning on

[1] http://webdatacommons.org/structureddata/#results-2021-1.
[2] http://lod-a-lot.lod.labs.vu.nl/.

the Web entails the need for explainable machine learning because of its known societal advantages [8].

A popular family of ante-hoc explainable approaches for supervised learning on RDF knowledge graphs are class expression learning algorithms based on refinement operators [10,18,23]. Given a set of positive and negative examples, these approaches generate a class expression in a predefined description logic, which ideally describes the positive and not the negative examples. However, a large body of literature on learning class expressions [13,17,23] suggests that these approaches do not scale to the size of datasets found on the Web. This is mostly due to their instance retrieval function–which computes the instances of given class expressions [10,18,23]–being unable to retrieve instances in a time-efficient manner [18,23]. The authors of [7] suggest that this weakness can be addressed by converting class expressions into SPARQL queries.

SPARQL[3] is the designated language for querying RDF knowledge graphs. A recent advancement in querying processing is the introduction of worst-case optimal multi-way join algorithms [19]. Worst-case optimal multi-way join algorithms have been adopted by the semantic web community [15] and are now being used by state-of-the-art knowledge graph storage solutions (e.g., triple stores) for the efficient evaluation of SPARQL queries [2,5]. However, the use of multi-way joins for SPARQL is mostly limited to conjunctive queries. However, as demonstrated in Sect. 3, class expression learning with SPARQL does not only deal with conjunctive queries; in particular, it requires the efficient evaluation of queries containing *negation*.

The hypothesis behind this work is that we can exploit multi-way joins to implement a time-efficient retrieval function for class expression learning. To this end, as in previous works (e.g., [13,17]), we set our focus on the description logic $\mathcal{ALC}$ and present a multi-way join algorithm for the efficient evaluation of SPARQL queries generated by $\mathcal{ALC}$ class expressions. Such SPARQL queries include union graph patterns and negation, which is captured by `FILTER NOT EXISTS` patterns. To the best of our knowledge, there have not been any works for SPARQL that consider the evaluation of negation in multi-way join plans.

The main contributions of this work are the following: (i) Inspired by worst-case-optimal multi-way join algorithms and their efficiency in evaluating conjunctive queries, we present a multi-way join algorithm for the evaluation of SPARQL queries generated from $\mathcal{ALC}$ class expressions. Our approach relies on theoretical foundations of SPARQL to enable the use of multi-way joins in queries involving negation. (ii) We have identified an issue with the mapping of class expressions to SPARQL queries presented in [7] and present a solution that addresses this particular issue. (iii) We have implemented the proposed algorithm in a state-of-the-art triple store and present the results of an extensive comparison of our implementation on multiple benchmark datasets of varying sizes with multiple state-of-the-art triple stores using SPARQL queries that correspond to $\mathcal{ALC}$ class expressions. In our evaluation, we use four datasets, ranging from

[3] https://www.w3.org/TR/2013/REC-sparql11-query-20130321/.

96K to 2.1M triples, that are commonly used to evaluate learning algorithms. To evaluate the scalabilty of our approach, we also use a dataset consisting of more than 40M triples. The results of our experimental evaluation show that our approach outperforms its competition across all datasets and that it is the only one that efficiently handles query workloads in the largest dataset.

The rest of this work is structured as follows: In Sect. 2, we provide background knowledge on the topics that are covered in this paper. In Sect. 3, we cover the translation of $\mathcal{ALC}$ class expressions to SPARQL queries. We present the proposed multi-way join algorithm for SPARQL queries generated from $\mathcal{ALC}$ class expressions in Sect. 4. Our experimental results are presented in Sect. 5. We discuss related works in Sect. 6 and conclude in Sect. 7.

2 Preliminaries

We begin first with a brief overview of the description logic $\mathcal{ALC}$. Second, we describe the problem of class expression learning. Third, we cover definitions and properties of SPARQL queries that are required in this work. Last, we briefly summarize worst-case optimal multi-way join algorithms.

2.1 The Description Logic $\mathcal{ALC}$

A description logic is a decidable fragment of first-order predicate logic that uses only unary and binary predicates [4]. The set of unary predicates, binary predicates and constants correspond to the set of named concepts N_C, roles N_R, and individuals N_I of a description logic, respectively. Like in recent works on class expression learning (e.g., [13,17]), we focus on the description logic $\mathcal{ALC}$ [25]. Its syntax and semantics are provided in Table 1. In this paper, $\mathcal{C}$ denotes all valid $\mathcal{ALC}$ concepts C under the construction rules: $C ::= A \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \exists r.C \mid \forall r.C$, where $A \in N_C$ and $r \in N_R$.

Knowledge bases in $\mathcal{ALC}$ are often defined as $\mathcal{K} = (\mathit{Tbox}, \mathit{Abox})$. All axioms in *Tbox* are of the form $A \sqsubseteq B$ or $A \equiv B$. *Abox* contains the relationships between individuals $a, b \in N_I$ via roles $r \in N_R$ and membership relationships between N_I and $\mathcal{C}$. Following previous works on class expression learning [13,17,18], we adopt closed world semantics. Under the closed world assumption, the ABox of a knowledge base is treated as the knowledge base's model $\mathcal{I}$ [18]. In addition, under closed world semantics, checking whether an instance belongs to a particular concept or retrieving the individuals of a particular concept is similar to querying in classical databases [7,14].

2.2 Class Expression Learning

Class expression learning is a family of supervised machine learning algorithms. Given a knowledge base $\mathcal{K} = (\mathit{Tbox}, \mathit{Abox})$, a set of positive examples E^+, and a set of negative examples E^-, with $E^+ \cup E^- \subseteq N_I$ and $E^+ \cap E^- = \emptyset$, class expression learning algorithms aim to learn a class expression H that ideally

Table 1. Syntax and semantics of $\mathcal{ALC}$. $\mathcal{I}$ denotes an interpretation and $\Delta^{\mathcal{I}}$ its domain.

Construct	Syntax	Semantics
Top concept	$\top$	$\Delta^{\mathcal{I}}$
Bottom concept	$\bot$	$\emptyset$
Atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
Role	r	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
Conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Existential restriction	$\exists\, r.C$	$\{x \mid \exists\, (x^{\mathcal{I}}, y^{\mathcal{I}}) \in r^{\mathcal{I}} \wedge y^{\mathcal{I}} \in C^{\mathcal{I}}\}$
Universal restriction	$\forall\, r.C$	$\{x \mid \forall\, (x^{\mathcal{I}}, y^{\mathcal{I}}) \in r^{\mathcal{I}} \implies y^{\mathcal{I}} \in C^{\mathcal{I}}\}$

describes all of the individuals of E^+ and not any of the individuals of E^- [18]. To find the most appropriate H for a particular pair of E^+ and E^-, a class expression learning problem is often transformed into a search problem within a quasi-ordered space $(\mathcal{C}, \preceq)$ [11,18,31], where $\preceq$ is often the subsumption relation $\sqsubseteq$ between concepts [17]. The traversal of the search problem's space is usually conducted using a downward refinement operator $\rho : \mathcal{C} \rightarrow 2^{\mathcal{C}}$ such that $\rho(C) \subseteq \{C' \in \mathcal{C} \mid C' \sqsubseteq C, C' \neq C\}$ for all $C \in \mathcal{C}$ [17].

The quality of a class expression H (i.e., finding whether H describes the individuals of E^+ and not the individuals of E^-) is commonly computed by a heuristic function, such as CELOE [18]. Internally, such heuristic functions use a retrieval function $\mathcal{R} : \mathcal{C} \rightarrow 2^{N_I}$, which returns all individuals in N_I that are instances of the provided class expression H. As discussed in Sect. 2.1, we assume that retrieval operations are carried out under closed world semantics. As the size of an input knowledge base grows, executing retrieval operations against reasoners becomes one of the main computational bottlenecks [7,18,23] of learning algorithms. The goal of this work is to accelerate class expression learning by reducing the runtimes of retrieval operations.

2.3 Semantics and Properties of SPARQL

Here, we provide the semantics and the properties of SPARQL queries that use those features of the language that are used in queries generated by $\mathcal{ALC}$ class expressions (Sect. 3). In particular, our focus is on SPARQL queries consisting of *triple patterns*, *basic graph patterns*, *union graph patterns*, and *negation* in the form of *`FILTER NOT EXISTS`* patterns. Note that SPARQL queries are defined under bag semantics. In this work, we adopt set semantics (i.e., queries contain `DISTINCT`), which are in line with the semantics of $\mathcal{ALC}$. Let $\mathbf{I}$ be an infinite set of IRIs, $\mathbf{B}$ an infinite set of blank nodes, and $\mathbf{L}$ an infinite set of literals. The sets $\mathbf{I}$, $\mathbf{B}$ and $\mathbf{L}$ are pairwise disjoint and their union, denoted as $\mathbf{T} = \mathbf{I} \cup \mathbf{B} \cup \mathbf{L}$, is the set of all terms. Furthermore, let $\mathbf{V}$ be an infinite set of variables. An RDF graph is a set of triples and is defined as $G = \{(s, p, o) \mid s \in (\mathbf{I} \cup \mathbf{B}), p \in \mathbf{I}, o \in \mathbf{T}\}$, where s, p, and o stand for *subject*, *predicate*, and *object*, respectively.

Triple, Basic and Union Graph Patterns. The following is based on [15,21]. A triple pattern tp is a triple (s, p, o), where $s \in (\mathbf{I} \cup \mathbf{L} \cup \mathbf{V})$, $p \in (\mathbf{I} \cup \mathbf{V})$, and $o \in (\mathbf{I} \cup \mathbf{L} \cup \mathbf{V})$. The set of variables of a triple pattern is denoted as $var(tp)$. Since blank nodes behave as variables, we do not consider them in triple patterns [15,21]. A basic graph pattern (BGP) is a set of triple patterns. The semantics of SPARQL queries are defined using mappings, which are partial functions assigning terms to variables. A mapping is formally defined as $\mu : \mathbf{V} \rightarrow \mathbf{T}$ and its domain is denoted as $dom(\mu)$. With $\mu(tp)$, we denote the RDF triple obtained by replacing the variables of $var(tp)$ with their corresponding values in μ. Two mappings μ_1 and μ_2 are compatible ($\mu_1 \sim \mu_2$), iff $\mu_1(?v) = \mu_2(?v)$ for every variable $?v \in dom(\mu_1) \cap dom(\mu_2)$. Two mappings μ_1 and μ_2 are not compatible ($\mu_1 \nsim \mu_2$), if for any $?v \in dom(\mu_1) \cap dom(\mu_2)$, $\mu_1(?v) \neq \mu_2(?v)$. Given two sets of mappings Ω_1 and Ω_2 the join operation is defined as $\Omega_1 \bowtie \Omega_2 = \{\mu_1 \cup \mu_2 \mid \mu_1 \in \Omega_1, \mu_2 \in \Omega_2 \text{ and } \mu_1 \sim \mu_2\}$ and the union operation is defined as $\Omega_1 \cup \Omega_2 = \{\mu \mid \mu \in \Omega_1 \text{ or } \mu \in \Omega_2\}$. The evaluation of a triple pattern tp and a BGP P over an RDF graph G is denoted as $[\![tp]\!]_G = \{\mu \mid dom(\mu) = var(tp) \text{ and } \mu(tp) \in G\}$ and $[\![P]\!]_G = [\![tp_1, \ldots, tp_n]\!]_G = [\![tp_1]\!]_G \bowtie \ldots \bowtie [\![tp_n]\!]_G$, respectively. In SPARQL, graph patterns are constructed recursively. Triple patterns and BGPs are graph patterns. The set of variables of a graph pattern P is denoted as $var(P)$. The conjunction and union between two graph patterns P_1 and P_2 are also graph patterns and are evaluated over G as $[\![P_1 \ AND \ P_2]\!]_G = [\![P_1]\!]_G \bowtie [\![P_2]\!]_G$ and $[\![P_1 \ UNION \ P_2]\!]_G = [\![P_1]\!]_G \cup [\![P_2]\!]_G$, respectively.

Negation. The following definitions are based on [1]. In SPARQL, there are multiple ways to express negation [1]. Here, we focus on patterns of negation that are expressed using `FILTER NOT EXISTS` patterns, which are used for converting $\mathcal{ALC}$ class expressions to SPARQL queries (Sect. 3). `FILTER NOT EXISTS` can be provided with either a subquery or a graph pattern. In this work, we are interested only in the latter case (Sect. 3). Given a graph pattern $P = (P_1 \ FNE \ P_2)$, where FNE stands for `FILTER NOT EXISTS`, the set of correlated variables $corVars(P)$ is defined as $corVars(P) = var(P_1) \cap var(P_2)$; P_1 and P_2 are correlated if $corVars(P) \neq \emptyset$. In this work, we focus only on queries for which P_1 and P_2 are always correlated (Sect. 3). If P_1 and P_2 are correlated, P_1 is evaluated before P_2 and P_2 is evaluated after each correlated variable in P_2 is replaced with its corresponding value obtained in the evaluation of P_1 [1]. The set of certain variables of a graph pattern P, denoted as $cVars(P)$, consists of the variables that are always bound in the solution mapping of P [24]. The set of certain variables of a graph pattern P is defined recursively [1,24]: (i) if P is a triple pattern tp, then $cVars(P) = var(tp)$, (ii) if $P = (P_1 \ AND \ P_2)$, then $cVars(P) = cVars(P_1) \cup cVars(P_2)$, (iii) if $P = (P_1 \ UNION \ P_2)$, then $cVars(P) = cVars(P_1) \cap cVars(P_2)$, (iv) if $P = (P_1 \ FILTER \ R)$, then $cVars(P) = cVars(P_1)$ with R being a built-in condition (e.g., integer addition), and (v) if

Algorithm 1. Generic Join

```
1: // Execution is resumed after a **yield** operation
2: **function** GenericJoin($P$, $G$, $X$)    ▷ $P$: BGP, $G$: RDF Graph, $X$: Mapping
3:     **if** $var(P) = \emptyset$ **then yield** $X$ and **return**    ▷ All variables are evaluated
4:     $?x \leftarrow$ a variable from $var(P)$    ▷ Select a variable to be evaluated
5:     $K \leftarrow \bigcap_{tp \in P \mid ?x \in var(tp)} \{\mu(?x) \mid \mu \in [\![tp]\!]_G\}$    ▷ All possible values of $?x$
6:     **for all** $k \in K$ **do**    ▷ Iterate over the possible values of $?x$
7:         $X(?x) \leftarrow k$    ▷ Store $k$ in the solution mapping
8:         $P' \leftarrow$ assign $k$ to all occurrences of $?x$ in $P$
9:         **yield all** GenericJoin($P'$, $G$, $X$)    ▷ proceeds with the next $k$ afterwards
```

$P = (P_1\ \mathit{FNE}\ P_2)$, then *cVars(P)* $=$ *cVars*(P_1). A graph pattern is *fne-safe*, if for every subpattern $P = (P_1\ \mathit{FNE}\ P_2)$ holds that *corVars(P)* $\subseteq$ *cVars*(P_2). Given two sets of mappings Ω_1 and Ω_2 the difference operation is defined as $\Omega_1 \setminus \Omega_2 = \{\mu_1 \in \Omega_1 \mid \forall \mu_2 \in \Omega_2, \mu_1 \nsim \mu_2\}$. For fne-safe patterns, the negation is defined as $[\![P_1\ \mathit{FNE}\ P_2]\!]_G = [\![P_1]\!]_G \setminus [\![P_2]\!]_G$. The notion of fne-safety ensures that nested *FNE* patterns can be evaluated using the difference operation [1].

Filter Rewriting Rule and Union Normal Form. A graph pattern $((P_1\ \mathit{FILTER}\ R)\ \ \mathit{AND}\ P_2)$ can be rewritten to a pattern $((P_1\ \mathit{AND}\ P_2)$ $\mathit{FILTER}\ R)$, if every variable of R is also a certain variable of P_1 or if R and P_2 do not share any variables [24]. A graph pattern P is in *UNION normal form* if it is in the form $(P_1\ \mathit{UNION}\ P_2\ \mathit{UNION}\ \ldots\ \mathit{UNION}\ P_n)$ and each P_i, for $1 \leq i \leq n$, is UNION-free [21]. Every *AND-UNION-FILTER* graph pattern P is equivalent to a graph pattern P', which is in UNION normal form [21].

2.4 Worst-Case Optimal Multi-way Join Algorithms

In recent years, worst-case optimal multi-way join algorithms [19] have been the subject of many research works in the database literature. This is due to their runtime complexity being bounded by the worst-case size of the result of the input query [3] and their ability to achieve state-of-the-art performance in the evaluation of conjunctive queries (e.g., [2,5,15]). Their main characteristic is that, contrary to conventional binary joins that carry out joins on two operands at a time, their evaluation follows a variable elimination process. This evaluation process resembles a backtracking search, does not store any intermediate results and enables the incremental output of solution mappings. A worst-case optimal multi-way join algorithm for the evaluation of BGPs (conjunctive queries) based on Generic Join [20] is shown in Algorithm 1.

3 Mapping $\mathcal{ALC}$ Class Expressions to SPARQL Queries

As discussed in Sect. 2.2, one of the main computational bottlenecks for class expression learning algorithms is the execution of retrieval operations against reasoners. To alleviate this issue, Bin et al. [7] propose a mapping for the conversion of class expressions to SPARQL queries. This mapping is shown in the first six entries of Table 2—including the struck through text. These conversions enable learning algorithms to carry out retrieval operations against a triple store instead of a reasoner.

Table 2. The mapping of $\mathcal{ALC}$ expressions to SPARQL queries. The struck through entry is the erroneous mapping for $\forall r.C$ class expressions proposed in [7]. Our proposed mapping for $\forall r.C$ class expressions is shown in the table's last entry.

$\mathcal{ALC}$ Class Expression C_i	SPARQL Graph Pattern $\tau(C_i, ?var)$
A	{ $?var$ `rdf:type` A . }
$\neg C$	{ $?var$ $?p$ $?o$. `FILTER NOT EXISTS` { $\tau(C, ?var)$ } }
$C_1 \sqcap \cdots \sqcap C_n$	{ $\tau(C_1, ?var)$. $\tau(C_2, ?var)$ . .... $\tau(C_n, ?var)$ }
$C_1 \sqcup \cdots \sqcup C_n$	{ { $\tau(C_1, ?var)$ } `UNION` ... `UNION` { $\tau(C_n, ?var)$ } }
$\exists r.C$	{ $?var$ r $?s$. $\tau(C, ?s)$ }
~~$\forall r.C$~~	~~{ ?var r ?s0 . { SELECT ?var (COUNT(?s1) AS ?c1) WHERE { ?var r ?s1 . τ(C, ?s1) } GROUP BY ?var } { SELECT ?var (COUNT(?s2) AS ?c2) WHERE { ?var r ?s2 } GROUP BY ?var } FILTER(?c1 = ?c2) }~~
$\forall r.C$	{ $?var$ $?p$ $?o$. `FILTER NOT EXISTS` { $?var$ r $?s$. `FILTER NOT EXISTS` { $\tau(C, ?s)$ } } }

During the development of our multi-way join algorithm for class expression learning (Sect. 4), we identified an issue with the mapping presented in [7]. In particular, the SPARQL queries corresponding to class expressions of the type $\forall r.C$ did not return the expected results. As per the semantics of $\mathcal{ALC}$, the set of instances corresponding to a class expression $\forall r.C$ should also contain those individuals that do not have any r-successors [22, Remark 18]. For example, assuming a knowledge base capturing the concepts of family members, the set of individuals corresponding to the concept $\forall hasChild.Male$ should also contain those individuals that do not have any children. However, this was not the case with the SPARQL queries corresponding to such class expressions, as they did not take individuals not having any r-successors into account [7, Section 3 of the corresponding technical report]. To alleviate this issue we propose a new mapping for $\forall r.C$ expressions, which is presented in the last entry of Table 2 and replaces the struck through rule. The proposed graph pattern for $\forall r.C$ class expressions is constructed using the concept equivalence rule $\forall r.C \equiv \neg\exists r.\neg C$ [22, Section 5.1]. For individuals that do not have any r-successors, the first `FILTER NOT EXISTS` is always evaluated to true. For individuals that have at least one r-successor, the first `FILTER NOT EXISTS` is evaluated to true, if all of their r-successors belong to C, which is tested by the seconds `FILTER NOT EXISTS`. Note that applying a combination of the transformation rules corresponding to $\neg C$ and $\exists r.C$ would yield a query that is semantically equivalent to the updated query.

The SPARQL queries shown in Table 2 use only those features of SPARQL that were discussed in Sect. 2.3. Furthermore, all of the queries that involve patterns of negation (`FILTER NOT EXISTS`) are fne-safe and hence, can be evaluated using the difference operator for sets of mappings. To prove that the newly proposed mapping for $\forall r.C$ follows the semantics of $\mathcal{ALC}$, the existing proofs for $\neg C$ and $\exists r.C$, which are provided in the technical report of [7], can be used.

4 Negation in Multi-way Joins

The efficiency of worst-case optimal multi-way join algorithms in evaluating basic graph pattern queries has been demonstrated in recent works (e.g., [2,5,15]). However, to the best of our knowledge, there have not been any efforts for SPARQL that include patterns of negation in multi-way join plans. In this section, we present our algorithm for the efficient evaluation of SPARQL queries corresponding to $\mathcal{ALC}$ class expressions. Our algorithm follows the sketch provided in [28] for the evaluation of Datalog rules containing negation and incorporates the evaluation of `FILTER NOT EXISTS` patterns in multi-way join plans. The two main ideas behind the proposed algorithm are the following. First, the evaluation of union graph patterns should take advantage of multi-way joins and their efficiency in evaluating basic graph patterns [16]. To this end, we rely on the definitions provided in Sect. 2.3 to rewrite each query generated by $\mathcal{ALC}$ class expressions to a semantically equivalent query that is in UNION normal form. Second, instead of waiting for a graph pattern to be fully evaluated, `FILTER NOT EXISTS` patterns should be evaluated as soon as their correlated variables (Sect. 2.3) are bound to a particular term.

4.1 Rewriting Rule for Negation and UNION Normal Form

As discussed in Sect. 2.3, a graph pattern $((P_1\ \mathit{FILTER}\ R)\ \ \mathit{AND}\ P_2)$ can be rewritten to a semantically equivalent graph pattern $((P_1\ \mathit{AND}\ P_2)\ \mathit{FILTER}\ R)$, if every variable of R is also a certain variable of P_1 or R and P_2 do not share any variables [24]. The above rewriting rule is also applicable to `FILTER NOT EXISTS` patterns. Note that the SPARQL standard treats `FILTER NOT EXISTS` patterns as filter expressions.

Proposition 1. *Let $P = ((P_1\ \mathit{FNE}\ P_3)\ \ \mathit{AND}\ P_2)$ and $P' = ((P_1\ \mathit{AND}\ P_2)\ \mathit{FNE}\ P_3)$ be fne-safe graph patterns. P and P' are semantically equivalent, if $\mathit{corVars}(P_1\ \mathit{FNE}\ P_3) = \mathit{corVars}((P_1\ \mathit{AND}\ P_2)\ \mathit{FNE}\ P_3)$.*

Proof. *For $[\![P]\!]_G = [\![P']\!]_G$, we need $(([\![P_1]\!]_G \setminus [\![P_3]\!]_G) \bowtie [\![P_2]\!]_G) = (([\![P_1]\!]_G \bowtie [\![P_2]\!]_G) \setminus [\![P_3]\!]_G)$. As per the semantics (Sect. 2.3), for the equality to hold, we need $(dom([\![P_1]\!]_G) \cap dom([\![P_3]\!]_G)) = (dom([\![P_1]\!]_G \bowtie [\![P_2]\!]_G) \cap dom([\![P_3]\!]_G))$, which holds because $\mathit{corVars}(P_1\ \mathit{FNE}\ P_3) = \mathit{corVars}((P_1\ \mathit{AND}\ P_2)\ \mathit{FNE}\ P_3)$.*

Intuitively, for the equivalence to hold, any variable of P_2 that does not appear in P_1 should also not appear in P_3. To enable the rewriting of $P = ((P_1\ \mathit{FNE}\ P_3)\ \mathit{AND}\ P_2)$ to a semantically equivalent pattern $P' = ((P_1\ \mathit{AND}\ P_2)\ \mathit{FNE}\ P_3)$, we propose the replacement of each variable of P_3 that is not in $\mathit{corVars}(P_1\ \mathit{FNE}\ P_3)$ with a unique variable that is not used in P. This way, we ensure that P_2 and P_3 do not share any variables that do not appear in P_1.

By following the SPARQL standard and treating `FILTER NOT EXISTS` patterns as `FILTER` expressions, we can apply the UNION normal form provided for *AND-UNION-FILTER* graph patterns (Sect. 2.3) to SPARQL queries generated by $\mathcal{ALC}$ class expressions. By applying the UNION normal form and the rewriting rule proposed above to the queries of Table 2, we end up dealing with queries that are disjunctions of graph patterns, where each graph pattern is either a BGP or a set of triple patterns alongside patterns of negation. Our proposed algorithm, which is presented below, is able to evaluate the resulting queries by carrying out a series of multi-way joins.

Algorithm 2. Multi-Way Join for Class Expression Learning in $\mathcal{ALC}$

```
1: function MWJ($P$, $G$, $X$)    ▷ $P$: Graph Pattern, $G$: RDF Graph, $X$: Mapping
2:     // $P$ is a single graph pattern or a $\mathit{UNION}$ of $\mathit{UNION}$-free graph patterns
3:     for all UNION-free patterns $P_i$ of $P$ do
4:         if $P_i$ is a BGP then yield all GenericJoin($P_i$, $G$, $X$)    ▷ Algorithm 1
5:         else yield all MWJFNE($P_i$, $G$, $X$)    ▷ $P_i$ contains negation
6: function MWJFNE($P$, $G$, $X$) ▷ $P$: Graph Pattern, $G$: RDF Graph, $X$: Mapping
7:     $P' \leftarrow P$    ▷ Copy pattern to preserve original
8:     for all $\mathit{FNE}$ patterns $P_i$ of $P$ do
9:         Let $P_i = P_l\ \mathit{FNE}\ P_r$    ▷ $P_l$ is a set of triple patterns
10:        if $\mathit{corVars}(P_i) = \emptyset$ then    ▷ Correlated variables are evaluated (fne-safety)
11:            if EvalFNE($P_r$, $G$, $X$) is false then
12:                return ▷ The current mapping $X$ does not yield a solution mapping
13:            else remove ($\mathit{FNE}\ P_r$) from $P'$    ▷ successfully evaluated
14:    // All $\mathit{FNE}$ patterns that were evaluated returned true and were removed
15:    if there are no more $\mathit{FNE}$ patterns in $P'$ then
16:        yield all GenericJoin($P'$, $G$, $X$) and return
17:    $?x \leftarrow$ a variable from $\mathit{cVars}(P')$    ▷ Select a certain variable to be evaluated
18:    // Evaluate the selected variable $?x$ using the triple patterns of $P'$
19:    $K \leftarrow \bigcap_{tp \in P' \mid ?x \in var(tp)} \{\mu(?x) \mid \mu \in [\![tp]\!]_G\}$
20:    for all $k \in K$ do    ▷ Iterate over the possible values of $?x$
21:        $X(?x) \leftarrow k$    ▷ Store $k$ in the solution mapping
22:        $P'' \leftarrow$ assign $k$ to all occurrences of $?x$ in the triple patterns of $P'$
23:        yield all MWJFNE($P''$, $G$, $X$)  ▷ after yielding proceeds with the next $k$
24: function EvalFNE($P$, $G$, $X$)
25:    // Uses the values assigned to the correlated variables to evaluate $P$
26:    $P' \leftarrow \forall ?v \in \mathit{var}(P) \cap \mathit{dom}(X)$, assign $X(?v)$ to $?v$ in the triple patterns of $P$
27:    $X' \leftarrow$ new empty mapping ; $X'(?v) \leftarrow X(?v)$
28:    if MWJ($P'$, $G$, $X'$) yields a solution then return false
29:    else return true
```

4.2 Multi-way Join Algorithm

The proposed algorithm for the evaluation of SPARQL queries generated by $\mathcal{ALC}$ class expressions is shown in Algorithm 2. The algorithm's entry point is the function `MWJ` (lines 1–5), which takes a graph pattern that is already in UNION normal form as input. `MWJ` iterates over all UNION-free graph patterns P_i of the input graph pattern P and for each P_i calls the appropriate function for its evaluation. If P_i is a BGP, it is evaluated by Generic Join (line 4), otherwise it is evaluated by the function `MWJFNE` (line 5), which is responsible for the evaluation of graph patterns containing `FILTER NOT EXISTS` patterns.

The function `MWJFNE` (lines 6–23) iterates first over the `FILTER NOT EXISTS` patterns ($P_i = P_l\ FNE\ P_r$) of the provided graph pattern P (lines 8–13). For each P_i having its correlated variables evaluated (lines 11–13), the function `EvalFNE` is called (line 11). `EvalFNE` (lines 24–29) checks whether the current solution mapping X is a solution of P_i. This is done by evaluating P_r (line 28) after all the occurrences of the correlated variables of P_i are replaced with their corresponding term in the triple patterns of P_r (line 26). If the evaluation of P_r yields at least one solution, `EvalFNE` is evaluated to false, which leads to the active solution mapping in `MWJFNE` to be discarded (line 13). Recall that the mappings of P_l and P_r should not be compatible (Sect. 2.3). If all `FILTER NOT EXISTS` patterns are successfully evaluated, `MWJFNE` proceeds with the evaluation of P' (lines 15–23). Note that successfully evaluated `FILTER NOT EXISTS` patterns are removed from P' (line 13) and hence, are not evaluated multiple times. If P' is a BGP, it is evaluated by Generic Join (line 16). Otherwise, the evaluation proceeds in a similar fashion to Generic Join (lines 17–23). If there are remaining `FILTER NOT EXISTS` patterns in P', the evaluation focuses on the certain variables of P', i.e., on the variables appearing only in triple patterns (line 19). For each possible value of the selected certain variable (lines 20–23), `MWJFNE` is called recursively.

The proposed algorithm integrates negation in multi-way join plans by calling `MWJ` within `EvalFNE` (line 28). In addition, provided a solution mapping, it does not completely evaluate the right hand side of `FILTER NOT EXISTS` patterns. Instead, it terminates its evaluation once a solution mapping is found, thus avoiding storing intermediate results. The use of the UNION normal form may result in a larger number of joins. However, this can be beneficial, as the joins may limit the intermediate mappings generated by the original union patterns.

4.3 Implementation

We have implemented the proposed algorithm within the tensor-based triple store Tentris [5]. We chose Tentris because it supports multi-way joins [5] and achieves state-of-the-art performance in the evaluation of basic graph patterns [6]. However, Tentris is not able to evaluate SPARQL queries generated by $\mathcal{ALC}$ class expressions, due to its missing support for `FILTER NOT EXISTS` patterns. By implementing our proposed algorithm within Tentris, we further improve the state of the art, as demonstrated by the experimental results (Sect. 5). The performance of multi-way joins is affected by the order in which variables are evaluated [15]. Tentris dynamically selects a variable at each recursive step of the algorithm using cardinality estimations. We modified its selection process to take the number of `FILTER NOT EXISTS` patterns a particular variable appears into account to break ties (i.e., provided two variables with the same cardinality estimation, the one that appears in a `FILTER NOT EXISTS` pattern is selected). Last, we apply the UNION normal form to the provided queries while parsing them. Henceforth, we refer to our implementation as TentrisALC.

5 Experimental Results

We evaluated the performance of our implementation using SPARQL queries corresponding to $\mathcal{ALC}$ class expressions on five datasets of varying sizes. The experiments that are presented below were carried out on a Debian 10 server with an AMD EPYC 7282 CPU, 256GB RAM and a 2TB Samsung 970 EVO Plus SSD. Supplementary material—including datasets, binaries, queries, scripts, and configurations—is available online.[4]

5.1 Systems, Setup and Execution

As the learning of class expressions using SPARQL is carried out over HTTP [7], we compared the performance of TentrisALC against the performance of the following triple stores that provide a SPARQL compliant HTTP endpoint: (i) Blazegraph 2.1.6.RC, (ii) Fuseki 4.10.0, (iii) GraphDB 10.3.3, and (iv) Virtuoso 7.2.10. In our experiments, we also wanted to include MilleniumDB[5], commit: 442e650 [29], which uses multi-way joins for the evaluation of basic graph patterns. However, we did not include it, as it does not evaluate queries having union graph patterns within `FILTER NOT EXISTS` patterns correctly. Each triple store was configured following its respective documentation. The experiments were executed over HTTP using the benchmark execution framework IGUANA 3.3.3 [9]. For each dataset, each query was executed once during the warmup phase. After the warmup phase, the sets of queries were executed three consecutive times. The query timeout was set to three minutes. As in [6], we measured the performance of the triple stores using the following metrics: (i) QPS, i.e., the number of queries executed per second, (ii) pAvgQPS, i.e., the penalized average QPS and (iii) QMPH, i.e., the number of query mixes executed per hour. A query mix is a set or a multiset of queries. The metric QMPH captures the number of times a particular query mix is evaluated in an hour. This means that the lower the runtimes of individual queries are, the higher the QMPH value is. Queries that failed (e.g., timed out or returned an error code) are penalized with a runtime of three minutes.

[4] https://github.com/dice-group/alc2sparql-bench.
[5] https://github.com/MillenniumDB/MillenniumDB.

5.2 Datasets and Queries

Our evaluation comprises five datasets (i.e., knowledge graphs). Four of these datasets, namely Carcinogenesis, Mutagenesis, Premier League and Vicodi, are frequently used to evaluate class expression learning algorithms (e.g., [13,17]) and are available in [17,30]. The largest of these datasets, namely Premier League, contains 2.1M triples. To evaluate the scalability of our approach, we used a subset of the English version of YAGO4 [26], which contains more than 40M triples. As in previous works that follow closed-world semantics (e.g., [7,13]), we first materialized the inferences of the knowledge graphs. Table 3 reports the statistics of the knowledge bases after the materialization process. For the class expression learning datasets, we generated 300 unique $\mathcal{ALC}$ class expressions using a slightly modified version of the learning problem generator of [17]. This modified version does not prioritize simple class expressions. For YAGO4English, due to the generator not scaling to its size, we randomly created 300 unique class expressions by recursively applying the construction rules of $\mathcal{ALC}$. During the generation of class expressions for YAGO4English, we focused only on the properties of the dataset that come from schemas.org (i.e., we did not consider properties coming from bioschemas.org or the RDF/S vocabulary). Schema.org has a richer taxonomy than bioschemas.org, which resulted in $\forall r.C$ and $\exists r.C$ class expressions being more diverse. All class expressions were mapped to SPARQL queries using the mapping of Table 2. For YAGO4English, we ended up having to remove eight queries, as IGUANA was not able to handle them properly (non-escaped characters in IRIs). The last column of Table 3 shows the number of queries having at least one `FILTER NOT EXISTS` pattern.

Table 3. The statistics of the datasets used in the evaluation

	#Triples	#Distinct Subjects	#Distinct Predicates	#Distinct Objects	# Queries \w Negation
Carcinogenesis	157K	22.5K	25	23.2K	169
Mutagenesis	96K	14.2K	16	15K	208
Premier League	2.1M	11.5K	217	12.5K	209
Vicodi	405K	33.4K	14	35.2K	137
YAGO4English	40.2M	7.2M	104	3M	209

5.3 Results and Discussion

The results of our evaluation on the datasets for class expression learning are shown in Tables 4 and 5. The results on YAGO4English are shown in Table 6. Many queries of YAGO4English return more than 7M results, due to their corresponding class expression covering the whole set of individuals. As Virtuoso has a limit of 2^{20} results [5], we did not evaluate it on YAGO4English.

Table 4. The results on class expression learning datasets (cold run). The column failed reports the number of queries for which the corresponding system failed (e.g., timed out) at least once.

	Carcinogenesis			Mutagenesis		
	QMPH	pAvgQPS	failed	QMPH	pAvgQPS	failed
Blazegraph	68.122	49.221	0	40.911	31.637	0
Fuseki	43.426	96.789	0	28.580	58.097	0
GraphDB	28.003	139.800	0	21.156	102.190	0
TentrisALC (ours)	**1410.674**	**792.274**	**0**	**1128.076**	**545.604**	**0**
Virtuoso	148.622	372.766	0	113.312	268.588	0
	Premier League			Vicodi		
	QMPH	pAvgQPS	failed	QMPH	pAvgQPS	failed
Blazegraph	16.832	44.261	0	2.508	48.152	0
Fuseki	3.254	85.898	2	1.399	101.890	9
GraphDB	5.708	139.975	0	1.756	128.904	5
TentrisALC (ours)	**2463.874**	**902.343**	**0**	**715.107**	**773.800**	**0**
Virtuoso	82.245	390.305	0	6.227	411.529	0

The results show that TentrisALC achieves the highest pAvgQPS and QMPH values across all datasets. To ensure the that difference in the performance is statistically significant, we performed the Wilcoxon signed-rank test using the penalized QPS values achieved by each system in each query. The null hypothesis (i.e., the performances of TentrisALC and the baseline systems come from the same distribution) was rejected for all systems in all datasets (p-value $p < 0.001$).

The scalability of our approach can already be observed in the datasets for class expression learning. Table 5 shows that, in the smaller datasets for class expression learning, TentrisALC achieves a value of QMPH that is 10 times higher than the second best system, namely Virtuoso. In Premier League and Vicodi, TentrisALC performs 37 and 126 times better than Virtuoso in terms of QMPH, respectively. This is due to TentrisALC being able to efficiently evaluate queries having `FILTER NOT EXISTS` patterns. More specifically, in Vicodi, TentrisALC achieves a QMPH value that is 7.6 times higher than the second best system (Virtuoso) in the set of queries that do not have any `FILTER NOT EXISTS` patterns. In the set of queries that contain negation, TentrisALC achieves a QMPH value that is 128 times higher than the second best value (Virtuoso). This shows that the overall results are mostly affected by the systems' ability to efficiently evaluate queries with `FILTER NOT EXISTS` patterns. This observation becomes more evident in YAGO4English (Table 6), where all systems apart from TentrisALC timed out in multiple queries. In particular, Fuseki and GraphDB timed out in more than half of the queries. More importantly, the timeouts occurred only in queries that contain `FILTER NOT EXISTS` patterns. As mentioned earlier, many queries in YAGO4English return more than 7M results. In

Table 5. The results on class expression learning datasets (warm runs). The column failed reports the number of queries for which the corresponding system failed (e.g., timed out) at least once.

	Carcinogenesis			Mutagenesis		
	QMPH	pAvgQPS	failed	QMPH	pAvgQPS	failed
Blazegraph	72.920	58.134	0	42.615	37.090	0
Fuseki	45.509	133.754	0	28.940	82.716	0
GraphDB	31.263	246.155	0	22.802	162.238	0
TentrisALC (ours)	**1605.557**	**1571.377**	**0**	**1224.473**	**1015.727**	**0**
Virtuoso	149.928	734.093	0	114.822	532.871	0
	Premier League			Vicodi		
	QMPH	pAvgQPS	failed	QMPH	pAvgQPS	failed
Blazegraph	17.303	51.415	0	2.524	58.075	0
Fuseki	3.237	110.457	2	1.381	144.729	14
GraphDB	5.823	257.314	0	1.763	220.353	5
TentrisALC (ours)	**3176.005**	**1984.633**	**0**	**761.052**	**1554.764**	**0**
Virtuoso	84.486	1150.088	0	5.991	762.763	0

Table 6. The results on the largest dataset, namely YAGO4English. The column failed reports the number of queries for which the corresponding system failed (e.g., timed out) at least once. Virtuoso is not included due to its hard limit of 2^{20} results.

	YAGO4English (cold run)			YAGO4English (warm runs)		
	QMPH	pAvgQPS	failed	QMPH	pAvgQPS	failed
Blazegraph	0.131	18.275	58	0.131	25.985	66
Fuseki	0.116	36.425	168	0.116	66.201	168
GraphDB	0.123	46.120	161	0.123	100.297	161
TentrisALC (ours)	**1.342**	**207.281**	**0**	**1.351**	**435.361**	**0**

queries with large result sets, the runtime can be heavily impacted by the results' enumeration. An example of a class expression that captures the efficiency of our approach is $\neg(\forall exampleOfWork.Prueba\_Villafranca\_de\_Ordizia)$. The corresponding SPARQL query includes three nested `FILTER NOT EXISTS` patterns and returns only 161 solutions (i.e., the results' enumeration did not impact the query's execution time). TentrisALC required 8.3 s on average to evaluate this query, whereas all other systems timed out.

The efficiency of our approach lies in its ability to evaluate `FILTER NOT EXISTS` patterns without having to materialize intermediate results and its ability to terminate the evaluation of such patterns once a single mapping is found. Regarding the size of the datasets, to the best of our knowledge, recent works on class expression learning have focused only on small scale datasets that contain up to a few million triples (e.g., Premier League). As demonstrated above, our approach is able to scale to large datasets and we believe that it will enable learning algorithms to be deployed on large scale knowledge graphs.

6 Related Work

Class expression learning has been extensively investigated (e.g., [12,18,27]). In recent years, several works have focused on accelerating the process of learning class expressions. For instance, Westphal et al. [31] accelerate class expression learning by introducing a heuristic function that is based on simulated annealing. With their meta-heuristics, they are able to reduce the number of instance retrieval operations that are required to reach a goal concept. Kouagou et al. [17] accelerate class expression learning by accurately predicting lengths of possible goal concepts. By this, they avoid instance retrieval operations on lengthy concepts. In [11], the authors employ deep reinforcement learning to learn non-myopic heuristic functions, i.e., heuristic functions that take future rewards into account. These non-myopic heuristic functions accelerate class expression learning, as they are able to efficiently steer the search process towards goal states. Our work builds upon [7] that showed that class expression learning can be accelerated by converting class expressions to SPARQL queries. Our work focuses on improving the runtimes of retrieval operations and hence can be used in combination with the approaches described above.

Hogan et al. [15] were the first to formalize worst-case optimal join algorithms for the evaluation of basic graph pattern SPARQL queries. Several works have employed such algorithms for the evaluation of conjunctive queries [2,5,15] and demonstrated their efficiency. Recently, a multi-way join algorithm for the evaluation of conjunctive regular path queries based on the evaluation process of worst-case optimal join algorithms was proposed in [16]. This algorithm also makes use of the UNION normal form for AND-UNION patterns (i.e., it does not consider filters). One of the first worst-case optimal multi-way join algorithms was presented in [28]. As mentioned in Sect. 4, a sketch for the evaluation of Datalog rules containing negation is provided in [28]. However, to the best of our knowledge, there have not been any works for SPARQL that integrate the evaluation of negation in multi-way join plans.

7 Conclusion And Future Work

We presented a multi-way join algorithm for the evaluation of SPARQL queries corresponding to $\mathcal{ALC}$ class expressions. The main characteristic of our algorithm is the inclusion of `FILTER NOT EXISTS` patterns (i.e., negation) in multi-way join plans. Its purpose is to accelerate class expression learning in $\mathcal{ALC}$ by reducing the runtimes of instance retrieval operations. The experimental results on five datasets show that our implementation outperforms its competition across all datasets and that it is the only one scaling to the largest dataset, which contains more than 40M triples. In the future, we will extend our approach to support more expressive description logics (e.g., $\mathcal{SROIQ}$).

Acknowledgments. This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 860801 and Horizon Europe research and innovation programme under grant agreement No 101070305. It has also been supported by the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW) within the project SAIL under the grant no NW21-059D.

References

1. Angles, R., Gutierrez, C.: Negation in SPARQL. In: Proceedings of the 10th Alberto Mendelzon International Workshop on Foundations of Data Management, Panama City, Panama, 8–10 May 2016, vol. 1644. CEUR Workshop Proceedings (2016)
2. Arroyuelo, D., Hogan, A., Navarro, G., Reutter, J.L., Rojas-Ledesma, J., Soto, A.: Worst-case optimal graph joins in almost no space. In: SIGMOD 2021: International Conference on Management of Data, Virtual Event, China, 20–25 June 2021, pp. 102–114 (2021)
3. Atserias, A., Grohe, M., Marx, D.: Size bounds and query plans for relational joins. In: 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, 25–28 October 2008, Philadelphia, PA, USA, pp. 739–748 (2008)
4. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
5. Bigerl, A., Conrads, F., Behning, C., Sherif, M.A., Saleem, M., Ngonga Ngomo, A.-C.: Tentris – a tensor-based triple store. In: Pan, J.Z., et al. (eds.) ISWC 2020. LNCS, vol. 12506, pp. 56–73. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-62419-4_4
6. Bigerl, A., Conrads, L., Behning, C., Saleem, M., Ngomo, A.N.: Hashing the hypertrie: space- and time-efficient indexing for SPARQL in tensors. In: Sattler, U., et al. (eds.) The Semantic Web - 21st International Semantic Web Conference, Virtual Event, 23–27 October 2022, Proceedings. LNCS, vol. 13489, pp. 5–73. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19433-7_4
7. Bin, S., Bühmann, L., Lehmann, J., Ngomo, A.N.: Towards SPARQL-based induction for large-scale RDF data sets. In: ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August–2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016). FAIA, vol. 285, pp. 1551–1552 (2016)
8. Burnett, M.: Explaining AI: fairly? well? In: Proceedings of the 25th International Conference on Intelligent User Interfaces, pp. 1–2 (2020)
9. Conrads, F., Lehmann, J., Saleem, M., Morsey, M., Ngonga Ngomo, A.-C.: Iguana: a generic framework for benchmarking the read-write performance of triple stores. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10588, pp. 48–65. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68204-4_5
10. d'Amato, C.: Machine learning for the semantic web: lessons learnt and next research directions. Semant. Web **11**(1), 195–203 (2020)
11. Demir, C., Ngomo, A.N.: Neuro-symbolic class expression learning. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th–25th August 2023, Macao, SAR, China, pp. 3624–3632 (2023)

12. Fanizzi, N., d'Amato, C., Esposito, F.: DL-FOIL concept learning in description logics. In: Železný, F., Lavrač, N. (eds.) ILP 2008. LNCS (LNAI), vol. 5194, pp. 107–121. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85928-4_12
13. Heindorf, S., et al.: EvoLearner: learning description logics with evolutionary algorithms. In: WWW 2022: The ACM Web Conference 2022, Virtual Event, Lyon, France, 25 April–29 2022, pp. 818–828. ACM (2022)
14. Hogan, A., et al.: Knowledge graphs. ACM Comput. Surv., 71:1–71:37 (2021)
15. Hogan, A., Riveros, C., Rojas, C., Soto, A.: A worst-case optimal join algorithm for SPARQL. In: Ghidini, C., et al. (eds.) ISWC 2019. LNCS, vol. 11778, pp. 258–275. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30793-6_15
16. Karalis, N., Bigerl, A., Heidrich, L., Sherif, M.A., Ngomo, A.N.: Efficient evaluation of conjunctive regular path queries using multi-way joins. In: Meroño Peñuela, A., et al. (eds.) ESWC 2024, Part I. LNCS, vol. 14664, pp. 218–235. Springer, Cham (2024)
17. Kouagou, N.J., Heindorf, S., Demir, C., Ngomo, A.N.: Learning concept lengths accelerates concept learning in ALC. In: Groth, P., et al. (eds.) ESWC 2022. LNCS, vol. 13261, pp. 236–252. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-06981-9_14
18. Lehmann, J.: Learning OWL Class Expressions, Studies on the Semantic Web, vol. 6. IOS Press (2010)
19. Ngo, H.Q., Porat, E., Ré, C., Rudra, A.: Worst-case optimal join algorithms. J. ACM, 16:1–16:40 (2018)
20. Ngo, H.Q., Ré, C., Rudra, A.: Skew strikes back: new developments in the theory of join algorithms. SIGMOD Rec. **42**(4), 5–16 (2013)
21. Pérez, J., Arenas, M., Gutierrez, C.: Semantics and complexity of SPARQL. ACM Trans. Database Syst. **34**(3), 16:1–16:45 (2009)
22. Rudolph, S.: Foundations of description logics. In: Polleres, A., et al. (eds.) Reasoning Web 2011. LNCS, vol. 6848, pp. 76–136. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23032-5_2
23. Sarker, M.K., Hitzler, P.: Efficient concept induction for description logics. In: The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, 27 January–1 February 2019, pp. 3036–3043 (2019)
24. Schmidt, M., Meier, M., Lausen, G.: Foundations of SPARQL query optimization. In: Segoufin, L. (ed.) Database Theory - ICDT 2010, 13th International Conference, Lausanne, Switzerland, 23–25 March 2010, Proceedings, pp. 4–33. ACM International Conference Proceeding Series. ACM (2010)
25. Schmidt-Schauß, M., Smolka, G.: Attributive concept descriptions with complements. Artif. Intell. **48**(1), 1–26 (1991)
26. Pellissier Tanon, T., Weikum, G., Suchanek, F.: YAGO 4: a reason-able knowledge base. In: Harth, A., et al. (eds.) ESWC 2020. LNCS, vol. 12123, pp. 583–596. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49461-2_34
27. Tran, A.C., Dietrich, J., Guesgen, H.W., Marsland, S.: Parallel symmetric class expression learning. J. Mach. Learn. Res. **18**, 64:1–64:34 (2017)
28. Veldhuizen, T.L.: Triejoin: a simple, worst-case optimal join algorithm. In: Proceedings 17th International Conference on Database Theory (ICDT), Athens, Greece, 24–28 March 2014, pp. 96–106. OpenProceedings.org (2014)
29. Vrgoč, D., et al.: MillenniumDB: an open-source graph database system. Data Intell., 1–39 (2023)

30. Westphal, P., Bühmann, L., Bin, S., Jabeen, H., Lehmann, J.: SML-bench - a benchmarking framework for structured machine learning. Semant. Web **10**(2), 231–245 (2019)
31. Westphal, P., Vahdati, S., Lehmann, J.: A simulated annealing meta-heuristic for concept learning in description logics. In: Katzouris, N., Artikis, A. (eds.) ILP 2021. LNCS, vol. 13191, pp. 266–281. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-97454-1_19

LayeredLiNGAM: A Practical and Fast Method for Learning a Linear Non-gaussian Structural Equation Model

Hirofumi Suzuki(✉)

Artificial Intelligence Laboratory, Fujitsu Limited, Tokyo, Japan
suzuki-hirofumi@fujitsu.com

Abstract. Structural equation models (SEMs) have been widely used to analyze causal relationships between variables via graphs. Linear non-Gaussian acyclic model (LiNGAM) is a type of SEM mainly assuming that the graph is a directed acyclic graph (DAG), the relationships are linear, and the noises follow non-Gaussian distributions. DirectLiNGAM is a popular LiNGAM learning method with applications in various fields. However, DirectLiNGAM has computational difficulty on large-scale data with many variables. In this study, we point out that the bottleneck of DirectLiNGAM is in estimating a *causal order* of variables. We also propose an algorithm that improves the computational complexity of estimating a causal order. The main idea is to construct a DAG from multiple *layers*, and we name the algorithm *LayeredLiNGAM*. As a result, the computational complexity of estimating a causal order is reduced from $O(Cd^3)$ to $O((C + d)Td^2)$. We here denote the number of variables by d and the number of detected layers by T. Furthermore, C is the computational complexity required to compute independence between two variables. Experimental results show that LayeredLiNGAM is faster than DirectLiNGAM without quality degradation of learned DAGs on synthetic and real-world datasets.

Keywords: Structural Equation Model · Linear Non-Gaussian Acyclic Model · LiNGAM · Causal Discovery

1 Introduction

Structural equation models (SEMs) have been widely used to analyze causal relationships between variables via graph structures, i.e., the nodes correspond one-to-one to the variables and the arcs indicate causal relationships. There are wide range of applications including pharmacy [8], epidemiology [9], biology [10], and genetics [14]. Among various SEM formulations [5,11,13,15], *Linear non-Gaussian acyclic model* (LiNGAM) [11] is a popular variant. LiNGAM is a simple SEM by the assumption that the graph is a directed acyclic graph (DAG), the relationships are linear, the noises follow non-Gaussian distributions with zero means and non-zero variances, and the noises are independent of each other.

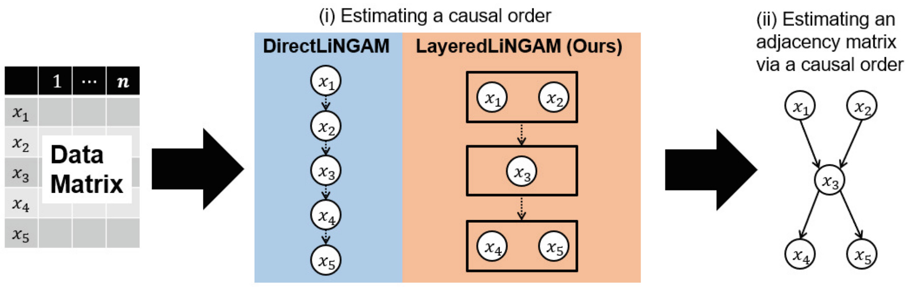

(i) Estimating a causal order
DirectLiNGAM
LayeredLiNGAM (Ours)
(ii) Estimating an adjacency matrix via a causal order
1
⋯
n
Data Matrix
x_1
x_2
x_3
x_4
x_5

Fig. 1. The concept of LayeredLiNGAM.

DirectLiNGAM [12] is a representative method for learning LiNGAM. The algorithm consists of two steps (i) estimating a *causal order* and (ii) estimating a weighted adjacency matrix of the DAG. Especially, DirectLiNGAM has validity by a theoretical framework derived from some observations of LiNGAM. However, DirectLiNGAM has difficulty with large-scale data having many variables in terms of computation time. This paper points out that the bottleneck of DirectLiNGAM is in estimating a causal order from both theoretical and empirical aspects. Furthermore, we aim to improve its computational complexity.

As a main idea, we take advantage of the fact that we can decompose a DAG into multiple *layers*. Introducing the concept of layers, we can regard DirectLiNGAM as estimating a causal order with the restriction that each layer has exactly one node. Therefore, we generalize the theoretical framework of DirectLiNGAM and propose a new algorithm so that each layer is allowed to consist of multiple nodes. It improves the computational complexity of estimating a causal order from $O(Cd^3)$ to $O((C+d)Td^2)$. We here denote the number of variables by d and the number of detected layers by T. In addition, C is the computational complexity required to compute independence between two variables. Note that the obvious lower bound of C is the sample size of the input data. Furthermore, the sample size should be greater than d for accurate learning. Thus, our improvement is more effective when the DAG consists of fewer layers relative to the number of nodes.

Contributions. In this paper, we propose *LayeredLiNGAM* as a practical and fast learning method for LiNGAM to handle large-scale data consisting of many variables. The contributions of this paper are as follows:

– We show that the bottleneck of DirectLiNGAM is the computation time for estimating a causal order both theoretically and empirically. In the theoretical aspect, we discuss the computational complexity of DirectLiNGAM. In the empirical aspect, we show experimental evidence.
– We introduce the concept of layers and generalize the theoretical framework that guarantees the validity of DirectLiNGAM. From the generalization, we derive a new LiNGAM learning algorithm named LayeredLiNGAM. Figure 1 shows the concept of LayeredLiNGAM briefly.

- We also discuss the computational complexity of LayeredLiNGAM and show its superiority compared to DirectLiNGAM. Furthermore, by experiments on synthetic datasets, we show that LayeredLiNGAM faithfully follows its computational complexity.
- Our experimental results show that LayeredLiNGAM performs faster than DirectLiNGAM without degrading the quality of learned DAGs in many cases and with improving the quality in some cases. The speedups are more than 25 and 5 times for synthetic and real-world datasets.

2 Related Work

There are some existing studies that have leveraged the concept of layers in learning SEMs [4,17]. While they can handle more general SEMs than the linear model, they require some properties in the variances of variables conditioned on their parent variables: all the variables have an equal conditional variance [4], or each conditional variance is derived from a quadratic function of the conditional mean [17]. Such properties of conditional variances are unrealistic as the number of variables increases. Furthermore, the method of Zhou et al. [17] has a disadvantage that the algorithm needs some modifications depending on the assumed distribution of variables. Although our LayeredLiNGAM leverages the concept of layers, it requires no additional assumption beyond LiNGAM and no process modification depending on the assumed distribution of variables.

Ruixuan et al. [16] have leveraged the concept of layers in learning LiNGAM. Hereinafter, we refer to this method as RuixuanLiNGAM in this paper. RuixuanLiNGAM also requires no additional assumption beyond LiNGAM and no adaptive modification of the algorithm. The difference between RuixuanLiNGAM and LayeredLiNGAM is whether the algorithm estimates the layers of DAGs in a bottom-up (starting from sink nodes) or top-down (starting from root nodes) fashion. RuixuanLiNGAM adopts a bottom-up fashion and leverages a different theoretical framework to DirectLiNGAM. In contrast, LayeredLiNGAM adopts a top-down fashion that purely extends the DirectLiNGAM. On the computational complexity, RuixuanLiNGAM takes $O((C + Rd)Td^2)$ time in estimating a causal order where R is the number of steps for a precision matrix estimation such as Graphical LASSO [3]. Therefore, we expect that LayeredLiNGAM is faster than RuixuanLiNGAM. Our experiments focus on the superiority of LayeredLiNGAM compared to DirectLiNGAM and RuixuanLiNGAM.

3 Preliminaries

This section describes LiNGAM and its learning algorithm, DirectLiNGAM. Hereinafter, for simplicity, let $[n] := \{1, \ldots, n\}$ for any positive integer $n \in \mathbb{N}$.

3.1 LiNGAM

LiNGAM is a type of SEM representing a data generation process using DAG. For a d-dimensional random vector $\mathbf{x} = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d$, we consider a weighted adjacency matrix of a DAG with d nodes $\mathbf{B} = (b_{ij})_{d \times d} \in \mathbb{R}^{d \times d}$. Each element b_{ij} represents the direct causal effect from a variable x_j to another x_i. Without loss of generality, we assume that each observed variable x_i has zero mean. Then, the following equation expresses LiNGAM:

$$\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{e}, \tag{1}$$

where $\mathbf{e} = (e_1, \ldots, e_d)^\top \in \mathbb{R}^d$ is a random noise vector. We assume that each e_i follows a non-Gaussian distribution with zero mean and a non-zero variance. Furthermore, all e_i are assumed to be independent of each other. That is, there are no latent confounding variables.

We can order all the variables according to their precedence relationships. We call such an order of variables a *causal order* and denote it by a bijection $\mathcal{CO}\colon [d] \to [d]$. That is, $\mathcal{CO}(i) < \mathcal{CO}(j) \Rightarrow b_{ij} = 0$ holds for any $i, j \in [d]$. Then, the Eq. (1) is equivalent to

$$x_i = \sum_{j \in [d], \mathcal{CO}(j) < \mathcal{CO}(i)} b_{ij} x_j + e_i.$$

Furthermore, if x_i has no precedent variable and its observations satisfy $x_i = e_i$, we call x_i an *exogenous variable*. According to the assumption of acyclicity and no latent confounding variables, a causal order and an exogenous variable exist.

3.2 DirectLiNGAM

DirectLiNGAM [12] is a learning algorithm for LiNGAM. The algorithm consists of two steps (i) estimating a causal order and (ii) estimating an adjacency matrix. We here focus on two known lemmas to guarantee the validity of the step (i). The first lemma indicates the criteria for determining exogenous variables, i.e., how to find a variable that can be first in the causal order. The second lemma indicates that LiNGAM is preserved for the residuals of the simple regressions by one exogenous variable. Those statements are as follows.

Lemma 1 ([12]). *Assume the input data $\mathbf{x}$ follows LiNGAM, i.e., $\mathbf{x}$ satisfies all the model assumptions and has infinite samples. Let $r_i^{(j)}$ be the residual when x_j regreses x_i. That is, $r_i^{(j)} = x_i - \frac{\mathrm{cov}(x_i, x_j)}{\mathrm{var}(x_j)} x_j$ $(i \neq j)$. Then, x_j is an exogenous variable if and only if x_j is independent of all residuals $r_i^{(j)}$ $(i \neq j)$.*

Lemma 2 ([12]). *Assume the input data $\mathbf{x}$ follows LiNGAM. Furthermore, assume that x_j is an exogenous variable. Let $\mathbf{r}^{(j)} := (r_i^{(j)})_{i \neq j}^\top$ be the vector collecting the residuals by x_j for all x_i $(i \neq j)$. Similarly, let $\mathbf{e}^{(j)} := (e_i)_{i \neq j}^\top$. Then, there is a matrix $\mathbf{B}^{(j)}$ such that LiNGAM is preserved on $\mathbf{r}^{(j)} = \mathbf{B}^{(j)}\mathbf{r}^{(j)} + \mathbf{e}^{(j)}$.*

Transforming the Eq. (1), we can write $\mathbf{x} = \mathbf{A}\mathbf{e}$ where $\mathbf{A} = (\mathbf{I} - \mathbf{B})^{-1}$. Note that, because $\mathbf{B}$ represents a DAG and can be permuted to be *strictly* lower triangular, $(\mathbf{I} - \mathbf{B})$ is invertible. $\mathbf{A} = (a_{ij})_{d \times d}$ corresponds to the transitive closure of the DAG represented by $\mathbf{B}$, an element a_{ij} $(i \neq j)$ represents the total causal effect from x_j to x_i, and $a_{ii} = 1$. According to the proof of Lemma 2 [12], we can take $\mathbf{B}^{(j)} = \mathbf{I} - (\mathbf{A}^{(j)})^{-1}$ where $\mathbf{A}^{(j)}$ is the matrix with removing j-th row and j-th column from $\mathbf{A}$. Note that since $\mathbf{A}$ is lower triangular and invertible, $\mathbf{A}^{(j)}$ is also lower triangular and invertible. Furthermore, from the structure of $\mathbf{A}^{(j)}$, we immediately see that the possible causal orders of $\mathbf{x}$ and $\mathbf{r}^{(j)}$ are equivalent by ignoring x_j. Thus, we obtain the following corollary.

Corollary 1 ([12]). *Assume the input data* $\mathbf{x}$ *follows LiNGAM. Furthermore, assume that* x_j *is an exogenous variable. Then, for any causal order* $\mathcal{CO}$ *of* $\mathbf{x}$*, there is a causal order* $\mathcal{CO}_j$ *of* $\mathbf{r}^{(j)}$ *such that* $\mathcal{CO}(k) < \mathcal{CO}(l) \Leftrightarrow \mathcal{CO}_j(k) < \mathcal{CO}_j(l)$ *holds for any* $k \neq j$ *and* $l \neq j$*, i.e.,* $\mathbf{r}^{(j)}$ *preserves the possible causal orders of* $\mathbf{x}$*.*

By Lemma 2 and Corollary 1, we see that a causal order can be estimated by recursively finding an exogenous variable for the residuals. That is, if the algorithm selects x_j as an exogenous variable in the current step, then the algorithm replaces $\mathbf{x}$ by $\mathbf{r}^{(j)}$ before proceeding to the next step and removes x_j. The algorithm repeats the above operation until all variables are selected. As a result, $\mathcal{CO}(i)$ becomes the number of steps before x_i is selected.

As a quantified measure of exogenous variables, i.e., computing independence, DirectLiNGAM leverages mutual information. Let $K \subseteq [d]$ be the subset of indices whose corresponding variables are unordered. For any $i, j \in K$, $M_{ij} := I(x_j, r_i^{(j)}) - I(x_i, r_j^{(i)})$ indicates the precedence between x_i and x_j as follows: x_i precedes x_j if $M_{ij} > 0$, the precedence between x_i and x_j is arbitrary if $M_{ij} = 0$, and x_j precedes x_i if $M_{ij} < 0$. Let $m_i := \sum_{j \in K, i \neq j} \min(0, M_{ij})^2$. Then, because exogenous variables should precede more variables, the algorithm selects x_{j^*} as an exogenous variable for $j^* \in \arg\min_{j \in K} m_j$.

After estimating a causal order $\mathcal{CO}$, the algorithm estimates an adjacency matrix $\mathbf{B}$ as follows: b_{ij} becomes a coefficient of a sparse regression such that the precedent variables $\{x_j \mid \mathcal{CO}(j) < \mathcal{CO}(i)\}$ regress x_i. A recommended sparse regression method is Adaptive LASSO [18]. Algorithm 1 shows the pseudocode of DirectLiNGAM. Note that, $\mathbf{X} \in \mathbb{R}^{d \times n}$ denotes the data matrix of n samples, and $\mathbf{R}^{(j)}$ denotes the matrix collecting the residuals by x_j for $\{x_i\}_{i \neq j, i \in K}$. Furthermore, AdaptiveLASSO($\mathbf{X}_i$, $\mathbf{X}_P$) is the process that returns a sparse coefficient vector resulting from $\{x_j\}_{j \in P}$ regresses x_i.

Computational Complexity. Let us confirm the computational complexity of DirectLiNGAM. The computation of all the residual matrices $\mathbf{R}^{(j)}$ $(j \in K)$ takes $O(n|K|^2)$ time by the simple regression. When the computation of M_{ij} takes $O(C)$ time, the computation of all m_j $(j \in K)$ takes $O(C|K|^2)$ time. Note that $C \geq n$ holds. Thus, estimating a causal order takes $O((n + C)\sum_{k=1}^{d} k^2) = O(Cd^3)$ time. AdaptiveLASSO takes $O(n|P|^2)$ time by using LARS algorithm [2]. Thus, estimating an adjacency matrix takes $O(n \sum_{p=1}^{d} p^2) = O(nd^3)$ time. As a result, DirectLiNGAM takes $O(Cd^3)$ time in total.

Algorithm 1. DirectLiNGAM

```
Input: A data matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$ of $d$ variables and $n$ samples
Output: A weighted adjacency matrix $\mathbf{B} \in \mathbb{R}^{d \times d}$ of a DAG
 1: $K \leftarrow [d]$, $\mathbf{X}^{(1)} \leftarrow \mathbf{X}$
 2: for $t = 1, 2, \dots, d$ do
 3:     Compute $\mathbf{R}^{(j)}$ for all $j \in K$ over $\mathbf{X}^{(t)}$
 4:     Compute $m_j$ for all $j \in K$
 5:     Find $j^* \in \arg\min_{j \in K} m_j$
 6:     $\mathcal{CO}(j^*) \leftarrow t$
 7:     $K \leftarrow K \setminus \{j^*\}$, $\mathbf{X}^{(t+1)} \leftarrow \mathbf{R}^{(j^*)}$
 8: $\mathbf{B} \leftarrow d \times d$ zero matrix
 9: for $i = 1, 2, \dots, d$ do
10:     $P = \{j \in [d] \mid \mathcal{CO}(j) < \mathcal{CO}(i)\}$
11:     $(b_{ij})_{j \in P} \leftarrow$ AdaptiveLASSO($\mathbf{X}_i$, $\mathbf{X}_P$)
12: return $\mathbf{B}$
```

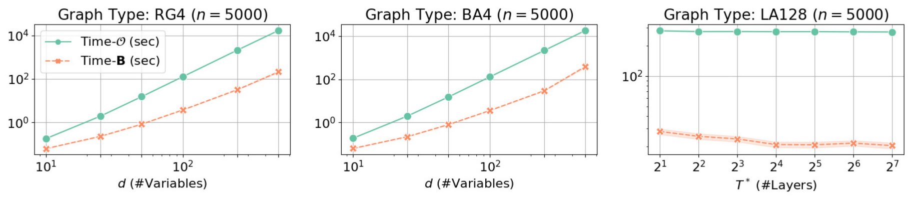


Fig. 2. Computation times of DirectLiNGAM on synthetic datasets. Time-$\mathcal{O}$ and Time-**B** are the computation time for estimating a causal order and an adjacency matrix, respectively. See Sect. 5 for the detailed settings.

Research Question. From the above discussion, we can infer that estimating a causal order (which takes $O(Cd^3)$ time) is dominant in the computation time of DirectLiNGAM. Therefore, our research question is how to speed up estimating a causal order. On the other hand, surprisingly, we obtain the lower bound $C = n$ by computing M_{ij} using maximum entropy approximation [6]. In other words, the theoretical result already appears to be tight. However, our preliminary experiments (see Fig. 2) show that estimating a causal order is a few to 50 times slower than estimating an adjacency matrix. It indicates the presence of overhead in maximum entropy approximation. Therefore, we aim to speed up DirectLiNGAM in practice by improving the term d^3 of $O(Cd^3)$ for estimating a causal order.

4 LayeredLiNGAM

Since DirectLiNGAM determines the causal order for each variable one by one, the term d^3 inevitably appears in the computational complexity. Therefore, we consider ordering multiple variables simultaneously to speed up estimating a causal order. The strategy originates from the decomposition of the DAG into multiple *layers*. Each layer corresponds to a subset of variables whose causal order can be arbitrarily among them.

Now, we decompose the DAG into T layers $L_1, \ldots, L_T \subseteq \{x_1, \ldots, x_d\}$. Note that $L_s \cap L_t = \emptyset$ $(s \neq t)$ and $\bigcup_{t \in [T]} L_t = \{x_i\}_{i \in [d]}$ hold. Then, we introduce a function $\mathcal{LO}\colon [d] \to [T]$ such that $x_i \in L_t \Leftrightarrow \mathcal{LO}(i) = t$ for all $i \in [d]$. We call $\mathcal{LO}$ a *layered causal order* if it satisfies the following equation:

$$x_i = \sum_{j \in [d], \mathcal{LO}(j) < \mathcal{LO}(i)} b_{ij} x_j + e_i.$$

If we restrict $|L_t| = 1$ for any layer $t \in [T]$, then a layered causal order is equivalent to a causal order. Thus, we can say that layered causal order is a generalization of causal order.

We propose a new LiNGAM learning algorithm that estimates a layered causal order $\mathcal{LO}$ instead of a causal order $\mathcal{CO}$. First, to design the algorithm with validity, we generalize Lemma 2 and Corollary 1. Then, we present the new algorithm and discuss its computational complexity with superiority over DirectLiNGAM. We name the proposed algorithm *LayeredLiNGAM*.

4.1 Generalization of Lemma 2

We can regard that Lemma 2 originates from restricting $|L_t| = 1$ for all $t \in [T]$. We show a similar lemma for the general cases such that $|L_t| \geq 1$. Furthermore, by the proof of the new lemma, we immediately obtain a new corollary that generalizes Corollary 1.

Lemma 3. *Assume the input data* $\mathbf{x}$ *follows LiNGAM. Furthermore, assume that* $X_J := \{x_j\}_{j \in J}$ *is a set of exogenous variables where* $J \subseteq [d]$ $(J \neq \emptyset)$. *Let* r_i^J *be the residual when* X_J *regresses* x_i. *Let* $\mathbf{r}^J := (r_i^J)_{i \notin J}^\top$ *be the vector collecting the residuals by* X_J *for all* x_i $(i \notin J)$. *Similarly, let* $\mathbf{e}^J := (e_i)_{i \notin J}^\top$. *Then, there is a matrix* $\mathbf{B}^J$ *such that LiNGAM is preserved on* $\mathbf{r}^J = \mathbf{B}^J \mathbf{r}^J + \mathbf{e}^J$.

Proof. Without loss of generality, we assume that $\mathbf{B}$ is a strictly lower triangular matrix whose diagonal elements are zero, i.e., the rows and columns are arranged according to a causal order. Then, we can assume that $J = \{1, \ldots, k\} \subseteq [d]$. Since X_J is a set of exogenous variables, we have $x_j = e_j$ for all $j \in J$.

Let $\bar{\mathbf{A}}^J$ be a $(d-k) \times k$ matrix that takes the last $(d-k)$ rows and the first k columns of $\mathbf{A}$. Let $\mathbf{A}^J$ be a $(d-k) \times (d-k)$ matrix that takes the last $(d-k)$ rows and the last $(d-k)$ columns of $\mathbf{A}$. Then, we can rewrite the equation $\mathbf{x} = \mathbf{A}\mathbf{e}$ as follows:

$$\mathbf{x} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \bar{\mathbf{A}}^J & \mathbf{A}^J \end{bmatrix} \mathbf{e} \tag{2}$$

From the model assumption, for any $i \in [d] \setminus J$ and $j \in J$, a_{ij} is equal to the regression coefficient when e_j regresses x_i, and we can write $a_{ij} = \frac{\mathrm{cov}(x_i, e_j)}{\mathrm{var}(e_j)}$.

Thus, we have

$$\forall i \in [d] \setminus J,\ r_i^J = x_i - \sum_{j \in J} \frac{\mathrm{cov}(x_i, e_j)}{\mathrm{var}(e_j)} e_j = x_i - \sum_{j \in J} a_{ij} e_j. \tag{3}$$

Let $\mathbf{r} := (e_1, \dots, e_k, r_{k+1}^J, \dots r_d^J)^\top$. From the equations (2) and (3), by removing the causal effects of exogenous variables X_J from the equation $\mathbf{x} = \mathbf{A}\mathbf{e}$, we have

$$\mathbf{r} = \mathbf{x} - \begin{bmatrix} \mathbf{O} & \mathbf{O} \\ \bar{\mathbf{A}}^J & \mathbf{O} \end{bmatrix} \mathbf{e} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \bar{\mathbf{A}}^J & \mathbf{A}^J \end{bmatrix} \mathbf{e} - \begin{bmatrix} \mathbf{O} & \mathbf{O} \\ \bar{\mathbf{A}}^J & \mathbf{O} \end{bmatrix} \mathbf{e} = \begin{bmatrix} \mathbf{I} & \mathbf{O} \\ \mathbf{O} & \mathbf{A}^J \end{bmatrix} \mathbf{e}.$$

In the above equation, X_J does not affect any other variable. Thus, by removing X_J, we obtain $\mathbf{r}^J = \mathbf{A}^J \mathbf{e}^J$. Since $\mathbf{A}$ is lower triangular and invertible, $\mathbf{A}^J$ is also lower triangular and invertible. Thus, we obtain a new SEM $\mathbf{r}^J = \mathbf{B}^J \mathbf{r}^J + \mathbf{e}^J$ where $\mathbf{B}^J = \mathbf{I} - (\mathbf{A}^J)^{-1}$. Furthermore, $\mathbf{B}^J$ represents a DAG due to the structure of $\mathbf{A}^J$. Therefore, the lemma holds.

Corollary 2. *Assume the input data* $\mathbf{x}$ *follows LiNGAM. Furthermore, assume that* $X_J := \{x_j\}_{j \in J}$ *is a set of exogenous variables where* $J \subseteq [d]$ $(J \neq \emptyset)$*. Then, for any layered causal order* $\mathcal{LO}$ *of* $\mathbf{x}$*, there is a layered causal order* $\mathcal{LO}_J$ *of* $\mathbf{r}^J$ *such that* $\mathcal{LO}(k) < \mathcal{LO}(l) \Leftrightarrow \mathcal{LO}_J(k) < \mathcal{LO}_J(l)$ *holds for any* $k, l \notin J$*, i.e.,* $\mathbf{r}^J$ *preserves the possible layered causal orders of* $\mathbf{x}$*.*

4.2 Algorithm

By Lemma 3 and Corollary 2, we see that a layered causal order can be estimated by recursively finding a set of exogenous variables for the residuals. That is, if the algorithm selects X_J as a set of exogenous variables in the current step, then the algorithm replaces $\mathbf{x}$ by $\mathbf{r}^J$ before proceeding to the next step and removes X_J. The algorithm repeats the above operation until all variables are selected. As a result, $\mathcal{LO}(i)$ becomes the number of steps before x_i is selected.

An algorithmic issue is the search for a set of exogenous variables X_J over the unordered variables $\{x_i\}_{i \in K}$. We solve the issue by defining

$$J := \left\{ j \in K \;\middle|\; \sqrt{\frac{m_j}{|K| - 1}} \leq \epsilon \right\},$$

where $\epsilon \in \mathbb{R}_{\geq 0}$ is a threshold parameter. Note that $\sqrt{\frac{m_j}{|K|-1}}$ is the root mean square with respect to $\min(0, M_{ij})$, i.e., its scale is immutable regardless of K.

As a result, X_J contains variables that seem more exogenous. If there is no $j \in K$ such that $\sqrt{\frac{m_j}{|K|-1}} \leq \epsilon$, we set $J = \arg\min_{j \in K} m_j$. After estimating a layered causal order $\mathcal{LO}$, the algorithm estimates an adjacency matrix $\mathbf{B}$. We conduct it by replacing $\mathcal{CO}$ with $\mathcal{LO}$ in line 10 of Algorithm 1. Therefore, we can write the pseudocode of LayeredLiNGAM as Algorithm 2. Note that $\mathbf{R}^J$ is the matrix collecting the residuals by X_J for $\{x_i\}_{i \in K \setminus J}$.

Algorithm 2. LayeredLiNGAM

```
Input: A data matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$ and a threshold parameter $\epsilon$
Output: A weighted adjacency matrix $\mathbf{B} \in \mathbb{R}^{d \times d}$ of a DAG
 1: $K \leftarrow [d]$, $\mathbf{X}^{(1)} \leftarrow \mathbf{X}$, $t \leftarrow 1$
 2: while $K \neq \emptyset$ do
 3:     Compute $\mathbf{R}^{(j)}$ for all $j \in K$ over $\mathbf{X}^{(t)}$
 4:     Compute $m_j$ for all $j \in K$
 5:     $J \leftarrow \left\{ j \in K \,\middle|\, \sqrt{\frac{m_j}{|K|-1}} \leq \epsilon \right\}$
 6:     if $J = \emptyset$ then
 7:         $J \leftarrow \arg\min_{j \in K} m_j$
 8:     $\mathcal{LO}(j) \leftarrow t$ for all $j \in J$
 9:     $K \leftarrow K \setminus J$, $\mathbf{X}^{(t+1)} \leftarrow \mathbf{R}^J$, $t \leftarrow t+1$
10: $\mathbf{B} \leftarrow d \times d$ zero matrix
11: for $i = 1, 2, \ldots, d$ do
12:     $P = \{j \in [d] \mid \mathcal{LO}(j) < \mathcal{LO}(i)\}$
13:     $(b_{ij})_{j \in P} \leftarrow$ AdaptiveLASSO($\mathbf{X}_i$, $\mathbf{X}_P$)
14: return $\mathbf{B}$
```

Computational Complexity. We discuss the computational complexity of estimating a layered causal order. Note that there is no difference in computational complexity of estimating an adjacency matrix between DirectLiNGAM and LayeredLiNGAM. The main difference from DirectLiNGAM is the computation of the residual matrix R^J by multiple regression. However, it can be done in $O(n|J||K| + |J|^2|K|) = O(n|K|^2 + |K|^3)$ time using the least-squares. Thus, when $C \geq |K|$, the bottleneck is still the computation of all m_j $(j \in K)$ that takes $O(C|K|^2)$ time. Let T be the number of iterations in lines 2 through 9 of Algorithm 2. It is equal to the number of layers detected by LayeredLiNGAM. Then, estimating a layered causal order takes $O((n + C)\sum_{t=1}^{T} k_t^2 + \sum_{t=1}^{T} k_t^3) = O((C + d)Td^2)$ time where k_t is the size of K at t-th iteration. Especially, we have $O(CTd^2)$ when $C \geq d$ or $n \geq d$ hold. Since $T \leq d$ holds, we expect that LayeredLiNGAM is faster than DirectLiNGAM when the estimated DAG consists of a small number of layers relative to the number of nodes.

4.3 Adaptive Thresholding

The behavior of LayeredLiNGAM can vary depending on the threshold parameter ϵ. Specifically, as ϵ increases, the algorithm estimates each layer as a larger variable set. Then, as the computation time decreases with ϵ increases, the output DAG is more different from that of DirectLiNGAM. If the input data matrix is always ideal for LiNGAM, we may be able to fix ϵ to a proper constant value. However, it is an unrealistic strategy in practice. Therefore, we consider how to determine a reasonable value of ϵ while running the algorithm.

A simple strategy is to use the minimum meaningful value multiplied by a constant value. That is, we take an alternative constant parameter $\delta \geq 1$

and use $\hat{\epsilon} := \delta \cdot \min_{j\in K}\left\{\sqrt{\frac{m_j}{|K|-1}}\right\}$ as ϵ for each iteration. However, small ϵ leads to no speedup effect, and large ϵ may cause quality degradation. Therefore, we take an upper bound and a lower bound for ϵ: given two constant parameters $\epsilon_{\min}, \epsilon_{\max} \in \mathbb{R}_{\geq 0}$ where $\epsilon_{\min} \leq \epsilon_{\max}$, we compute the bounded value $\epsilon = \min\{\max\{\epsilon_{\min}, \hat{\epsilon}\}, \epsilon_{\max}\}$. In Sect. 5, we experimentally find better values of δ, $\epsilon_{\min}$, and $\epsilon_{\max}$.

5 Experiments

We conducted experiments to confirm the actual behavior of LayeredLiNGAM and its superiority over DirectLiNGAM and RuixuanLiNGAM. Note that we used maximum entropy approximation for computing independence between two variables to achieve $C = n$. The implementation[1] was in Python 3.10 with leveraging `numpy` and `numba` for standard acceleration tools. The computer environment consisted of Ubuntu 22.04.2 LTS, Intel(R) Xeon(R) CPU E5-2667 v4, and 128GB RAM.

Although there is an existing library [7] for DirectLiNGAM, to be fair to the implementation of the other algorithms, we implemented DirectLiNGAM on our own. Furthermore, we have confirmed that our DirectLiNGAM implementation is more than ten times faster than the existing one and still produces comparable output. In implementing RuixuanLiNGAM, we need a subroutine to estimate a precision matrix such as Graphical LASSO [3]. However, because RuixuanLiNGAM with Graphical LASSO was very slow, we used the empirical precision matrix for ease.

5.1 Datasets and Evaluation Metrics

Synthetic Datasets. We used synthetic datasets based on three different types of sparse random DAGs. After generating DAGs, we drew the value b_{ij} of each directed edge uniformly at random from $[-2.0, -0.5] \cup [0.5, 2.0]$. Then, we drew each random noise e_i from a Laplace distribution with mean 0 and variance 1. Finally, we generated samples based on the formula (1). The variations of sparse random DAGs are as follows:

– **RG4** originated from an undirected graph with random $2d$ edges, i.e., the expected degree of each node was four. We orientated each edge according to a randomly generated causal order.
– **BA4** originated from an undirected graph of Barabási-Albert model [1] that added four random edges from a newly generated node to existing nodes. We orientated each edge according to a randomly generated causal order.
– **LA128** was a DAG with $d = 128$ nodes and $T^* \in \{2, 4, 8, 16, 32, 64, 128\}$ layers. Each layer exactly contained $\frac{d}{T^*}$ nodes. We connected each node to just its above and below layers only.

[1] The source code is available at https://doi.org/10.6084/m9.figshare.25425235.v1.

Real-World Datasets. We used gene expression cancer RNA-Seq (GENE for short) dataset[2] and yprop_4_1 (YPROP for short) dataset[3] taken from UCI repository and OpenML, respectively. GENE contains 801 samples of patients and 20531 variables of gene expressions. YPROP is a drug design dataset consisting of 8885 samples and 252 variables. For each dataset, we resampled variables and obtained ten different datasets: First, the variables were classified into d clusters using the hierarchical clustering of Ward's method. Second, we resampled d variables by randomly selecting one variable from each of the d clusters. We set $d = 500$ for GENE dataset and $d = 100$ for YPROP dataset. Finally, we standardized all the resampled datasets.

Evaluation Metrics. We evaluated the competitors by the following metrics.

– **Time-$\mathcal{O}$** and **Time-B** are the computation times of estimating a (layered) causal order and an adjacency matrix, respectively.
– **Precision** and **Recall** are computed as $\frac{|E_{\text{tr}} \cap E_{\text{le}}|}{|E_{\text{le}}|}$ and $\frac{|E_{\text{tr}} \cap E_{\text{le}}|}{|E_{\text{tr}}|}$, respectively. E_{tr} and E_{le} are the edge set of the true DAG and learned DAG, respectively.
– **Structural Hamming Distance (SHD)** is the number of edge insertions, deletions, and flips required to convert the estimated DAG to the true DAG.
– **Bayesian Information Criterion (BIC)** is defined as $-2\ell(\mathbf{X}, \mathbf{B}) + m \log n$ where $\ell(\mathbf{X}, \mathbf{B})$ is the log-likelihood and m is the number of estimated edges.
– **Root Mean Squared Error (RMSE)** is the loss in terms of the regression model and is defined as $\sqrt{\frac{1}{nd}\|\mathbf{X} - \mathbf{BX}\|_2^2}$.

On the quality metrics, higher values are better for Precision and Recall, and lower values are better for SHD, BIC, and RMSE. We used Precision, Recall, and SHD for synthetic datasets because we knew the true DAGs. BIC and RMSE were used for GENE and YPROP datasets because these datasets have no known true DAG.

5.2 Determining Threshold Parameters

Before specific comparisons of the competitors, we experimentally determined reasonable threshold parameters δ, $\epsilon_{\min}$, and $\epsilon_{\max}$ for LayeredLiNGAM. We here observed the behaviors when δ was given and ϵ was fixed, respectively. The candidate values were $\delta \in \{0, 1, \ldots, 20\}$ and $\epsilon \in \{2^{-k} \mid k \in \{0, 1, \ldots, 20\}\}$. Furthermore, we used synthetic datasets derived from RG4/BA4 of $d = 100$ and LA128 of $T^* = 16$ with the fixed sample size $n = 5000$. Figure 3 shows the results.

We focused on parameters with low computation time and high quality. The reasonable interval of δ for each graph type were $[4, 12]$ for RG4, $[11, 18]$ for BA4, and $[9, 13]$ for LA128. Since the intersection of these intervals was $[11, 12]$, we adopted its midpoint as $\delta = 11.5$. The reasonable interval of ϵ was $[2^{-12}, 2^{-8}]$ for all the graph types. Therefore, we set $\epsilon_{\min} = 2^{-12}$ and $\epsilon_{\max} = 2^{-8}$.

[2] https://archive.ics.uci.edu/dataset/401/gene+expression+cancer+rna+seq.
[3] https://www.openml.org/search?type=data&sort=runs&id=416&status=active.

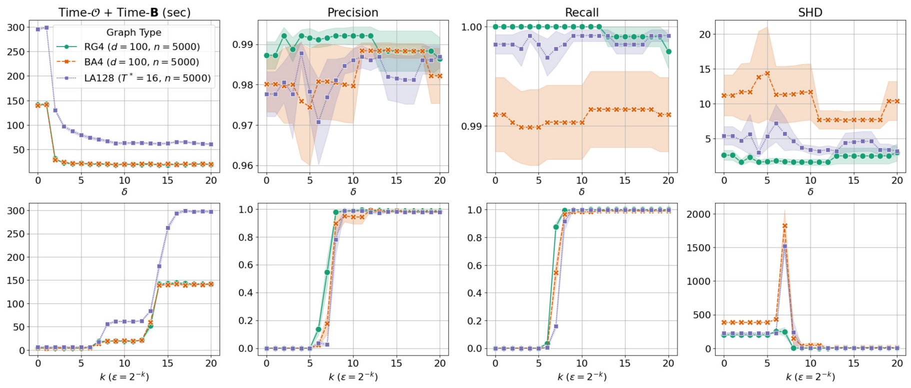


Fig. 3. Results of LayeredLiNGAM with various threshold parameters. We generated ten random instances for each graph type. We plotted the means and standard errors of four evaluation metrics total computation time (the sum of time-$\mathcal{O}$ and Time-$\mathbf{B}$), Precision, Recall, and SHD.

5.3 Results on Synthetic Datasets

We conducted observations over synthetic datasets in the following two ways. (a) Varied the number of variables in RG4/BA4 and the number of layers in LA128 and fixed the sample size to sufficient ($d \in \{10, 25, 50, 100, 250, 500\}$, $T^* \in \{2, 4, 8, 16, 32, 64, 128\}$, and $n = 5000$). (b) Varied the sample size and fixed the number of variables in RG4/BA4 and the number of layers in LA128 ($d = 100$, $T^* = 16$, and $n \in \{10, 25, 50, 100, 250, 500, 1000, 2500, 5000\}$). Figure 4 and 5 show the results of (a) and (b), respectively.

From the results on RG4/BA4 in Fig. 4, estimating causal orders by LayeredLiNGAM was faster than that by DirectLiNGAM and achieved speedups of more than 50 times in the cases of $d = 500$. It was comparable to the computation times for estimating adjacency matrices. Furthermore, LayeredLiNGAM achieved speedups more than 25 times in the total computation times. Since there was no difference among the competitors in the computation times for estimating adjacency matrices, we obtained dramatic impacts of the fast estimation of causal orders by LayeredLiNGAM.

From the results for LA128 in Fig. 4, we observed that the computation times for LayeredLiNGAM were close to that of DirectLiNGAM as the number of layers increased. From Fig. 5, we observed that the acceleration by LayeredLiNGAM did not work well in the cases of $n \leq d$ but worked significantly in the cases of $n > d$. Therefore, we confirmed that LayeredLiNGAM performed faithfully with respect to the theoretical computational complexity $O((C + d)Td^2)$.

Focusing on the quality metrics, LayeredLiNGAM achieved comparable results to DirectLiNGAM in almost all cases and improvements in some cases. From Fig. 5, LayeredLiNGAM achieved sufficient quality with a larger sample size ($n \geq 1000 \simeq 10d$) for any graph type. We observed significant quality degradations in LayeredLiNGAM in only the cases of LA128 with $n \in \{250, 500\}$

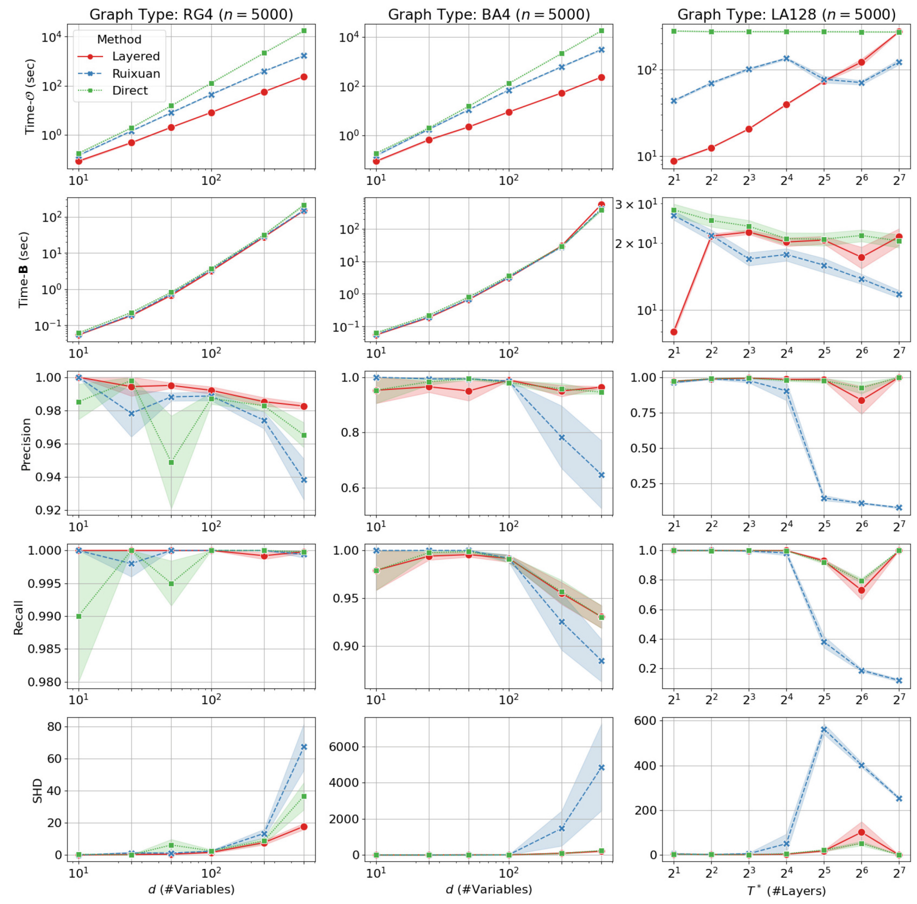


Fig. 4. Results on synthetic datasets that we focused on the number of variables $d \in \{10, 25, 50, 100, 250, 500\}$ and layers $T^* \in \{2, 4, 8, 16, 32, 64, 128\}$ with the fixed sample size $n = 5000$. We generated ten random instances for each setting. We plotted the means and standard errors of five evaluation metrics Time-$\mathcal{O}$, Time-$\mathbf{B}$, Precision, Recall, and SHD.

in Fig. 5. However, since LA128 is a special DAG and sufficient samples solve quality degradation, the above observation is not an issue.

The results of RuixuanLiNGAM are as follows. RuixuanLiNGAM was faster than DirectLiNGAM but a few to ten times slower than LayeredLiNGAM. Furthermore, RuixuanLiNGAM obtained a more significant quality degradation as the number of variables increased compared to the other methods. Therefore, the top-down fashion may be more appropriate than the bottom-up fashion for estimating layers in learning LiNGAM.

In summary, on synthetic datasets, LayeredLiNGAM performed faster than DirectLiNGAM without degrading quality in almost all cases and with improv-

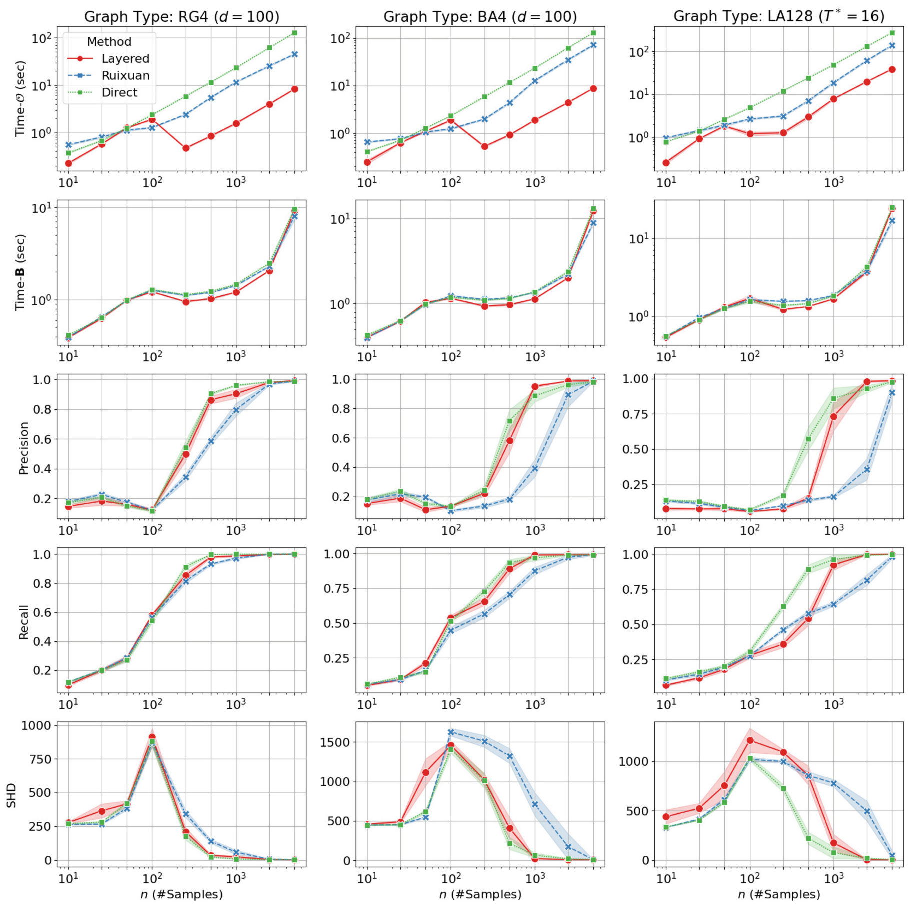


Fig. 5. Results on synthetic datasets that we focused on the sample size $n \in \{10, 25, 50, 100, 250, 500, 1000, 2500, 5000\}$ with the fixed number of variables $d = 100$ for RG4/BA4 and the fixed number of layers $T^* = 16$ for LA128. We generated ten random instances for each setting. We plotted the means and standard errors of five evaluation metrics Time-$\mathcal{O}$, Time-$\mathbf{B}$, Precision, Recall, and SHD.

ing quality in some cases. In addition, we found that LayeredLiNGAM was a more reasonable acceleration method of DirectLiNGAM compared to RuixuanLiNGAM. These results show that LayeredLiNGAM is a more practical LiNGAM learning method than DirectLiNGAM and RuixuanLiNGAM when the input data follows LiNGAM and has a sufficient size of samples.

5.4 Results on Real-World Datasets

We conducted ten trials for each GENE and YPROP dataset via bootstrap sampling. We measured BIC and RMSE on the original rather than the bootstrapped

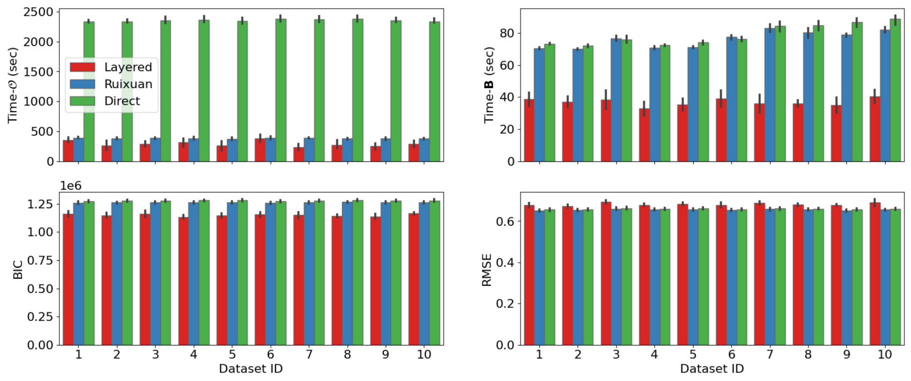


Fig. 6. Results on GENE datasets. We tried ten random bootstrap samplings for each dataset. We plotted the means and standard errors of four evaluation metrics Time-$\mathcal{O}$, Time-$\mathbf{B}$, BIC, and RMSE.

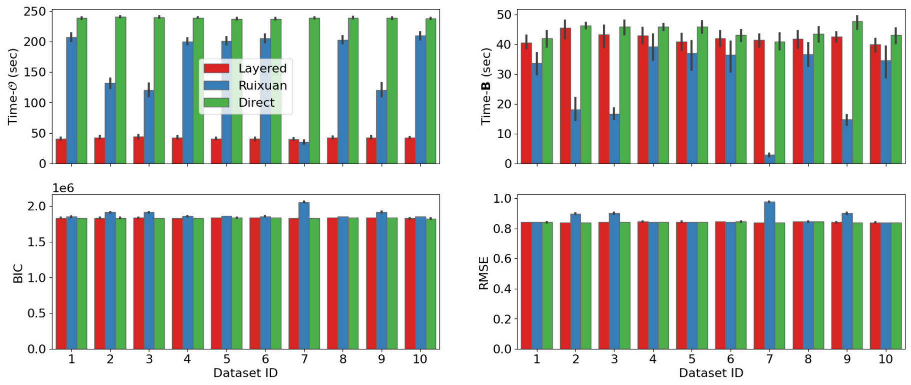


Fig. 7. Results on YPROP datasets. We tried ten random bootstrap samplings for each dataset. We plotted the means and standard errors of four evaluation metrics Time-$\mathcal{O}$, Time-$\mathbf{B}$, BIC, and RMSE.

dataset. As mentioned above, we did not measure Precision, Recall, and SHD because there is no known true DAG. Figure 6 and 7 show the results on GENE and YPROP datasets, respectively.

LayeredLiNGAM estimated causal orders more than five times faster than DirectLiNGAM on both GENE and YPROP datasets. In the total computation times, the speedups by LayeredLiNGAM were more than five and three times on GENE and YPROP datasets, respectively. Thus, we achieved desirable accelerations by LayeredLiNGAM on real-world datasets.

The results on the quality of learned DAGs are as follows. On GENE datasets, LayeredLiNGAM was slightly inferior in RMSE but significantly superior in BIC. On YPROP datasets, LayeredLiNGAM achieved almost comparable BIC and RMSE to DirectLiNGAM. Thus, we succeeded to avoid undesirable quality degradations of learned DAGs on LayeredLiNGAM.

RuixuanLiNGAM performed as fast as LayeredLiNGAM on GENE datasets and faster than DirectLiNGAM on YPROP datasets. However, its BIC and RMSE were comparable or significantly inferior to DirectLiNGAM. Therefore, the top-down fashion can be more appropriate than the bottom-up fashion for estimating layers in learning LiNGAM on real-world datasets.

In summary, on real-world datasets, LayeredLiNGAM performed faster than DirectLiNGAM without undesirable quality degradations of learned DAGs. On the other hand, RuixuanLiNGAM could accelerate learning LiNGAM but would obtain quality degradations. These results show that LayeredLiNGAM is a more practical LiNGAM learning method than DirectLiNGAM and RuixuanLiNGAM on real-world datasets.

6 Conclusion

This paper focused on a representative LiNGAM learning method called DirectLiNGAM. First, we pointed out that the bottleneck of DirectLiNGAM is in estimating causal order both theoretically and empirically. Second, we proposed LayeredLiNGAM improving the computational complexity of DirectLiNGAM by leveraging the concept of layers in DAG structures. To derive the algorithm of LayeredLiNGAM, we generalized the theoretical framework of DirectLiNGAM. We also conducted experiments on both synthetic and real-world datasets. The experimental results showed that LayeredLiNGAM succeeded in learning faster than DirectLiNGAM when the estimated DAG consisted of a small number of layers relative to the number of nodes. Furthermore, LayeredLiNGAM faithfully followed its computational complexity and obtained no undesirable quality degradation of learned DAG in almost all cases and quality improvements in some cases. An interesting open research question is how to make the output of LayeredLiNGAM closer to DirectLiNGAM while maintaining the computation time of LayeredLiNGAM.

References

1. Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. Rev. Mod. Phys. **74**, 47–97 (2002)
2. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. Ann. Stat. **32**(2), 407–499 (2004)
3. Friedman, J., Hastie, T., Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso. Biostatistics **9**(3), 432–41 (2008)
4. Gao, M., Ding, Y., Aragam, B.: A polynomial-time algorithm for learning nonparametric causal graphs. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 11599–11611. Curran Associates, Inc. (2020)

5. Hoyer, P.O., Janzing, D., Mooij, J.M., Peters, J., Scholkopf, B.: Nonlinear causal discovery with additive noise models. In: Neural Information Processing Systems (2008)
6. Hyvärinen, A., Smith, S.M.: Pairwise likelihood ratios for estimation of non-gaussian structural equation models. J. Mach. Learn. Res. **14**, 111–152 (2013)
7. Ikeuchi, T., Ide, M., Zeng, Y., Maeda, T.N., Shimizu, S.: Python package for causal discovery based on lingam. J. Mach. Learn. Res. **24**, 14:1–14:8 (2023)
8. Michoel, T., Zhang, J.D.: Causal inference in drug discovery and development. Drug Discovery Today, 103737 (2022)
9. Rosenström, T.H., et al.: Pairwise measures of causal direction in the epidemiology of sleep problems and depression. PLoS ONE **7** (2012)
10. Sachs, K., Perez, O.D., Pe'er, D., Lauffenburger, D.A., Nolan, G.P.: Causal protein-signaling networks derived from multiparameter single-cell data. Science **308**, 523–529 (2005)
11. Shimizu, S., Hoyer, P.O., Hyvärinen, A., Kerminen, A.J.: A linear non-gaussian acyclic model for causal discovery. J. Mach. Learn. Res. **7**, 2003–2030 (2006)
12. Shimizu, S., et al.: DirectLiNGAM: a direct method for learning a linear non-gaussian structural equation model. J. Mach. Learn. Res. **12**, 1225–1248 (2011)
13. Zeng, Y., Shimizu, S., Matsui, H., Sun, F.: Causal discovery for linear mixed data. In: Conference on Causal Learning and Reasoning (2022)
14. Zhang, B., et al.: Integrated systems approach identifies genetic nodes and networks in late-onset Alzheimer's disease. Cell **153**, 707–720 (2013)
15. Zhang, K., Hyvärinen, A.: On the identifiability of the post-nonlinear causal model. In: Conference on Uncertainty in Artificial Intelligence (2009)
16. Zhao, R., He, X., Wang, J.: Learning linear non-gaussian directed acyclic graph with diverging number of nodes. J. Mach. Learn. Res. **23**, 269:1–269:34 (2021)
17. Zhou, W., He, X., Zhong, W., Wang, J.: Efficient learning of quadratic variance function directed acyclic graphs via topological layers. J. Comput. Graph. Stat. **31**, 1269–1279 (2021)
18. Zou, H.: The adaptive lasso and its oracle properties. J. Am. Stat. Assoc. **101**, 1418–1429 (2006)

Enhanced Bayesian Optimization via Preferential Modeling of Abstract Properties

A. V. Arun Kumar(✉), Alistair Shilton, Sunil Gupta, Santu Rana, Stewart Greenhill, and Svetha Venkatesh

Applied Artificial Intelligence Institute (A^2I^2), Deakin University, Geelong, Australia
a.anjanapuravenkatesh@deakin.edu.au

Abstract. Experimental (design) optimization is a key driver in designing and discovering new products and processes. Bayesian Optimization (BO) is an effective tool for optimizing expensive and black-box experimental design processes. While Bayesian optimization is a principled data-driven approach to experimental optimization, it learns everything from scratch and could greatly benefit from the expertise of its human (domain) experts who often reason about systems at different abstraction levels using physical properties that are not necessarily directly measured (or measurable). In this paper, we propose a human-AI collaborative Bayesian framework to incorporate expert preferences about unmeasured abstract properties into the surrogate modeling to further boost the performance of BO. We provide an efficient strategy that can also handle any incorrect/misleading expert bias in preferential judgments. We discuss the convergence behavior of our proposed framework. Our experimental results involving synthetic functions and real-world datasets show the superiority of our method against the baselines.

Keywords: Machine Learning · Bayesian Optimization · Gaussian Process · Expert Feedback · Preferential Modeling

1 Introduction

Experimental design is the workhorse of scientific design and discovery. Bayesian Optimization (BO) has emerged as a powerful methodology for experimental design tasks [1,2] due to its sample-efficiency in optimizing expensive black-box functions. In its basic form, BO starts with a set of randomly initialized designs and then sequentially suggests the next design until the target objective is reached or the optimization budget is depleted. Theoretical analyses [3,4] of BO methods have provided mathematical guarantees of sample efficiency in the form of sub-linear regret bounds. While BO is an efficient optimization method, it only uses data gathered during the design optimization process. However, in real world

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_14.

A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14946, pp. 234–250, 2024.
https://doi.org/10.1007/978-3-031-70365-2_14

experimental design tasks, we also have access to human experts [5] who have enormous knowledge about the underlying physical phenomena. Incorporating such valuable knowledge can greatly accelerate the sample-efficiency of BO.

Previous efforts in BO literature have incorporated expert knowledge on the shape of functions [6], form of trends [7], priors over optima [8] and model selection [9], which require experts to provide very detailed knowledge about the black-box function. However, most experts understand the process in an approximate or qualitative way, and usually reason in terms of the intermediate abstract properties - the expert will compare designs, and reason as to why one design is better than another using high level abstractions. For instance, consider the design of a spacecraft shield (Whipple shield) consisting of 2 plates separated by a gap to safeguard the spacecraft against micro-meteoroid and orbital debris particle impacts. The design efficacy is measured by observing the penetration caused by hyper-velocity debris. An expert would reason why one design is better than another and accordingly come up with a new design to try out. As part of their domain knowledge, human experts often expect the first plate to shatter the space debris while the second to absorb the fragments effect. Based on these abstract intuitions, the expert will compare a pair of designs by examining the shield penetration images and ask: Does the first plate shatter better *(Shattering)*? Does the second plate absorb the fragments better *(Absorption)*? The use of such abstractions allows experts to predict the overall design objective thus resulting in an efficient experimental design process. It is important to note that measuring such abstractions is not usually feasible and only expert's qualitative inputs are available. *Incorporating such abstract properties in BO for the acceleration of experimental design process is not well explored.*

In this paper, we propose a novel human-AI collaborative approach - **B**ayesian **O**ptimization with **A**bstract **P**roperties (BOAP) - to accelerate BO by capturing expert inputs about the abstract, unmeasurable properties of the designs. Since expert inputs are usually qualitative [10] and often available in the form of design preferences based on abstract properties, we model each abstract property via a latent function using the qualitative pairwise rankings. We note that eliciting such pairwise preferences about designs does not add significant cognitive overhead for the expert, in contrast to asking for explicit knowledge about properties. We fit a separate rank Gaussian process [11] to model each property. Our framework allows enormous flexibility for expert collaborations as it does not need the exact value of an abstract property, just its ranking. A schematic of our proposed BOAP framework is shown in Fig. 1.

Although we anticipate that experts will provide accurate preferences on abstract properties, the expert preferential knowledge can sometimes be misleading. Therefore to avoid such undesired bias, we use two models for the black-box function. The first model uses a "main" Gaussian Process (GP) to model the black-box function in an augmented input space where the design variables are augmented with the estimated abstract properties modeled via their respective rank GPs. The second model uses another "main" GP to model the black-box function using the original design space *without* any expert inputs. At each iteration, we use predictive likelihood-based model selection to choose the "best" model that has higher probability of finding the optima.

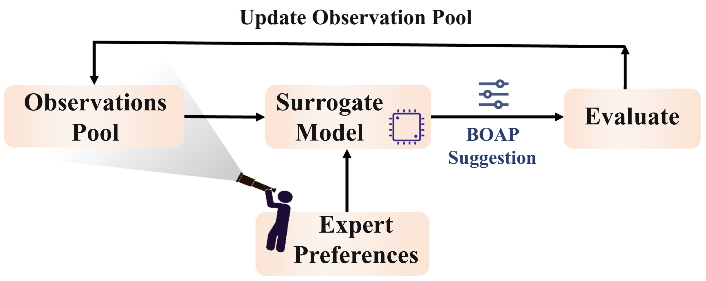


Fig. 1. A schematic representation of our proposed framework Bayesian Optimization with Abstract Properties (BOAP).

Our contributions are: **(i)** we propose a novel human-AI collaborative BO algorithm (BOAP) for incorporating the expert pairwise preferences on abstract properties via rank GPs (Sect. 3), **(ii)** we provide a brief discussion on the convergence behavior of our proposed BOAP method (Sect. 4), **(iii)** we provide empirical results on both synthetic optimization problems and real-world design optimization problems to prove the usefulness of BOAP framework (Sect. 5).

2 Background

Notations

We use lower case bold fonts $\mathbf{v}$ for vectors and v_i for each element in $\mathbf{v}$. $\mathbf{v}^\intercal$ is the transpose. We use upper case bold fonts (and bold greek symbols) $\mathbf{M}$ for matrices and M_{ij} for each element in $\mathbf{M}$. abs$(\cdot)$ is the absolute value. $|\cdot|$ is the determinant. $\mathbb{N}_n = \{1, 2, \cdots, n\}$. $\mathbb{R}$ for Reals. $\mathcal{X}$ is a index set and $\mathbf{x} \in \mathcal{X}$.

2.1 Bayesian Optimization

Bayesian Optimization (BO) [12,13] provides an elegant framework for finding the global optima of an expensive black-box function $f(\mathbf{x})$, given as $\mathbf{x}^\star \in \text{argmax}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$, where $\mathcal{X}$ is a compact search space. BO is comprised of two main components: (i) a surrogate model (usually a Gaussian Process [11]) of the unknown objective function $f(\mathbf{x})$, and (ii) an Acquisition Function $u(\mathbf{x})$ [14] to guide the search for optima.

Gaussian Process. A Gaussian Process (GP) [11] is a flexible, non-parametric distribution over functions. It is a preferred surrogate model because of its simplicity and tractability, in contrast to other surrogate models such as Student-t process [15] and Wiener process [16]. A GP is defined by a prior mean function $\mu(\mathbf{x})$ and a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The function $f(\mathbf{x})$ is modeled using a GP as $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$. If $\mathcal{D}_{1:t} = \{\mathbf{x}_{1:t}, \mathbf{y}_{1:t}\}$ denotes a set of observations, where $y = f(\mathbf{x}) + \eta$ is the observation corrupted with noise

$\eta \in \mathcal{N}(0, \sigma_\eta^2)$ then, according to the properties of GP, the observed samples $\mathcal{D}_{1:t}$ and a new observation $(\mathbf{x}_\star, f(\mathbf{x}_\star))$ are jointly Gaussian. Thus, the posterior distribution $f(\mathbf{x}_\star)$ is $\mathcal{N}(\mu(\mathbf{x}_\star), \sigma^2(\mathbf{x}_\star))$, where $\mu(\mathbf{x}_\star) = \mathbf{k}^\intercal[\mathbf{K} + \sigma_\eta^2\mathbf{I}]^{-1}\mathbf{y}_{1:t}$, $\sigma^2(\mathbf{x}_\star) = k(\mathbf{x}_\star, \mathbf{x}_\star) - \mathbf{k}^\intercal[\mathbf{K} + \sigma_\eta^2\mathbf{I}]^{-1}\mathbf{k}$, $\mathbf{k} = [k(\mathbf{x}_\star, \mathbf{x}_1) \cdots k(\mathbf{x}_\star, \mathbf{x}_t)]^\intercal$, and $\mathbf{K} = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j \in \mathbb{N}_t}$.

Acquisition Functions. The acquisition function selects the next point for evaluation by balancing the exploitation vs exploration (i.e. searching in high value regions vs highly uncertain regions). Some popular acquisition functions include Expected Improvement (EI) [17], GP-UCB [3] and Thompson Sampling (TS) [18]. A standard BO algorithm is provided in Sect. 8 of the supplementary material[1].

2.2 Rank GP Distributions

[19] demonstrated that humans are better at providing qualitative comparisons than absolute magnitudes. Thus modeling latent human preferences is crucial when optimization objectives are in domains such as A/B testing of web designing [20], recommender systems [21], players skill rating [22] and many more. [23] proposed a non-parametric Bayesian algorithm for learning instance or label preferences. We now discuss modeling pairwise preference relations using rank GPs.

Consider a set of n distinct training instances denoted by $X = \{\mathbf{x}_i \; \forall i \in \mathbb{N}_n\}$ based on which pairwise preference relations are observed. Let $P = \{(\mathbf{x} \succ \mathbf{x}') \mid \mathbf{x}, \mathbf{x}' \in X\}$ be a set of pairwise preference relations, where the notation $\mathbf{x} \succ \mathbf{x}'$ expresses the preference of instance $\mathbf{x}$ over $\mathbf{x}'$. For example, the pair $\{\mathbf{x}, \mathbf{x}'\}$ can be two different spacecraft shield designs and $\mathbf{x} \succ \mathbf{x}'$ implies that spacecraft design $\mathbf{x}$ is preferred over $\mathbf{x}'$. [23] assume that each training instance is associated with an unobservable latent function value $\{\bar{f}(\mathbf{x})\}$ measured from an underlying hidden preference function $\bar{f} : \mathbb{R}^d \to \mathbb{R}$, where $\mathbf{x} \succ \mathbf{x}'$, implies $\bar{f}(\mathbf{x}) > \bar{f}(\mathbf{x}')$. Employing an appropriate GP prior and likelihood, user preference can be modeled via rank Gaussian process distributions.

Preference learning has been used in BO literature [24,25]. [24] proposed Preferential BO (PBO) to model the unobserved objective function using a binary design preferential feedback. [26] modified PBO to compute posteriors via skew GPs. [27] proposed a preference learning based BO to model preferences in a multi-objective setup using multi-output GPs. [28] proposed a preference learning with Siamese Networks to capture preferences in a Multi-task learning setup. All these works incorporate preferences about an unobserved objective function. However, in this paper, we use preference learning to model expert preferences about the intermediate abstract (auxiliary) properties. Our latent model learned using such preferential data is then used as an input to model the main objective function.

[1] The supplementary material of BOAP is accessible online at the following link: https://doi.org/10.1007/978-3-031-70365-2_14.

3 Framework

This paper addresses the global optimization of an expensive, black-box function f, *i.e.,* we aim to find the global optima ($\mathbf{x}^\star$) of the unknown objective function f represented as:

$$\mathbf{x}^\star \in \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmax}} f(\mathbf{x}) \tag{1}$$

where $f : \mathcal{X} \to \mathbb{R}$ is a noisy and expensive objective function. For example, f could be a metric signifying the strength of the spacecraft shield. The motivation of this research work is to model f by capturing the cognitive knowledge of experts in making preferential decisions based on the inherent non-measurable abstract properties of the possible designs. The objective here is same as that of standard BO *i.e.,* to find the optimal design ($\mathbf{x}^\star$) that maximizes the unknown function f, but in the light of expert preferential knowledge on abstract properties. The central idea is to use preferential feedback to model and utilize the underlying higher-order properties that underpin preferential decisions about designs. We propose **B**ayesian **O**ptimization with **A**bstract **P**roperties (**BOAP**) for the optimization of f in the light of expert preferential inputs. First, we discuss expert knowledge about abstract properties. Next, we discuss GP modeling of f with preferential inputs, followed by a model-selection step that is capable of overcoming a futile expert bias in preferential knowledge. A complete algorithm for BOAP is presented in Algorithm 1 at the end of this section.

3.1 Expert Preferential Inputs on Abstract Properties

In numerous scenarios, domain experts reason the output of a system in terms of higher-order properties $\omega_1(\mathbf{x}), \omega_2(\mathbf{x}), \ldots$ of a design $\mathbf{x} \in \mathcal{X}$. However, these abstract properties are rarely measured, only being accessible via expert preferential inputs. For instance, a material scientist designing spacecraft shield can easily provide her pairwise preferences on the properties such as shattering, shock absorption, *i.e., "this design absorbs shock better than that design"*, in contrast to specifying the exact measurements of shock absorption. These properties can be simple physical properties or abstract combinations of multiple physical properties which an expert uses to reason about the output of a system. We propose to incorporate such qualitative properties accessible to the expert in the surrogate modeling of the given objective function to further accelerate the sample-efficiency of BO.

Let $\omega_{1:m}(\mathbf{x})$ be a set of m abstract properties derived from the design $\mathbf{x} \in \mathcal{X}$. For property ω_i, design $\mathbf{x}$ is preferred over design $\mathbf{x}'$ if $\omega_i(\mathbf{x}) > \omega_i(\mathbf{x}')$. We denote the set of preferences provided on ω_i as $P^{\omega_i} = \{(\mathbf{x} \succ \mathbf{x}') \,\text{if}\, \omega_i(\mathbf{x}) > \omega_i(\mathbf{x}') \mid \mathbf{x} \in \mathcal{X}\}$.

Rank GPs for Abstract Properties. We capture the aforementioned expert preferential data for each of the abstract properties $\omega_{1:m}$ individually using m separate rank Gaussian process distributions [23]. In conventional GPs the observation model consists of a map of input-output pairs. In contrast, the observation model of a rank (preferential) Gaussian process ($\mathcal{GP}$) consists of a set of instances and a set of pairwise preferences between those instances. The central idea here is to capture the ordering over a set of n instances $X = \{\mathbf{x}_i \,|\, \forall i \in \mathbb{N}_n\}$ by learning latent preference functions $\{\omega_i \,|\, \forall i \in \mathbb{N}_m\}$. We denote such a rank GP modeling abstract property ω_i by the notation $\mathcal{GP}_{\omega_i}$.

Let $\mathcal{X} \in \mathbb{R}^d$ be a $d-$dimensional compact search space and $X = \{\mathbf{x}_i \,|\, \forall i \in \mathbb{N}_n\}$ be a set of n training instances. Let $\boldsymbol{\omega} = \{\omega(\mathbf{x})\}$ be the unobservable latent preference function values associated with each of the instances $\mathbf{x} \in X$. Let P be the set of p pairwise preferences between instances in X, defined as:

$$P = \{(\mathbf{x} \succ \mathbf{x}')_j \text{ if } \omega(\mathbf{x}) > \omega(\mathbf{x}') \mid \mathbf{x} \in X, \forall j \in \mathbb{N}_p\}$$

where ω is the latent preference function. The observation model for the rank GP distribution $\mathcal{GP}_\omega$ modeling the latent preference function ω is given as:

$$\bar{\mathcal{D}} = \{\mathbf{x}_{1:n}, P = \{(\mathbf{x} \succ \mathbf{x}')_j \; \forall \mathbf{x}, \mathbf{x}' \in X, j \in \mathbb{N}_p\}\}$$

We follow the probabilistic kernel approach for preference learning [23] to formulate the likelihood function and Bayesian probabilities. Imposing non-parametric GP priors on the latent function values $\boldsymbol{\omega}$, we arrive at the prior probability of $\boldsymbol{\omega}$ given by:

$$\mathcal{P}(\boldsymbol{\omega}) = (2\pi)^{-\frac{n}{2}} |\mathbf{K}|^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}\boldsymbol{\omega}^\intercal \mathbf{K}^{-1} \boldsymbol{\omega}\right) \tag{2}$$

With suitable noise assumptions $\mathcal{N}(0, \tilde{\sigma}_\eta^2)$ on inputs and the preference relations $(\mathbf{x}, \mathbf{x}')_{1:p}$ in P, the Gaussian likelihood function based on [29] is:

$$\mathcal{P}((\mathbf{x} \succ \mathbf{x}')_i | \omega(\mathbf{x}), \omega(\mathbf{x}')) = \Phi\big(z_i(\mathbf{x}, \mathbf{x}')\big) \tag{3}$$

where Φ is the c.d.f of standard normal distribution and $z(\mathbf{x}, \mathbf{x}') = \frac{\omega(\mathbf{x}) - \omega(\mathbf{x}')}{\sqrt{2\tilde{\sigma}_\eta^2}}$. Based on Bayes theorem, the posterior distribution of the latent function given the data is given by:

$$\mathcal{P}(\boldsymbol{\omega}|\bar{\mathcal{D}}) = \frac{\mathcal{P}(\boldsymbol{\omega})}{\mathcal{P}(\bar{\mathcal{D}})} \mathcal{P}(\bar{\mathcal{D}}|\boldsymbol{\omega})$$

where $\mathcal{P}(\boldsymbol{\omega})$ is the prior distribution (Eq. (2)), $\mathcal{P}(\bar{\mathcal{D}}) = \int \mathcal{P}(\bar{\mathcal{D}}|\boldsymbol{\omega})\mathcal{P}(\boldsymbol{\omega})\, d\boldsymbol{\omega}$ is the evidence of model parameters including kernel hyperparameters, and $\mathcal{P}(\bar{\mathcal{D}}|\boldsymbol{\omega})$ is the probability of observing the pairwise preferences given the latent function values $\boldsymbol{\omega}$, which can be computed as a product of the likelihood (Eq. (3)) *i.e.,* $\mathcal{P}(\bar{\mathcal{D}}|\boldsymbol{\omega}) = \prod_p \mathcal{P}((\mathbf{x} \succ \mathbf{x}')_p | \omega(\mathbf{x}), \omega(\mathbf{x}'))$. We find the posterior distribution using Laplace approximation and the Maximum A Posteriori estimate (MAP) $\boldsymbol{\omega}_{\text{MAP}}$ as the mode of posterior distribution. We can find the MAP using Newton-Raphson descent given by:

$$\boldsymbol{\omega}^{\text{new}} = \boldsymbol{\omega}^{\text{old}} - \mathbf{H}^{-1}\mathbf{g}|_{\boldsymbol{\omega}=\boldsymbol{\omega}^{\text{old}}} \tag{4}$$

where the Hessian $\mathbf{H} = [\mathbf{K} + \tilde{\sigma}_\eta^2 \mathbf{I}]^{-1} + \mathbf{C}$, and the gradient $\mathbf{g} = \nabla_{\boldsymbol{\omega}} \log \mathcal{P}(\boldsymbol{\omega}|\bar{\mathcal{D}}) = -[\mathbf{K} + \tilde{\sigma}_\eta^2 \mathbf{I}]^{-1}\boldsymbol{\omega} + \mathbf{b}$, given $b_j = \frac{\partial}{\partial \omega(\mathbf{x}_j)} \sum_p \ln \Phi(z_p)$ and $C_{ij} = \frac{-\partial^2}{\partial \omega(\mathbf{x}_i) \partial \omega(\mathbf{x}_j)} \sum_p \ln \Phi(z_p)$.

Hyperparameter Optimization. Kernel hyperparameters (θ) are crucial to optimize the generalization performance of the GP. We perform the model selection for our rank-GPs by maximizing the corresponding log-likelihood in the light of latent values $\boldsymbol{\omega}_{\text{MAP}}$. In contrast to the evidence maximization mentioned in [23] *i.e.,* $\theta^{\star}_{\omega} = \text{argmax}_{\theta_\omega} \mathcal{P}(\bar{\mathcal{D}}|\theta_\omega)$, we find the optimal kernel hyperparameters by maximizing the log-likelihood ($\bar{\mathcal{L}}$) of rank GPs *i.e.,* $\theta^{\star}_{\omega} = \text{argmax}_{\theta_\omega} \bar{\mathcal{L}}$. The closed-form of log-likelihood of the rank GP is given as:

$$\bar{\mathcal{L}} = -\frac{1}{2}\boldsymbol{\omega}^{\intercal}_{\mathbf{MAP}}[\mathbf{K}+\tilde{\sigma}^2_\eta \mathbf{I}]^{-1}\boldsymbol{\omega}_{\text{MAP}} - \frac{1}{2}\log|\mathbf{K}+\tilde{\sigma}^2_\eta \mathbf{I}| - \frac{n}{2}\log(2\pi) \tag{5}$$

3.2 Augmented GP with Abstract Property Preferences

To account for property preferences in modeling f, we augment the input $\mathbf{x}$ of a conventional GP modeling f with the mean predictions obtained from m rank GPs ($\mathcal{GP}_{\omega_{1:m}}$) as auxiliary inputs capturing the property preferences $\omega_{1:m}$, in other words, instead of modeling GP directly on $\mathbf{x}$ we model on $\tilde{\mathbf{x}} = [\mathbf{x}, \mu_{\omega_1}(\mathbf{x}), \cdots, \mu_{\omega_m}(\mathbf{x})]$, where μ_{ω_i} is the predictive mean computed using:

$$\mu_{\omega_i}(\mathbf{x}) = \mathbf{k}^{\intercal}[\mathbf{K}+\sigma^2_\eta \mathbf{I}]^{-1}\boldsymbol{\omega}_{\text{MAP}}$$

where $\mathbf{k} = [k(\mathbf{x}, \mathbf{x}_1), \cdots, k(\mathbf{x}, \mathbf{x}_n)]^{\intercal}$, $\mathbf{K} = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j\in\mathbb{N}_n}$ and $\mathbf{x}_i \in X$. To handle different scaling levels in rank GPs, we normalize its output in the interval $[0, 1]$, such that $\mu_{\omega_i}(\mathbf{x}) \in [0, 1]$.

Although we model $\tilde{\mathbf{x}}$ using mean predictions $\mu_{\omega_i}(\mathbf{x})$, the uncertainty estimates were not (directly) considered in the modeling. The GP predictive variance tends to be high outside of the neighborhood of observations, indicating the uncertainty in our beliefs on the model. Therefore, a data point with high predictive variance $(\sigma_{\omega_1}(\mathbf{x}))^2$ in rank GP indicates the model uncertainty. We incorporate this uncertainty in our main GP modeling $\tilde{\mathbf{x}}$ such that the effects of predicted abstract properties $\mu_{\omega_i}(\mathbf{x})$ are appropriately reduced when the model is uncertain *i.e.* when $(\sigma_{\omega_i}(\mathbf{x}))^2$ is high.

To achieve this, we formulate the feature-wise lengthscales as a function of predictive uncertainty of the augmented dimensions to control their *importance* in the overall GP. Note that augmented features can be detrimental when the model is uncertain. To address this potential problem, we use a spatially varying kernel [6] that treats the lengthscale as a function of the input, rather than a constant. A positive definite kernel with spatially varying lengthscale is given as:

$$k(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{d} \sqrt{\frac{2l(x_i)l(x'_i)}{l^2(x_i) + l^2(x'_i)}} \exp\left(-\sum_{i=1}^{d} \frac{(x_i - x'_i)^2}{l^2(x_i) + l^2(x'_i)} \right) \tag{6}$$

where $l(\cdot)$ is the lengthscale function and $\mathbf{x} \in \mathbb{R}^d$. In our proposed framework, we model $\tilde{\mathbf{x}} \in \mathbb{R}^{d+m}$ and use lengthscale as a function $l(\cdot)$ only for the newly augmented (m) dimensions and retain the lengthscales of the original (d) dimensions to standard constant values *i.e.* $l(x_i) = l_i \;\; \forall i \in \mathbb{N}_d$. Therefore the overall kernel hyperparameter set is given as $\theta = [l_1, \cdots, l_d, l_{\omega_1}(\mathbf{x}), \cdots, l_{\omega_m}(\mathbf{x})]$. As we need lengthscale function to reflect the model uncertainty, we set $l_{\omega_i}(\mathbf{x}) = \alpha_i \tilde{\sigma}_{\omega_i}(\mathbf{x})$, where $\tilde{\sigma}_{\omega_i}(\mathbf{x})$ is the normalized standard deviation of the rank GP predicted for the abstract property ω_i and α_i is a scale parameter that is tuned using the standard GP log-marginal likelihood in conjunction with other kernel parameters. The aforementioned lengthscales ensure that the data points $\tilde{\mathbf{x}}$ with high model uncertainty have higher lengthscale on the augmented dimensions and thus are treated as less important.

The objective function is modeled on the concatenated inputs $\tilde{\mathbf{x}} \in \mathbb{R}^{d+m}$ using the spatially varying kernel (Eq. (6)) $k(\tilde{\mathbf{x}}, \tilde{\mathbf{x}})$ $\forall \tilde{\mathbf{x}} \in \mathbb{R}^{d+m}$ and we denote this function with augmented inputs $\tilde{\mathbf{x}}$ as human-inspired objective function $h(\tilde{\mathbf{x}})$. The GP ($\mathcal{GP}_h$) constructed in the light of expert preferential data is then used in BO to find the global optima of $h(\tilde{\mathbf{x}})$, given as:

$$\mathbf{x}^\star \in \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmax}}\, h(\tilde{\mathbf{x}}) \tag{7}$$

The observation model is $\mathcal{D} = \{(\mathbf{x}, y = h(\tilde{\mathbf{x}}) \approx f(\mathbf{x}))\}$ *i.e.* the human-inspired objective function $h(\tilde{\mathbf{x}})$ is a simplified $f(\mathbf{x})$ with auxiliary features in the input, thus we observe the $h(\tilde{\mathbf{x}})$ via $f(\mathbf{x})$. The kernel hyperparameters associated with $\mathcal{GP}_h$ are denoted as θ_h given as $\theta_h = \{l_{1:d}, \alpha_{1:m}\}$.

3.3 Overcoming Inaccurate Expert Inputs

Up to this point we have assumed that expert input is accurate and thus likely to accelerate BO. However, in some cases this feedback may be inaccurate, and potentially slowing optimization. To overcome such bias and encourage exploration we maintain 2 models, one of which is augmented by expert abstract properties (we refer to this as Human Arm-$\mathfrak{h}$) and an un-augmented model (we refer to this as Control Arm-$\mathfrak{f}$), and use predictive likelihood to select the arm at each iteration.

The control arm models f directly by observing the function values at suggested candidate points. Here, we fit a standard GP ($\mathcal{GP}_f$) based on the data collected *i.e.,* $\mathcal{D} = \{(\mathbf{x}, y = f(\mathbf{x}) + \eta)\}$ where $\eta \sim \mathcal{N}(0, \sigma_\eta^2)$ is the Gaussian noise. The GP distribution ($\mathcal{GP}_f$) with hyperparameters $\theta_f = \{l_{1:d}\}$ may be used to optimize f using a BO algorithm.

At each iteration t, we compare the predictive likelihoods ($\mathcal{L}_t$) of both the human augmented arm (Arm-$\mathfrak{h}$) and the control arm (Arm-$\mathfrak{f}$) to select the arm to pull for suggesting the next promising candidate for the function evaluation. Then, we use Thompson Sampling (TS) strategy [18] to draw a sample S_t from the GP distribution of the arm pulled and find its corresponding maxima given as:

$$\mathbf{x}_t^{\mathfrak{h}} = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmax}}\,(S^{\mathfrak{h}}(\tilde{\mathbf{x}})); \quad \mathbf{x}_t^{\mathfrak{f}} = \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmax}}\,(S^{\mathfrak{f}}(\mathbf{x})) \tag{8}$$

The arm with maximum predictive likelihood is chosen at each iteration and we observe f at the suggested location *i.e.,* $(\mathbf{x}_t^{\mathfrak{h}}, f(\mathbf{x}_t^{\mathfrak{h}}))$ or $(\mathbf{x}_t^{\mathfrak{f}}, f(\mathbf{x}_t^{\mathfrak{f}}))$. Then rank GPs are updated to capture the preferences with respect to the new suggestion

$\mathbf{x}_t^{\mathfrak{h}}$ or $\mathbf{x}_t^{\mathfrak{f}}$. This process continues until the evaluation budget T is exhausted. A complete flowchart of our framework is shown in Fig. 2. Additional details of BOAP framework are provided in the supplementary material (Sect. 9).

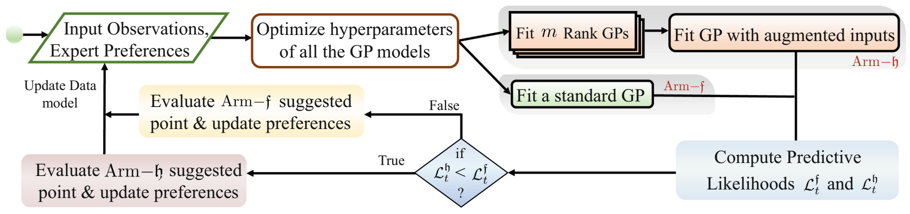

Input Observations, Expert Preferences
Optimize hyperparameters of all the GP models
Fit m Rank GPs
Fit GP with augmented inputs
Arm$-\mathfrak{h}$
Fit a standard GP
Arm$-\mathfrak{f}$
Update Data model
Evaluate Arm$-\mathfrak{f}$ suggested point & update preferences
False
if $\mathcal{L}_t^{\mathfrak{h}} < \mathcal{L}_t^{\mathfrak{f}}$?
True
Evaluate Arm$-\mathfrak{h}$ suggested point & update preferences
Compute Predictive Likelihoods $\mathcal{L}_t^{\mathfrak{f}}$ and $\mathcal{L}_t^{\mathfrak{h}}$

Fig. 2. A complete process flowchart of our proposed BOAP framework.

Algorithm 1. BO with Preferences on Abstract Properties (BOAP)

Input: Sampling Budget T, Initial Samples: $\mathcal{D}_{1:t'} = \{\mathbf{x}_{1:t'}, \mathbf{y}_{1:t'}\}$, Expert Preferences: $P^{\omega_i} = \{(\mathbf{x} \succ \mathbf{x}')_{1:p} \mid \forall i \in \mathbb{N}_m\}$

```
1.  for $t = t' + 1, \cdots, T$ iterations do
2.    optimize hyperparameters $\Theta_t^\star = \{\theta_{\omega_{1:m}}^\star, \theta_h^\star, \theta_f^\star\}$ and update $\mathcal{GP}_{\omega_{1:m}}, \mathcal{GP}_h, \mathcal{GP}_f$
3.    compute predictive likelihoods $\mathcal{L}_t^{\mathfrak{h}}$ and $\mathcal{L}_t^{\mathfrak{f}}$ for Arm-$\mathfrak{h}$ and Arm-$\mathfrak{f}$
4.    if $\mathcal{L}_t^{\mathfrak{h}} > \mathcal{L}_t^{\mathfrak{f}}$, then
5.      draw a random sample $S_t^{\mathfrak{h}}$ from Arm-$\mathfrak{h}$ using Thompson Sampling
6.      maximize $S_t^{\mathfrak{h}}$ to find $\mathbf{x}_t^{\mathfrak{h}} = \underset{\mathbf{x} \in \mathcal{X}}{\text{argmax}}\,(S_t^{\mathfrak{h}}(\tilde{\mathbf{x}}))$
7.      $\mathbf{x}_t = \mathbf{x}_t^{\mathfrak{h}}$
8.    else,
9.      draw a random sample $S_t^{\mathfrak{f}}$ Arm-$\mathfrak{f}$ using Thompson Sampling
10.     maximize $S_t^{\mathfrak{f}}$ to find $\mathbf{x}_t^{\mathfrak{f}} = \underset{\mathbf{x} \in \mathcal{X}}{\text{argmax}}\,(S_t^{\mathfrak{f}}(\mathbf{x}))$
11.     $\mathbf{x}_t = \mathbf{x}_t^{\mathfrak{f}}$
12.   evaluate $f$ at $\mathbf{x}_t$ to obtain $y_t = f(\mathbf{x}_t) + \eta_t$
13.   augment data $\mathcal{D} = \mathcal{D} \cup (\mathbf{x}_t, y_t)$ and update expert preferences $P^{\omega_{1:m}}$ w.r.t $\mathbf{x}_t$
14.   $\mathbf{x}^\star = \underset{(\mathbf{x}, y) \in \mathcal{D}}{\text{argmax}}\; y$
15. end for
16. return $\mathbf{x}^\star$
```

4 Convergence Remarks

In this section we discuss the convergence of our BOAP algorithm in terms of regret bounds. As we are dealing with human expert feedback in our algorithm, it is difficult to make absolute statements as we are reliant on the accuracy of the feedback given and the knowledge of the expert involved, which may be limited if the objective must explore less thoroughly understood areas of the search space (so the expert learns alongside the GP model). Nevertheless, with minimal assumptions we may draw some conclusions that help us to better understand the impact of expert feedback, which is important not only to better understand the potential of BOAP to accelerate convergence but also to give insight into possible future directions.

BOAP may be understood as kernel learning in practice - the core distinction between the human and control arms is that the human arm features an evolving kernel (6). Assuming for simplicity that the human arm comes to dominate over time (as measured by likelihood) then the influence of the kernel (6) on the convergence of the BO algorithm is measured through the maximum information gain $\gamma_T(d)$, where d is the input dimension. For Thompson sampling type algorithm the cumulative regret $R_T = \sum_t f(\mathbf{x}^\star) - f(\mathbf{x}_t)$ is typically [4,30] bounded as $R_T = \mathcal{O}(\sqrt{T\gamma_T(d)})$ (up to log factors), where the maximum information gain $\gamma_T(d)$ is governed by the kernel K through the eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots$ of the covariance matrix $\mathbf{K}_T$ evaluated on the observations $\{(\mathbf{x}_t, y_t) : t \leq T\}$ [3]:

$$\gamma_T(d) \leq \frac{\frac{1}{2}}{1-\frac{1}{e}} \max_{(m_t : \sum_t m_t = T)} \sum_{t=1}^{|\mathcal{D}|} \log\left(1 + \sigma^{-2} m_t \lambda_t\right) \tag{9}$$

Moreover in [3] it is shown that the asymptotic behavior of $\gamma_T(d)$ is controlled by the dimension d of the input. Our key insight for the kernel (6) is that we can drop features with lengthscales over a threshold without overly perturbing the kernel, effectively replacing d with the number of features (d_{eff}) having lengthscales below the threshold, bounding the resulting error so introduced.

Assuming that **(a)** expert observations obey a simple convergence assumption $\max_{\mathbf{x}\in\mathcal{X},t} K_{\omega_i}(\mathbf{x}, \mathbf{x}_t) = \mathcal{O}(g(T))$, where $g(T) \to 0$ as $T \to \infty$, and **(b)** as $T \to \infty$, only $d_{\text{eff}} < d$ of the lengthscales (augmenting or otherwise) satisfy $l_d, l_{\omega_i} < l_{\max}$, then, for the kernel (6), $\gamma_T(d) \leq \breve{\gamma}_T(d_{\text{eff}}) + \mathcal{O}(\frac{d_{\text{eff}}}{l_{\max}^2}) + \mathcal{O}(g(T))$. In this expression $\breve{\gamma}_T(d_{\text{eff}})$ is the maximum information gain for a d_{eff} dimensional SE kernel, *i.e.* [3] $\breve{\gamma}_T(d_{\text{eff}}) = \mathcal{O}((\log T)^{d_{\text{eff}}+1})$. Thus we would expect cumulative regret to satisfy:

$$R_T = \mathcal{O}\left(\sqrt{T\left((\log T)^{d_{\text{eff}}+1} + \frac{d_{\text{eff}}}{l_{\max}^2}\right)}\right) \tag{10}$$

That is, the regret bound for BOAP, assuming the human arm dominates as $T \to \infty$, is the the regret bound for BO with effective dimension d_{eff} plus a term that scales as the ratio of d_{eff} and the cut-off lengthscale $l_{\max}^2$. The more effectively the augmenting features are able to summarize the data in a useful way that renders other features superfluous (*i.e.* minimizes d_{eff}), the tighter the regret bound becomes. A detailed discussion on the maximum information gain and the regret bounds is provided in the supplementary material (Sect. 10)

5 Experiments

We evaluate the performance of BOAP method using synthetic benchmark function optimization problems and real-world optimization problems arising in advanced battery manufacturing processes. We have considered the following experimental settings for BOAP. We use the popular Automatic Relevance Determination (ARD) kernel [31] for the construction of both the rank GPs and the conventional (un-augmented) GPs. For rank GPs, we tune ARD kernel hyperparameters $\theta_d = \{l_d\}$ using max-likelihood estimation (Eq. (5)). For the augmented GP modeling $\tilde{\mathbf{x}}$, we use a spatially varying kernel with a parametric lengthscale function (See discussion in Sect. 3.2). As we normalize the bounds, we tune l_d (the lengthscale for the *un*-augmented features) in the interval $[0.1, 1]$ and the scale parameter $\boldsymbol{\alpha}$ (for the auxiliary features) in the interval $(0, 2]$. Further, we set signal variance $\sigma_f^2 = 1$ as we standardize the outputs.

We compare the performance of BOAP algorithm with the following state-of-the-art baselines. **(i) BO-TS:** a standard Bayesian Optimization (BO) with Thompson Sampling (TS) strategy, **(ii) BO-EI:** BO with Expected Improvement (EI) acquisition function, and **(iii)** BOAP - Only Augmentation **(BOAP-OA)**: Here we run our algorithm without the 2-arm scheme and we only use augmented input for GP modeling. This method shows the effectiveness of expert's inputs. We evaluate the performance of our method against the baselines by plotting the simple regret ($\mathcal{R}_t$) given by: $\mathcal{R}_t = f(\mathbf{x}^\star) - \max_{\mathbf{x} \in \mathcal{D}_{1:t}} f(\mathbf{x})$, where $f(\mathbf{x}^\star)$ is the true optima of the objective function. We do not consider any preference based BO methods [24,26] as baselines, because the preferences are provided directly on the objective function, as opposed to abstract properties that are not measured directly. The additional details of our experimental setup are provided in the supplementary material (Sect. 11).

5.1 Synthetic Experiments

We evaluate BOAP framework in the global optimization of synthetic benchmark functions [32]. The list of synthetic functions used are provided in Table 1.

Emulating Preferential Expert Inputs: As discussed in Sect. 3.1, we fit a rank GP using the expert preferences provided on designs based on their cognitive knowledge. In all our synthetic experiments we set $m = 2$, *i.e.,* we model two abstract properties $\{\omega_1, \omega_2\}$ for the considered synthetic function. We expect the expert to know the higher order abstract features of each design $\mathbf{x} \in \mathcal{X}$. We construct rank GPs by emulating the expert preferences based on such high level features of the given synthetic function. The possible set of high level features of the synthetic functions are mentioned in Table 1. We generate preference list P^{ω_i} for each high level feature of the designs by comparing its utility. We start with $p = \binom{t'}{2}$ preferences in P, that gets updated in every iteration of the optimization process. We construct rank GP surrogates $\{\mathcal{GP}_{\omega_1}, \mathcal{GP}_{\omega_2}\}$ using P^{ω_1} and P^{ω_2}.

For a given $d-$dimensional problem, we have considered $t' = d + 3$ initial observations and allocate $T = 10 \times d + 5$ budget. We repeat all our synthetic experiments 10 times with random initialization and report the average simple regret [12] (along with its standard error) as a function of iterations. The convergence plots obtained for the optimization of synthetic functions after 10 runs are shown in Fig. 3. It is evident from the convergence results that our proposed BOAP method has outperformed the standard baselines by a huge margin, thereby proving its superiority. Further, it is also observed that BOAP-OA, a BOAP variant without the 2-arm bandit strategy and just the augmented GP ($\mathcal{GP}_h$), has a superior performance when compared to the baselines (BO-EI and BO-TS), thereby indicating the usefulness of expert inputs in significantly improving the performance of Bayesian optimization algorithm.

Table 1. Details of the synthetic optimization benchmark functions. Analytical forms are provided in the 2nd column and the last column depicts the high level features used by a simulated expert.

Functions	$f(\mathbf{x})$	High Level Features
Benchmark-1D	$\exp^{(2-\mathbf{x})^2} + \exp^{\frac{(6-\mathbf{x})^2}{10}} + \frac{1}{\mathbf{x}^2+1}$	$\omega_1 = \exp^{(2-\mathbf{x})^2}$, $\omega_2 = \frac{1}{\mathbf{x}^2}$
Rosenbrock-3D	$\sum_{i=1}^{d-1} [100 \times (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$\omega_1 = (x_3 - x_2^2)^2 + (x_2 - x_1^2)^2$
		$\omega_2 = (x_2 - 1)^2 + (x_1 - 1)^2$
Griewank-5D	$\sum_{i=1}^{d} \left[\frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \right]$	$\omega_1 = \sum_{i=1}^{d} x_i^2$, $\omega_2 = \prod_{i=1}^{d} \cos x_i$

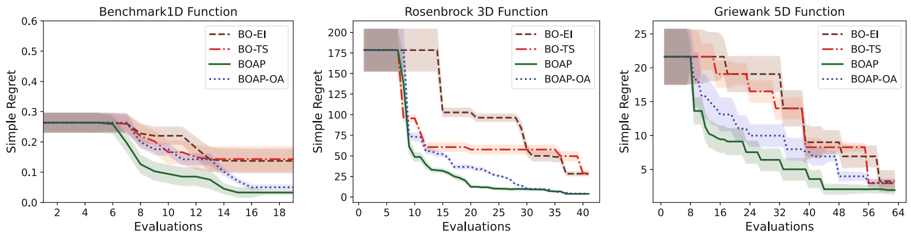


Fig. 3. Simple regret vs iterations for robustness experiments using synthetic multi-dimensional benchmark functions. We plot the average regret (along with its standard error) obtained after 10 random repeated runs.

To demonstrate the robustness of our approach we have conducted additional experiments by accounting for the inaccuracy or poor choices in expert preferential knowledge. Here, we show the robustness of our BOAP approach in two scenarios. First, we show the performance of our proposed approach when the higher order abstract properties are poorly selected. Second, we incorporate noise in the expert preferential feedback by flipping the expert preference between two inputs (designs) with a probability δ. We now discuss in detail the aforementioned two variations of our proposed method.

Inaccurate Abstract Properties (BOAP-IA). In the first variation, we assume that the expert poorly selects the human abstraction features. Table 2 depicts the synthetic functions considered and the corresponding (poorly chosen or uninformative) human abstraction features (ω_1 and ω_2). BOAP-IA uses such inaccurate human abstract features while augmenting the original input space.

Table 2. Selection of abstract (uninformative) features by a simulated human expert. The human abstraction (high level) features shown in the 3rd column are deliberately selected to be uninformative.

Functions	$f(\mathbf{x})$	Human Abstraction Features
Benchmark-1D	$\exp^{(2-\mathbf{x})^2} + \exp^{\frac{(6-\mathbf{x})^2}{10}} + \frac{1}{\mathbf{x}^2+1}$	$\omega_1 = \sin \mathbf{x}$, $\omega_2 = \cos \mathbf{x}$
Rosenbrock-3D	$\sum_{i=1}^{d-1} [100 \times (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$\omega_1 = \sin \mathbf{x}$, $\omega_2 = \cos \mathbf{x}$
Griewank-5D	$\sum_{i=1}^{d} \left[\frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \right]$	$\omega_1 = \sin \mathbf{x}$, $\omega_2 = \mathbf{x}^3$

Noisy Expert Preferences (BOAP-NP). In the second variation, we account for the inaccurate expert preferential knowledge by introducing an error in human expert preferential feedback. To do this, we flip the preference ordering with a probability δ i.e., $P^{\omega,\delta} = \{(\mathbf{x}_i \succ \mathbf{x}_j) \,|\, \mathbf{x}_i, \mathbf{x}_j \in \mathbf{x}_{1:n}, \nu_{ij}\, \omega(\mathbf{x}_i) > \nu_{ij}\, \omega(\mathbf{x}_j)\}$, where ν_{ij} is drawn from a random distribution such that it is $+1$ with probability $1 - \delta$, -1 with probability δ. In this set of experiments we have set the probability $\delta = 0.3$.

We evaluate the performance by computing the simple regret after $10 \times d$ iterations. The empirical results for BOAP with inaccurate features (BOAP-IA) and BOAP framework with noisy preferences (BOAP-NP) are presented in Fig. 4. Although the expert preferential knowledge is noisy and inaccurate, it is significant from the results that our proposed BOAP framework outperforms the standard baselines. We believe that the superior performance of BOAP variants is due to the model selection based safeguard mechanism that uses 2-arm scheme to intelligently select the arm with the maximum predictive likelihood to suggest the next sample.

5.2 Real-World Experiments

We demonstrate the performance of BOAP in two real-world optimization use-cases in Lithium-ion battery manufacturing that are proven to be very complex and expensive in nature, thus providing a wide scope for the optimization. Further, battery scientists often reveal additional knowledge about the abstract properties in the battery design space and thus providing a rich playground for the evaluation of our framework. We refer to the supplementary material (Sect. 11.2) for the detailed experimental setup.

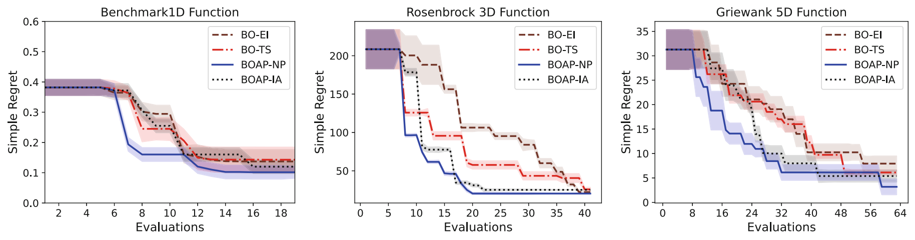


Fig. 4. Simple regret vs iterations for the synthetic multi-dimensional benchmark functions. We plot the average regret (along with its standard error) obtained after 10 random repeated runs.

Optimization of Electrode Calendering. In this experiment, we consider a case study on the calendering process proposed in [33]. The authors analyzed the effect of parameters such as calendering pressure (ε_{cal}), electrode porosity and electrode composition on the electrode properties such as electrolyte conductivity, tortuosity (both in solid phase (τ_{sol}) and liquid phase (τ_{liq})), Current Collector (CC), Active Surface (AS), etc. We define an optimization paradigm using the data grid published in [33].

We use our proposed BOAP framework to optimize the electrode calendering process by maximizing the *Active Surface* of electrodes by modeling two abstract properties: **(i)** Property 1 (ω_1): *Tortuosity in liquid phase* τ_{liq}, and **(ii)** Property 2 (ω_2): *Output Porosity* (OP). We simulate the expert pairwise preferential inputs $\{P^{\omega_{\tau_{\text{liq}}}}, P^{\omega_{\text{OP}}}\}$ by comparing the actual measurements reported in the dataset published in [33]. We consider 4 initial observations and maximize the active surface of the electrodes for 50 iterations. We compare the performance of our proposed BOAP framework by plotting the average simple regret (along with its standard error) after 10 repeated runs with random initialization. The convergence results obtained for the electrode optimization are shown in Fig. 5a.

Electrode Manufacturing Optimization. The best battery formulation and the optimal selection of process parameters is crucial for manufacturing long-life and energy-dense batteries. [34] analyzed the manufacturing of Lithium-ion graphite based electrodes and reported the process parameters in manufacturing a battery along with the output charge capacities of the battery measured after certain charge-discharge cycles. In our experiment, we use BOAP to optimally select the manufacturing process parameters to design a battery with maximum endurance *i.e.,* a battery that can retain the maximum charge after certain charge-discharge cycles. We consider *Anode Thickness* (AT) and *Active Mass* (AM) as abstract properties $\{\omega_{\text{AT}}, \omega_{\text{AM}}\}$ to maximize the battery endurance $E = \frac{D_{50}}{D_5}$, where D_{50} and D_5 are the discharge capacities of the cell at 50th and 5th cycle, respectively. We consider 4 initial observations and maximize the endurance of the cell for 50 iterations. We compare the performance by plotting the average simple regret versus iterations after 10 random repeated runs. The convergence results obtained for maximizing the endurance is shown in Fig. 5b.

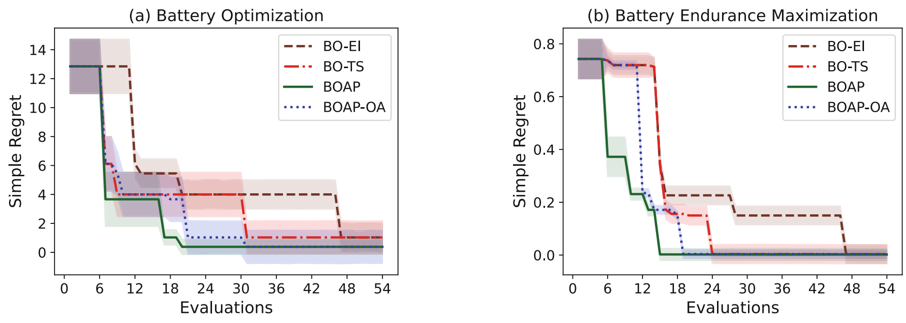


Fig. 5. Simple regret vs iterations for battery manufacturing optimization experiments: **(a)** Optimization of electrode calendering process **(b)** Optimization of the battery endurance.

It is evident from Fig. 5 that BOAP is superior to the baselines due to its ability to model the abstract properties of the battery designs that can be beneficial in accelerating BO performance. Similar to the trends observed in the synthetic experiments, BOAP-OA with just the augmented inputs has outperformed the standard baselines (BO-EI and BO-TS), thereby proving again the benefits of expert inputs in boosting the optimization performance. The supplementary material along with the necessary implementation details and the code snippets are available at https://github.com/mailtoarunkumarav/BOAP.

6 Conclusion

We present a novel approach for human-AI collaborative BO for modeling the expert inputs on abstract properties to further improve the sample-efficiency of BO. Experts provide preferential inputs about the abstract and unmeasurable properties. We model such preferential inputs using rank GPs. We augment the inputs of a standard GP with the output of such auxiliary rank GPs to learn the underlying preferences in the instance space. We use a 2-arm strategy, a key safeguard that provides assurance to utilize only relevant and accurate expert preferential inputs in the modeling, thus overcoming any futile expert bias. We discuss the convergence of our proposed BOAP framework. The experimental results show the superiority of our proposed BOAP algorithm.

Acknowledgements. This research was partially supported by the Australian Government through the Australian Research Council's Discovery Project funding scheme (project DP210102798). The views expressed herein are those of the authors and are not necessarily those of the Australian Government or Australian Research Council.

Disclosure of Interests. Prof. Svetha Venkatesh has received research grants from Australian Research Council's Discovery Project funding scheme.

References

1. Martinez-Cantin, R.: BayesOpt: a Bayesian optimization library for nonlinear optimization, experimental design and bandits. J. Mach. Learn. Res. **15**(1), 3735–3739 (2014)
2. Greenhill, S., Rana, S., Gupta, S., Vellanki, P., Venkatesh, S.: Bayesian optimization for adaptive experimental design: a review. IEEE access **8**, 13937–13948 (2020)
3. Srinivas, N., Krause, A., Kakade, S.M., Seeger, M.W.: Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. IEEE Trans. Inf. Theory **58**(5), 3250–3265 (2012)
4. Chowdhury, S.R., Gopalan, A.: On kernelized multi-armed bandits. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 844–853. PMLR, International Convention Centre, Sydney, Australia (Aug 2017)
5. Swersky, K.: Improving Bayesian optimization for machine learning using expert priors. University of Toronto (Canada) (2017)
6. Venkatesh, A.K.A., Rana, S., Li, C., Gupta, S., Shilton, A., Venkatesh, S.: Bayesian optimization for objective functions with varying smoothness. In: Australasian Joint Conference on Artificial Intelligence, pp. 460–472 (2019)
7. Li, C., et al.: Accelerating experimental design by incorporating experimenter hunches. In: 2018 IEEE International Conference on Data Mining (ICDM), pp. 257–266 (2018). https://doi.org/10.1109/ICDM.2018.00041
8. Hvarfner, C., Stoll, D., Souza, A., Lindauer, M., Hutter, F., Nardi, L.: πBO: Augmenting Acquisition Functions with User Beliefs for Bayesian Optimization. arXiv preprint arXiv:2204.11051 (2022)
9. Venkatesh, A.K.A., Rana, S., Shilton, A., Venkatesh, S.: Human-AI Collaborative Bayesian optimization. In: Advances in Neural Information Processing Systems (2022)
10. Nguyen, Q.P., Tay, S., Low, B.K.H., Jaillet, P.: Top-k ranking Bayesian optimization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 9135–9143 (2021)
11. Williams, C.K., Rasmussen, C.E.: Gaussian Processes For Machine Learning, vol. 2. MIT press Cambridge, MA (2006)
12. Brochu, E., Cora, V.M., De Freitas, N.: A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599 (2010)
13. Frazier, P.I.: A tutorial on Bayesian optimization. arXiv preprint arXiv:1807.02811 (2018)
14. Kushner, H.J.: A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. J. Basic Eng. **86**(1), 97–106 (03 1964). https://doi.org/10.1115/1.3653121
15. Shah, A., Wilson, A., Ghahramani, Z.: Student-t processes as alternatives to Gaussian processes. In: Artificial intelligence and statistics, pp. 877–885 (2014)
16. Zhang, Z., Si, X., Hu, C., Lei, Y.: Degradation data analysis and remaining useful life estimation: a review on Wiener-process-based methods. Eur. J. Oper. Res. **271**(3), 775–796 (2018)
17. Mockus, J., Tiesis, V., Zilinskas, A.: The application of Bayesian methods for seeking the extremum. In: Towards Global Optimization, vol. 2, pp. 117–129 (September 1978)

18. Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika **25**(3–4), 285–294 (1933)
19. Kahneman, D., Tversky, A.: Prospect theory: An analysis of decision under risk. In: Handbook of the Fundamentals of Financial Decision Making: Part I, pp. 99–127. World Scientific (2013)
20. Siroker, D., Koomen, P.: A/B testing: The most powerful way to turn clicks into customers. John Wiley & Sons (2015)
21. Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
22. Herbrich, R., Minka, T., Graepel, T.: TrueSkill: a Bayesian skill rating system. In: Advances in Neural Information Processing Systems **19** (2006)
23. Chu, W., Ghahramani, Z.: Preference learning with Gaussian processes. In: Proceedings of the 22nd International Conference on Machine learning, pp. 137–144 (2005)
24. González, J., Dai, Z., Damianou, A., Lawrence, N.D.: Preferential Bayesian optimization. In: International Conference on Machine Learning, pp. 1282–1291. PMLR (2017)
25. Mikkola, P., Todorović, M., Järvi, J., Rinke, P., Kaski, S.: Projective preferential Bayesian optimization. In: International Conference on Machine Learning, pp. 6884–6892. PMLR (2020)
26. Benavoli, A., Azzimonti, D., Piga, D.: Preferential Bayesian optimization with skew Gaussian processes. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1842–1850 (2021)
27. Astudillo, R., Frazier, P.: Multi-attribute Bayesian optimization with interactive preference learning. In: International Conference on Artificial Intelligence and Statistics, pp. 4496–4507. PMLR (2020)
28. Huang, D., Filstroff, L., Mikkola, P., Zheng, R., Kaski, S.: Bayesian optimization augmented with actively elicited expert knowledge. arXiv preprint arXiv:2208.08742 (2022)
29. Thurstone, L.L.: A law of comparative judgment. In: Scaling, pp. 81–92. Routledge (2017)
30. Kandasamy, K., Krishnamurthy, A., Schneider, J., Póczos, B.: Parallelised Bayesian optimisation via thompson sampling. In: International Conference on Artificial Intelligence and Statistics, pp. 133–142 (2018)
31. Neal, R.M.: Bayesian learning for neural networks, vol. 118. Springer Science & Business Media (2012)
32. Surjanovic, S., Bingham, D.: Virtual library of simulation experiments: Test Functions and Datasets (2017). http://www.sfu.ca/~ssurjano. Accessed 10 Apr 2024
33. Duquesnoy, M., Lombardo, T., Chouchane, M., Primo, E.N., Franco, A.A.: Data-driven assessment of electrode calendering process by combining experimental results, in silico mesostructures generation and machine learning. J. Power Sources **480**, 229103 (2020)
34. Drakopoulas, S.X., et al.: Formulation and manufacturing optimization of Lithium-ion graphite-based electrodes via machine learning. Cell Reports Phys. Sci. **2**(12), 100683 (2021)

Enhancing LLM's Reliability by Iterative Verification Attributions with Keyword Fronting

Yize Sui, Jing Ren(✉), Huibin Tan, Huan Chen, Zhaoye Li, and Ji Wang

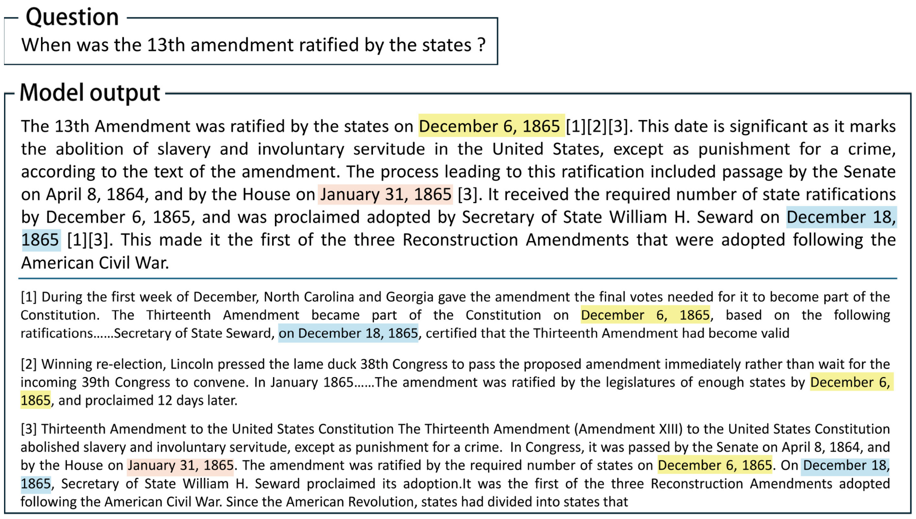
Department of Intelligent Data Science, College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China
{suiyize18,renjing,tanhb_,chenhuan14,lizhaoye23,wj}@nudt.edu.cn

Abstract. Retrieval-augmented text generation attribution is of great significance for knowledge-intensive tasks as it can enhance the credibility and verifiability of large language models (LLMs). However, existing research often ignores the adverse effect of "Middle Loss" in lengthy input contexts on answer correctness, and the potential negative impact of unverified citations on the quality of attribution. To address these challenges, we propose a framework **IVAKF** (**I**terative **V**erified **A**ttribution with **K**eyword **F**ronting), which better utilizes long context information and integrates attribution verification throughout the whole process of response generation. Specifically, for the "Middle Loss" issue, we employ a keyword fronting strategy with Named Entity Recognition (NER), guiding the model's attention to focus on key entities and their relationship with other parts. As for the issue of poor attribution quality, we design a verification-based iterative optimization algorithm, which continuously updates candidate statements and citations until it produces a satisfactory output result. Experiments on three public knowledge-intensive datasets demonstrate that the proposed framework significantly improves the quality of the final response. It improved answer correctness by 6.4%, and citation quality by 9.1% than the baselines.

Keywords: Text Generation Attribution · Retrieval Augmented Generation · Large Language Models

1 Introduction

The rapid development of LLMs has facilitated their widespread application across numerous domains, bringing new opportunities for knowledge-intensive industries such as law and healthcare. In these application scenarios, the core of model reliability lies in the verifiability of its output content. However, with the continuous advancement of LLMs, the so-called "hallucination" [10] phenomenon has sparked widespread concern and apprehension among users regarding the credibility and safety of their outputs. This phenomenon is characterized by the



Question

When was the 13th amendment ratified by the states ?

Model output

The 13th Amendment was ratified by the states on December 6, 1865 [1][2][3]. This date is significant as it marks the abolition of slavery and involuntary servitude in the United States, except as punishment for a crime, according to the text of the amendment. The process leading to this ratification included passage by the Senate on April 8, 1864, and by the House on January 31, 1865 [3]. It received the required number of state ratifications by December 6, 1865, and was proclaimed adopted by Secretary of State William H. Seward on December 18, 1865 [1][3]. This made it the first of the three Reconstruction Amendments that were adopted following the American Civil War.

[1] During the first week of December, North Carolina and Georgia gave the amendment the final votes needed for it to become part of the Constitution. The Thirteenth Amendment became part of the Constitution on December 6, 1865, based on the following ratifications......Secretary of State Seward, on December 18, 1865, certified that the Thirteenth Amendment had become valid

[2] Winning re-election, Lincoln pressed the lame duck 38th Congress to pass the proposed amendment immediately rather than wait for the incoming 39th Congress to convene. In January 1865......The amendment was ratified by the legislatures of enough states by December 6, 1865, and proclaimed 12 days later.

[3] Thirteenth Amendment to the United States Constitution The Thirteenth Amendment (Amendment XIII) to the United States Constitution abolished slavery and involuntary servitude, except as punishment for a crime. In Congress, it was passed by the Senate on April 8, 1864, and by the House on January 31, 1865. The amendment was ratified by the required number of states on December 6, 1865. On December 18, 1865, Secretary of State William H. Seward proclaimed its adoption.It was the first of the three Reconstruction Amendments adopted following the American Civil War. Since the American Revolution, states had divided into states that

Fig. 1. An illustration of LLM's response with citations

distortion or fabrication of facts in the generated content, as well as the lack of authoritative sources.

To enhance the discernibility and verifiability of text generation system outputs, text generation attribution methods have emerged. This approach aims to enable the text generation model to provide corresponding evidence when generating a response. Evidence usually comes in the form of citations or documents to substantiate claims or statements. Figure 1 presents a case that an LLM enhances its response by integrating evidence from identifiable sources to support its answer. This evidence ensures that the relevant statements are logically derived from the underlying corpus, enabling the general audience to understand and verify this information. Some studies have attempted to directly apply LLMs to generate attributions [4,27], yet resulting in unsatisfactory response. Other methods [14,28] focus on integrating the attribution process with information retrieval tasks, enhancing attribution by retrieving relevant documents from external sources to improve model performance. Meanwhile, researchers are also dedicated to establishing criteria for evaluating citation quality [21], aiming to assess the credibility of the content generated by text generation systems.

Despite substantial progress, existing methods for retrieval-augmented text generation attribution still exhibit noteworthy drawbacks that need to be addressed. These methods typically utilize the entire retrieved document as input, and lack effective means of verifying the attribution statements. To be specific, current language models struggle to effectively leverage information from lengthy input contexts [16]. Even models explicitly designed for handling long contexts experience a considerable drop in performance when extracting relevant information from extensive contexts. This implies that models may fail to consider crucial information when dealing with extended inputs, which in turn impacts the accuracy of the output. Moreover, using unverified attribution to explain the relationship between statements and citations poses a potential reliability risk. Without a dependable attribution verification mechanism, there is a risk of inaccurately attributing statements to irrelevant citations. In conclusion, the development of effective long context parsing and citation verification mechanisms is essential for enhancing LLMs.

To tackle the aforementioned issues, this paper introduces a novel framework called IVAKF (Iterative Verification Attribution with Keyword Fronting) for enhancing answers and attributions in LLMs. To address information loss in lengthy input contexts, IVAKF incorporates an explicit keyword fronting mechanism. Initially, the NER model [13] is utilized to identify entity information, considered essential clues within retrieved documents. Subsequently, these entities are explicitly prioritized by positioning them ahead of the original text during the construction of contextual information. Regarding the attribution verification challenge, IVAKF devises a new citation verification mechanism capable of effectively capturing the intricate relationship between statements and their supporting evidence. It leverages a Natural Language Inference (NLI) model to evaluate the logical coherence that candidate evidence offers to statements. Through iterative attribution generation, IVAKF enables LLMs to maintain high-quality attribution outcomes, rectify false statements, and address inconsistent references continuously. Experimental results demonstrate that the proposed approach significantly enhances the overall accuracy and credibility of attribution.

In summary, the main contributions of our paper include:

1. Introduction of the novel framework IVAKF, which enhances LLMs through the integration of a keyword fronting strategy and a verification-based iterative optimization algorithm, leading to the generation of verified high-quality outputs and attributions. Both algorithms are designed for easy integration as plug-ins for LLMs.
2. Development of a keyword fronting strategy to address the issue of information omission in lengthy input contexts.
3. Proposition of a verification-based iterative optimization algorithm that enables a thorough exploration of the inherent logical relationships between statements and candidate evidence. This algorithm facilitates the verification and refinement of mismatched statements and citations.
4. Experimental validation conducted on three publicly available knowledge-intensive task datasets. The results demonstrate that IVAKF offers significant advantages in terms of answer correctness and citation quality compared to current attribution methods.

2 Related Work

2.1 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a typical method that enhances the capabilities of LLMs by integrating external knowledge bases and has been widely used for efficiently handling knowledge-intensive tasks. Initially, this enhancement relied mainly on unstructured data [1,11], such as plain text. The scope of the retrieval unit varied from words, phrases to document paragraphs with different granularities, which, despite improving the precision of features to increase accuracy, also made the retrieval process more complex. This method was later further expanded to cover the use of structured data. RET-LLM [18] utilizes previous dialogues to build a knowledge graph memory bank, laying the foundation for future references. KnowledGPT [25] enhances the performance of RAG models in terms of knowledge richness and contextual relevance by generating search queries aimed at the knowledge base and storing the retrieved knowledge in a personalized library. Recently, research trends using the content generated by LLMs themselves to support retrieval and enhancement have become increasingly popular [26]. However, tasks based on these retrieval-augmented Generation methods inevitably have the problem of so-called "intermediate loss". Specifically, when language models process long text, they tend to show a weaker ability to recognize information in the middle of the text. To mitigate this problem in the attribution task, we implemented a keyword front strategy to explicitly focus important information.

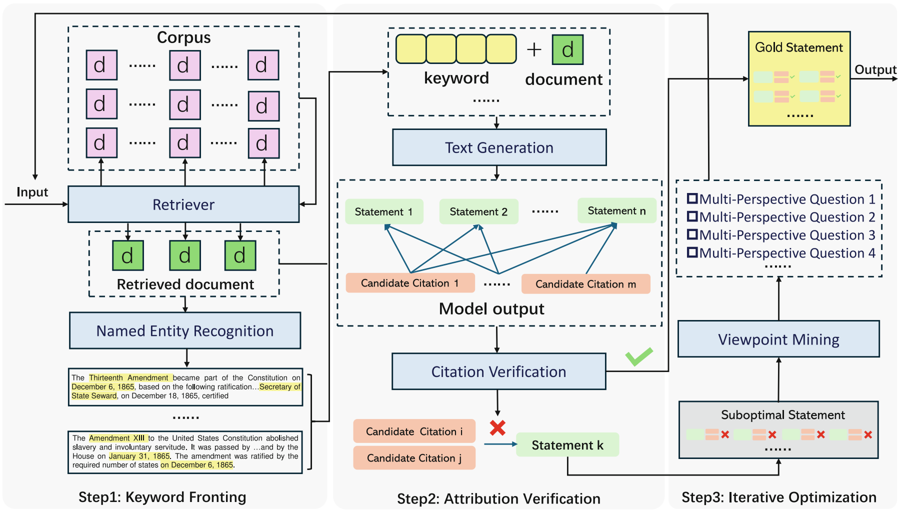
2.2 Text Generation Attribution

Text Generation Attribution refers to the process of tracing the information or decisions produced by a model back to its original materials or input features, requiring appropriate and sufficient citation support for each perspective and statement of the model. The ways in which models handle attribution are mainly divided into two categories: direct model-driven attribution and retrieval-driven attribution. In direct model-driven attribution, the cited documents originate from the model itself and are used to corroborate the generated answer. Sun et al. [24] found that by requiring the model to perform self-attribution, the factual basis of the generated text was enhanced, improving performance in downstream tasks. However, Gravel et al. [8] pointed out that even when the model correctly answers the question, the evidence it provides is often incorrect in most cases, offering knowledge sources unrelated to the current topic. Therefore, in retrieval-driven attribution, the model generates answers with citations based on retrieved documents. Among these, Li et al. [15] used knowledge graphs as sources of evidence. However, it should be noted that retrieval is not inherently equivalent to attribution. To enhance the supportiveness of documents to the answer, LLatreivel [14] continuously updates retrieval results until it is confirmed that the retrieved documents sufficiently support the answer to the question. RARR [6] first generates an answer and then performs the attribution operation. Different from previous methods, we introduce a candidate citation verification mechanism during the attribution generation process, repeatedly ensuring that the citation indeed supports the statement rather than establishing the citation for the statement all at once.

3 Methodology

3.1 Task Formalization

In general, we can formalize the text generation attribution task as follows. Specifically, given a question q and a corpus of text passages $\mathcal{D}$. The goal of the attribution task is to produce an output $\mathcal{S}$, where $\mathcal{S}$ is a set of n distinct statements $s_1, \ldots, s_n$. Each statement s_i cites a series of paragraphs $\mathcal{C}_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,n}\}$, thus the statement-citation pair within the model response can be represented as $(s_i \Leftarrow \mathcal{C}_i)$, where $c_{i,j} \in \mathcal{D}$. Model output's citation set C covers all elements of any $\mathcal{C}_i$, and ensures that each element occurs only once. For ease of narration, we adopt a format akin to citing references in academic papers for attribution, that is, placing the citation numbers within square brackets, such as [1] [2].



The Thirteenth Amendment became part of the Constitution on December 6, 1865, based on the following ratification…Secretary of State Seward, on December 18, 1865, certified

......

The Amendment XIII to the United States Constitution abolished slavery and involuntary servitude. It was passed by …and by the House on January 31, 1865. The amendment was ratified by the required number of states on December 6, 1865.

Fig. 2. The overall framework of IVAKF.

3.2 Overall Framework

Compared to traditional attribution paradigms, our proposed IVAKF framework follows a "fronting + verification + optimization" approach, with a heightened focus on mitigating the "Middle Loss" issue within lengthy input contexts and is dedicated to enhancing the consistency of statement-citation pairs. On the other hand, if correct statements lack the support of appropriate citations, they appear unconvincing. In light of this, we have implemented a full-process optimization cycle to more effectively produce accurate answers and appropriate citation matches. Figure 2 demonstrates our IVAKF framework, whose workflow mainly comprises the following three steps:

Table 1. Keywords extraction from top-2 retrieved documents. Entities $\mathcal{E}_1$,$\mathcal{E}_2$ extracted by NER models from different text passages d_1^{top2},d_2^{top2} related to question q are taken as keywords.

q	When was the 13th amendment ratified by the states ?
d_1^{top2}	ratification did not imply federal power to legislate on the status of former slaves. During the first week of December, $\boxed{\textit{North Carolina}}$ and $\boxed{\textit{Georgia}}$ gave the amendment the final votes needed for it to become part of the Constitution. The Thirteenth Amendment became part of the Constitution on December 6, 1865, based on the following ratifications: Having been ratified by the legislatures of three-fourths of the several states (27 of the 36 states, including those that had been "in rebellion"), Secretary of State $\boxed{\textit{Seward}}$, on December 18, 1865, certified that the $\boxed{\textit{Thirteenth Amendment}}$ had become valid, to all intents and
$\mathcal{E}_1$	"Georgia", "North Carolina", "Seward", "Thirteenth Amendment"
d_2^{top2}	$\boxed{\textit{Thirteenth Amendment to the UnitedStates Constitution}}$ The Thirteenth Amendment (Amendment XIII) to the $\boxed{\textit{United States Constitution}}$ abolished slavery and involuntary servitude, except as punishment for a crime. In Congress, it was passed by the Senate on April 8, 1864, and by the House on January 31, 1865. The amendment was ratified by the required number of states on December 6, 1865. On December 18, 1865, Secretary of State $\boxed{\textit{William H.Seward}}$ proclaimed its adoption. It was the first of the three $\boxed{\textit{Reconstruction Amendments}}$ adopted following the $\boxed{\textit{American Civil War}}$. Since the $\boxed{\textit{American Revolution}}$, states had divided into states that
$\mathcal{E}_2$	"William H. Seward", "Thirteenth Amendment to the United States Constitution", "American Revolution", "United States Constitution", "Reconstruction Amendments", "American Civil War"

1. Keyword Fronting: Key information from external knowledge is extracted through NER models at first, and integrated into the process of constructing contextual prompts.
2. Attribution Verification: Based on the prompts enriched with key information obtained from the first step, raw statements and candidate citations are generated, followed by verification of the extent to which the candidate citations support the raw statements.
3. Iterative Optimization: In the previous step, statements that did not pass attribution verification will undergo iterative statement corrections and citation updates.

3.3 Keyword Fronting

Extract Document Keywords. For effectively capture key information from the retrieved documents, we propose using a NER model to extract paragraph entities as keywords. In our experiments, we chose the span-marker-mbert-base-multinerd model [5] for this NER task. This model is a fine-tuned version of SpanMarker based on the bert-base-multilingual-cased model, specifically trained for multilingual named entity recognition, and has demonstrated high precision and recall in the field of multilingual named entity recognition. Specifically, the retriever R searches for the top-k most suitable documents $\mathcal{D}^{topk} = \left\{d_1^{topk}, d_2^{topk}, \ldots, d_k^{topk}\right\}$ based on the question q as follows:

$$\mathcal{D}^{topk} = \mathrm{R}(q, \mathcal{D}, k) = \text{Top-k}_{d_i^{topk} \in \mathcal{D}} \, \text{simi-score}(q, d_i^{topk}), \tag{1}$$

where simi-score$(\cdot, \cdot)$ is the similarity score of R. Then we require the NER model to extract a set of entities $\mathcal{E}_i = \{e_{i,1}, e_{i,2}, \ldots, e_{i,n}\}$ from each document d_i^{topk}, where $d_i^{topk} \in \mathcal{D}^{topk}$. So we have

$$\mathcal{E}_i = NER_{\mathcal{D}^{topk} \subseteq \mathcal{D}}(d_i^{topk}). \tag{2}$$

Table 1 shows an example of keyword extraction. From this example, we can clearly see that for the question "When was the 13th amendment ratified by the states ?" the retriever returned two paragraphs and successfully identified their respective relevant entities. These entities, as keywords, effectively carry most of the information content of the entire paragraph.

Construct Keyword-Based Documents. After completing the entity extraction task for all documents in $\mathcal{D}^{topk}$, we need to efficiently utilize the obtained entity information $\mathcal{E}_i$. According to research by Liu et al. [16], LLMs can process information more effectively when it is located at the beginning or end rather than in the middle. Therefore, when constructing the context, we explicitly place the entity information in front of the corresponding document. We use the ordered text pair set $\mathcal{D}^{topk^*}$ consisting of entity information $\mathcal{E}_i$ and document d_i^{topk} to represent the result of the construction, which can be defined as:

$$\mathcal{D}^{topk^*} = \left\{(\mathcal{E}_i, d_i^{topk}) \mid \mathcal{E}_i = \{e_{i,1}, e_{i,2}, \ldots, e_{i,n}\}, d_i^{topk} \in \mathcal{D}^{topk}\right\} \tag{3}$$

We did not simply discard the original document because relying on keywords alone does not adequately capture the context of the entire document. Context is the key to understanding the relationship between entities in a text, and to generate an excellent model response, both of them are needed to achieve perfection.

3.4 Attribution Verification

The study [29] notes that direct model-driven attribution is only about 50% successful in providing the correct answer to a question, and its recommended attribution sources are only 14% true. For the four generative search engines that used retrieval-driven attribution, on average, only 51.5% of the generated sentences were fully supported by the citations, while 74.5% of the citations supported the relevant generated sentences [17]. In our study, we do not consider the one-off generated statements and attributions as the final output. Conversely, we filter out those statement-citation pairs that are logically untenable through the verification process, so as to reconstruct these suboptimal responses in subsequent steps. The details of this step are as follows:

Statement and Candidate Citation Generation. We follow Gao et al. [7] use the instruction I and demonstrations to guide LLMs to generate answers and citations to documents $\mathcal{D}^{topk}$, the instruction looks like this "Instruction: Refer to the provided search content and keywords to write high-quality answers to the given question and reference them correctly using [1][2][3] whenever possible." For a specific question q, we concatenate it with the top-k relevant documents $\mathcal{D}^{topk^*}$ constructed in the previous step that contain keyword information $\mathcal{E}_i \, (1 \leq i \leq k)$, as well as the selected few-shot prompts, all serving as input to the LLM. The model output includes a set of candidate statements $\mathcal{S}^{'}$ and their pointers to the citation, which can be represented by statement-citation pairs as follows:

$$(s_i^{'} \Leftarrow \mathcal{C}_i^{'}) = \text{LLM}_{\text{few-shot}}(I, q, \mathcal{D}^{topk^*}), \tag{4}$$

where $s_i^{'} \in \mathcal{S}^{'}$ and supported by a set of candidate citations $\mathcal{C}_i^{'}$ $(\mathcal{C}_i^{'} \subseteq \mathcal{D}^{topk})$, which is a LLMs determination that may not always be accurate. Therefore, we will next verify these citations to determine whether they indeed support the corresponding statements.

Leveraging NLI Models for Effective Verification. Inspired by natural language inference tasks, we utilize a NLI model to assess whether candidate citations logically support the current statement. The core of this model lies in determining the semantic entailment relationship between two sentences or phrases, namely "premise" and "hypothesis". In addition, we adopt a strict standard that requires the text to only faithfully reflect its original content, without considering its "correctness" towards the "real world." This makes the task definition clearer because judging the truthfulness of general "real world" facts is subjective, depending on an individual's knowledge, values, and beliefs. This definition also demonstrates similar rigor in tasks involving textual entailment, question answering, summarization, and other understandings based on given foundational texts without considering conflicts with other world knowledge [9]. Citation validation uses binary labeling, which means that the NLI model outputs 1 if the premise (candidate citations $\mathcal{C}_i^{'}$) entails the hypothesis (candidate statement $s_i^{'}$), and 0 otherwise.

$$NLI_{\mathcal{C}_i{'} \subseteq \mathcal{D}^{topk}, s_i{'} \in \mathcal{S}'}(\mathcal{C}_i^{'}, s_i^{'}) = \begin{cases} 1 \text{ if } \mathcal{C}_i^{'} \text{ entails } s_i^{'}, \\ 0 \text{ otherwise.} \end{cases} \tag{5}$$

In an ideal scenario, all statement-citation pairs should be labeled as 1 by the NLI model, indicating that these statements are backed by precise citations and not just unfounded conjecture. If a label of 0 occurs, it only signifies that the statement cannot be supported by the corresponding citations, and we must not hastily dismiss these statements as factual errors and exclude them from the final response. This is because the output generated by the model may depend not only on external non-parametric knowledge but also on its own internal parametric knowledge. This internal knowledge could be correct and relevant yet uncaptured by the retrieved documents, or it could be omitted due to the "Middle Loss" issue.

3.5 Iterative Optimization

Careful consideration reveals that for all the candidate statements $\mathcal{S}'$, the reason for the verification failure of the statement-citation pairs could be that the statement itself is a "hallucination" or the citation collection lacks references related to the statement. To correct this "hallucination" and update the citation collection, we need to broaden the scope of the information collected. This is a process of continuous iterative optimization.

Algorithm 1. Iterative Optimization of IVAKF

```
Input: corpus $\mathcal{D}$, original query $q$, gold statement set $G$, suboptimal statement set $B$,
    the NLI model, the NER model, the LLM, the retriever $R$, the maximum iteration
    $T$
Output: gold statement set $G$ and suboptimal statement set $B$
 1: $t \leftarrow 0$
 2: Initialize G and B
 3: while $t < T$ do
 4:     if $t = 0$ then
 5:         $\mathcal{D}^{topk} \leftarrow \mathrm{R}(q, \mathcal{D}, k)$
 6:     else
 7:         $\mathcal{Q}^* \leftarrow \mathrm{LLM}_{\text{few-shot}}(I_q, q, S_f)$
 8:         $\mathcal{D}^{topk} \leftarrow \mathrm{R}(\mathcal{Q}^*, \mathcal{D}, k)$
 9:         $B \leftarrow B - (s_f \Leftarrow \mathcal{C}_f)$
10:     end if
11:     $\mathcal{D}^{topk^*}$ $\leftarrow$ Use NER model to extract keywords and put in front of $\mathcal{D}^{topk}$
12:     $(s_i{}' \Leftarrow \mathcal{C}_i{}')_{NEW} \leftarrow \mathrm{LLM}_{\text{few-shot}}(I, \mathcal{Q}^*, \mathcal{D}^{topk^*})$
13:     if $NLI_{\mathcal{C}_i{}' \subseteq \mathcal{D}^{topk^*}}(\mathcal{C}_i{}', s_i{}') = 1$ then
14:         $G \leftarrow G \cup (s_v \Leftarrow \mathcal{C}_v)_{NEW}$
15:     else
16:         $B \leftarrow B \cup (s_f \Leftarrow \mathcal{C}_f)_{NEW}$
17:     end if
18:     $t = t + 1$
19: end while
20: return $G$ and $B$
```

Our approach for iterative optimization is presented in Algorithm 1. Specifically, each verified statement-citation pair will be saved into a Gold Statement

Set G and denoted as $(s_v \Leftarrow \mathcal{C}_v)$. Similarly, each unvalidated statement-citation pair $(s_f \Leftarrow \mathcal{C}_f)$ will be saved to the Suboptimal Statement Set B and proceed to the next round of iterative optimization. Firstly, we will conduct viewpoint mining through instruction to guide the LLM to generate multi-perspective questions $\mathcal{Q}^* = \{q_1^*, \dots q_m^*\}$about the original query q and all the suboptimal statements S_f, which is given by:

$$\mathcal{Q}^* = \mathrm{LLM}_{\text{few-shot}}(I_q, q, S_f), \tag{6}$$

where I_q represents the instruction input to the LLM, specifically saying: "Based on the original query and the following statements, please generate no more than K multi-perspective questions that help to reveal more information." Then, the retriever retrieves relevant documents $\mathbb{D}$ from external corpus based on these multi-perspective questions, and LLM generates some new statement-citation pairs $(s_i^{'} \Leftarrow \mathcal{C}_i^{'})_{NEW}$ based on these newly obtained documents. The $(s_i^{'} \Leftarrow \mathcal{C}_i^{'})_{NEW}$ also need to undergo attribution verification. Finally, the gold statement set is updated, and the iterative process continues until the preset maximum number of iterations is reached. Notably, before generating new statements in each round, the newly retrieved documents also need to undergo keyword fronting operations.

4 Experiments

4.1 Experimental Setup

Datasets. We conducted experiments on three publicly available knowledge-intensive question answering task datasets, including ASQA [22], ELI5 [3], and NQ [12]. Among them, ASQA and ELI5 are datasets specifically designed for long-form question answering, requiring answers to be composed of multiple sentences to comprehensively address all aspects of a question. In contrast, NQ is the first dataset created based on naturally occurring queries, comprising 300,000 real-world questions along with their corresponding human-annotated answers from Wikipedia pages. We randomly selected 1,000 examples from the development set of each dataset as the basis for performance evaluation.

Evaluation Metrics. In order to evaluate the correctness of the answers, we adopted the evaluation criteria proposed by Gao et al. [7] when evaluating the long-form QA datasets ASQA and ELI5. For ASQA, we use Exact Match recall (*EM-R*) to calculate the recall of correct short answers. When processing the ELI5, the NLI model is used to measure whether the model prediction entails the sub-claims of the gold answer (*Claim*). Following Sun et al. [23], we use Exact Match (*EM*) scores as correctness evaluation metrics for open-domain QA dataset NQ. In addition, to assess the quality of citations in responses, we focused on calculating *Citation Recall* [7], *Citation Precision* [7], and the *Citation F1* [14] score. *Citation Recall* evaluates the proportion of all statements in the output that are supported by cited passages, while *Citation Precision* checks the portion of irrelevant citations among all citations.

Baselines. The five baseline methods proposed in ALCE [7] are chosen for comparison: (i) answers with citations are generated based on top-k documents (VANILLA), (ii) answers with citations are generated based on the summaries, which are summarized from the top-k documents (SUMM), (iii) answers with citations are generated based on relevant snippets, which are extracted from the top-k documents (SNIPPET), (iv) use high temperature to generate four unique responses and outputs the one with the highest citation recall (RERANK), (v) generate answers with citations directly without any retrieved documents (CLOSEDBOOK). Additionally, we also examine (vi) LLatrieval [14], an attribution method in which the LLM continuously updates the retrieval results until it verifies that the retrieved documents can support answering the question.

Implementation Details. We use LLMs with different parameter sizes and context window for evaluation, specifically, gpt-3.5-turbo-0613, gpt-3.5-turbo-16k-0613 and the open-source model Vicuna-13B [2]. For the natural language inference task, we chose the currently best performing TRUE model [20]. In addition, the 2018-12-20 Wikipedia snapshot was used as the retrieval corpus, and GTR [19] was chosen as the dense retriever. In terms of model decoding methods, ChatGPT uses a sampling with temperature 0.5, while Vicuna uses Nucleus sampling and sets the top-p parameter to 0.95. Unless otherwise stated, gpt-3.5-turbo-0613 was used in all experiments in this paper, and the number of multi-perspective questions was set to 2, and 3 related documents were retrieved for each multi-perspective question. The initial number of documents retrieved is 5, and the maximum number of iterations is limited to 5.

4.2 Main Results

Table 2 presents the results of our IVAKF framework and baseline attribution approach in three knowledge-intensive datasets. We can see that IVAKF performs better than baseline on all three datasets. This shows the effectiveness of IVAKF in improving the correctness of model output and improving the quality of attribution. Specifically, when evaluated using the three LLMs, Vicuna-13B, gpt-3.5-turbo-0613, and gpt-3.5-turbo-16k-0613, respectively, IVAKF improved by 6.4%, 5.4%, and 4.5% in terms of correct responses compared to the optimal baseline, and in terms of citation quality then improved by 9.1%, 8.7% and 8.3%, respectively. These experimental results clearly show that our approach can achieve significant performance improvements for LLMs with different parameter sizes and context window sizes.

The baseline approach often optimizes one performance at the expense of others. SUMM and SNIPPET have shown an improvement in correctness, thanks to their strategy of using summaries or snippets instead of full documents, which dramatically reduces the context length while preserving core information as much as possible, thereby mitigating the "Middle Loss" problem and reducing noise. However, overcompressing the text can be detrimental to the quality of the citations. Similarly, CLOSEDBOOK maintains a high level of answer correctness by completely eliminating the interference of external knowledge, but

Table 2. Comparisons between IVAKF and baselines on knowledge-intensive QA task.

	ASQA				ELI5				NQ				Overall	
	Correct	Citation			Correct	Citation			Correct	Citation			Correct	Citation F1
	EM-R	Rec	Pc	F1	Claim	Rec	Prec	F1	EM	Rec	Prec	F1		
Vicuna-13B														
VANILLA	31.40	52.31	51.18	51.74	11.20	16.46	14.27	15.29	53.20	59.78	53.51	56.47	31.93	41.17
SUMM	42.53	49.75	50.71	50.23	10.26	13.23	12.41	12.81	56.40	48.65	47.71	48.18	36.40	37.07
SNIPPET	40.02	46.59	48.52	47.54	13.35	20.57	18.30	19.37	56.40	40.26	38.13	39.17	36.60	35.36
RERANK	34.86	64.27	63.03	63.64	11.47	29.87	36.41	32.82	55.80	61.23	55.64	58.30	34.04	51.59
CLOSEDBOOK	33.84	15.42	15.42	15.42	17.16	5.30	5.30	5.30	56.00	24.32	24.32	24.32	35.67	15.01
LLatrieval	43.64	59.96	60.45	60.20	12.42	27.85	34.68	30.89	58.30	60.02	56.34	58.12	38.12	49.74
IVAKF	**44.67**	**66.83**	**66.23**	**66.52**	**17.67**	**38.53**	**44.17**	**41.16**	**59.40**	**63.59**	**59.10**	**61.26**	**40.58**	**56.31**
gpt-3.5-turbo-0613														
VANILLA	40.35	70.13	71.17	70.65	13.05	53.43	51.12	52.25	61.20	62.38	56.26	59.16	38.20	60.69
SUMM	42.14	64.38	62.94	63.65	11.87	47.60	49.20	48.39	61.50	48.61	42.85	45.55	38.50	52.53
SNIPPET	39.85	61.14	59.26	60.19	13.56	46.21	46.29	46.25	63.00	56.74	50.47	53.42	38.80	53.29
RERANK	38.55	75.01	76.84	75.91	12.68	74.36	68.54	71.33	61.00	70.30	66.34	68.26	37.41	71.83
CLOSEDBOOK	39.13	27.94	27.94	27.94	18.23	14.66	14.66	14.66	62.50	34.62	34.62	34.62	39.95	25.74
LLatrieval	42.21	78.36	75.23	76.76	16.74	73.02	69.87	71.41	65.50	71.41	65.55	68.35	41.48	72.17
IVAKF	**44.05**	**84.25**	**81.02**	**82.60**	**18.68**	**78.82**	**75.21**	**76.97**	**68.50**	**76.34**	**75.09**	**75.71**	**43.74**	**78.43**
gpt-3.5-turbo-16k-0613														
VANILLA	38.76	72.44	71.57	72.00	13.91	48.32	47.97	48.14	60.50	63.12	55.94	59.31	37.72	59.82
SUMM	41.23	66.14	61.31	63.63	12.55	44.38	49.15	46.64	62.20	49.31	40.76	44.63	38.66	51.63
SNIPPET	40.32	63.14	61.28	62.20	14.05	43.76	41.36	42.53	61.80	58.10	49.32	53.35	38.72	52.69
RERANK	39.62	78.26	77.19	77.72	15.14	70.29	69.17	69.73	61.80	72.69	68.62	70.60	38.85	72.68
CLOSEDBOOK	38.63	26.44	26.44	26.44	18.35	14.16	14.16	14.16	60.50	33.43	33.43	33.43	39.16	24.68
LLatrieval	43.25	80.22	77.14	78.65	16.21	72.02	69.39	70.68	63.80	70.03	68.39	69.20	41.09	72.84
IVAKF	**44.54**	**83.55**	**81.12**	**82.32**	**18.53**	**78.63**	**76.54**	**77.57**	**65.80**	**76.34**	**77.22**	**76.78**	**42.96**	**78.89**

its citation quality is lower due to the lack of citation support. In contrast, RERANK stands out in citation quality due to its multi-sampling strategy. It is worth noting that LLatrieval adopts a similar iterative approach to this paper and makes significant progress in both correctness and citation quality. But its iteration is only limited to the statement generated before retrieval phase, does not involve direct attribution of quality assurance. The iterative optimization of IVAKF covers the whole process, and clearly introduces the attribution verification mechanism, which realizes the mutual enhancement of correctness and citation quality.

4.3 Ablation Studies

To analyze in depth the specific impact of each component in IVAKF on overall performance, we conducted a series of ablation experiments on the ASQA dataset. By removing each component one by one, we were able to accurately assess their respective contributions. In this process, "w/o KF" represents the removal of the keyword fronting strategy; "w/o VM" means that viewpoint mining is not implemented to generate multi-perspective questions during the iteration process, but only the current statement is utilized to iterate. Considering that attribution verification is used jointly with iterative optimization, therefore, "w/o AV & IO" means that the verification-based iterative optimization is removed.

Table 3. Ablation Study on ASQA.

	Correct	Citation		
	EM	Rec	Pc	F1
IVAKF	**44.05**	**84.25**	**81.02**	**82.60**
-w/o KF	41.27	81.15	77.37	79.21
-w/o VM	40.42	76.26	73.51	74.86
-w/o AV & IO	40.13	69.15	71.21	70.16

The experimental results shown in Table 3 clearly show that the model performance is adversely affected once any component is removed. In particular, the performance of the model deteriorates significantly in the absence of citation validation and iterative optimization support. This is mainly because the model cannot judge the appropriateness of candidate citations without the process of attribution verification. At the same time, the lack of iterative optimization also hinders the possibility of improving the quality of attribution. In addition, the removal of keyword lead steps also weakens the performance of the model, mainly because the lack of explicit prompts for keywords increases the risk of ignoring key intermediate information in a long context. Finally, the removal of the viewpoint mining operation limits the model's ability to retrieve relevant documents, thus affecting the overall performance.

4.4 Impact of Hyperparameters

In this section, we explore the effects of different parameters on IVAKF performance. Where M represents the number of generated multi-perspective questions and N represents the number of documents that need to be retrieved for each multi-perspective question. As shown in Table 4, we find that providing a more comprehensive perspective of the question and increasing the number of relevant documents helps LLM to utilize more comprehensive information, which in turn improves the correctness of output and the quality of citations. However, simply increasing M and N does not consistently produce better results; When their value exceeds a certain threshold, performance deteriorates. This is mainly because a large pool of documents can introduce noisy information, which can have a negative impact on both the statement generation and citation generation processes. In addition, we observed that obtaining 5 initial documents was better than 10 documents during the retrieval of the original question, suggesting that the model may not be able to efficiently utilize all the information when dealing with overly long contexts. In addition, we studied the effects of a few

Table 4. Hyperparameter analysis of IVAKF on ASQA.

	5-passage				10-passage			
	Correct	Citation			Correct	Citation		
	EM	Rec	Pc	F1	EM	Rec	Pc	F1
N = 1								
M = 1	41.82	76.67	74.48	75.56	39.76	77.52	76.35	76.93
M = 2	43.38	81.46	78.92	80.17	41.65	78.19	76.32	77.24
M = 3	42.32	79.32	75.32	77.27	41.92	76.82	74.59	75.69
N = 3								
M = 1	43.52	80.45	76.47	78.41	40.74	81.21	78.92	80.05
M = 2	**44.05**	**84.25**	**81.02**	**82.60**	42.13	80.26	79.61	79.93
M = 3	44.68	82.13	78.35	80.20	39.54	79.33	78.21	78.76
N = 5								
M = 1	41.92	80.72	79.24	79.97	36.99	78.91	78.51	78.71
M = 2	43.09	82.65	80.02	81.31	38.41	78.96	79.44	79.20
M = 3	42.24	80.64	77.38	78.98	37.62	76.93	78.68	77.80

shot prompt on answering questions and generating multi-perspective questions. Table 5 and Table 6 shows the templates. In general, the 0-shot and 1-shot sample sizes are not large enough for the language model to fully understand the learning context. When the sample size is increased to 2-shot and 3-shot, the language model is able to generate better-quality responses and a rich variety of multi-perspective questions. However, if the number of samples is too large, excessive costs will be incurred.

4.5 The Performance of the Iteration

In this section, we will analyze the impact of multiple iterations on the performance of three datasets. Specifically, we plot the performance change line in Fig. 3. From this figure, we can see that the correctness and citation quality of IVAKF improve with the increase of the maximum number of iterations, which confirms the effectiveness of the multi-iteration optimization strategy in our method. At the same time, we also found that due to the limitation of the content richness of the corpus, although increasing the maximum number of iterations could indeed improve the performance, the growth trend was not linear all the time, but tended to flatline after reaching a certain stage, indicating that the performance began to approach saturation. Therefore, considering the issue of token consumption, we recommend that the iteration parameter setting should preferably not exceed 5.

Table 5. Few-shot prompt template for LLMs to generate statements and candidate citations.

Instructions: Refer to the provided search content and keywords to write high-quality answers to the given question and reference them correctly using [1] [2] [3] whenever possible.

[Example Query 1]: Number of branches of oriental bank of commerce in india ?

[Doc 1]: (Keywords: OBC, 31 March 2010, Public Sector Bank, India, ...) as on 31 March 2010 making it the seventh largest Public Sector Bank in India. On 14 August 2004, OBC amalgamated Global Trust Bank (GTB) ... The acquisition brought with it 103 branches, which ...

[Doc 2]:(Keywords:{keyword1, Keyword2, Keyword3, ...}){Retrieved Document}

[Doc 3]:(Keywords:{keyword1, Keyword2, Keyword3, ...}){Retrieved Document}

...

[Answer]: In August 2004, Oriental Bank of Commerce in India united with Global Trust Bank, which was ... the acquisition brought with it 103 branches [1], which brought OBC's branch total to 1092. According to ..., it has 2390 branches [2]. On April 1st, 2020, Oriental Bank ... the total amount of branches to 11,437 [3].

[Example Query 2]: ...

[Example Query 3]: ...

...

[Original Query]: When was the 13th amendment ratified by the states ?

Table 6. Few-shot prompt template for LLMs to generate multi-perspective questions.

Instructions: Based on the original query and the following statements, please generate no more than K multi-perspective questions that help to reveal more information.

[Example Query 1]: Number of branches of oriental bank of commerce in india ?

[statement 1]:On April 1st, 2020, Oriental Bank of Commerce ... brings the total amount of branches to 11,437.

[statement 2]:...

...

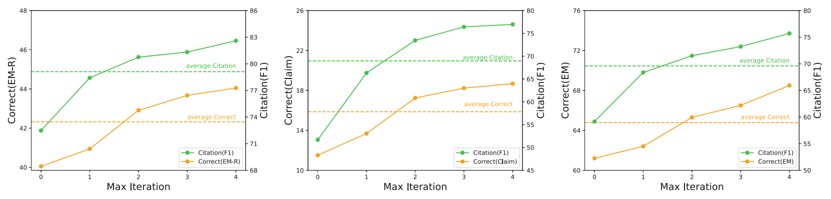
[multi-perspective questions]: "What is the total number of branches of the newly formed bank after the merger of OBC, United Bank of India and Punjab National Bank on 1 April 2020 ?", "According to the ...", ...

[Example Query 2]: ...

[Example Query 3]: ...

...

[Original Query]: When was the 13th amendment ratified by the states ?



(a) ASQA (b) ELI5 (c) NQ

Fig. 3. Performance analysis with different max iteration.

5 Conclusion

In this paper, we design and propose IVAKF, a novel framework for the text generation attribution. This framework adopts a keyword fronting strategy at first, focusing on extracting key information from long texts, and ensuring that these important document features can be accurately identified, extracted, and effectively applied during the text generation process. Moreover, compared with the traditional text generation attribution process, IVAKF also combines attribution verification with iterative optimization. This algorithm enables the attribution system to accurately evaluate the rationality of the statement and its citation, and to perform iterative optimization in time, so as to improve the correctness of response and the quality of attribution. We conducted extensive experiments on knowledge-intensive tasks across three different datasets to verify the effectiveness of the IVAKF framework. The experimental results indicate that the framework significantly enhances the performance of text generation models in terms of attribution accuracy and generated text quality, demonstrating its advantages in dealing with text generation attribution tasks.

Acknowledgements. This work is partially supported by the National Natural Science Foundation of China (No. 91948303-1, No. 61803375, No. 12002380, No. 62106278, No. 62101575, No. 61906210), the National University of Defense Technology Foundation (No. ZK20-52) and the Independent and open subject fund (grant no.202201-06) from State Key Laboratory of High Performance Computing.

References

1. Borgeaud, S., et al.: Improving language models by retrieving from trillions of tokens. In: International Conference on Machine Learning, pp. 2206–2240. PMLR (2022)
2. Chiang, W.L., et al.: Vicuna: an open-source chatbot impressing GPT-4 with 90%* chatgpt quality (2023). https://lmsys.org/blog/2023-03-30-vicuna/
3. Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., Auli, M.: ELI5: long form question answering. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 3558–3567 (2019)
4. Fierro, C., et al.: Learning to plan and generate text with citations. arXiv preprint arXiv:2404.03381 (2024)

5. Fu, J., Huang, X., Liu, P.: Spanner: named entity re-/recognition as span prediction. arXiv preprint arXiv:2106.00641 (2021)
6. Gao, L., et al.: RARR: researching and revising what language models say, using language models. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 16477–16508 (2023)
7. Gao, T., Yen, H., Yu, J., Chen, D.: Enabling large language models to generate text with citations. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 6465–6488 (2023)
8. Gravel, J., D'Amours-Gravel, M., Osmanlliu, E.: Learning to fake it: limited responses and fabricated references provided by ChatGPT for medical questions. Mayo Clinic Proc. Digit. Health **1**(3), 226–234 (2023)
9. Honovich, O., et al.: True: re-evaluating factual consistency evaluation. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 3905–3920 (2022)
10. Ji, Z., et al.: Survey of hallucination in natural language generation. ACM Comput. Surv. **55**(12), 1–38 (2023)
11. Jiang, Z., et al.: Active retrieval augmented generation. arXiv preprint arXiv:2305.06983 (2023)
12. Kwiatkowski, T., et al.: Natural questions: a benchmark for question answering research. Trans. Assoc. Comput. Linguist. **7**, 453–466 (2019)
13. Li, J., Sun, A., Han, J., Li, C.: A survey on deep learning for named entity recognition. IEEE Trans. Knowl. Data Eng. **34**(1), 50–70 (2020)
14. Li, X., Zhu, C., Li, L., Yin, Z., Sun, T., Qiu, X.: Llatrieval: LLM-verified retrieval for verifiable generation. arXiv preprint arXiv:2311.07838 (2023)
15. Li, X., Cao, Y., Pan, L., Ma, Y., Sun, A.: Towards verifiable generation: a benchmark for knowledge-aware language model attribution. arXiv preprint arXiv:2310.05634 (2023)
16. Liu, N.F., et al.: Lost in the middle: how language models use long contexts. Trans. Assoc. Comput. Linguist. **12**, 157–173 (2024)
17. Liu, N.F., Zhang, T., Liang, P.: Evaluating verifiability in generative search engines. In: The 2023 Conference on Empirical Methods in Natural Language Processing (2023)
18. Modarressi, A., Imani, A., Fayyaz, M., Schütze, H.: RET-LLM: towards a general read-write memory for large language models. arXiv preprint arXiv:2305.14322 (2023)
19. Ni, J., et al.: Large dual encoders are generalizable retrievers. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pp. 9844–9855 (2022)
20. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
21. Rashkin, H., et al.: Measuring attribution in natural language generation models. Comput. Linguist. **49**(4), 777–840 (2023)
22. Stelmakh, I., Luan, Y., Dhingra, B., Chang, M.W.: ASQA: factoid questions meet long-form answers. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pp. 8273–8288 (2022)
23. Sun, H., et al.: Allies: prompting large language model with beam search. In: The 2023 Conference on Empirical Methods in Natural Language Processing (2023)
24. Sun, Z., Wang, X., Tay, Y., Yang, Y., Zhou, D.: Recitation-augmented language models. In: The Eleventh International Conference on Learning Representations (2023). https://openreview.net/forum?id=-cqvvvb-NkI

25. Wang, X., et al.: Knowledgpt: enhancing large language models with retrieval and storage access on knowledge bases. arXiv preprint arXiv:2308.11761 (2023)
26. Wang, Y., Li, P., Sun, M., Liu, Y.: Self-knowledge guided retrieval augmentation for large language models. arXiv preprint arXiv:2310.05002 (2023)
27. Weller, O., Marone, M., Weir, N., Lawrie, D., Khashabi, D., Van Durme, B.: " according to..." prompting language models improves quoting from pre-training data. arXiv preprint arXiv:2305.13252 (2023)
28. Xu, S., Pang, L., Shen, H., Cheng, X., Chua, T.S.: Search-in-the-chain: towards the accurate, credible and traceable content generation for complex knowledge-intensive tasks. arXiv preprint arXiv:2304.14732 (2023)
29. Zuccon, G., Koopman, B., Shaik, R.: Chatgpt hallucinates when attributing answers. In: Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region, pp. 46–51 (2023)

Reconstructing the Unseen: GRIOT for Attributed Graph Imputation with Optimal Transport

Richard Serrano[1(✉)], Charlotte Laclau[2], Baptiste Jeudy[1], and Christine Largeron[1]

[1] Laboratoire Hubert Curien, 42000 Saint-Étienne, France
{richard.serrano,baptiste.jeudy,christine.largeron}@univ-st-etienne.fr
[2] Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France
charlotte.laclau@telecom-paris.fr

Abstract. In recent years, there has been a significant surge in machine learning techniques, particularly in the domain of deep learning, tailored for handling attributed graphs. Nevertheless, to work, these methods assume that the attributes values are fully known, which is not realistic in numerous real-world applications. This paper explores the potential of Optimal Transport (OT) to impute missing attributes on graphs. To proceed, we design a novel multi-view OT loss function that can encompass both node feature data and the underlying topological structure of the graph by utilizing multiple graph representations. We then utilize this novel loss to train efficiently a Graph Convolutional Neural Network (GCN) architecture capable of imputing all missing values over the graph at once. We evaluate the interest of our approach with experiments both on synthetic data and real-world graphs, including different missingness mechanisms and a wide range of missing data. These experiments demonstrate that our method is competitive with the state-of-the-art in all cases and of particular interest on weakly homophilic graphs.

Keywords: Attributed Graph · Missing Data Imputation · Optimal Transport

1 Introduction

Graphs have become an indispensable tool for modeling and solving a wide range of practical problems. From transportation networks to protein-protein interactions, graphs provide a natural and versatile representation framework for modeling relationships of various kinds. In this context, so-called attributed

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_16.

graphs possess a valuable set of information about the objects whose relationships they model, e.g., the personal information of users for online social networks, or the properties of atoms for a molecule. However, for real-world applications, attributed graphs often suffer from missing data [17]. For instance, it is quite common for traditional online social networks, usually modeled as graphs of users, to have missing attribute values, as some of the information filled in by users is not mandatory (e.g., gender, age). This will result in missing node attributes in a graph.

Imputing missing data is a long-standing challenge in statistics that has been at the forefront of research and practice for decades [19]. This problem arises when datasets contain missing or incomplete information, which can lead to biased analyses, reduced statistical power, and inaccurate results.

Missing data may be the result of different factors, also referred to in the literature as *mechanisms* [25]. The simplest case is MCAR (Missing Completely At Random), where the missingness of the attribute values (true or false) can be modeled by i.i.d. random variables. For instance, a social platform's localization data might be lost due to occasional technical glitches. In the MAR (Missing At Random) case, the missing data probability depends on other observed attributes. For example, in social media profiles, MAR might occur if the probability of users omitting their hobbies depends on their gender. MNAR (Missing Not At Random) is the most challenging case, where missing data probability depends on unobserved variables. For instance, users may choose not to share their income due to personal factors, unaccounted for in the dataset. Depending on the mechanism, imputation can be more or less difficult.

In graph data, one approach to address missing node attributes is to cast the problem as imputing missing information in tabular data, given that node attributes are structured as a matrix. However, this method does not use the crucial structural dependencies inherent in graph data. Therefore, addressing missing information on graphs requires methods that respect and leverage the underlying graph structure [10].

Most imputation methods on graphs assume homophily [24], but as it turns out, real graphs are sometimes heterophilic [34]. A graph is homophilic if nodes sharing similar attributes tend to be more often connected than those with different attributes: "birds of a feather flock together" [20]. In this paper, we are interested in the problem of the imputation of missing node attributes, whatever the nature of the graph, homophilic or heterophilic, or the missingness mechanism (MCAR or MNAR). To proceed, we propose to exploit the Optimal Transport (OT) theory to impute missing node attributes on graphs. In recent years, OT has proved remarkably successful in machine learning notably for missing values imputation on tabular data [23] and more recently with applications on graphs including node embedding [33], fair edge prediction [18], graph prediction [2], to name a few. The intuition of using OT for imputation [23] is that the distance between two random samples from the data distribution should be small (using Wasserstein distance from optimal transport theory). Thus, a good imputation of missing values should minimize this distance between many pairs of random samples. This goal is particularly suitable for OT-based distances, as they can be (and have already been) used in gradient-based optimization as valid loss terms due to their attractive differentiability properties [2,21]. However, the classic Wasserstein distance does not take the graph topology into account.

Contributions. We propose to use a novel distance, *multi-view Wasserstein* (MultiW), able to take into account any representation of the graph, such as attributes, topology, but also hierarchy, spectral decomposition, and more. This distance is also more flexible and more computationally efficient than similar multi-view loss functions, such as FGW [29] or OTT [13]. The MultiW distance is used as a loss to train a Graph Neural Network (GNN) model able to impute missing attributes on a graph. The main distinctive feature of our approach is its ability to use the trained imputer on new nodes without the need to be retrained (contrary to state-of-the-art FP [24]). This feature is of particular interest as many graphs, such as social networks, are inherently dynamic.

Summing up, our contributions embed (1) the creation of an efficient Multi-view loss function referred to as MultiW; (2) a framework for graph missing attributes imputation `GRIOT` (**GR**aph **I**mputation with **O**ptimal **T**ransport) ; an extensive empirical study of the performance of our approach and the most recent state-of-the-art methods on a very wide variety of scenarios, whereas most studies generally focus on a very small subset of these scenarios. The code is available online[1].

2 Related Works

Over the years, researchers have developed a wide range of techniques to tackle the problem of missing data imputation [26,28,32]. Despite the progress made, the field of research for data imputation remains dynamic due to evolving data complexity. In this paper, we specifically address missing node attribute values in data structured as graphs. There has been a renewal of interest in graphs, driven particularly by the rise of Graph Neural Networks (GNNs) that require complete attributes.

While various approaches like SAT [3], GCNMF [27], and PaGNN [11] have aimed to adapt GNNs to this context, they primarily emphasize task performance rather than imputation quality. Conversely, some methods employ GNNs for graph completion [1,22], focusing on attribute matrix reconstruction over task performance, but they often encounter scalability challenges.

Finally, to the best of our knowledge, Feature Propagation (FP) [24] is currently the state-of-the-art method for missing node attribute imputation. FP is a diffusion-based attribute reconstruction approach that allows imputation on the graphs upstream to the node classification task. As such, it is not tied to any particular model or architecture for solving the final task. However, similarly to the aforementioned approaches, FP assumes a strong attribute-based homophily in the graph to impute the missing attributes. This means that nodes

[1] Code and additional material available at: github.com/RichardSrn/GRIOT.

in the graph are more likely to be connected to other nodes that have similar attributes.

Our approach is at the crossroads of these methods. Much like FP, we impute the missing node attributes matrix in an initial pre-processing step. However, our approach involves training an imputer that relies on GNNs. Our framework is also mainly different in the design of the loss function, which simultaneously takes into account graph topology and node attributes, potentially assigning different weights to each. This distinctive property makes our approach well-suited to graphs with low homophily, and for complex missing data mechanisms, in contrast with SOTA methods, which have not been evaluated in this context before, and which are outperformed by `GRIOT` according to our experiments.

3 Multi-view Optimal Transport Loss for Attribute Imputation

Hereafter, we define our notations and provide a reminder about Optimal Transport (OT) and the Wasserstein distance. We then propose a novel loss function that extends OT to enable graph attribute imputation.

3.1 Notations

We denote by $\mathcal{G} = (\mathcal{V}, \mathcal{E}, F)$, an undirected and attributed graph, where $\mathcal{V} = (v_i)_{1 \leq i \leq n}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges represented by an adjacency matrix $A = (a_{ij})_{i,j=1}^{n} \in \{0,1\}^{n \times n}$, such that $a_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$, 0 otherwise and $F = (f_i)_{i \leq n} \in \mathbb{F}^{n \times d}$ are the node attribute vectors. In this paper, we consider real or binary attributes: $\mathbb{F} = \mathbb{R}$ or $\mathbb{F} = \{0,1\}$. In the context of missing data, some values in F are not observable. These missing values are encoded as 0 in a binary mask $\Omega \in \{0,1\}^{n \times d}$. The ground truth values are denoted F^{gt} and $F = F^{gt} \odot \Omega + \mathtt{NaN} \odot (1 - \Omega)$, where $\odot$ is the Hadamard product and `NaN` denotes missing values. The missing value imputation problem is to recover an approximation $\hat{F}$ of F^{gt} from $\mathcal{G}$ and Ω. In the following, a *view* ζ_j of a graph $\mathcal{G}$ is a collection of n vectors in a z_i dimensional space. Formally, ζ_i is a $n \times z_i$ matrix, with its ℓ-th vector associated to the ℓ-th node of $\mathcal{G}$ (the nodes attribute matrix F is such a view).

There exist multiple tasks associated with graph analysis including node classification, edge prediction, and community detection to name a few. In this paper, we consider node classification as our auxiliary task; missing data imputation being the primary task. Therefore, we assume a label associated with each node: $\mathcal{C} : \mathcal{V} \to \{1, \cdots, k\}$. The goal of node classification is to learn a classification function that maps each node v_i in the graph to a class label, i.e., we aim to learn $\epsilon : \mathcal{V} \to \{1, \cdots, k\}$ s.t. $\epsilon(v_i) = C(v_i), \forall i \in \{1, \cdots, n\}$. Note that because our imputation procedure is done before solving the task, one can easily apply our method to any subsequent task.

3.2 Optimal Transport and Wasserstein Distance

We present theoretical notions from Optimal Transport (OT) [30] restricted to two weighted sets of data points in $\mathbb{R}^d$, useful for understanding the sequel.

Given two weighted sets of data points (X, w_1) and (Y, w_2) where $X = \{x_i\}_{i \le n_1} \in \mathbb{R}^{d \times n_1}$ and $Y = \{y_j\}_{j \le n_2} \in \mathbb{R}^{d \times n_2}$ are composed respectively of n_1 and n_2 vectors of size d. The weight vector w_1 of size n_1 (resp. w_2 of size n_2) defines the weights of each vector of X (resp. Y). Each weight vector is a discrete probability distribution on X (resp. Y), meaning that all the weights are positive, and they sum to one.

The goal of OT is to find a transport plan of minimal cost between (X, w_1) and (Y, w_2). This minimal cost is called the Wasserstein distance between (X, w_1) and (Y, w_2). A transport plan is represented by a matrix π such that $\pi_{i,j}$ is the weight transported from data point x_i to data point y_j. The constraints are that the total weight transported from x_i must sum to $w_1(i)$ and the total weight transported to y_j must sum to $w_2(j)$. More formally, π must belong to the set of valid transportation plans $\Pi(w_1, w_2)$:

$$\Pi(w_1, w_2) = \{\pi \in \mathbb{R}_+^{n_1 \times n_2} \mid \pi \mathbb{1}_{n_2} = w_1,\ \pi^T \mathbb{1}_{n_1} = w_2\}, \tag{1}$$

where $\mathbb{1}_n$ is the vector $(1, 1, ..., 1)^T$ of dimension n, and $\cdot^T$ the transposition operation.

The costs $m_{i,j}$ of transporting one unit of weight from x_i to y_j are given in a matrix $M^{X,Y} = (m_{i,j})_{1 \le i \le n_1, 1 \le j \le n_2}$. This matrix is usually computed as the ℓ_2 norm between the points of X and Y.

Finally, the optimal plan is found by solving the following regularized minimization problem:

$$\pi((X, w_1), (Y, w_2)) = \underset{\pi \in \Pi(w_1, w_2)}{\operatorname{argmin}} \langle M^{X,Y}, \pi \rangle_F + \varepsilon H(\pi) \tag{2}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product, ε is the regularization hyperparameter, and H is the entropy.

The Wasserstein distance [12] is the cost associated with the optimal plan:

$$\mathcal{W}((X, w_1), (Y, w_2)) = \min_{\pi \in \Pi(w_1, w_2)} \langle M^{X,Y}, \pi \rangle_F + \varepsilon H(\pi) \tag{3}$$

Being equipped with all the necessary material from OT theory, we can now move on to the description of MultiW, our newly introduced loss function.

3.3 Definition of the MultiW Loss Function

Graphs are complex objects, that can be represented through various means, each offering distinct insights. For example, an adjacency matrix captures pairwise node connections, reflecting first-order proximity. On the other hand, the Laplacian matrix [4] conveys information on node degrees and their relationships, providing a deeper understanding of the graph's structure beyond pairwise connections. Finally, the node attribute matrix is yet another way to represent the nodes of the graph.

To capture all these properties at once, we aim to design a loss function that can simultaneously leverage these diverse views of a graph to impute missing data in the feature matrix. Note that while our focus is on missing attributes, our approach could be extended to the imputation of missing information on any other view, notably the adjacency matrix for link completion.

Motivation for OT. We opt for OT theory as it provides a clever way of estimating the discrepancy between distributions. We assume that the distribution of imputed values should resemble that of the known values, especially when taking into account multiple views of the graph, i.e. the optimal transport distance between these distributions should be small.

General Definition. Let us consider a graph $G = (\mathcal{E}, \mathcal{V})$ with n nodes and $\zeta = (\zeta_i)_{i \leq q}$ be q views representing G in different spaces, such that for all i, ζ_i is a $n \times z_i$ matrix. Let $\alpha = (\alpha_i)_{1 \leq i \leq q} \in [0,1]^q$ be the views' weights such that $\sum_{i=1}^{q} \alpha_i = 1$. To proceed, we quantify the distance between random subgraphs to evaluate the gap between the distributions. Therefore, we consider two subgraphs of $\mathcal{G}$ respectively obtained by randomly selecting two subsets of nodes $\mathcal{V}^1$ and $\mathcal{V}^2$ from $\mathcal{V}$, and their respective views $(\zeta_i^1)_{1 \leq i \leq q}$ and $(\zeta_i^2)_{1 \leq i \leq q}$.

To compute an optimal transport between $\mathcal{V}^1$ and $\mathcal{V}^2$ with the q views of ζ, one could solve the OT problem q times independently, and get q different transport plans. In our case, we are interested in solving the OT problem in a way that takes into account the q views simultaneously, resulting in a single transportation plan.

With MultiW, we propose to solve a unique optimization problem considering all views at once:

$$\text{MultiW}_\alpha((\zeta_i^1)_{i \leq q}, (\zeta_i^2)_{i \leq q}) = \min_{\pi \in \Pi(w_1, w_2)} \sum_{i=1}^{q} \alpha_i \langle M^{\zeta_i^1, \zeta_i^2}, \pi \rangle_F + \varepsilon H(\pi) \tag{4}$$

However, this issue is not solvable in a reasonable time with existing tools, such as the `POT` [9] library, because of our q optimization objectives. Nonetheless, as a direct consequence of the linearity of the Frobenius inner product, we have:

$$\sum_{i=1}^{q} \alpha_i \langle M^{\zeta_i^1, \zeta_i^2}, \pi \rangle_F = \langle \sum_{i=1}^{q} \alpha_i M^{\zeta_i^1, \zeta_i^2}, \pi \rangle_F \tag{5}$$

Hence, using the linearity, the q optimization problems, defined in Eq. (4), can be simplified to one as defined in Eq. (6):

Definition 1. *Multi-view Wasserstein. Given a graph $\mathcal{G}$, and q views $\zeta = (\zeta_i)_{i \leq q}$. For any two subgraphs corresponding to the subsets of nodes $\mathcal{V}_1$ and $\mathcal{V}_2$, and their respective views $(\zeta_i^1)_{1 \leq i \leq q}$ and $(\zeta_i^2)_{1 \leq i \leq q}$; given the weights of nodes of* *the subgraphs w_1 and w_2; given $\alpha = (\alpha_i)_{1\leq i\leq q} \in [0,1]^q$ such that $\sum_{i=1}^{q} \alpha_i = 1$, let $M^{\zeta_i^1,\zeta_i^2}$ be the cost matrix between the subgraphs, according to the i-th view, then:*

$$\text{MultiW}_\alpha((\zeta_i^1)_{i\leq q}, (\zeta_i^2)_{i\leq q}) = \min_{\pi\in\Pi(w_1,w_2)} \langle \sum_{i=1}^{q} \alpha_i M^{\zeta_i^1,\zeta_i^2}, \pi\rangle_F + \varepsilon H(\pi) \tag{6}$$

Remark: The weights w_1 (resp. w_2) can be set to a uniform distribution $\forall i \in [\![1, n_1]\!]$, $(w_1)_i = \frac{1}{n_1}$ or proportional to the nodes' degree.

The optimization problem presented in Eq. (6) can be solved by the Sinkhorn-Knopp's fixed point iteration algorithm [6], and the solution is a well-defined transport plan.

3.4 Instantiation of MultiW Loss with Attributes and Structure

In this work, our focus lies on imputing missing node attributes in a graph. Therefore, we use Eq. (6) to integrate two views. These two views encapsulate both the structural characteristics and the node attributes matrix F associated with the graph. To represent the structural aspects, a straightforward representation is through the adjacency matrix A. To effectively incorporate this representation into our loss function, we propose the computation of a proximity matrix $P \in \mathbb{N}^{n\times n}$ which is defined as $P = (p_{i,j})$ where $p_{i,j}$ is the geodesic path length between the nodes v_i and v_j. Now, based on the two views P and F, we can leverage the MultiW loss to estimate the mean distance between random subgraphs of $\mathcal{G}$. We operate under the assumption that nodes with comparable roles in the graph exhibit similar attribute distributions, a well-imputed graph thus yielding a smaller MultiW distance.

Let us consider $\mathcal{G}^1$, a subgraph of G, and the corresponding vertices $\mathcal{V}^1$. Let $P_1 \in \mathbb{N}^{|\mathcal{V}^1|\times n}$ be the sub-matrix of P associated to $\mathcal{G}^1$ defined as $P_1 = \{p_{i,j} | v_i \in \mathcal{V}_1, v_j \in \mathcal{V}\}$. Unlike a sub-adjacency-matrix, P_1 is a rectangular matrix, representing the relative position of each node in $\mathcal{V}_1$ to every other node in $\mathcal{V}$ as shown in Fig. 1. Let F^1 be the sub-features-matrix associated to $\mathcal{G}^1$. Similarly, P^2, F^2 are views corresponding to a random sub-graph $\mathcal{G}^2$.

The distance between $\mathcal{G}^1$ and $\mathcal{G}^2$ can be computed as:

$$\text{MultiW}_\alpha((F_1, P_1), (F_2, P_2)) = \min_{\pi\in\Pi(w_1,w_2)} \langle M_\alpha, \pi\rangle_F + \varepsilon H(\pi) \tag{7}$$

where $M_\alpha = (1-\alpha)M^{F_1,F_2} + \alpha M^{P_1,P_2}$, and $(1-\alpha)$, α are the weights attributed to, respectively, the features and the structure.

Remark: The Fused-Gromov-Wasserstein (FGW) [29] could also be used to compute the distance between $\mathcal{G}^1$ and $\mathcal{G}^2$. However, FGW transports pairs of nodes on pairs of nodes yielding a $O(n^4)$ complexity of the Frobenius product computation of the loss versus $O(n^2)$ for MultiW. Moreover, FGW only takes into account the edges of the two subgraphs contrary to MultiW which uses the proximity matrix with all other nodes of the whole graph. Finally, the complexity of MultiW grows linearly with the number of views considered, which makes it very efficient when compared to other multi-views OT approaches such as Optimal Tensor Transport (OTT) [13].

4 Imputing Missing Attributes with MultiW Loss

Next, we describe the architecture that we designed to train an imputer that optimizes the GRIOT loss with the ability to impute all features in parallel.

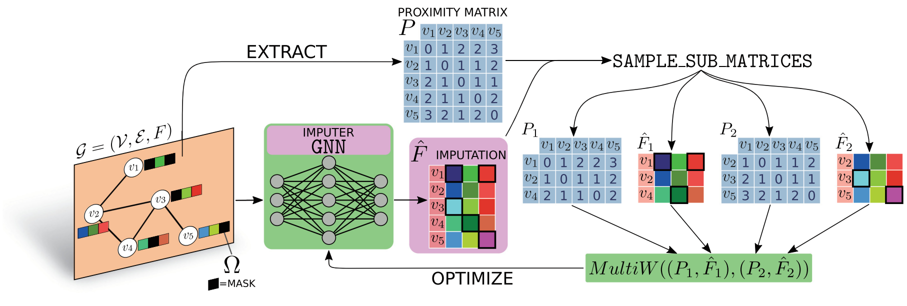


Fig. 1. Architecture of the GRIOT Framework. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes attributes F, and a mask of missing data Ω as input, GRIOT decomposes in 2 main elements: the GCN Imputer and the MultiW Loss Function used to optimize it. The framework outputs the last imputed attributes matrix $\hat{F}$ and the GNN imputer trained and ready to be reused on new nodes with missing attributes.

4.1 Architecture of GRIOT

The overall architecture of GRIOT is presented in Fig. 1, a detailed pseudocode is shown in the additional material (see Algorithm GRIOT), and the code is available online (see footnote 1).

Input of GRIOT. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a matrix of attributes F, and a mask Ω of missing features GRIOT aims at reconstructing the node attributes matrix, $\hat{F}$. Our imputer takes as input the adjacency (A) and the feature (F) matrices. However, the missing values of F must be filled in at the first iteration. To this end, we initialize $\hat{F}$ by imputing the missing values with normal random values such that $(\mu_j, \sigma_j)_{j \leq d}$ are the average and standard deviation of each feature over observed ones.

$$\forall i \leq n,\ \forall j \leq d,\ \hat{F}_{i,j} = \begin{cases} F_{i,j}, \text{ if } \omega_{i,j} == 1 \text{ (i.e. } F_{ij} \text{ is observed)} \\ \hat{F}_{i,j} \sim \mathcal{N}(\mu_j, \sigma_j) \text{ otherwise} \end{cases} . \quad (8)$$

The loss takes as input the proximity matrix P computed from the adjacency matrix such that $\forall 1 \leq i \leq n$, $\forall 1 \leq j \leq n$, $p_{i,j}$ is the length of the geodesic path between the nodes v_i and v_j, and the imputed feature matrix $\hat{F}$.

Architecture and Training of the Imputer. The imputer is a Graph Convolutional Network (GCN) [16] that takes as input $\hat{F} \in \mathbb{F}^{n\times d}$ after the initialization of the missing values and A. The output of the GCN is the imputed feature matrix, F^{imp}. Formally, we get the following equations for a GCN with $\ell = 1, \cdots, L$ layers:

$$\begin{aligned} H_1 &= \sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}\hat{F}\Theta_1) \\ H_{\ell+1} &= \sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{-1/2}H_\ell\Theta_{\ell+1}) \\ F^{imp} &= \sigma'(H_L\Theta_{L+1}^T + \Theta_{L+1}) \end{aligned}$$

where $\hat{A} = A + I$ denotes the adjacency matrix with inserted self-loops, $\hat{D}$ is the diagonal degree matrix, $(H_i)_{i\leq L}$ are hidden states, σ and σ' are activation functions, and $(\Theta_i)_{i\leq L+2}$ are the learned parameters.

Finally, the mask of observed features is applied to the imputed matrix such that only missing features are replaced:

$$\hat{F} = F \odot \Omega + F^{imp} \odot (1 - \Omega) \tag{9}$$

where $\odot$ is the element-wise product. The GCN is trained by minimizing the MultiW loss (see Definition 1). To compute the loss, we consider two subgraphs $\mathcal{G}^1$ and $\mathcal{G}^2$ from $\mathcal{G}$, defined from two subsets of nodes randomly drawn from $\mathcal{V}$, and as explained in Sect. 3.4, we use the MultiW loss function to estimate the distance between them. This operation is carried out with n_p couples of random subgraphs, and sum as $loss = \sum_{i=1}^{n_p} \text{MultiW}((F_i^1, P_i^1), (F_i^2, P_i^2))$, before being back-propagated through the imputer. The whole process is repeated `epochs` times, with a new imputation at each epoch.

Output of `GRIOT`. The output of `GRIOT` is the last version of the imputed feature matrix $\hat{F}$, and the imputer itself. The trained imputer offers the possibility to perform inductive feature imputation. Indeed, an important difference between our approach and current SOTA methods is that it goes beyond one-shot imputation on a graph and can impute missing features on new nodes.

4.2 Accelerating the Imputation

The training of machine learning models as imputers has been studied in the literature on tabular data, but these approaches are generally based on the Round-Robin principle of sequentially training one ML model imputer for each feature (not parallelizable as each ML imputer depends on the previous ones). As a result, the Round-Robin principle suffers from scalability and efficiency problems that limit its use for high-dimensional data. Indeed, all real graphs considered in this paper have in the order of magnitude of 10^3 attributes.

In contrast, our approach can impute all features simultaneously using one GNN imputer, providing a parallelizable solution, especially when utilizing GPUs.

Complexity Analysis. The theoretical time complexity analysis of GRIOT yields a complexity of $O(d \times m \times n^2 \times \texttt{n\_epochs} \times \texttt{n\_pairs})$, where d is the number of features, m is the number of views, n is the batch size, n_epochs and n_pairs are parameters of Algorithm GRIOT (c.f. additional material).

5 Experimental Analysis

Now, we propose an extensive comparison of GRIOT against different SOTA approaches while covering a rich variety of scenarios: two different masking strategies (MCAR and MNAR), different percentages of missing information, and different levels of homophily. In addition, our evaluation unfolds across two dimensions: the quality of the values imputed by the different approaches and their impact on node classification. Note that, most recent works focus on the latter aspect for homophilic graph and with MCAR masking. Overall, our goal is to answer the following research questions:

Table 1. Datasets summary. h_{obs} is the observed homophily, h_{exp} is the expected homophily from a random graph and $h_{ratio} = h_{obs}/h_{exp}$. + (resp. −) denotes a strong (resp. weak) homophily level. k is the number of classes.

	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	$\mathbb{F}$	k	sparsity	h_{obs}	h_{exp}	h_{ratio}
SBM2$^-$ (low hom.)	796	2407	$[0,1]^3$	4	2.8%	0.13	0.34	0.38
SBM1$^\sim$ (med hom.)	794	1939			3.1%	0.55	0.61	1.11
SBM0$^+$ (high hom.)	770	2085			3.1%	0.90	0.36	2.50
SBM	794	1939	$[0,1]^3$	4	2.5%	0.76	0.35	2.19
CORNELL$^-$	127	159	$\{0,1\}^{1,702}$	5	94.3%	0.13	0.32	0.42
TEXAS$^-$	129	171			95.0%	0.13	0.42	0.31
WISCONSIN$^-$	168	232			93.6%	0.17	0.30	0.55
CITESEER$^+$	2,120	3,679	$\{0,1\}^{3,702}$	6	99.1%	0.74	0.19	3.90
CORA$^+$	2,485	5,069	$\{0,1\}^{1,433}$	7	98.7%	0.80	0.18	4.50
PUBMED$^+$	19,717	44,324	$[0,2]^{499}$	3	89.9%	0.80	0.36	2.20

1. Can we outperform state-of-the-art methods by using the OT theory in GRIOT on graph features imputation, and on node classification?
2. Does the masking strategy impact the imputation?
3. Theoretically, GRIOT can impute new nodes without recycling. How does it behave in such a scenario?

In this section, we first describe the experimental protocol setup to address these questions and then present the results obtained for all aforementioned scenarios.

5.1 Experimental Protocol

Baselines. We include both *naive* and SOTA baselines. For the naive baseline, we compare with a K-nearest-neighbors-based principle and simply impute the *average* of the features of nodes directly connected to the node presenting missing values. The strong baselines comprise three models. The first one, `PaGNN` [11], considers the mask Ω when classifying nodes. The second baseline, OT-TAB [23] is also an OT-based imputation approach but for tabular data only leaving out the graph's structure. It corresponds to two algorithms, a one-shot imputation referred to as OT-TAB and a Round-Robin-based version referred to as OT-TAB-rr that, similar to us, allows the training of an imputer. Note that we could not run the OT-TAB-rr algorithm on all datasets for time complexity reasons. The third and strongest baseline is Feature Propagation (FP) [24] which is currently the best approach among SOTA for this task.

Datasets. We evaluate our algorithm and baseline methods on diverse graphs, both synthetic and real, covering various homophily levels as summarized in Table 1. Homophily is defined as the tendency for nodes with similar attributes to be more likely connected [14]. Table 1 presents the expected homophily h_{exp} derived from a randomly drawn graph with equivalent node count and edge probability, alongside the observed homophily h_{obs} in the given graph. The ratio $h_{ratio} = \frac{h_{obs}}{h_{exp}}$ provides a key insight, where a value exceeding 1 signifies high homophily, while a value below 1 indicates weak homophily.

We use the Stochastic Block Model (SBM) to generate synthetic graphs with different homophily levels, spanning from low to medium and high homophily. Each dataset has four imbalanced clusters, and specific parameters for generation can be found in the online code repository (see footnote 1). In addition to the synthetic graphs, our evaluation encompassed three datasets from the WebKnowledgeBase (WebKB) [5] (weakly homophilic, marked with a superscript "−" sign) and three citation graphs from the citation network (Planetoid) datasets [31] (highly homophilic, marked with a superscript "+" sign).

Missing Data Masking. We use two strategies to build the mask Ω: Missing Completely At Random (MCAR) and Missing Not At Random (MNAR). For MCAR, we draw $\Omega = (\omega_{i,j})_{i \leq n,\ j \leq d}$ i.i.d. such that $\forall (i,j),\ \omega_{i,j} \sim \mathcal{B}(p)$ follows a Bernoulli distribution, where p is the probability of values being set to 0 in the feature matrix. MCAR exhibits no correlation with the data or graph structure. In MNAR, we consider the self-masked context, where $P(\omega_{i,j} = 0)$ depends on the values of the unobserved data itself (extreme values are more likely to be missing) and the characteristics of the graph's structure (nodes with lower degree are more prone to have missing data). Finally, we also vary the level of missing data during experiments, maintaining consistency at 20%, 50%, and 80%.

Evaluation Metrics. Evaluation metrics fall into two categories. The first category assesses the quality of the imputed values by comparing the imputed matrix $\hat{F}$ to ground truth values F^{gt}, using Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), where lower is better. The second evaluates imputation's impact on node classification done with a Graph Neural Network after imputation and is based on accuracy and ROC-AUC scores, where higher is better.

We use the Mann-Whitney U test with a 5% p-value over 5 runs to determine significant results.

Remark: Results presented in the following are a subset of our experiments. Additional results are available in the additional material.

Imputer Architecture. To build the reconstructed feature matrix $\hat{F}$, we employ a Graph Neural Network (GNN). The imputer accepts as input the edge index $\mathcal{E}$ and the imputed features $\hat{F}$, which have dimensions $n \times d$. Consequently, both the input and output of the imputer are of size d. We optimized the imputer's architecture through cross-validation, resulting in a model with 2 layers of Graph Convolutional Network (GCN) and 1 linear layer, with respective dimensions $(d, \lceil\sqrt{d}\rceil)$, $(\lceil\sqrt{d}\rceil, \lceil\sqrt{d}\rceil)$, and $(\lceil\sqrt{d}\rceil, d)$. This architecture enhances graph abstraction and improves overall reconstruction quality. Furthermore, we introduced a 50% dropout rate to increase the model's robustness against overfitting.

The imputer's parameters are optimized using the Adam optimizer [15], with a learning rate set to 0.01 and a weight decay of 10^{-5}.

Classifier Architecture. To evaluate the performance of all imputers on the node classification task, we define a GNN classifier. This classifier takes $(\mathcal{E}, \hat{F})$

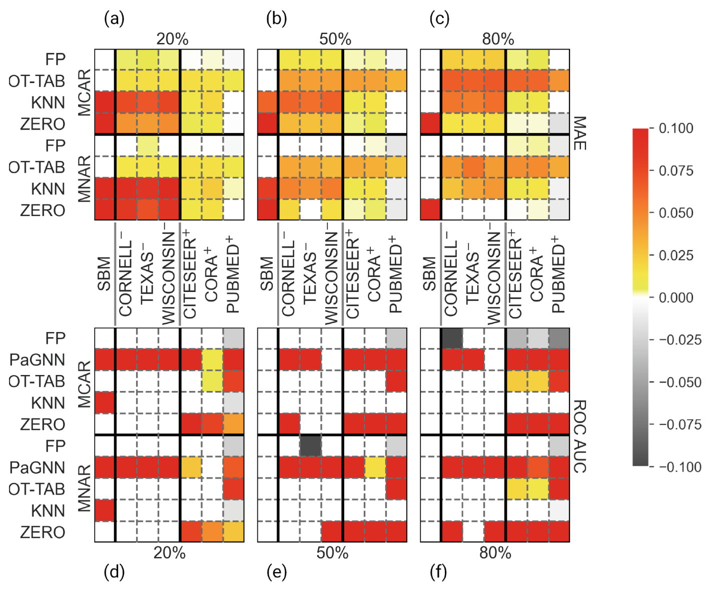


Fig. 2. Comparison of GRIOT v.s. baselines. (a,b,c) **imputation** MAE↓, (d,e,f) **classification** ROC AUC↑, on multiple datasets with (1) varying degrees of missing data: (a,d) 20%, (b,e) 50%, and (c,f) 80%, and (2) varying missingness mechanisms: MCAR (upper parts of the matrices) and MNAR (lower parts). In the visualization, white squares denote non-significant differences, grayscale squares indicate instances where GRIOT is outperformed, and colored squares represent significant improvements of GRIOT over baseline methods, with colors ranging from yellow (small differences) to red (larger differences).

as input, where $\hat{F}$ has dimensions $n \times d$, and each node is assigned to one of k distinct classes. The classifier consists of 2 `Cheb` layers [7] and 1 linear layer, with dimensions $(d, \lceil\sqrt{d}\rceil)$, $(\lceil\sqrt{d}\rceil, \lceil\sqrt{d}\rceil)$, and $(\lceil\sqrt{d}\rceil, k)$, respectively.

We determined the type of layers through cross-validation, although we spent a short time optimizing the layers' hyperparameters, as the primary goal was to assess the impact of imputation on classification performance.

5.2 Imputation Quality v.s. Node Classification Accuracy

We start by assessing the performances of all methods, with a distinction between the accuracy of the imputed values and the impact on node classification accuracy.

Imputed Values. Looking at the quality of the imputed values, Fig. 2 (a, b, c) shows the MAE obtained by all baselines, with a distinction between homophilic and heterophilic graphs with different masking strategies, MCAR in the upper parts and MNAR in the lower parts, and percentages of missing attributes varying from 20% (a), 50% (b) to 80% (c). This figure presents pairwise comparisons between `GRIOT` and each of the baselines; colored squares (from yellow to red) correspond to the case when `GRIOT` significantly outperforms other baselines. Overall, `GRIOT` performs much better for this particular task than its competitors, and this difference is even more significant on heterophilic graphs. However, we note that despite the high sparsity of the attributes, the strategy that consists of imputing a zero in place of all missing attributes, indicated by ZERO, performs poorly. Now, taking a closer look at the masking strategy, we can see that the results of KNN and OT-TAB are consistent with the ones obtained when using MCAR or MNAR. Finally, we observe that the differences in FP performances become more pronounced as the percentage of missing data increases.

Node Classification. Figure 2 (d, e, f) presents the AUC score differences between all baselines versus `GRIOT`, it reads the same way as Fig. 2 (a, b, c). We observe that, although `GRIOT` showed better imputation performance compared to baselines, this superiority does not systematically translate into a noticeable improvement in the node classification task. Indeed, when dealing with 20% and 50% of missing attributes, we see that `GRIOT` obtains comparable AUC with almost all baselines, including FP on all datasets. We also note that `GRIOT` is always outperforming PaGNN. Now moving on to the 80% of missing attributes, FP is obtaining better results for all the homophilic graphs. This comes as no surprise, as FP was shown to perform extremely well in these extreme types of scenarios [24]. Finally, it is worth remarking that, with 80% of missing attributes, `GRIOT` significantly outperforms OT-TAB on the homophilic graphs.

Are Both Objectives Always Aligned? To get more insights into the correlation between imputation quality and node classification, we take a closer look at the hyper-parameter α of `GRIOT`, which determines the importance of each view. A value of α close to 0 emphasizes the attribute information, while a value close to 1 prioritizes the topology. Table 2, shows the α corresponding to the

Table 2. Average of the best α over the type of graph and the tasks: imputation and node classification.

Masking	SBM		WebKB$^-$		Planetoid$^+$	
	MCAR	MNAR	MCAR	MNAR	MCAR	MNAR
Imput.	0.50	0.42	0.31	0.50	0.14	0.33
Classif.	0.42	0.42	0.25	0.31	0.33	0.58

best results, selected on a validation set. We distinguish between WebKB and Planetoid graphs.

For WebKB graphs, we notice that α tends to be higher during the imputation task compared to classification. This suggests an increased emphasis on the graph structure during the imputation optimization and a decreased emphasis during classification. Conversely, the opposite trend is observed for Planetoid graphs. Additionally, on the SBM graph, which has a balanced homophily, α remains relatively stable, hovering around 0.50. Smaller values of α during the imputation of homophilic graphs imply that the structure is considered less crucial than with the heterophilic graphs. However, when it comes to classifying heterophilic graphs, it appears beneficial for the network to downplay the importance of structure, as it can potentially lead to misleading outcomes.

Moreover, we note that α is generally higher for real graphs (and similar for SBM) in the MNAR scenario, indicating a greater emphasis on topology than in the MCAR scenario. We recall that the MNAR mechanisms tend to mask extreme values and attributes of weakly connected nodes. Thus, information loss on attributes caused by the MNAR mechanism is likely being counterbalanced during the network training by leveraging more of the graph's structure.

For homophilic graphs, we observed only a small decrease of AUC when focusing on MAE for the cross-validation (1.22%) against a decrease of 16% on heterophilic ones. We believe that this is a particularly interesting finding as the performances of recently proposed models are mostly evaluated on homophilic graphs using node classification accuracy.

5.3 Imputing Missing Values for Unseen Nodes

The key feature of our approach is the ability to impute missing values on new nodes dynamically added to the graph without having to retrain our imputer from scratch. To evaluate this feature, we propose to build a test set composed of nodes that were fully removed from the graph during the training of `GRIOT`. We report results in Fig. 3 and focus on the most heterophilic graphs, as we have shown that these are the most complex ones. For the MAE, we observe that the results of our imputer on unseen nodes are consistent with the ones obtained when all nodes are known from training time, with a small increase in MAE only for extreme cases (80% of missing data). However, this increase in MAE does not coincide with a decrease in AUC score. Indeed, for AUC the results are comparable most of the time, and the imputer is even getting slightly better results on unseen nodes in the majority of the scenarios.

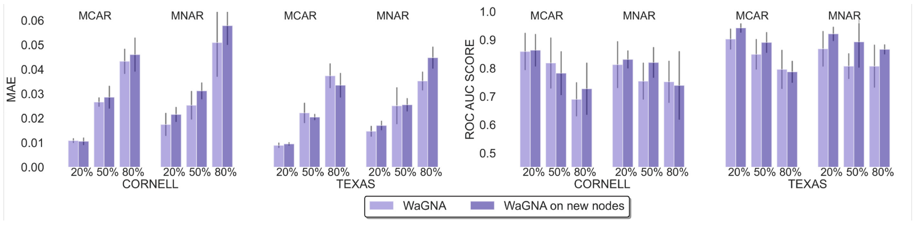


Fig. 3. Performance in terms of (left) MAE and (right) AUC score of GRIOT and GRIOT tested on nodes not present at training time for all settings.

Table 3. Imputation time complexity (in seconds) for all graphs. We report results for the case of 50% missing data.

Model	CITESEER	CORA	PUBMED	CORNELL	TEXAS	WISCONSIN
GRIOT-CPU	846 ± 162	338 ± 6	1096 ± 216	186 ± 10	195 ± 64	244 ± 67
GRIOT-GPU	571 ± 48	236 ± 6	819 ± 189	194 ± 18	148 ± 44	168 ± 67
FP	14 ± 1	4 ± 0	18 ± 1	$.3 \pm .0$	$.3 \pm .1$	$.3 \pm .0$
OT-TAB	489 ± 17	160 ± 7	597 ± 32	79.2 ± 1	78.0 ± 0	43 ± 0
OT-TAB-rr (estimation)	$5.5e6$	$9.0e5$	$8.3e5$	$1.7e5$	$1.6e5$	$1.9e5$

5.4 Time Complexity

Table 3 displays the running times. GRIOT is comparable in processing time to OT-TAB [23], which is less complex as it does not involve imputer training and fills missing values in one step. Additionally, we were not able to compare with OT-TAB-rr, based on Round-Robin, as running time was always exceeding multiple days per run. Unsurprisingly, GRIOT is considerably slower than FP, which does not require any learning process. Finally, FP is faster than GRIOT as it does not require training an imputer (one-shot imputation). Bearing this in mind and taking CITESEER as an example, it means that in a dynamic environment GRIOT becomes more efficient than FP if more than 40 new nodes requiring attribute imputation appear in the graph after training, and more efficient than OT-TAB after just 1 new node.

Summing Up Our Results. When summarizing the comparison between GRIOT and state-of-the-art methods (Table 4), we observe that GRIOT significantly enhances data reconstruction in 68% of the scenarios and improves the

Table 4. Each cell shows the number of times GRIOT (a) underperforms, (b) shows no significant changes, or (c) outperforms compared to strong baselines when averaged over all scenarios.

	(a) underperform	(b) similar	(c) outperform
MAE	6%	26%	68%
ROC	8%	56%	37%

classification task in 37% of the cases. However, it remains on par with the best-performing method, FP if we consider its impact on node classification. Our method is particularly relevant in dynamic environments such as social graphs, where the number of new users is constantly increasing. The use of a trained imputer improves adaptability and guarantees our efficiency in dynamic environments. Furthermore, we hope that our findings provide interesting insights to the community regarding the counterintuitive observations made between the quality of imputed values and subsequent node classification, offering the potential for valuable advances in the understanding and optimization of imputation strategies for various applications.

6 Conclusion and Perspectives

We introduce GRIOT, a framework employing OT and the MultiW metric for missing attribute imputation in attributed graphs. Key features include support for multiple graph representations, efficient parallelization of the imputation, and a trained imputer for new nodes. Finally, GRIOT is also independent of the task at hand and can therefore be used for tasks other than node classification. Our extensive experiments, spanning synthetic and real-world data with diverse missingness mechanisms, demonstrate the competitiveness of GRIOT in addressing the challenge of missing attributes, particularly in weakly homophilic graphs. Another way to better adapt to heterophilic graphs would be to use an imputer architecture that relies less on the homophilic assumption than GCN. We have begun to investigate the potential use of convolutional graph transformers [8] in place of GCN, but have so far been unable to achieve better results. The future perspective of this work will also consist in deepening our study of the impact of different structural views for tasks other than node classification.

Acknowledgment. This work has been funded by a public grant from the French National Research Agency (ANR) under the "France 2030" investment plan, which has the reference EUR MANUTECH SLEIGHT - ANR-17-EURE-0026.

Ethical Statement. Datasets used in this article are publicly accessible, and the code is public, making the results reproducible (Code and additional material available at: github.com/RichardSrn/GRIOT). This paper presents work in machine learning, a field that can have potential societal consequences, but none of which we feel must be specifically highlighted here.

References

1. Berg, R.V.D., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263 (2017)
2. Brogat-Motte, L., Flamary, R., Brouard, C., Rousu, J., d'Alché-Buc, F.: Learning to predict graphs with Fused Gromov-Wasserstein barycenters. In: ICML, pp. 2321–2335 (2022)
3. Chen, X., Chen, S., Yao, J., Zheng, H., Zhang, Y., Tsang, I.W.: Learning on attribute-missing graphs. IEEE PAMI **44**(2), 740–757 (2022)
4. Chung, F.R.: Spectral Graph Theory, vol. 92. American Mathematical Society (1997)
5. Craven, M., et al.: Learning to extract symbolic knowledge from the world wide web. AAAI/IAAI **3**(3.6), 2 (1998)
6. Cuturi, M.: Sinkhorn distances: lightspeed computation of optimal transport. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
7. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Advances in Neural Information Processing Systems, vol. 29 (2016)
8. Dwivedi, V.P., Bresson, X.: A generalization of transformer networks to graphs. CoRR abs/2012.09699 (2020). https://arxiv.org/abs/2012.09699
9. Flamary, R., et al.: POT: python optimal transport. J. Mach. Learn. Res. **22**(78), 1–8 (2021)
10. Huisman, M.: Imputation of missing network data: some simple procedures. In: Encyclopedia of Social Network Analysis and Mining, pp. 707–715 (2014)
11. Jiang, B., Zhang, Z.: Incomplete graph representation and learning via partial graph neural networks. arXiv preprint arXiv:2003.10130 (2020)
12. Kantorovich, L.V.: Mathematical methods of organizing and planning production. Manage. Sci. **6**(4), 366–422 (1960)
13. Kerdoncuff, T., Emonet, R., Perrot, M., Sebban, M.: Optimal tensor transport. In: AAAI Conference on Artificial Intelligence, vol. 36, pp. 7124–7132 (2022)
14. Khanam, K.Z., Srivastava, G., Mago, V.: The homophily principle in social network analysis: a survey. Multimedia Tools Appl. **82**(6), 8811–8854 (2023)
15. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
17. Kossinets, G.: Effects of missing data in social networks. Soc. Netw. 247–268 (2006)
18. Laclau, C., Redko, I., Choudhary, M., Largeron, C.: All of the fairness for edge prediction with optimal transport. In: AISTATS, vol. 130, pp. 1774–1782 (2021)
19. Little, R.J., Rubin, D.B.: Statistical Analysis with Missing Data, vol. 793. Wiley, Hoboken (2019)
20. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: homophily in social networks. Ann. Rev. Sociol. **27**(1), 415–444 (2001)
21. Mensch, A., Blondel, M., Peyré, G.: Geometric losses for distributional learning. In: International Conference on Machine Learning, pp. 4516–4525 (2019)

22. Monti, F., Bronstein, M., Bresson, X.: Geometric matrix completion with recurrent multi-graph neural networks. In: NIPS, vol. 30 (2017)
23. Muzellec, B., Josse, J., Boyer, C., Cuturi, M.: Missing data imputation using optimal transport. In: ICML, pp. 7130–7140 (2020)
24. Rossi, E., Kenlay, H., Gorinova, M.I., Chamberlain, B.P., Dong, X., Bronstein, M.M.: On the unreasonable effectiveness of feature propagation in learning on graphs with missing node features. In: Learning on Graphs Conference (2022)
25. Schafer, J.L., Graham, J.W.: Missing data: our view of the state of the art. Psychol. Methods **7**(2), 147 (2002)
26. Stekhoven, D.J., Bühlmann, P.: MissForest-non-parametric missing value imputation for mixed-type data. Bioinformatics **28**(1), 112–118 (2011)
27. Taguchi, H., Liu, X., Murata, T.: Graph convolutional networks for graphs containing missing features. Futur. Gener. Comput. Syst. **117**, 155–168 (2021)
28. Van Buuren, S., Groothuis-Oudshoorn, K.: Mice: multivariate imputation by chained equations in R. J. Stat. Softw. **45**, 1–67 (2011)
29. Vayer, T., Chapel, L., Flamary, R., Tavenard, R., Courty, N.: Fused Gromov-Wasserstein distance for structured objects: theoretical foundations and mathematical properties. Algorithms **13**(9), 212 (2020)
30. Villani, C., et al.: Optimal Transport: Old and New, vol. 338. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-71050-9
31. Yang, Z., Cohen, W., Salakhudinov, R.: Revisiting semi-supervised learning with graph embeddings. In: ICML, pp. 40–48 (2016)
32. Yoon, J., Jordon, J., van der Schaar, M.: GAIN: missing data imputation using generative adversarial nets. In: Proceedings of ICML, vol. 80, pp. 5689–5698 (2018)
33. Zhang, J., Xiao, X., Huang, L.K., Rong, Y., Bian, Y.: Fine-tuning graph neural networks via graph topology induced optimal transport. In: IJCAI, pp. 3730–3736 (2022)
34. Zheng, X., Liu, Y., Pan, S., Zhang, M., Jin, D., Yu, P.S.: Graph neural networks for graphs with heterophily: a survey. arXiv preprint arXiv:2202.07082 (2022)

Introducing Total Harmonic Resistance for Graph Robustness Under Edge Deletions

Lukas Berner(✉) and Henning Meyerhenke

Department of Computer Science, Humboldt-Universität zu Berlin,
Unter den Linden 6, 10099 Berlin, Germany
{lukas.berner,meyerhenke}@hu-berlin.de

Abstract. Assessing and improving the robustness of a graph G are critical steps in network design and analysis. To this end, we consider the optimisation problem of removing k edges from G such that the resulting graph has minimal robustness, simulating attacks or failures.

In this paper, we propose total harmonic resistance as a new robustness measure for this purpose – and compare it to the recently proposed forest index [Zhu et al., IEEE Trans. Inf. Forensics and Security, 2023]. Both measures are related to the established total effective resistance measure, but their advantage is that they can handle disconnected graphs. This is also important for originally connected graphs due to the removal of the k edges. To compare our measure with the forest index, we first investigate exact solutions for small examples. The best k edges to select when optimizing for the forest index lie at the periphery. Our proposed measure, in turn, prioritizes more central edges, which should be beneficial for most applications. Furthermore, we adapt a generic greedy algorithm to our optimization problem with the total harmonic resistance. With this algorithm, we perform a case study on the Berlin road network and also apply the algorithm to established benchmark graphs. The results are similar as for the small example graphs above and indicate the higher suitability of the new measure.

Keywords: Graph robustness optimization · infrastructure protection · total harmonic resistance · forest index · effective resistance

1 Introduction

The analysis of network[1] topologies, a major subarea of data science on network data, is key to understanding the functionality, dynamics, and evolution of networks [5,26]. An important property of a network in this context is its *robustness*, i. e., its ability to withstand failures of its components (or the extent of this ability) [5]. As an example, a typical question is whether a network remains (mostly) connected if a certain fraction of its vertices and/or edges are

[1] We use the terms *network* and *graph* interchangeably.

deleted [26, Ch. 15]. Despite the widespread use of vertex deletions, edge deletions can be more appropriate depending on the modeled phenomenon. Examples include, among others, road blocks in street or public transportation networks; pollution in water distribution networks; disruption of gas pipelines, energy grids, or computer/telecommunication networks. Such deletions may occur as a result of failure or of an attack; robustness thus is a critical design issue that arises in many application areas [13], e. g., various public infrastructures [9,19,33,37].

Due to economic reasons, it is unrealistic that all network components can be protected with the same effort. Thus, with the protection of critical infrastructure as application in mind, we consider the following optimization problem: given a graph $G = (V, E)$ and a budget of k graph edges to be removed, find the subset $S \subset E$ such that the robustness of $G' = (V, E \setminus S)$ is minimized. This problem, which we call k-GRoDel (short for *graph robustness problem with k deletions*), models a concurrent attack (or failure). The solution indicates which set of edges should be particularly safeguarded, e. g., segments in a water distribution network. Clearly, for a particular application, one must also instantiate this generic problem with a sensible notion of robustness.

Not surprisingly, numerous robustness measures have been proposed in the literature [5,33]. For the related problem of optimizing the robustness by *adding* k edges (called k-GRIP in Ref. [30], short for *graph robustness improvement problem*), *total effective resistance* was established as a meaningful robustness measure in various scenarios [12,29,31,36]. Effective resistance is a pairwise metric on the vertices of G; intuitively, it becomes small if there are many short paths between two vertices. Two disconnected vertices have infinite effective resistance, though. When total graph resistance were used in k-GRoDel, a trivial solution to maximize it would thus be to disconnect G. Yet, from an application's point of view, disconnecting a small part from the vast majority of the graph may be less problematic than a bottleneck (or a disconnection) between two large parts. Liu et al. [22] handles this issue by demanding that G is still connected after edge removal. Given an infrastructure scenario, this is a rather unnatural assumption. Zhu et al. [38] address the issue by proposing the forest index, $R_f(G)$, as robustness measure. Instead of effective resistance, $R_f(G)$ sums up the closely related forest distance [11] for all vertex pairs. Forest distance is derived from the number of certain rooted forests in G. It yields finite distance values also for disconnected vertex pairs.

Contribution. We show in this paper that k-GRoDel using the forest index favors peripheral edges in many networks. We deem this behavior unintuitive and, most importantly, undesirable for most applications. That is why we propose *total harmonic resistance* $R_h(\cdot)$ instead. This measure adds up the reciprocal of effective resistance for all vertex pairs, leading to a zero contribution of disconnected pairs (details in Sect. 2). The use of $R_h(\cdot)$ may seem like a straightforward extension to handle deletions given that the use of reciprocals is known for the popular (harmonic) closeness centrality (based on the ordinary graph distance) to handle disconnectedness [26]. Nonetheless, we are to our knowledge the first to investigate this notion of robustness (also see Sect. 3).

To substantiate the higher suitability of $R_h(\cdot)$ compared to the forest index in k-GRoDel, we first examine optimal solutions for small graphs with the two measures (Sect. 4). They clearly show that $R_h(\cdot)$ favors more central edges than the forest index and also finds balanced cuts in examples with suitable k.

Since exact solutions for either k-GRoDel measure are expensive to compute, we adapt in Sect. 5 the general greedy algorithm used in several previous papers for related problems. For $R_h(\cdot)$, our greedy algorithm differentiates between bridge edges and other edges. When a bridge edge is removed, a simple update operation for the Laplacian pseudoinverse does not work. For these cases, we thus provide specialized update functions. For the forest index, we derive a connection to total effective resistance, which allows the re-use of optimized Laplacian pseudoinverse solvers instead of more general matrix inversion solvers.

Our experiments (Sect. 6) include a case study on the road network of (a part of) Berlin, Germany, as well as numerous public benchmark graphs used before in related work. They show: (i) visually, the case study results indicate that the greedy solution for $R_h(\cdot)$ prefers more central edges than the one with the forest index. Maybe not surprisingly, one can find an even better solution regarding $R_h(\cdot)$ by choosing natural cut edges (river bridges in the road network) manually, which underlines the expressiveness of the new measure; (ii) for the benchmark graphs, a ranking based on closeness centrality confirms that greedy solutions of $R_h(\cdot)$ lead to more central edges than the forest index in most cases, too.

2 Problem Statement and a New Robustness Measure

2.1 Problem Statement and Notation

The input to k-GRoDel is an integer $k \in \mathbb{N}$ and a simple undirected graph $G = (V, E)$ with $|V| = n$, $|E| = m$. Given $S \subset E$, let $G' = (V, E \setminus S)$ be the graph with the edges from S removed. k-GRoDel aims at finding S with $|S| = k$ such that the robustness of G' is minimized (i. e., $R_h(G')$ is minimized or $R_f(G')$ is maximized, respectively).

We use well-known matrix representations of graphs. $\mathbf{L}_G = \mathbf{D} - \mathbf{A} \in \mathbb{R}^{n \times n}$ is the Laplacian matrix of G, where $\mathbf{D}$ is the vertex degree matrix and $\mathbf{A}$ is the adjacency matrix. $\mathbf{L}_G$ is symmetric (since G is undirected) and has zero row and column sums ($\mathbf{L1} = 0 = \mathbf{1}^T\mathbf{L}$). Since $\mathbf{L}_G$ is not invertible, the Moore-Penrose pseudoinverse $\mathbf{L}^\dagger$ is used instead (cf. [8]). When G has multiple components, $\mathbf{L}_G$ is a (permuted) block diagonal matrix where each block corresponds to one component of G.

2.2 Robustness Measures

Effective Resistance. Viewing the graph as an electrical circuit where each edge is a resistor, the effective resistance is the potential difference between two nodes u

and v when injecting [extracting] a unit current at u [v] [17]. It can be computed via $\mathbf{L}^\dagger$: $\mathbf{r}_G(u,v) = \mathbf{L}_G^\dagger[u,u] - 2\mathbf{L}_G^\dagger[u,v] + \mathbf{L}_G^\dagger[v,v]$ for nodes in the same component of G. For disconnected pairs, the resistance is infinite. As a robustness measure, one can take the sum over all pairwise effective resistances to compute the total effective resistance $R_r(G) = \sum_{u<v} \mathbf{r}_G(u,v)$, which has previously been used as optimization target for k-GRIP [29,31,36] in graphs with only one component. Combining both equations above results in a simple trace-based formula [8]:

$$R_r(G) = n \cdot \mathrm{tr}(\mathbf{L}_G^\dagger). \tag{1}$$

Forest Index (FI). To address the issue of disconnected graphs, other robustness measures are required. The *forest index*, based on the *forest distance* [11] $d_G^f(\cdot,\cdot)$, was proposed by Zhu et al. [38]. Similar to effective resistance, the forest distance is based on the *forest matrix* $\Omega = (L+I)^{-1}$, with $d_G^f(u,v) = \Omega[u,u] - 2\Omega[u,v] + \Omega[v,v]$. The forest distance is closely related to effective resistance (for details see Sect. 5), but yields finite values also for disconnected vertex pairs. Similar to total effective resistance, the forest index is the sum of the forest distance (instead of the effective resistance) of all ordered vertex pairs (u,v):

$$R_f(G) := \sum_{u<v} d_G^f(u,v). \tag{2}$$

With an argument analogous to the one for total effective resistance, the forest index can be expressed using the trace as well:

$$R_f(G) = n \cdot \mathrm{tr}(\Omega) - n. \tag{3}$$

Total Harmonic Resistance (THR). We now propose a new measure to handle disconnected graphs, *total harmonic resistance*. This measure is again based on the effective resistance; this time one sums up the reciprocal of all pairwise effective resistances – therefore *harmonic*:

$$R_h(G) := \sum_{u<v} \frac{1}{\mathbf{r}_G(u,v)}. \tag{4}$$

For vertex pairs where the effective resistance is infinite (i. e., vertices lie in different components), we define the reciprocal to be zero. The reciprocity in this sum makes computations more difficult compared to the other two metrics.

3 Related Work

Due to its high relevance in numerous application areas as well as a rich assortment of research questions, robustness in networks has been an active research area for several decades [13]. We thus point the interested reader to recent surveys for a broader overview [13,27]. Concerning robustness measures, the survey by Freitas et al. [13] categorizes them into three classes: (i) based on structural (combinatorial) properties, (ii) spectral properties of the adjacency matrix, and (iii) spectral properties of the Laplacian matrix. Total effective resistance belongs to the third class as it can be computed by the sum of the Laplacian (inverse) eigenvalues. Chan and Akoglu [10] propose a budget-constrained edge rewiring mechanism to address six different spectral measures – a related optimization problem, yet different from k-GRoDel. Note that Oehlers and Fabian [27] focus on communication networks and use a more fine-grained categorization than Freitas et al. within their context.

Failures of components can result from various reasons, e. g., from natural disasters, attacks, or wear. The targeted attack models surveyed by Freitas et al. [13] refer to vertex removals and are based on vertex degrees and centrality scores. In general, vertex removals are the predominant failure model in the literature; Newman [26, Ch. 15] discusses percolation (removal of a fraction of the nodes), for example. An important question in this context is after which fraction the graph becomes disconnected or, more generally, when the giant component dissolves. One can address this question analytically in generative models (e. g., [26]) and/or empirically with real-world data (e. g., [6]). As a prime example, an influential paper by Albert et al. [1] and follow-up work led to the popular belief that scale-free networks are "robust-yet-fragile", i. e., robust against uniform vertex deletion and fragile against targeted attacks that remove high-degree vertices. Recent work by Hasheminezhad and Brandes [15] puts this view into a more nuanced perspective: robustness depends primarily on the graph's minimum degree, not a power-law degree distribution.

As mentioned in Sect. 1, edge deletions are natural to model failures in numerous applications. Liu et al. [22], who study the problem of minimizing one node's information centrality when removing k edges, argue that edge deletions are less intrusive than vertex deletions and that they provide a more fine-granular control of disruptions. To measure how easy two vertices can reach each other via alternative paths, numerous works use effective resistance [12,22,29,30,36], whereas Zhu et al. [38] use forest distance [11] as summands of their forest index, a metric related to effective resistance. Both robustness measures express with lower values that more alternative pathways exist. A small total forest distance (as well as a small total effective resistance) thus means that many vertex pairs can reach each other via many alternative short paths. Obviously, this is a desirable property for a robustness measure in a number of applications, e. g., when it comes to routing information or goods [27]. Forest distance has recently been used for forest closeness centrality [14,16]. There and when used as part of the forest index, it has the advantage (compared to the ordinary graph distance or effective resistance) to be able to handle disconnected graphs without changes.

An exact solution of the k-GRoDel problem with total harmonic resistance or a related measure is likely infeasible for instances of non-trivial size: (i) similar optimization problems have been shown to be $\mathcal{NP}$-hard [14,18], including the single-vertex variant of Liu et al. [22] with information centrality, and (ii) mathematical programming, even when applied to related problems with simpler objective functions, can usually solve instances with only hundreds or at most a few thousand vertices in reasonable time [2]. Empirically, however, the related problem of adding k edges to minimize total effective resistance can be solved adequately (yet in general not optimally) with a standard greedy algorithm [35]. we developed in our previous work [30,31] heuristics to accelerate the greedy algorithm for the k-GRIP problem (with usually tolerable losses in solution quality). Even more closely related, Liu et al. [22] and Zhu et al. [38] use greedy strategies to identify k edges to delete from G while optimizing for a robustness measure (information centrality vs forest index). We thus expect an adapted greedy algorithm to work similarly well for our variant of k-GRoDEL.

4 Comparison of Exact Solutions

To investigate the difference between forest index and total harmonic resistance as robustness measures for k-GRoDEL, we analyze exact solutions for a collection of small examples. These examples consist of different graph classes: grid graphs and variants thereof, random graphs [26, Part III] generated using the Barabási-Albert model (parameters: $k = 3$, $n_{max} = 18$), and random graphs generated with the Watts-Strogatz model (parameters: $n = 16$, deg $= 3$, $p = 0.7$). For each example, we compute the exact solutions of the optimization problem for both robustness measures. Results for the grid-like grahps are visualized in Fig. 1. Due to symmetry in the grid-like graphs, there are often multiple optimal solutions; for simplicity, we only show one of these solutions.

Visually, the figures suggest that the forest index (FI) finds edges in the periphery, while THR finds central edges. THR also seems to be more robust regarding changes to low-degree nodes in the periphery of a graph: THR finds the same solution for the 4x7 grid and for the hotdog grid, while the FI solution changes.

To further support our claim about periphery vs center, we compute a centrality score for an edge set as follows: Given the closeness centrality $c(\cdot)$ of all nodes in $G = (V, E)$ and a set of edges $S \subset E$, we rank the nodes by their closeness centrality and convert their rank into a relative (quantile) score $s \in [0, 1]$, where the most central node has score 1 and the least central node has score 0. Then, for each edge $e = (u, v) \in S$, we take the mean score of both incident nodes and call this the score of that edge $s(e)$. Edges which are central in the graph have a larger score than less central edges. Finally, we define the score of S as the mean of all edge scores in the set. Scores for all solution sets are listed in Table 1. The centrality scores of the solutions further support our claim: for all graphs of all three types, the score of the THR solution is higher (i. e., more central) than in the FI solution. This even holds when comparing the best FI solution to the worst THR solution in this metric.

Discussion. We would like to note that the observed behavior of the forest index is not according to our original intuition (which was similar to the one given by Zhu et al. [38]) *before* working on this paper. Broadly speaking, we generally expected the forest index to be maximized when the number of disconnected

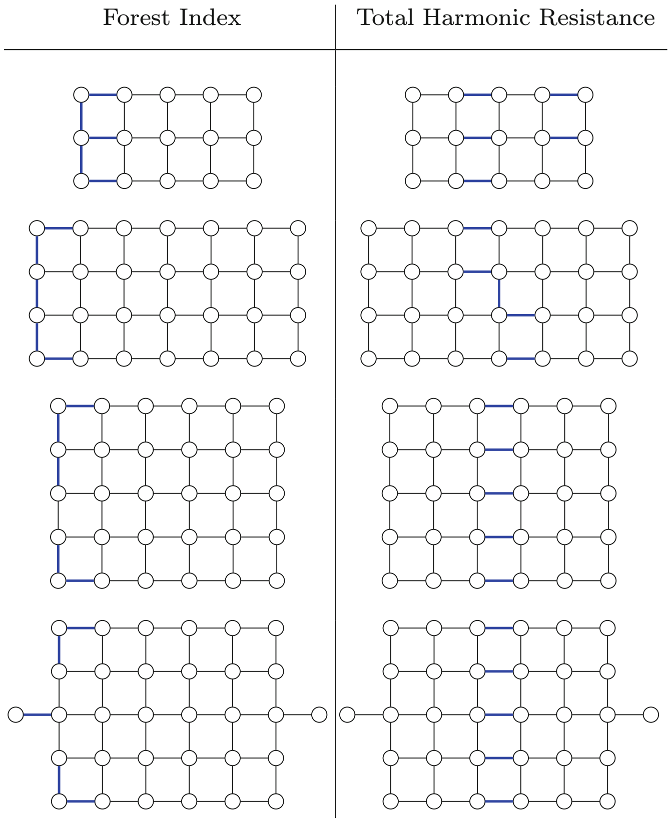


Fig. 1. Optimal solutions for $k = 5$ on grid-like graphs using FI (left column) and THR (right column) as resistance measures. Edges highlighted in blue belong to the solution set. (Color figure online)

vertex pairs is maximized, because this leads to many high terms in the sum. The optimal solution (for appropriate k) on a grid graph would then be a balanced cut in the middle. Instead, the optimal solution when using the forest index is a set of edges at the boundary of the grid, disconnecting just a few vertices from a large component. While such peripheral edges may be desirable in some applications, we argue that in most scenarios more central edges – whose deletion ideally even leads to several connected components – are beneficial from an attacker's point of view. We further explore this in our case study on (parts of) the Berlin road network in Sect. 6. To be able to process non-trivial instances, we propose to adapt a generic greedy algorithm in the next section.

5 Greedy Heuristic for *k*-GRoDel

We adapt the general greedy algorithm previously used for many related problems. The basic idea of this algorithm is to iteratively pick the edge with best marginal loss until k edges are found (see Algorithm 1).

Table 1. Solution set centrality scores as defined in Sect. 4. Since multiple optimal solutions exist for some graphs, the score is computed for each solution and aggregated in this table.

graph	BA1		BA2		BA3		grid5x3		grid5x6	
opt	FI	THR	FI	THR	FI	THR	FI	THR	FI	THR
min	0.31	0.40	0.29	0.47	0.32	0.40	0.24	0.53	0.09	0.69
mean	0.33	0.40	0.29	0.47	0.34	0.40	0.24	0.60	0.10	0.69
max	0.37	0.40	0.29	0.47	0.36	0.40	0.24	0.67	0.13	0.69
graph	grid7x4		hotdog5x6		WS1		WS2		WS3	
opt	FI	THR	FI	THR	FI	THR	FI	THR	FI	THR
min	0.11	0.76	0.14	0.71	0.34	0.49	0.27	0.34	0.16	0.28
mean	0.11	0.76	0.14	0.71	0.34	0.49	0.27	0.34	0.16	0.28
max	0.11	0.76	0.14	0.71	0.34	0.49	0.27	0.34	0.16	0.28

The greedy algorithm starts by computing the Laplacian pseudoinverse, which is required to compute the effective resistance and hence the loss when removing an edge from G. Then, in the main loop, the algorithm iterates all edges in G and computes the loss for each one. The best edge is picked, and G and $\mathbf{L}^\dagger$ are updated. The main loop runs for k iterations.

To implement the greedy algorithm, we need a formula to compute the marginal loss when removing an edge (Line 7) as well as a way to update $\mathbf{L}^\dagger$ after choosing an edge to compute the objective function in the next iteration (Line 10). These depend on the robustness metric used and will be derived in the next section.

For submodular functions the greedy framework can be combined with lazy evaluation [24] to speed up the computation. This lazy evaluation stores all candidates in a priority queue with their most recent loss value and instead of evaluating the loss for all candidates in each iteration of the main loop, it iteratively evaluates (and updates) only the top candidates' loss value until the top candidate is a candidate that has been evaluated in the current iteration of the main loop. Effectively, lazy evaluation reduces the number of evaluations significantly, while still providing a quality guarantee for submodular problems. Even though k-GRoDel is not known to be submodular for THR and is not submodular for FI [38], we still apply this technique because practical experience has proven to lead to good results even for non-submodular problems [25].

Combining the lazy evaluation technique with the general greedy algorithm and THR-based loss and update function leads to `GreedyTHR`: first, compute $\mathbf{L}_G^\dagger$.

Algorithm 1. Greedy algorithm for k-GRoDel

```
1: function Greedy($G$, $k$)
2:     Input: Graph $G = (V, E)$, $k \in \mathbb{N}$
3:     Output: $G_k$ – graph after $k$ edge deletions
4:     $G_0 \leftarrow G$
5:     Compute $\mathbf{L}^\dagger$
6:     for $r \leftarrow 0, \ldots, k-1$ do                          ▷ main loop
7:         Compute loss($e$) $\forall e \in E$                     ▷ evaluation step
8:         $e^* \leftarrow \mathrm{argmax}_{e \in E}$ loss($e$)
9:         $G_{r+1} = G_r \setminus e^*$
10:        Update($\mathbf{L}^\dagger$, $G_{r+1}$)                          ▷ update step
11:    end for
12:    return $G_{r+1}$
13: end function
```

This takes $\mathcal{O}(n^3)$ time (with standard tools in practice). Then, compute the loss for all edges of G and set up a priority queue of all edges by their respective loss value. In the main loop, get the top entry from the priority queue (using lazy evaluation), remove that edge from G and update $\mathbf{L}^\dagger_{G_r}$.

5.1 Total Harmonic Resistance Loss After Deleting an Edge

We now derive an update formula and state the loss formula for THR, which are required for the greedy algorithm.

For the Update step (Line 10) there are efficient ways to compute $\mathbf{L}^\dagger_{G'}$: Removing an edge $e = \{a, b\} \in E$ from G results in $G' = (V, E \setminus \{e\})$ and $\mathbf{L}_{G'} = \mathbf{L}_G - (\mathbf{e}_a - \mathbf{e}_b)(\mathbf{e}_a - \mathbf{e}_b)^T$, where $\mathbf{e}_i$ is the i-th unit vector. One can apply the Sherman-Morrison-Formula [34] (which holds for $\mathbf{L}$ and $(\mathbf{e}_a - \mathbf{e}_b)$ as well) to write:

$$\begin{aligned}\mathbf{L}^\dagger_{G'} &= \mathbf{L}^\dagger_G + \frac{\mathbf{L}^\dagger_G(\mathbf{e}_a - \mathbf{e}_b)(\mathbf{e}_a - \mathbf{e}_b)^T\mathbf{L}^\dagger_G}{1 - (\mathbf{e}_a - \mathbf{e}_b)^T\mathbf{L}^\dagger_G(\mathbf{e}_a - \mathbf{e}_b)} \\ &= \mathbf{L}^\dagger_G + \frac{\mathbf{L}^\dagger_G(\mathbf{e}_a - \mathbf{e}_b)(\mathbf{e}_a - \mathbf{e}_b)^T\mathbf{L}^\dagger_G}{1 - \mathbf{r}_G(a, b)}\end{aligned} \tag{5}$$

There are limitations to using the Sherman-Morrison-Formula for updates: if the removed edge is a bridge, $\mathbf{r}_G(\cdot, \cdot) = 1$ [23] and hence the denominator in Eq. 5 is 0. In case the edge is not a bridge though, we can apply the Sherman-Morrison-Formula.

To handle the case of a bridge edge e, some more involved computation is required. Recall that $\mathbf{L}$ is a (permuted) block diagonal matrix where each block corresponds to a component of G (see Sect. 2). Removing e causes the corresponding block in $\mathbf{L}$ to be split into two blocks – one for each component. All other blocks of $\mathbf{L}$ are not modified by this edge removal. Since the pseudoinverse of a block diagonal matrix is the block matrix build from the pseudoinverse of each block, $\mathbf{L}_{G'}^{\dagger}$ can be found by computing the pseudoinverse of the two blocks related to e and re-using the other pseudoinverse blocks from $\mathbf{L}_G^{\dagger}$.

One has to keep track of a mapping from each node to its component (since in general $\mathbf{L}$ is permuted and we need to know which row/column belongs to which block) which takes $\mathcal{O}(n+m)$ time. For simplicity, we re-compute this after removing a bridge edge (because the pseudoinversion step dominates the running time), but in principle it is possible to dynamically update the connected components instead. The running time of the update step is either $\mathcal{O}(c^2)$ (non-bridge edge) or $\mathcal{O}(\max(n+m, c^3))$ (bridge edge), where c is the size of the block matrix (resp. component of G) that contains e.

For the loss function, the basic formula is $\text{loss}(a,b) := R_h(G) - R_h(G') = \sum_{u<v} \frac{1}{\mathbf{r}_G(u,v)} - \frac{1}{\mathbf{r}_{G'}(u,v)}$. This formula depends on values in $\mathbf{L}_G^{\dagger}$ and $\mathbf{L}_{G'}^{\dagger}$ (via $\mathbf{r}_G(u,v) = \mathbf{L}_G^{\dagger}[u,u] + \mathbf{L}_G^{\dagger}[v,v] - 2\mathbf{L}_G^{\dagger}[u,v]$). Since this is a sum of reciprocal values, deriving an efficient formula proves difficult; we do not know of a closed formula analogous to the forest index or effective resistance yet. Using the basic formula to compute the loss requires computing $\mathbf{L}_{G'}^{\dagger}$ which we have discussed above. Computing the loss when $\mathbf{L}_{G'}^{\dagger}$ is given takes $\mathcal{O}(n^2)$ time and the loss is computed up to $\mathcal{O}(km)$ times in the greedy algorithm (in the worst case, the loss is computed for each edge even though we use lazy evaluation). This leads to $\mathcal{O}(kmn^2 \cdot \max(n+m, c^3))$ time overall for the loss computation. The running time varies a lot depending on the size and number of the components in G and the number of bridge edges.

5.2 Forest Index Loss After Deleting an Edge

For our experiments, we are also interested in results from the greedy algorithm for FI. Since the experimental setup by Zhu et al. [38] is to our knowledge not publicly available, we implemented our own version of this algorithm. There are two differences in our implementation compared to the algorithm description given in their paper though: (i) we exploit a connection between forest index and effective resistance to convert the FI computation back to a problem that is based on the Laplacian matrix. This allows re-use of specialized Laplacian pseudoinverse solvers. (ii) we use the lazy evaluation technique described in Sect. 5, even though FI is not submodular. As mentioned, this technique usually yields good results even for non-submodular problems and in our preliminary experiments we observed no difference in the solution quality.

To derive a marginal loss formula for the forest index, we use a theorem on the connection between effective resistance and forest distance; it allows to reduce the forest index formula (based on forest distance) back to total effective resistance and this reduction facilitates the reuse of some other theorems and algorithms:

Theorem 1. *Given $G = (V, E)$, define the* augmented Graph $G_* = (V_*, E_*)$ *with a universal vertex u^* which is connected to all other vertices: $V_* = V \cup \{u^*\}$ and $E_* = E \cup \{(v, u^*) : v \in V\}$.*

Then $d_G^f(u,v) = \mathbf{r}_{G_*}(u,v)\ \forall u, v \in V$.

The proof (with a slight change in the forest distance definition that does not affect the validity of the result) can be found in Ref. [11, Proposition 7]. From Theorem 1 the following result can be derived:

Proposition 1. *The forest index can be written in terms of the effective resistance of the augmented graph:* $R_f(G) = n \cdot \mathrm{tr}(\mathbf{L}^\dagger_{G_*}) - (n+1) \cdot \mathbf{L}^\dagger_{G_*}[u^*, u^*]$.

Proof. See Appendix A.1. The main idea is to extend the forest index sum by adding a zero term, which then includes the trace of $\mathbf{L}^\dagger_{G_*}$.

Edge Removal. We can now use Proposition 1 to write the forest index $R_f(G)$ in terms of the augmented graph G_* and $\mathbf{L}^\dagger_{G_*}$, which allows us to compute the marginal loss via $\mathbf{L}^\dagger_{G_*}$ when removing an edge from G.

Removing an edge $\{a,b\} \in E$ from G results in $G' = (V, E \setminus \{\{a,b\}\})$ and $\mathbf{L}_{G'_*} = \mathbf{L}_{G_*} - (\mathbf{e}_a - \mathbf{e}_b)(\mathbf{e}_a - \mathbf{e}_b)^T$, where $\mathbf{e}_i$ is the i-th unit vector. Apply the Sherman-Morrison-Formula [34] to write:

$$\mathbf{L}^\dagger_{G'_*} = \mathbf{L}^\dagger_{G_*} + \frac{\mathbf{L}^\dagger_{G_*}(\mathbf{e}_a - \mathbf{e}_b)(\mathbf{e}_a - \mathbf{e}_b)^T \mathbf{L}^\dagger_{G_*}}{1 - \mathbf{r}_{G_*}(a,b)} \tag{6}$$

To calculate the loss$(a,b) := R_f(G') - R_f(G)$ when removing $e = \{a,b\}$ from G, we can use Eqs. (1) and (6) and the connection to total effective resistance (Proposition 1):

Proposition 2. *The marginal loss for the forest index when removing edge* (a,b) *from* G *is:*

$$\begin{aligned} \mathit{loss}(a,b) = {} & \frac{n}{1 - \mathbf{r}_{G_*}(a,b)} \cdot \left\| \mathbf{L}^\dagger_{G_*}[:,a] - \mathbf{L}^\dagger_{G_*}[:,b] \right\|^2 \\ & - \frac{n+1}{1 - \mathbf{r}_{G_*}(a,b)} \cdot (\mathbf{L}^\dagger_{G_*}[u^*,a] - \mathbf{L}^\dagger_{G_*}[u^*,b])^2. \end{aligned}$$

Proof. See Appendix A.2.

We use these loss and update formulae in our greedy algorithm, which allows us to re-use existing code. Running times for the loss and update computation are $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ respectively, which results in a overall running time of $\mathcal{O}(kmn)$ and $\mathcal{O}(kn^2)$ for k iterations of the greedy algorithm.

6 Experimental Results

6.1 Experimental Setup

We conduct experiments to evaluate the quality of the greedy solution for THR to the greedy solution for FI (`GreedyTHR` and `GreedyFI`). Our algorithms are implemented in C++ using the NetworKit toolkit [3] as a graph library. We also build upon the previous work in Refs. [30,31]. To solve linear systems and compute the pseudoinverse, we use the LAMG solver from NetworKit. SimExPal [4] is used to manage our experiments and analyze the results. All experiments are run on a machine with an Intel Xeon 6126 CPU and 192 GB RAM. Code and the experimental setup are available on github: https://github.com/bernlu/GRoDel-THR-FI.

Table 2 lists all networks used in our case study and benchmark study with their approximate number of nodes and edges. For the following analysis, we split them into two groups: *small* graphs with $|V| < 50K$ and *large* graphs with $|V| > 50K$. These networks are taken from SNAP [21], Networkrepository [32] and KONECT [20]. For our experiments, we perform preprocessing on these graphs to turn them into simple graphs by removing self-loops, multi-edges and edge weights; we use the largest connected component of each graph. We set the accuracy parameter ϵ of our LAMG solver (which we use to compute $\mathbf{L}^\dagger$) to 10^{-5}.

Table 2. Graph instances used for experiments, their vertex and edge counts after preprocessing, and the mean closeness centrality of the greedy THR and FI solutions.

Graph	$\|V\|$	$\|E\|$	THR	FI
euro-road	1K	1.3K	0.661	0.160
EmailUniv	1K	5.4K	0.405	0.526
air-traffic-control	1.2K	2.4K	0.579	0.063
inf-power	4K	6K	0.858	0.257
web-spam	4K	37K	0.457	0.039
Bcspwr10	5.3K	8.2K	0.301	0.740
Erdos992	6K	7.5K	0.643	0.691
Reality	6.8K	7.6K	0.825	0.508
Mitte-Berlin-Germany	1K	1.5K	0.648	0.334
Treptow-Köpenick-Berlin-Germany	3.6K	5.2K	0.733	0.283

6.2 Case Study: Berlin Districts

For our case study, we use the road networks of two Berlin districts (Table 2). We choose these networks because road networks in general are easy to visualize and understand intuitively; Berlin specifically has some rivers flowing through the city which create cuts for many districts. These river bridges make for a natural solution to k-GRoDel which we will use as a manually chosen solution to compare to the greedy solutions.

Our graphs are generated from OpenStreetMap [28] data using the osmnx [7] python library. We convert the data into a simple graph and use our NetworKit-based greedy algorithm to find the solution for both THR and FI.

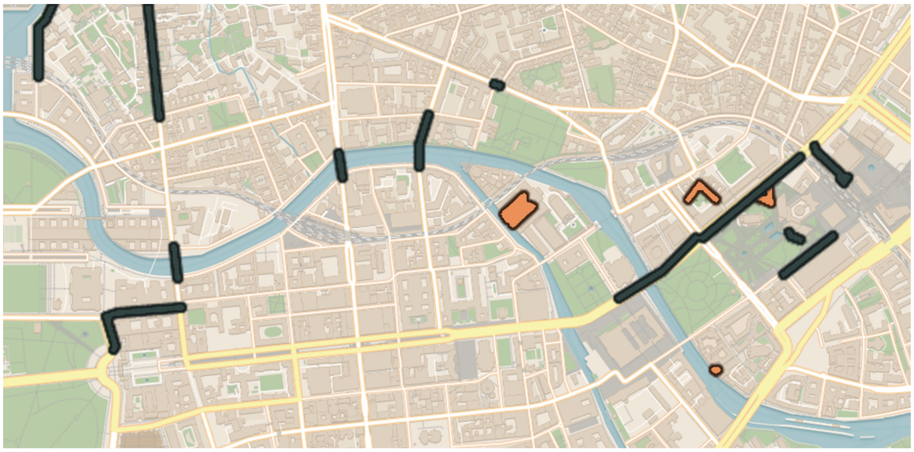


Fig. 2. Berlin case study result. Grey edges are the `GreedyTHR` solution; orange edges are the `GreedyFI` solution. The image is cropped – not all edges in the solution are displayed. `GreedyTHR` finds four of seven river bridges while `GreedyFI` mostly finds residential roads. Image created using OpenStreetMap [28]

The solutions for the *Mitte* district are drawn on a map for visual inspection (Fig. 2). One can clearly see that `GreedyTHR` finds some of the river bridges and other main roads, while `GreedyFI` finds less important streets. We also compare the solutions to the hand-picked solution that consists of the seven river bridges in this district (12 edges in total in our network because of multi-lane bridges). The THR of this manual solution is larger than that of the `GreedyTHR` solution, even though the greedy solution was computed for $k = 20$ edges – this further indicates that THR is a metric that prioritizes edges in a way we consider desirable. In contrast to the previous observation, the FI score of the manual solution is *worse* than the solution (of the same size) found by `GreedyFI`; this is another hint that FI prioritizes edges in the periphery of a network. We have also investigated other districts of Berlin, with very similar results: `GreedyTHR` finds some bridges and large streets; `GreedyFI` finds edges in the periphery and a manual choice of river bridges is better than the greedy solution.

6.3 Benchmark Results

For the benchmark graphs, we evaluate the results by comparing the average closeness centrality of the solutions using the same we method described for the exact solutions in Sect. 4.

Results are available in Table 2. For most benchmark graphs we observe that the `GreedyTHR` solution is considerably more central than the `GreedyFI` solution; on average, the `GreedyTHR` solution is about 25% more central in the closeness centrality metric. In the *Bcspwr10* graph `GreedyTHR` provides a considerably less central solution than `GreedyFI` – though there is no obvious reason for this result.

Regarding running times, with a timeout of 12 h we found solutions for graphs with up to 6.8K nodes or 13K edges. We observe that the network structure (esp. the amount of bridge edges) has significant impact on running times – which is expected given the two ways to compute the update step, where the update for bridge edges is much more expensive. As expected, running times for `GreedyFI` are 2-4 orders of magnitude lower than `GreedyTHR`. The reason for this is that we can use the much more efficient loss formula using the trace of $\mathbf{L}^\dagger$ for `GreedyFI` while we do not know of an analogous formula for `GreedyTHR`.

7 Conclusions

With the protection of large infrastructure in mind, we considered the k-GRoDel problem to identify a set of k particularly vulnerable edges in a graph. To this end, we proposed total harmonic resistance as objective function and compared it against the recently proposed forest index.

We show with small examples where we compute the exact solution that total harmonic resistance prioritizes more central edges than the forest index. We adapt the general greedy algorithm for similar optimization problems to k-GRoDel with total harmonic resistance and show in a case study on the Berlin road network that THR favors more central edges in larger examples as well. Finally, we run benchmark experiments which show that THR mostly favors more central edges than FI in a range of different network types. We note that the greedy algorithm for THR has higher time complexity than the greedy algorithm for FI and our experiments confirm this in practice.

In the future, we would like to focus on speeding up the greedy algorithm for THR by improving the update and loss formulae and by finding other, faster heuristics. These are highly complex problems because of the reciprocity in the objective function – which prevents re-use of most of the results and techniques used for related robustness measures like total effective resistance or forest index.

Acknowledgments. We would like to thank Rob Kooij from TU Delft for insightful discussions on total harmonic resistance and many related measures.

References

1. Albert, R., Jeong, H., Barabási, A.L.: Error and attack tolerance of complex networks. Nature **406**(6794), 378–382 (2000)
2. Angriman, E., Becker, R., D'Angelo, G., Gilbert, H., van der Grinten, A., Meyerhenke, H.: Group-harmonic and group-closeness maximization - approximation and engineering. In: Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX, pp. 154–168. SIAM (2021). https://doi.org/10.1137/1.9781611976472.12

3. Angriman, E., van der Grinten, A., Hamann, M., Meyerhenke, H., Penschuck, M.: Algorithms for large-scale network analysis and the networkit toolkit. In: Bast, H., Korzen, C., Meyer, U., Penschuck, M. (eds.) Algorithms for Big Data. LNCS, vol. 13201, pp. 3–20. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-21534-6_1
4. Angriman, E., et al.: Guidelines for experimental algorithmics: a case study in network analysis. Algorithms **12**(7), 127 (2019). https://doi.org/10.3390/a12070127
5. Barabási, A.L., Pósfai, M.: Network Science. Cambridge University Press, Cambridge (2016)
6. Beygelzimer, A., Grinstein, G., Linsker, R., Rish, I.: Improving network robustness by edge modification. Physica A **357**(3), 593–612 (2005). https://doi.org/10.1016/j.physa.2005.03.040. https://www.sciencedirect.com/science/article/pii/S0378437105003523
7. Boeing, G.: OSMnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. Comput. Environ. Urban Syst. **65**, 126–139 (2017)
8. Bozzo, E., Franceschet, M.: Resistance distance, closeness, and betweenness. Soc. Netw. **35**(3), 460–469 (2013). https://doi.org/10.1016/j.socnet.2013.05.003
9. Cats, O., Koppenol, G.J., Warnier, M.: Robustness assessment of link capacity reduction for complex networks: application for public transport systems. Reliab. Eng. Syst. Saf. **167**, 544–553 (2017)
10. Chan, H., Akoglu, L.: Optimizing network robustness by edge rewiring: a general framework. Data Min. Knowl. Disc. **30**, 1395–1425 (2016)
11. Chebotarev, P.Y., Shamis, E.: The forest metrics of a graph and their properties. Automation Remote Control C/C of Avtomatika I Telemekhanika **61**(8; Issu 2), 1364–1373 (2000)
12. Ellens, W., Spieksma, F., Van Mieghem, P., Jamakovic, A., Kooij, R.: Effective graph resistance. Linear Algebra Appl. **435**(10), 2491–2506 (2011)
13. Freitas, S., Yang, D., Kumar, S., Tong, H., Chau, D.H.: Graph vulnerability and robustness: a survey. IEEE Trans. Knowl. Data Eng. **35**(6), 5915–5934 (2022)
14. van der Grinten, A., Angriman, E., Predari, M., Meyerhenke, H.: New approximation algorithms for forest closeness centrality - for individual vertices and vertex groups. In: Proceedings of the 2021 SIAM International Conference on Data Mining, SDM 2021, pp. 136–144. SIAM (2021). https://doi.org/10.1137/1.9781611976700.16
15. Hasheminezhad, R., Brandes, U.: Robustness of preferential-attachment graphs. Appl. Netw. Sci. **8**(1), 32 (2023). https://doi.org/10.1007/s41109-023-00556-5
16. Jin, Y., Bao, Q., Zhang, Z.: Forest distance closeness centrality in disconnected graphs. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 339–348. IEEE Computer Society (2019). https://doi.org/10.1109/ICDM.2019.00044. https://doi.ieeecomputersociety.org/10.1109/ICDM.2019.00044
17. Klein, D., Randic, M.: Resistance distance. J. Math. Chem. **12**, 81–95 (1993). https://doi.org/10.1007/BF01164627
18. Kooij, R.E., Achterberg, M.A.: Minimizing the effective graph resistance by adding links is NP-hard. arXiv preprint arXiv:2302.12628 (2023)
19. Koç, Y., Warnier, M., Van Mieghem, P., Kooij, R.E., Brazier, F.M.: A topological investigation of phase transitions of cascading failures in power grids. Phys. A **415**, 273–284 (2014)
20. Kunegis, J.: KONECT: the koblenz network collection. In: Carr, L., et al. (eds.) 22nd International World Wide Web Conference, WWW 2013, pp. 1343–1350. International World Wide Web Conferences Steering Committee/ACM (2013). https://doi.org/10.1145/2487788.2488173

21. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection (2014). http://snap.stanford.edu/data
22. Liu, C., Zhou, X., Zehmakan, A.N., Zhang, Z.: A fast algorithm for moderating critical nodes via edge removal. IEEE Trans. Knowl. Data Eng. **36**(4), 1385–1398 (2024). https://doi.org/10.1109/TKDE.2023.3309987
23. Mavroforakis, C., Garcia-Lebron, R., Koutis, I., Terzi, E.: Spanning edge centrality: Large-scale computation and applications. In: Proceedings of the 24th International Conference on World Wide Web, pp. 732–742. International World Wide Web Conferences Steering Committee (2015)
24. Minoux, M.: Accelerated greedy algorithms for maximizing submodular set functions. In: Stoer, J. (ed.) Optimization Techniques, pp. 234–243. Springer, Heidelberg (1978). https://doi.org/10.1007/BFb0006528
25. Minoux, M.: Networks synthesis and optimum network design problems: models, solution methods and applications. Networks **19**(3), 313–360 (1989). https://doi.org/10.1002/net.3230190305
26. Newman, M.: Networks, 2nd edn. Oxford University Press, Oxford (2018)
27. Oehlers, M., Fabian, B.: Graph metrics for network robustness-a survey. Mathematics **9**(8) (2021). https://doi.org/10.3390/math9080895. https://www.mdpi.com/2227-7390/9/8/895
28. OpenStreetMap contributors: OpenStreetMap database (2017). https://www.openstreetmap.org
29. Pizzuti, C., Socievole, A.: A genetic algorithm for enhancing the robustness of complex networks through link protection. In: Aiello, L.M., Cherifi, C., Cherifi, H., Lambiotte, R., Lió, P., Rocha, L.M. (eds.) COMPLEX NETWORKS 2018. SCI, vol. 812, pp. 807–819. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05411-3_64
30. Predari, M., Berner, L., Kooij, R., Meyerhenke, H.: Greedy optimization of resistance-based graph robustness with global and local edge insertions. Soc. Netw. Anal. Mining (2023, to appear). Also available as arXiv preprint 2309.08271
31. Predari, M., Kooij, R., Meyerhenke, H.: Faster greedy optimization of resistance-based graph robustness. In: IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2022, Istanbul, Turkey, 10–13 November 2022, pp. 1–8. IEEE (2022)
32. Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015). http://networkrepository.com
33. Rueda, D.F., Calle, E., Marzo, J.L.: Robustness comparison of 15 real telecommunication networks: Structural and centrality measurements. J. Netw. Syst. Manage. **25**(2), 269–289 (2017)
34. Sherman, J., Morrison, W.J.: Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. Ann. Math. Stat. **21**(1), 124–127 (1950)
35. Summers, T., Shames, I., Lygeros, J., Dörfler, F.: Topology design for optimal network coherence. In: 2015 European Control Conference (ECC), pp. 575–580. IEEE (2015)
36. Wang, X., Pournaras, E., Kooij, R.E., Mieghem, P.V.: Improving robustness of complex networks via the effective graph resistance. Eur. Phys. J. B **87**, 1–12 (2014)
37. Yazdani, A., Jeffrey, P.: Complex network analysis of water distribution systems. Chaos **21**, 016111 (2011)
38. Zhu, L., Bao, Q., Zhang, Z.: Measures and optimization for robustness and vulnerability in disconnected networks. IEEE Trans. Inf. Forensics Secur. **18**, 3350–3362 (2023)

Counterfactual-Based Root Cause Analysis for Dynamical Systems

Juliane Weilbach[1,2](✉), Sebastian Gerwinn[1], Karim Barsim[1], and Martin Fränzle[2]

[1] Bosch Center for Artificial Intelligence, Renningen, Germany
juliane.weilbach@de.bosch.com
[2] Carl von Ossietzky University of Oldenburg, Oldenburg, Germany

Abstract. Identifying the underlying reason for a failing dynamic process or otherwise anomalous observation is a fundamental challenge, yet has numerous industrial applications. Identifying the failure-causing subsystem using causal inference, one can ask the question: "Would the observed failure also occur, if we had replaced the behaviour of a subsystem at a certain point in time with its *normal* behaviour?" To this end, a formal description of behaviour of the full system is needed in which such counterfactual questions can be answered. However, existing causal methods for root cause identification are typically limited to static settings and focusing on additive external influences causing failures rather than structural influences. In this paper, we address these problems by modelling the dynamic causal system using a Residual Neural Network and deriving corresponding counterfactual distributions over trajectories. We show quantitatively that more root causes are identified when an intervention is performed on the structural equation and the external influence, compared to an intervention on the external influence only. By employing an efficient approximation to a corresponding Shapley value, we also obtain a ranking between the different subsystems at different points in time being responsible for an observed failure, which is applicable in settings with large number of variables. We illustrate the effectiveness of the proposed method on a benchmark dynamic system as well as on a real world river dataset.

Keywords: Dynamic Root Cause Analysis · Counterfactual Inference · Dynamic Systems

1 Introduction

Explaining unexpected behaviour in terms of underlying causes is a difficult challenge with a broad range of applications. Such applications range from identifying

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_18.

potential problems in industrial processes to understanding influencing factors in anomalous weather phenomena. For example, within an assembly line of an industrial manufacturing plant, faster identification of root causes of increased scrap rate (the rate at which assembled products fail quality assessment audits) can minimize cost, increase production yield, and increase overall efficiency. If one can observe sufficiently many instances of anomalous behaviour or of faulty traces of a process, one option would be to perform correlation based analysis or causal discovery [14], thereby estimating the influencing factors to the variable "fault" [2,9,17]. Alternatively, causal inference can be used even if only a single anomalous observation is available [5,15]. Here, the identification of root causes is formulated in terms of a counterfactual query: "Would the observed failure also occur, if we had replaced the behaviour of a sub-system at a certain point in time with its *normal* behaviour?". Although such a causal inference approach can estimate a ranked score of each variable involved of being the underlying root cause, we address three main shortcomings of this approach in this paper:

Static Systems: Root cause analysis based on causal inference has been considered only in static environments [3,5,15]. To address this limitation, we fit a time-discretized version of an Ordinary Differential Equation (ODE) system, thereby obtaining a dynamic model. By deriving counterfactual distributions over trajectories we then employ similar strategies as in the static case.

Structural Influences: Existing causal inference methods using counterfactuals [5,15] focus on additive external influences causing failures rather than structural influences. While [2] also considers structural influences, the method is limited to linear models and does not include single time external influences. In this paper, we address this problem by allowing for interventions on the structural equation and the external influence.

Non-linear Systems: Existing methods for root cause analysis are typically limited to linear dynamic models. Here, we address this problem by allowing transition functions to be non-linear using a simple neural network architecture. Additionally, existing methods are limited to small systems as they rely on the computation of Shapley values, which scales exponentially with the number of variables. This becomes infeasible in a dynamic setting, since the corresponding causal graph – unrolled over time – would have an increasingly large number of nodes. While approximate methods for the computation of Shapley values have been proposed [10], we suggest a simple approximation to the Shapley value, which is applicable in settings with large number of variables.

The remainder of this paper is organized as follows: in Sect. 2, we review the related work mentioned above in more detail, and provide necessary background and notation in Sect. 3. In Sect. 4, we describe our method for identifying root causes. In Sect. 5, we first illustrate the mechanisms of the proposed method in a synthetic linear and non-linear setting before evaluating it on a benchmark dataset of [2] as well as on real data describing river levels as in [5]. In Sect. 6, we conclude the paper.

2 Related Work

The problem of identifying the root cause of a system failure or anomaly has been addressed in various domains, including healthcare [15], financial income distributions [4], reliability engineering [9], to name a few. In the context of time-series data, causal inference techniques have been used to qualitatively explain outliers using counterfactual trajectories [16]. To detect root causes affecting graphical structure or transition function Assad et al. [2] propose a method based on assessing the direct causal effect. Modelling such causal effect with linear models, Assad et al. [2] show that the total effects change if the underlying causal model changes. In turn, they can use this fact to identify structural changes in the causal model. However, the method is limited to linear models and does not include single time external influences. If more observations of anomalous data are available, the problem of identifying the root causes is also amenable to statistically estimate the correlation or causation of the different variables and time points onto the variable associated with the label "anomalous". To this end, Tonekaboni et al. [17] introduce feature importance in time (FIT), a scoring mechanism to quantify importance of features in a multi-variate time-series. The authors propose to assess feature importance based on their predictive power w.r.t. the outcome distribution, while accounting for temporal distributional shifts. The approach localizes important features over time and can thus be used to gain useful insights into the behaviour of dynamic systems. However, FIT does not leverage the causal structure of the underlying system and rather provides correlative explanations for the observed outcomes.

Best aligned with our approach, though, is the work by [5] which defines the problem of identifying root causes of a system failure as a counterfactual query. With this reformulation, the authors claim to be the first to propose actionable explanations to anomalous behaviour of underlying systems. In principle, counterfactual reasoning assumes, and leverages, complete causal knowledge of the underlying system in the form of a structural causal model (SCM). More precisely, the work in [5] assumes *invertible* functional causal models: models in which exogenous variables are computable from endogenous system observations. In fact, the authors leverage the default split between endogenous and exogenous variables in a graphical causal model to disentangle a node's inherited impact from its own contribution. They account for the notion of graded causation [7] and provide order-independent feature scoring using a game-theoretic concept commonly adopted in explainable machine learning [10], namely Shapley values [13]. With its computation complexity, their approach lacks direct applicability to dynamical systems. In our experiments, we compare against the linear model performing interventions on the exogenous variable analogously to [5].

3 Background and Notation

As mentioned in the introduction, we are interested in a counterfactual approach to identify the root cause of a system failure. In this section, we introduce the necessary concept from the literature and also introduce the notation we use throughout the paper. Following the notation from Peters et al. [12], we denote the sequence of observations of the system of interest by d-variate time series $(\mathbf{Y}_t)_{t\in\mathbb{Z}}$ where each $\mathbf{Y}_t$ for fixed t is the vector $(Y_t^1, ..., Y_t^d)$. Each Y_t^j represents the jth observable of a system at time t. By some abuse of notation, if we omit super- or subscripts, we refer to the full time series. That is, $\mathbf{Y} = (\mathbf{Y}_t)_{t\in\mathbb{Z}}$, $\mathbf{Y}^j = (Y_t^j)_{t\in\mathbb{Z}}$ and $\mathbf{Y}_t = (Y_t^j)_{j\in\{1,...,d\}}$. The full time causal graph $\mathcal{G}_t$ with a node for each time point and signal Y_t^j for $(j,t) \in 1, ..., d \times \mathbb{Z}$ has theoretically infinitely many nodes and is assumed to be acyclic, while the summary graph $\mathcal{G}$ with nodes $Y^1, ..., Y^d$ may be cyclic.

Definition 1. *(Structural causal model (SCM))* [12]
An SCM $\mathcal{M}(\mathcal{S}, P_N, \mathcal{G})$ is defined by a set of structural equations $\mathcal{S}$, an acyclic graph $\mathcal{G} = (\mathcal{Y}, \mathcal{E})$, and a set of independent noise variables $N^j \sim P_{N^j}, j \in \mathcal{Y}$. The structural equations for each node j are given by:

$$S^j := Y^j = f^j(Y^{PA(j)^{\mathcal{G}}}, N^j)$$

where $\mathcal{S} = \cup_{j\in\mathcal{E}}\{S^j\}$ is a set of structural collections, and $PA(j)^{\mathcal{G}} \subseteq \mathcal{E}$ denotes the parents of the node j according to the graph $\mathcal{G}$.

To describe dynamic processes, again following [12], we extend the above definition to the dynamic case by unrolling a causal graph over time as follows:

Definition 2. *(Dynamic SCM)*
In analogy to a static SCM, a dynamic SCM $\mathcal{M}(\mathcal{S}_t, P_{N_t}, \mathcal{G}_t)$ is given by an acyclic graph $\mathcal{G}_t$ and exogenous noise influences $N_t^j \sim P_{N_t^j}$ independent over each point in time t and variable j. $\mathcal{G}_t$ is referring to a graph consisting of an unrolled version of a summary graph $\mathcal{G}$. Following the notation of [12], the structural equations for node Y_t^j are given by:

$$S_t^j := Y_t^j = Y_{t-1}^j + f^j(Y_{t-1}^{PA(j)}, Y_{t-1}^j) + N_t^j$$

with $PA(j)$ being the parents of node j according to the summary graph $\mathcal{G}$ excluding the node itself. A notable difference from static SCMs is that the functional coupling f is constant over time[1].

Definition 3. *(Interventional Dynamic SCM) Let $\mathcal{J}$ be a set of interventions in which each element ξ can be of the following form:*

[1] Note that we restrict ourselves to additive noise in order to realize an *invertible* SCM, see [5].

$$\xi := do(P_{N_t^j}) = \tilde{P}_{N_t^j}, \qquad (1) \qquad \textit{or} \qquad \xi := do(S_t^j) = \tilde{S}_t^j \qquad (2)$$

where $\tilde{P}_{N_t^j}$ is a new noise distribution and $\tilde{S}_t^j$ is a new structural equation for the node j at time t. The interventional dynamic SCM is then defined by replacing either the noise distribution or structural equation within a given dynamic SCM $\mathcal{M}(\mathcal{S}_t, P_{N_t}, \mathcal{G}_t)$. Here Eq. 1 denotes a soft intervention on the noise distribution whereas Eq. 2 denotes an intervention on the structural intervention. We denote the resulting intervened dynamic SCM then by $M_{\mathcal{J}}(S_t, P_{N_t}, G_t)$.

As each SCM (interventional or not) defines structural equations and noise distributions, it can generate a trajectory of observations. We denote the distribution of the observations generated by the SCM as $P_{\mathcal{M}}$ and the distribution of the observations generated by the intervened SCM as $P_{\mathcal{M}_{\mathcal{J}}}$. Given an observed trajectory, we can now also define the counterfactual distribution describing hypothetical trajectories which would have been observed if an (alternative) intervention had been performed.

Abducted and Counterfactual SCMs. Let $\mathbf{Y}^F$ be an observed trajectory and $\mathcal{M}$ a given dynamic SCM. In order to construct a counterfactual dynamic SCM, we define the noise posterior distribution $P_{N_t^j}(N_t^j|\mathbf{Y}^F) = \delta(N_t^j - N_t^{F,j})$ by:

$$N_t^{F,j} = -Y_{t-1}^{F,j} - f^j(Y_{t-1}^{F,PA(j)}, Y_{t-1}^{F,j}) + Y_t^{F,j} \qquad (3)$$

where f^j is the structural equation of the node j and $PA(j)$ are the parents of the node j according to the summary graph $\mathcal{G}$. The resulting dynamic SCM, in which the noise distributions $P_{N_t^j}$ are replaced with the above defined noise posterior distributions, is then denoted as $\mathcal{M}^F$ indicating that the noise distributions are abducted from the observed trajectory $\mathbf{Y}^F$. In fact, when generating trajectories from this abducted SCM, it only generates the observed trajectory $\mathbf{Y}^F$ due to the above setting of the noise variables. In order to generate new counterfactual trajectories reflecting alternative outcomes, we need to perform an intervention on this abducted SCM, leading to the counterfactual SCM. That is, given an abducted SCM $\mathcal{M}^F$ and a set of interventions $\mathcal{J}$, we refer to the resulting interventional SCM $\mathcal{M}_{\mathcal{J}}^F$ as the counterfactual SCM. For example, when performing an intervention $do(P_{N_t^j}) = \tilde{P}_{N_t^j}$ on the noise distribution at a specific point in time t and a node j, the counterfactual SCM is defined by the following structural equations:

$$Y_s^e = Y_{s-1}^e + f^e(Y_{s-1}^{PA(e)}, Y_{s-1}^e) + N_s^e, \qquad \text{where} \qquad (4)$$

$$N_s^e \sim \begin{cases} \tilde{P}_{N_t^j} & \text{if } s = t \text{ and } e = j \\ \delta(N_t^j - N_t^{F,j}) & \text{otherwise} \end{cases} \qquad (5)$$

3.1 Root Cause

As we are interested in identifying a root cause, we state here more precisely what we mean by this term. We define a root cause as an intervention according to $M_{\mathcal{J}}(S_t, P_{N_t}, G_t)$ leading to a faulty behaviour. Here, we assume that a faulty behaviour can be detected or defined using a known classifier ϕ. This classifier maps a time series to a binary value, indicating whether the time series is faulty. Such classifier can either be given as a known test function (e.g. corresponding to an end-of-line test in an assembly line, an assertion in a software system, or a medical diagnosis) or can be learned from data (e.g. an outlier-score function learned on normal data).

Definition 4. *(Root cause) Given a classifier ϕ that determines whether an observed trajectory is faulty, we refer to a (set of) intervention(s) Ξ to be the root cause of a failure associated with the classifier ϕ, if observations $(\mathbf{Y}^F_{t,t=1,...T})_j$ from the interventional SCM $\mathcal{M}_{\{\Xi\}}$ are leading to an increased failure rate:*

$$\mathbb{E}_{\mathbf{Y}^F_{t,t=1,...T} \sim \mathcal{M}_{\Xi}}[\phi(\mathbf{Y}^F_{t,t=1,...T})] - \mathbb{E}_{\mathbf{Y}_{t,t=1,...T} \sim \mathcal{M}}[\phi(\mathbf{Y}_{t,t=1,...T})] > 0$$

Note that this corresponds to the average treatment effect of an intervention on the external influence or structural intervention. If the probability of a failure for an external intervention on the noise or structure is larger than without any intervention, we assume that the failure has an underlying root cause.

3.2 Shapley Value

Shapley values, originally defined to quantify the contribution of individual players to the outcome of a game, have been used by Budhathoki et al. [5] in a static setup to define a score for nodes being potential root causes of an observed fault. To this end, interventions (or possible root causes) are identified with players in a game whose outcome is determined by a value function that quantifyes the degree to which a set of interventions can increase the likelihood of correcting a failure (to be defined below).

Definition 5. *(Shapley value) The Shapley value [13] of a player i out of a set N of possible players to the outcome of a game characterized by the outcome function v is defined by:*

$$Sh(i) := \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} (v(S \cup \{i\}) - v(S))$$

Note that in order to calculate the Shapley value, one has to sum over exponentially many subsets of the set of possible players. This is feasible only for small sets of players. As in the context of root cause analysis in a dynamic setting, the set of players corresponds to the set of possible interventions ranging over all possible times and nodes within the unrolled graph of a dynamic SCM. Due to the exponential growth of the number of possible interventions, exact Shapley value estimation is computationally infeasible for dynamic SCMs, and we have to resort to an approximate version.

4 Method for Identifying Root Causes

Now that we have the necessary background, we can describe our method for identifying root causes in dynamic SCMs. The method is based on the following steps and is illustrated in Fig. 1. We want to identify the root cause that caused an observed failure in a system. To this end, we cast this problem in a counterfactual query: "Would the observed failure also occur if we had replaced the *faulty* behaviour of a sub-system at a certain point in time with its *normal* behaviour?". To answer this question after we observed a faulty observation $(\mathbf{Y}^F_{t,t=1,...T})_j$, as illustrated in *Inputs* in Fig. 1, we follow the steps of counterfactual distribution calculation: abduction, action and prediction [11]. However, in order to apply those steps, we need an SCM characterizing the normal and potentially the abnormal system. To characterize the normal system, we assume to have access to data representing the normal behaviour of the system, as shown in *Inputs* in Fig. 1. Additionally, we assume to have at least a summary graph $\mathcal{G}$ of the system. This summary graph can be obtained from expert knowledge

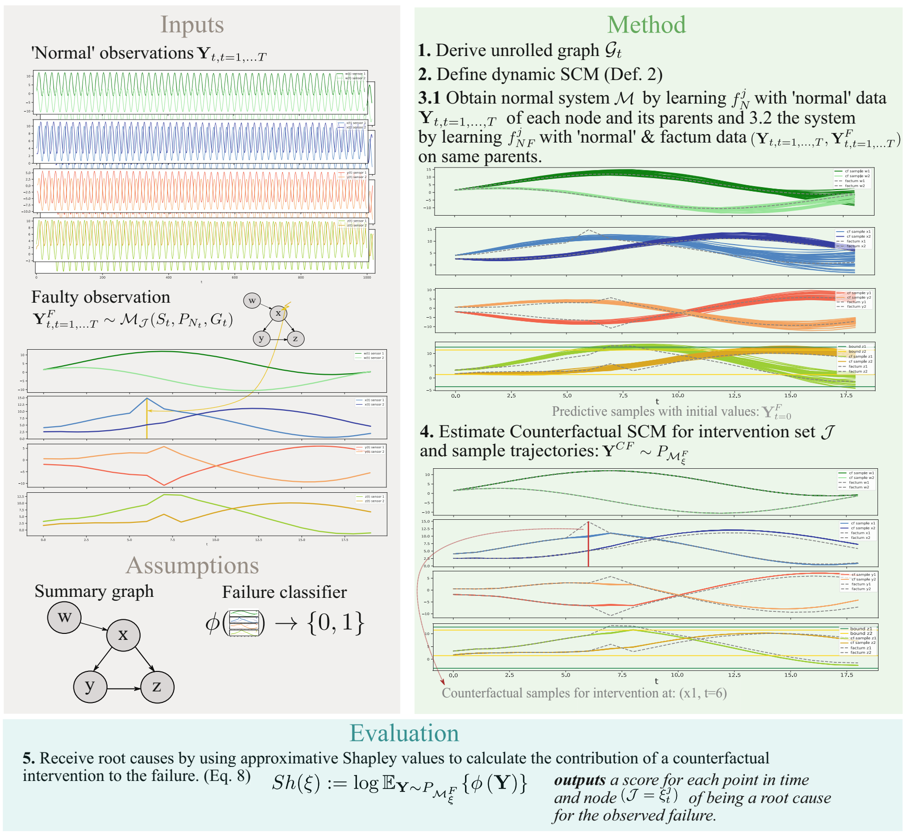

Inputs
'Normal' observations $\mathbf{Y}_{t,t=1,...T}$
Faulty observation
$\mathbf{Y}^F_{t,t=1,...T} \sim \mathcal{M}_{\mathcal{J}}(S_t, P_{N_t}, G_t)$
Assumptions
Summary graph
Failure classifier
$\phi(\cdot) \to \{0,1\}$
w
x
y
z
Method
1. Derive unrolled graph $\mathcal{G}_t$
2. Define dynamic SCM (Def. 2)
3.1 Obtain normal system $\mathcal{M}$ by learning f^j_N with 'normal' data $\mathbf{Y}_{t,t=1,...,T}$ of each node and its parents and 3.2 the system by learning f^j_{NF} with 'normal' & factum data $(\mathbf{Y}_{t,t=1,...,T}, \mathbf{Y}^F_{t,t=1,...T})$ on same parents.
Predictive samples with initial values: $\mathbf{Y}^F_{t=0}$
4. Estimate Counterfactual SCM for intervention set $\mathcal{J}$ and sample trajectories: $\mathbf{Y}^{CF} \sim P_{\mathcal{M}^F_\xi}$
Counterfactual samples for intervention at: (x1, t=6)
Evaluation
5. Receive root causes by using approximative Shapley values to calculate the contribution of a counterfactual intervention to the failure. (Eq. 8)
$Sh(\xi) := \log \mathbb{E}_{\mathbf{Y}\sim P_{\mathcal{M}^F_\xi}} \{\phi(\mathbf{Y})\}$
outputs a score for each point in time and node $(\mathcal{J} = \xi^j_t)$ of being a root cause for the observed failure.

Fig. 1. This figure shows an overview of the individual steps of our method.

or from data. Furthermore, as shown in *Assumptions* in Fig. 1, we assume that we know a function ϕ that classifies an observation into faulty or normal.

Fitting the model in step 3.1 of the *Method* part in Fig. 1 we obtain the normal behaviour system $\mathcal{M}$ by learning the functions f_N^j with the inputs being normal observations Y_t of each node and its parents of the summary graph $\mathcal{G}$. If for both, normal as well as abnormal data, a node and hence its transition function is not anomalous, the transition function would be identical for both settings. Therefore, in 3.2 we additionally fit a transition function f_{NF}^j with normal and factum data as input on the same parents and children as in 3.1 of the known graph $\mathcal{G}$ and with that we define the SCM $\mathcal{FM}$. We show predictive samples of $\mathcal{M}$ in the graph under 3.2.

Estimating the Counterfactual: In the abduction step, we first infer the noise distribution corresponding to the observed factum. We refer to the abducted SCMs $\mathcal{M}^F$ and $\mathcal{FM}^F$ by applying the factum as function input to f_N^j and f_{NF}^j and constructing the resulting noise posterior distributions as described in Eq. 3. We need to calculate the noise variables for both SCMs separately, because function couplings and noise variables are coupled. In the action step, we perform an intervention in $\mathcal{M}$ by $\xi_{\mathcal{M}} := \{do(P_{N_t^j}) = \tilde{P}_{N_t^j}\}$ (see Eq. 1), where we use the prediction error of our model to estimate the Gaussian noise variance:

$$\tilde{P}_{N_t^j} = \mathcal{N}(0, \sigma_{val}^2), \qquad \sigma_{val}^2 = \frac{1}{V}\frac{1}{T}\sum_v \sum_t (Y_{t+1}^{j,v} - f_j(Y_t^{Pa(j),v}))^2 \tag{6}$$

with $Y^{j,v}$ being a validation trajectory of the normal data, and V the number of validation trajectories. For an intervention in $\mathcal{FM}$ we intervene on the noise as before and we additionally intervene on the structure by $do(S_t^j) = \tilde{S}_t^j$ (see Eq. 2), which replaces the previous transition function f_{FN}^j with a new structural equation $\tilde{S}_t^j$ consisting of the transition function f_N^j originating from the "normal" SCM, obtained purely from training data $\xi_{\mathcal{FM}} := \{do(P_{N_t^j}) = \tilde{P}_{N_t^j}, do(S_t^j) = \tilde{S}_t^j\}$. After the construction of the corresponding counterfactual SCM we can then generate counterfactual trajectories under the different interventions $\mathbf{Y}^{CF} \sim P_{\mathcal{M}^F_{\xi_{\mathcal{M}}}}$, as illustrated in 4. in Fig. 1. If an external influence on node j at time t leads to an abnormal factum, an intervention of the above type should remove the abnormal behaviour and therefore lead to a normal trajectory.

Evaluation: To quantify how close these counterfactual samples are to normal trajectories, the trajectories are processed via a classifier function ϕ (Eq. 4). In turn, we receive a score for each counterfactual sample indicating whether the failure was removed by the counterfactual intervention ξ. We then average over multiple counterfactual samples. To rank interventions at different times and nodes, we can use Shapley values by identifying players with interventions and match-outcomes by the average normality of the counterfactual sample. Shapley values, however, scale exponentially and therefore we use the following simple approximation, which we obtain by ignoring interactions between different interventions, thereby only considering singleton intervention sets. Although mainly motivated by pure computational tractability, we can alternatively assume that perfectly synchronous occurrence of multiple root causes is very unlikely, thereby justifying the restriction to singleton intervention sets. Consequently we arrive the following simple expression of contribution score of individual interventions ξ for each point in time and node:

$$Sh(\xi) := \log \mathbb{E}_{\mathbf{Y} \sim P_{\mathcal{M}_\xi^F}} \{\phi(\mathbf{Y})\} \tag{7}$$

5 Experiments

In the following experiments, we evaluate the effectiveness of the proposed method for different synthetic and real world data-sets. As for synthetic data-sets, we consider both linear and non-linear dynamic systems with single point external failure-causing disturbances as well as a benchmark data-set for identifying structural causes for anomalies [2]. As for the real-world data-set, we are investigating dynamic water flow rate in rivers [1]. For our synthetic experiments, we perform two meta-experiments which analyze the influence on the model performance of varying root cause injections and how robust the model is against violating the assumption that the causal graph is known. We denote our models, a linear and a non-linear model both performing a counterfactual intervention on the external noise influence and on the structural equation with *Lin(*S_t^j, N_t^j*)* and *NLin(*S_t^j, N_t^j*)*. We compare against a linear layer model with counterfactual noise-influence intervention *Lin(*N_t^j*)*, similar to [5] as well as against EasyRCA [2] in the benchmarking experiment. For completeness, we additionally provide a nonlinear model *NLin(*N_t^j*)* with counterfactual noise-influence intervention. In order to model the non-linear dynamic SCM, for *NLin* we use a simple three-layer residual neural network (ResNet) with hyperbolic tangent activation functions and 128 neurons as latent layer.

5.1 Experimental Datasets

Linear Synthetic System: In our first data-set, we consider a linear multivariate system with additive Gaussian noise consisting of four nodes (w, x, y, z), each having two dimensions. The summary graph of the system is shown in *Assumptions* in Fig. 1. The structural equations of the system are of the form:

$$Y_t^j := A^i \mathbf{Y}_{t-1}^j + \sum_{k \in PA(j)} B^k \mathbf{Y}_{t-1}^k + C^l N_t^j, \qquad (N_t^j)_d \sim \mathcal{N}(0,1) \quad \forall d$$

with N_t^j being zero mean standard Gaussian noise. For this system we chose the transition matrices such that they generate a stable system by using eigenvalues smaller than 1 (see Appendix). To simulate a root cause, we inject an additive constant term at a single dimension of a node j at time t to the equation above. Instead of a learned anomaly scoring function, in this experiment, we assume to have access to a function that checks the validity of a given observation, similarly as it would be in a manufacturing scenario, in which an end of line test is performed [6]. Therefore, we examine if a failure on the "last" node in a manufacturing line (here "last" node in the summary graph is z) has occurred. To this end, we use a threshold function, fixed over time for each dimension of node z. More precisely, this classifier can be applied to any time-series observation $(\mathbf{Y}_t^j)_{t\in\{1,\ldots,T\},\, j\in\{w,x,y,z\}}$:

$$\phi(\mathbf{Y}) = 1 - \frac{1}{D_z}\sum_{k=1}^{D_z} \mathbb{1}_{[(\mu_z)_k-(\sigma_z)_k,(\mu_z)_k+(\sigma_z)_k]}(\mathbf{Y}_k^z)$$

Here, the dimension of node z is denoted with D_z. Note that this function provides a gradual feedback of how many of the dimensions in node z are outside of the pre-specified corridor given by the threshold function.

FitzHugh-Nagumo System: Next, to allow for non-linear dynamic behaviour, we are generating data of the FitzHugh-Nagumo system (FHN), which is cyclic with regard to its summary graph, but acyclic in the unrolled graph $\mathcal{G}_t$. Although being a multivariate system, as the two dimensions interact, the corresponding dynamic SCM consists of one node x with two dimensions:

$$\dot{x_1} = 3(x_1 - x_1^3/3 + x_2), \qquad \dot{x_2} = (0.2 - 3x_1 - 0.2x_2)/3$$

We chose the initial values as in [8] but with slightly reduced additive Gaussian noise variance $\sigma^2 = 0.0025$. The root cause is simulated similarly to the linear system by adding a constant to the difference equation at one dimension and time point. We classify an observation as faulty, if it deviates too much from a normal observation. As we have, in this setting, access to the ground truth, a normal observation is represented by a trajectory generated from the ground truth system. Consequently, the classifier consists of a time-varying threshold bound around each dimension of the normal observation without the injected root cause of node x. Denoting the expected trajectory from the system by $\mathbf{E}$ a given observation $\mathbf{Y}$ is then classified to be faulty if it does not deviate more than 10 standard deviations at any point in time from the expected trajectory: $\phi(\mathbf{Y}) = 1 - \prod_t \mathbb{1}_{[\mathbf{E}_t^x-10\sigma_x,\mathbf{E}_t^x+10\sigma_x]}(\mathbf{Y}_t^x)$.

5.2 Evaluation

When we have drawn counterfactual samples from our model, we calculate the approximate Shapley values (see Eq. 4) and use the ϕ function to evaluate each performed intervention based on whether it corrected the failure. The root cause is the intervention of the node j at time t that has the highest influence on the failure. If all counterfactual samples lead to the same ϕ evaluation for all interventions, then no unique root cause could be identified. However, due to random sampling of the counterfactual, this is an unlikely scenario (see for example Fig. 5.) Nevertheless, for the evaluation, we only require that the ground truth

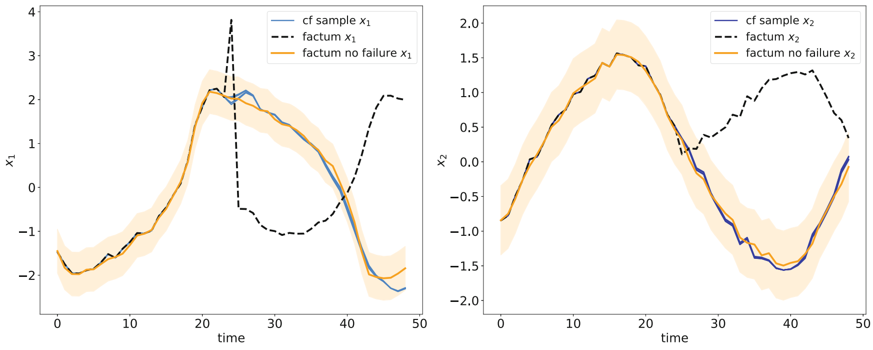


Fig. 2. The figure shows the counterfactual samples for the FHN system with injected root cause at $(j = x_1, t = 24)$. The injected root cause disrupts the system observation heavily (black dashed line). However, the counterfactual intervention performed by our model *NLin(*S_t^j, N_t^j*)* corrects the failure in both dimensions, such that it lies inside the threshold region (orange area). (Color figure online)

root cause is within the set of identified root causes. In Fig. 2 we show five counterfactual samples for the nonlinear FHN system at the actual root cause injection point. Although the injected root cause is fairly large with regard to the interval of the normal observation without failure (drawn as orange line), the counterfactual intervention performed by our model *NLin(*S_t^j, N_t^j*)* corrects the failure for both dimensions of x. In order to analyze root cause injections and how the identification capabilities of our model behave under varying injections, we performed an *Injection experiment* for the synthetic linear and nonlinear FHN system. External disturbances in dynamic systems may be propagated and thereby increase their impact. Alternatively, if the system is robust against incremental noise (as it is the case in the defined systems above due to the external noise influence even under the 'normal' conditions), it is not obvious how large an external influence at which point in time is noticeable. In Fig. 3, we show varying root cause injections for the linear synthetic system (varying constant added to the structural equation) over 20 randomly sampled facta with $T = 20$. It can be seen that the models intervening on the structure and the noise achieve a significantly higher identification score for large added constants. This could be due to a large root cause, in this setting leading to a factum with high distance to the normal data, which may lead to a divergence over time of the normal behaviour system $\mathcal{M}$. Note that we did a similar injection experiment for the FHN system, which can be found in the Appendix.

Assumption Violation. We probe our models on violation of the causal graph assumption for the linear synthetic system. For this, we modify the causal graph used by the underlying model through adding or removing random edges, while keeping the original summary graph for data generation. We use the same facta generated as $\sigma = 500$ in Fig. 3. In Table 1 it can be seen that removing edges for all models has a stronger impact on predictive performance than adding. As

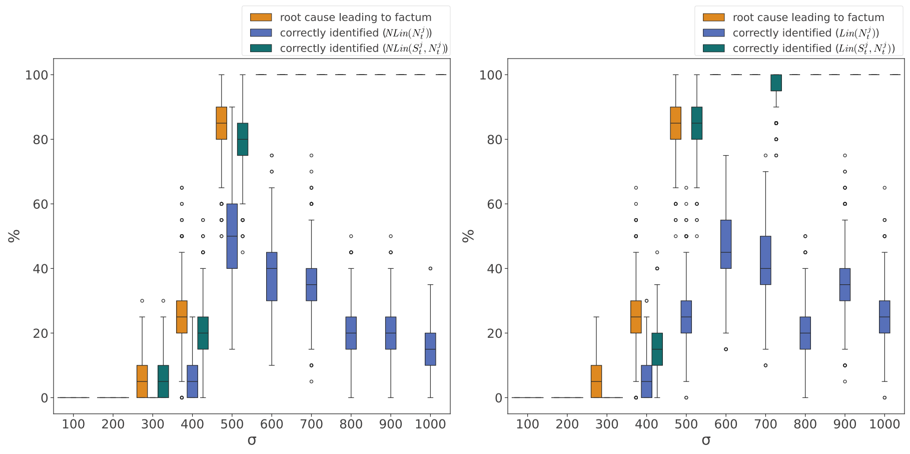


Fig. 3. The root cause was injected at a random node $j = x_1$ at $t = 6$ with varying constants in $[1, 10]$. The horizontal axis shows the injected constant in relation to the noise standard deviation denoted by σ. We report how many root causes could be identified in %.

Table 1. We show the **Accuracy** for the setup $\sigma = 500$ of the linear synthetic system (see Fig. 3) with varying number of *removed or added edges* of the summary graph $\mathcal{G}$ used by the models.

	$NLin(S_t^j, N_t^j)$	$NLin(N_t^j)$	$Lin(S_t^j, N_t^j)$	$Lin(N_t^j)$
nr. of removed edges				
1	0.47 ± 0.25	0.23 ± 0.18	0.47 ± 0.24	0.18 ± 0.15
2	0.29 ± 0.20	0.06 ± 0.06	0.47 ± 0.24	0.12 ± 0.10
nr. of added edges				
1	0.82 ± 0.15	0.12 ± 0.10	1.0 ± 0.0	0.18 ± 0.15
2	0.88 ± 0.10	0.0 ± 0.0	0.88 ± 0.10	0.06 ± 0.06

expected, *Lin(*(S_t^j, N_t^j)*)* performs best on this linear system, closely followed by *NLin(*(S_t^j, N_t^j)*)*. It must be mentioned that in a graph with only four edges, removing an edge is a major incision in the model assumption.

Linear EasyRCA Benchmark. We compare against the linear univariate benchmark of [2] consisting of six nodes and two types of root causes. The parametric root cause meaning they change the coefficient of the parent nodes to a random uniform sampled value. As a special case of the parametric setting, they inject structural root causes, which set the coefficient of the parent nodes to zero. Since EasyRCA excludes single time point root causes, in order to do a fair comparison, we only rank sets of interventions, where we intervene on all times for a given node and evaluating it accordingly by $Sh(\xi_0^j, ...\xi_T^j)$. In their work they inject on two nodes, where one is always the root node of the system and the other one a randomly chosen node. As in their benchmark comparison the root node root cause is excluded, we exclude it from the evaluation as well. In the evaluation they distinguish for parametric and structural root causes, but because our model makes no prediction about the type of root cause, it is sufficient if EasyRCA predicted root causes contain the true root cause, regardless of the type. To rate the normality of a given trajectory $\mathbf{Y}$, we make use of the learned dynamical SCM $\mathcal{M}$ which was fitted on normal observations of the system. More precisely, for the EasyRCA benchmark as well as the following River experiment, we used an outlier score similarly to [3], based on the learned dynamic SCM. That is, given a dynamic SCM $\mathcal{M}$ consisting of N nodes and providing the conditional distribution $p(\mathbf{Y}_t^j|\mathbf{Y}_t^{PA(j,t)})$ via the dynamics equation learned from normal observational data $(\mathbf{Y})_k$, we can define the following outlier score:

$$\phi(\mathbf{Y}) = \frac{1}{NT}\sum_{j,t} \log p(\mathbf{Y}_t^j|\mathbf{Y}_t^{PA(j,t)}) \tag{8}$$

In Table 2, it can be seen that in general the intervention (S_t^j, N_t^j) is preferable to an intervention only on (N_t^j). For the linear systems, the accuracy of *NLin*(S_t^j, N_t^j) and *Lin*(S_t^j, N_t^j) are similarly good, while *EasyRCA* shows lower performance in the factum $T = 100$ experiments. However, *Lin*(S_t^j, N_t^j) is inadequate for addressing the complexities of the nonlinear problem (Fig. 3).

Table 2. We report the **Accuracy** over 20 facta of the summary graph on a linear system and the FHN oscillator. In the lower part of the table we present the experimental results of the EasyRCA benchmark [2] comparing the accuracy for one factum over 30 graphs for different factum lengths T (here, normal data has the same size T. We excluded the 2000 factum length experiment of the EasyRCA benchmark for computational reasons. Additionally, note that since EasyRCA is univariate, it can not be applied to our synthetic systems.)

	NLin		Lin		EasyRCA
	(S_t^j, N_t^j)	(N_t^j)	(S_t^j, N_t^j)	(N_t^j)	
Lin. system	0.94 ± 0.05	0.59 ± 0.24	1.0 ± 0.0	0.29 ± 0.20	–
FHN oscillator	0.90 ± 0.09	0.65 ± 0.23	0.20 ± 0.16	0.15 ± 0.13	–
Lin. Parametric					
`Factum-100`	1.0 ± 0.0	0.97 ± 0.03	1.0 ± 0.0	0.97 ± 0.03	0.87 ± 0.12
`Factum-200`	0.97 ± 0.03	0.93 ± 0.06	1.0 ± 0.0	0.93 ± 0.06	0.93 ± 0.06
`Factum-500`	1.0 ± 0.0	0.97 ± 0.03	1.0 ± 0.0	0.97 ± 0.03	1.0 ± 0.0
`Factum-1000`	1.0 ± 0.0	1.0 ± 0.0	0.97 ± 0.03	0.83 ± 0.14	0.97 ± 0.03
Lin. Structural					
`Factum-100`	1.0 ± 0.0	0.87 ± 0.12	1.0 ± 0.0	0.83 ± 0.14	0.8 ± 0.16
`Factum-200`	0.90 ± 0.09	0.27 ± 0.20	0.70 ± 0.21	0.53 ± 0.25	0.90 ± 0.09
`Factum-500`	1.0 ± 0.0	1.0 ± 0.0	0.87 ± 0.12	1.0 ± 0.0	1.0 ± 0.0
`Factum-1000`	1.0 ± 0.0	0.8 ± 0.16	1.0 ± 0.0	1.0 ± 0.0	0.97 ± 0.03

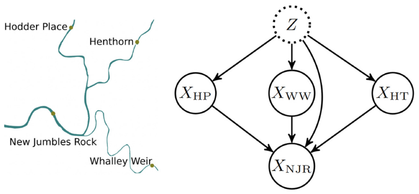


Fig. 4. With the geographical knowledge of the river flow, a summary graph can be inferred (Figure taken from [3]).

Real World River Experiment. We analyse our method on real-world data considering a univariate river experiment consisting of four nodes. The nodes represent measuring stations of the Ribble River in England (data is from [1]). These measuring stations are influenced by unknown external influences such as for example rain. For this reason, the summary graph includes an unobserved confounder Z that influences all nodes. This unobserved confounder affects the accuracy of our model when learning the normal system $\mathcal{M}$ from observational data. The nodes represent stations of the Ribble River that measure the flow rate. Although this data-set has been investigated in [3], as a result of our dynamic viewpoint, we consider a slightly different factum. They consider four time points as static facta and infer the root causes for these. In contrast, we consider an entire time series as factum and infer the root cause. In addition, we use a finer time resolution of 15-minute intervals instead of averaged daily values, which has the advantage that the resulting SCM is less prone to instantaneous effects due to aggregation within a time-window. The finer resolution means that we consider a shorter period of time, namely the three days from 16.03.2019 to 19.03.2019 in which the flow rate is particularly high. As training data, we use the same time span as [3] from 01.01.2010 to 31.12.2018. They provide a z-score threshold for the New jumbles rock station, which we use as ϕ in 8, see also Fig. 5. We find the Shapley values with the highest scores at station Henthorn, which is an upstream station of the New jumbles rock station. Although no ground truth root cause exist for this experiment since it is a real world example, the result is plausible both geographically and with regard to the time point. Nevertheless, the counterfactual intervention cannot correct the failure, as the counterfactual sample is not below the z-score threshold. This could be due to the fact that the influence of the unobserved confounders is particularly high.

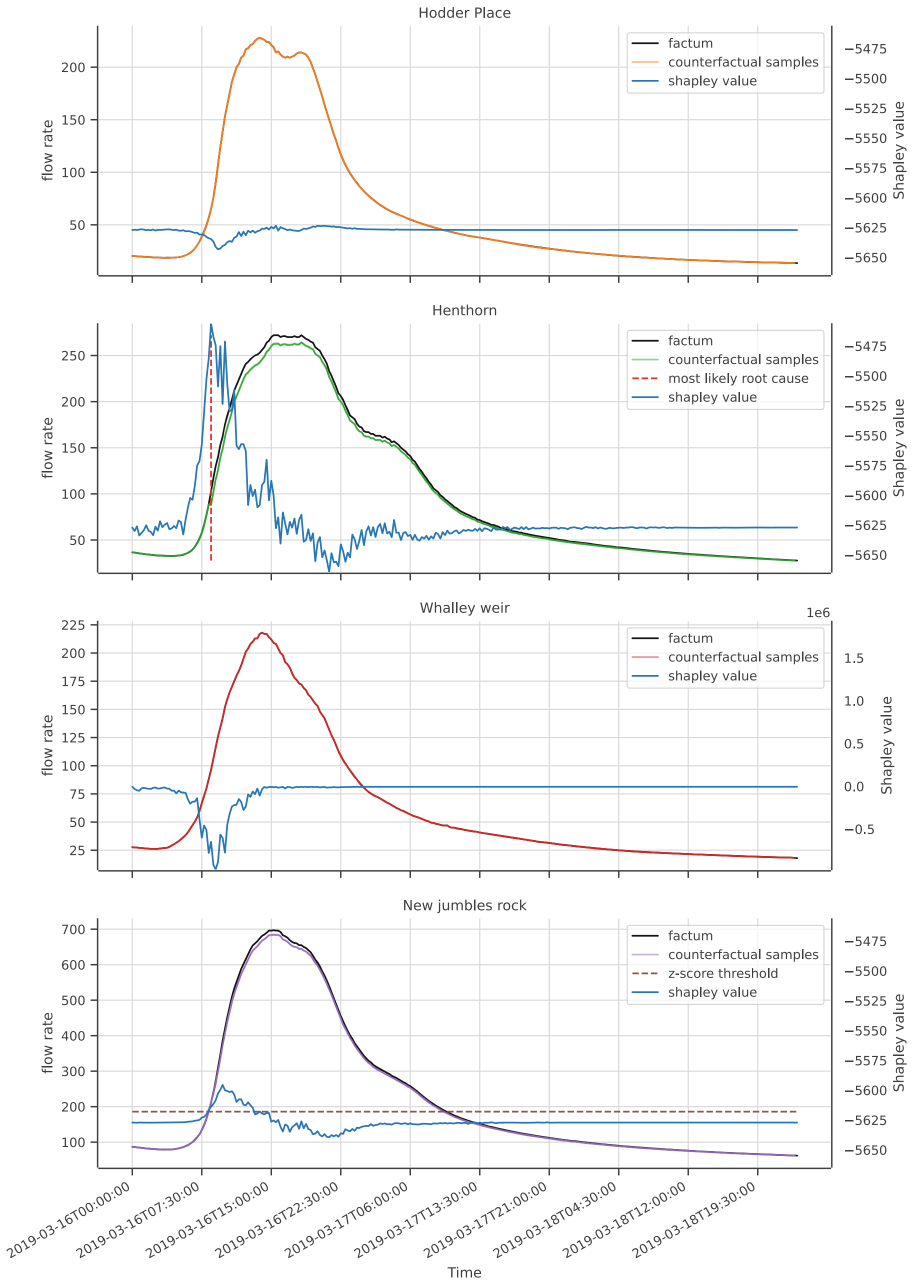


Fig. 5. We show five counterfactual samples (for each station) of our model $NLin(S_t^j, N_t^j)$ with the intervention at the predicted root cause at 08:30 on 16.03.2019. Additionally, we illustrate the resulting Shapley values for each time point, showing that right before the failure occurs the Shapley values increase.

6 Conclusion

In this paper, we have presented a method for identifying root causes in dynamic systems based on counterfactual reasoning. As the proposed method ranks individual interventions corresponding to individual nodes or sensors at particular times within a trajectory, our method is capable of exploiting not only the causal structure but also the natural direction of causality over time. By modelling temporal transitions with a non-linear neural network and a Shapley value approximation, we are able to remove important limitations of current counterfactual root cause analysis methods. While we demonstrated both on synthetic as well as real data the effectiveness of our method in identifying root causes in dynamic systems, there are several directions for further improvement. For example, our method is current limited to the assumption that the root cause consists of a single intervention and that the causal graphical structure is known as well as the absence of latent confounders. In future work, we plan to extend our method to identify multiple root causes and to include uncertainties in the graphical structure as well as potential latent confounders.

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

References

1. Hydrology data explorer. https://environment.data.gov.uk/hydrology/explore. Accessed 18 Mar 2024
2. Assaad, C.K., Ez-Zejjari, I., Zan, L.: Root cause identification for collective anomalies in time series given an acyclic summary causal graph with loops. PMLR (2023). https://proceedings.mlr.press/v206/assaad23a.html
3. Budhathoki, K., Janzing, D., Bloebaum, P., Ng, H.: Why did the distribution change? PMLR (2021). https://proceedings.mlr.press/v130/budhathoki21a.html
4. Budhathoki, K., Michailidis, G., Janzing, D.: Explaining the root causes of unit-level changes (2022). https://arxiv.org/abs/2206.12986
5. Budhathoki, K., Minorics, L., Bloebaum, P., Janzing, D.: Causal structure-based root cause analysis of outliers. PMLR (2022). https://proceedings.mlr.press/v162/budhathoki22a.html
6. Göbler, K., Windisch, T., Drton, M., Pychynski, T., Sonntag, S., Roth, M.: `causalAssembly`: generating realistic production data for benchmarking causal discovery (2024)
7. Halpern, J.Y., Hitchcock, C.: Graded causation and defaults. Brit. J. Phil. Sci. (2015)
8. Hegde, P., Çağatay Yıldız, Lähdesmäki, H., Kaski, S., Heinonen, M.: Variational multiple shooting for bayesian odes with gaussian processes (2022)
9. Ikram, A., Chakraborty, S., Mitra, S., Saini, S., Bagchi, S., Kocaoglu, M.: Root cause analysis of failures in microservices through causal discovery (2022)
10. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions (2017). https://papers.nips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html

11. Pearl, J., Mackenzie, D.: The book of why: the new science of cause and effect (2018). https://books.google.de/books?id=EmY8DwAAQBAJ
12. Peters, J., Janzing, D., Schölkopf, B.: Elements of causal inference - foundations and learning algorithms. In: Adaptive Computation and Machine Learning Series (2017)
13. Shapley, L.S.: A Value for N-Person Games, Santa Monica, CA (1952)
14. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search (2000)
15. Strobl, E.V., Lasko, T.A.: Identifying patient-specific root causes with the heteroscedastic noise model (2022)
16. Sulem, D., et al.: Diverse counterfactual explanations for anomaly detection in time series. arXiv preprint arXiv:2203.11103 (2022)
17. Tonekaboni, S., Joshi, S., Campbell, K., Duvenaud, D.K., Goldenberg, A.: What went wrong and when? instance-wise feature importance for time-series black-box models (2020)

Dropout Regularization in Extended Generalized Linear Models Based on Double Exponential Families

Benedikt Lütke Schwienhorst[1](✉), Lucas Kock[2], Nadja Klein[3], and David J. Nott[2]

[1] Department of Mathematics, University of Hamburg, Hamburg, Germany
benedikt.luetke.schwienhorst@uni-hamburg.de
[2] Department of Statistics and Data Science, National University of Singapore, Singapore, Singapore
[3] Research Center Trustworthy Data Science and Security and Department of Statistics, Technische Universität Dortmund, Dortmund, Germany

Abstract. Even though dropout is a popular regularization technique, its theoretical properties are not fully understood. In this paper we study dropout regularization in extended generalized linear models based on double exponential families, for which the dispersion parameter can vary with the features. A theoretical analysis shows that dropout regularization prefers rare but important features in both the mean and dispersion, generalizing an earlier result for conventional generalized linear models. To illustrate, we apply dropout to adaptive smoothing with B-splines, where both the mean and dispersion parameters are modeled flexibly. The important B-spline basis functions can be thought of as rare features, and we confirm in experiments that dropout is an effective form of regularization for mean and dispersion parameters that improves on a penalized maximum likelihood approach with an explicit smoothness penalty. An application to traffic detection data from Berlin further illustrates the benefits of our method.

Keywords: B-splines · double exponential families · dropout regularization · generalized linear models · nonparametric estimation · overdispersion and underdispersion

1 Introduction

Dropout regularization [12,20] was introduced in the context of neural networks and has been successfully implemented in a large number of applications [18, 21,22]. In its original formulation, dropout omits a randomly chosen subset of features during each iteration of stochastic gradient descent (SGD) optimization

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_19.

of the training loss. It has been generalized in various ways, and is an example of a broader class of methods using randomly corrupted features in model training [5,16].

This work considers dropout for extended generalized linear models (GLMs) based on double exponential families (DEFs). The DEF was introduced by [7] and generalizes the natural exponential family (EF) by incorporating an additional dispersion parameter. [7] also developed regression models where the distribution of the output follows a DEF, and both the mean and dispersion can vary with the features. The extended GLMs of [7] are related to extended quasi-likelihood methods [14] and are particularly useful for count data. If count data are modelled using a binomial or Poisson distribution, then there is no separate scale parameter, and the variance is a function of the mean. For real count data, the variance can be more or less than expected based on a binomial or Poisson mean-variance relationship. If the variance is larger than expected, this is referred to as overdispersion [17] and not taking it into account can result in unreliable inference [6]. Underdispersion, where the variance is less than expected, can also occur, but is less common. [17] state that it should be assumed that overdispersion is present unless proven otherwise. Extended GLMs for double binomial or double Poisson distributions are suitable alternatives to standard GLMs for count data exhibiting over- or underdispersion. We use dropout regularization in the mean and the dispersion to avoid overfitting and to prevent co-adaptation of features [11].

Overdispersed or underdispersed models for count data arise in important machine learning applications. One example is the use of Dirichlet-multinomial models, which generalize beta-binomial models, in topic modelling [4]. Flexible models for count data also arise in large-scale regression applications [10] and in mixture-of-experts models with GLM components. The subtleties that arise in the behaviour of dropout for DEF-GLMs will potentially arise in other overdispersed models, where the effect of regularization on the mean and variance needs to be considered jointly.

In conventional GLMs with canonical link functions, [23] have shown that dropout performs regularization which is first-order equivalent to L2 regularization, with a penalty matrix related to the empirical Fisher information. The form of the penalty favors rare but important features. In the neural network literature, previous work by [3] has shown that adding Gaussian noise to the training data is equivalent to L2 regularization (also known as Tikhonov regularization). [1] analyze dropout for both linear and non-linear networks, obtaining related results. [24] consider fast dropout based on analytically marginalizing the dropout noise. [25] consider deep network architectures and identify both explicit and implicit regularization effects of dropout training, where the explicit effect of regularization is approximated by an L2 penalty term. Despite the connections between L2 regularization and dropout, dropout can behave very differently from both L1 and L2 regularization in some circumstances [11].

Our work generalizes the study of [23] to extended GLMs based on DEFs. Our first contribution is to give a theoretical analysis of the behaviour of dropout for extended GLMs. We discuss the way that the regularization parameters for the mean and dispersion models interact, and the effect of over- or underdispersion on the regularization of the mean. We illustrate that dropout regularization in DEF-GLMs can be regarded as a form of L2 regularization, where the penalty matrix depends on the Fisher information for the mean and dispersion parameters. Understanding of the induced penalty requires considering penalization of the mean and dispersion parameters jointly, and penalization of the dispersion model behaves differently to penalization in the mean, with an asymmetry in the treatment of over- and underdispersion. Our paper contributes to the understanding of dropout regularization beyond its traditional application in neural networks, demonstrating its applicability in other model classes. Our second contribution is to consider the use of dropout regularization in nonparametric estimation of extended GLMs with B-splines, and to compare dropout with penalized maximum likelihood estimation (PMLE) [9] in this setting. [9] consider an explicit smoothness penalty based on second-order difference operators for regularization [8]. In accordance with our theoretical analysis, our experiments demonstrate that dropout regularization can be particularly effective when important B-spline basis functions are analogous to rare features in the mean and dispersion model. We also verify that the performance improves when there is only overdispersion and no underdispersion. After having confirmed the theory in our simulations, we illustrate the efficacy of dropout for DEF-GLMs with a real data set on traffic detection in Berlin.

The rest of this paper is organized as follows. Section 2 introduces the extended GLM based on DEFs and Sect. 3 analyzes the regularization induced by dropout in this model class. Then, an application to adaptive smoothing with B-splines based on simulated and real world data is discussed in Sect. 4. Section 5 gives a concluding discussion. Our code, together with the traffic detection data and the Appendix is publicly available on GitHub[1].

2 Generalized Linear Models Based on Double Exponential Families

2.1 Double Exponential Family

The distribution of a random vector Y is in a natural exponential family (EF) if its target density has the form

$$f_\theta(y) = \exp\left(\langle \theta, y\rangle - b(\theta) + c(y)\right), \tag{1}$$

where $c(y) = \log h(y)$ and $b(\theta) = \log C(\theta)^{-1}$ for known functions $C(\cdot)$ and $h(\cdot)$. The EF is parameterized through $\theta \in \Theta$. When Y is a random variable with density (1), $\mu := \mathbb{E}_\theta[Y] = b'(\theta)$ and $\mathbb{V}_\theta[Y] = b''(\theta)$, that is,

$$\mathbb{V}_\theta[Y] = \frac{\partial}{\partial\theta}\mathbb{E}_\theta[Y].$$

This shows that the variance is determined by the mean function only. Many popular distributions such as the Gaussian, Poisson and binomial have densities

[1] https://github.com/luetkeschwienhorst/dropoutinextendedGLMs.

of the form (1). As discussed earlier, the implied mean-variance relationship for binomial and Poisson models may be inappropriate for real data, since for the Poisson distribution $\mathbb{V}_\theta(Y) = \mathbb{E}_\theta(Y)$, and for a binomial distribution with N trials $\mathbb{V}_\theta(Y) = \mathbb{E}_\theta(Y)(N - \mathbb{E}_\theta(Y))/N$. For a general introduction and more details on EFs we refer to [15].

[7] extends the notion of the EF to the double exponential family (DEF) by introducing a dispersion parameter $\gamma > 0$. Target densities in the DEF are of the form

$$f_{\gamma,\theta}(y) := C(\gamma,\theta)\gamma^{\frac{1}{2}} f_\theta(y)^\gamma f_{\theta(y)}(y)^{1-\gamma}, \tag{2}$$

where f_θ and $f_{\theta(y)}$ are densities as in (1), C is a normalizing constant and $\theta(y) = (b')^{-1}(y)$ is the value of θ for which the mean is y, and we allow for $\theta(y) \in \mathbb{R} \cup \{-\infty, \infty\}$ to deal with boundary cases. The introduction of the dispersion parameter γ decouples the mean and variance of the underlying natural EF and [7] shows that for a random variable $Y \sim \text{DEF}(\theta, \gamma)$ with density (2), $\mathbb{E}[Y] \approx \mu$, $\mathbb{V}[Y] \approx b''(\theta)/\gamma$ and $C(\gamma, \theta) \approx 1$; see [7] for further discussion of the accuracy of these approximations. From the expression for the variance, $\gamma < 1$ ($\gamma > 1$) corresponds to overdispersion (underdispersion). Also, we will assume that the map $\theta \mapsto b'(\theta)$ is one-to-one, such that we can parameterize via μ instead of θ. We will write $f_{\gamma,\mu}$ and $Y \sim \text{DEF}(\mu, \gamma)$ instead of $f_{\gamma,\theta}$ and $Y \sim \text{DEF}(\theta, \gamma)$.

2.2 GLMs Based on DEFs

Consider observed data $\{(y_i, \boldsymbol{x}_i, \boldsymbol{z}_i)\}_{i=1}^n$ of responses $y_i \in \mathcal{Y} \subseteq \mathbb{R}$ and feature vectors $\boldsymbol{x}_i \in \mathbb{R}^{d_\mu}$ and $\boldsymbol{z}_i \in \mathbb{R}^{d_\gamma}$. Conditionally on the features, the responses will be modelled as observations of independent random variables Y_i with distributions from a DEF. The features $\boldsymbol{x}_i$ and $\boldsymbol{z}_i$ will appear in models for the mean and dispersion respectively. In the regression context it is convenient to rewrite the natural EF target density (1) for scalar Y_i as

$$f_{\theta,\phi/\nu_i}(y_i) = \exp\left(\frac{\theta_i y_i - b(\theta_i)}{\phi/\nu_i} + c\left(y_i, \phi/\nu_i\right)\right), \tag{3}$$

where ϕ is a fixed scale parameter, ν_i is a known weight and both ν_i and θ_i can vary between observations. The mean for density (3) is $\mu_i = b'(\theta_i)$, and the variance is $(\phi/\nu_i)b''(\theta_i)$. For a conventional GLM with a binomial response such as a logistic regression, the weight ν_i would be n_i, the number of binomial trials for the ith observation. For binomial and Poisson GLMs the scale ϕ is 1, but in a Gaussian linear regression the scale parameter ϕ is the variance of the response.

We leave dependence of all quantities on ϕ/ν_i implicit in our notation in the following discussion, retaining our previously established notation for DEFs. In our extended GLMs based on DEFs, we assume

$$Y_i \mid \theta_i, \gamma_i \sim \text{DEF}(\theta_i, \gamma_i),$$

where, as discussed below, θ_i and γ_i are functions of $\boldsymbol{x}_i$ and $\boldsymbol{z}_i$ respectively. Following [7], the DEF assumption implies

$$\mu_i := \mathbb{E}[Y_i \,|\, \theta_i, \gamma_i] \approx b'(\theta_i),$$
$$\sigma_i^2 := \mathbb{V}[Y_i \,|\, \theta_i, \gamma_i] \approx \frac{\phi b''(\theta_i)}{\gamma_i \nu_i},$$

and $C(\gamma_i, \theta_i) \approx 1$. We take μ_i and γ_i to be functions of linear predictors $\boldsymbol{x}_i^T \boldsymbol{\beta}$ and $\boldsymbol{z}_i^T \boldsymbol{\alpha}$, where $\boldsymbol{\beta} \in \mathbb{R}^{d_\mu}$ and $\boldsymbol{\alpha} \in \mathbb{R}^{d_\gamma}$ are unknown coefficient vectors. We choose a canonical link in the mean and a log-link for the dispersion so that

$$\theta_i = (b')^{-1}(\mu_i) = \boldsymbol{x}_i^T \boldsymbol{\beta},$$
$$\log(\gamma_i) = \boldsymbol{z}_i^T \boldsymbol{\alpha}.$$

The log-link for the dispersion ensures the dispersion parameter is nonnegative.

3 Dropout Regularization in GLMs Based on DEFs

3.1 Dropout Regularization

Dropout regularization randomly perturbs the observed feature vectors. Given some noise vector $\boldsymbol{\xi}$ and a noise function ν, an observed feature vector $\boldsymbol{x} \in \mathbb{R}^d$ is transformed into $\widetilde{\boldsymbol{x}} := \nu(\boldsymbol{x}, \boldsymbol{\xi})$. The random perturbation is unbiased, which means that $\mathbb{E}(\widetilde{\boldsymbol{x}}) = x$. In what follows we use multiplicative noise, $\nu(\boldsymbol{x}, \boldsymbol{\xi}) = \boldsymbol{x} \odot \boldsymbol{\xi}$, where $\odot$ is the elementwise product of two vectors. Assuming $\mathbb{E}[\boldsymbol{\xi}] = \mathbf{1}_d$, we have $\mathbb{E}[\widetilde{\boldsymbol{x}}] = \boldsymbol{x} \odot \mathbb{E}[\boldsymbol{\xi}] = \boldsymbol{x}$.

Typical choices for the distribution of $\boldsymbol{\xi}$ include Bernoulli dropout with i.i.d. $\xi_j \sim (1-\delta)^{-1}\text{Bernoulli}(1-\delta)$, where $\delta \in (0,1)$ is the dropout probability, and Gaussian dropout with i.i.d. $\xi_j \sim \mathcal{N}(1, \sigma^2)$, with noise variance σ^2 for $j = 1 \ldots, d$. We have described the process of random perturbation for a single feature vector. With many feature vectors, different random perturbations are performed independently for each one. Extended GLMs incorporate features in both the mean and dispersion models and these need not be the same, although they can be. When dropout is performed in both the mean and dispersion models, the perturbations will be independent in the mean and dispersion components. When the features enter into a model linearly, perturbing the features is equivalent to perturbing the parameters independently in the terms of the loss function, since $(\boldsymbol{x} \odot \boldsymbol{\xi})^T \boldsymbol{\beta} = \boldsymbol{x}^T(\boldsymbol{\beta} \odot \boldsymbol{\xi})$.

3.2 Dropout Regularization for the Mean Parameter

We first consider dropout for extended GLMs, where dropout is performed only in the mean and not the dispersion model. The argument closely follows the one given in [23] for the case of conventional GLMs. Dropout in both the mean and dispersion is more complex and is considered in the next subsection.

Based on the assumptions from Subsect. 2.2, dropout regularization in the mean model leads to the optimization problem

$$\min_{\boldsymbol{\beta}, \boldsymbol{\alpha}} \sum_{i=1}^{n} -\mathbb{E}\left[\ell_i\left(\boldsymbol{\beta} \odot \boldsymbol{\xi}_i, \boldsymbol{\alpha}\right)\right], \tag{4}$$

where we have written $\ell_i(\beta, \alpha)$ for the log-likelihood term for the ith observation, and the expectation is taken with respect to the distribution of the i.i.d. vectors $\boldsymbol{\xi}_i$ with $\mathbb{E}[\boldsymbol{\xi}_i] = \mathbf{1}_{d_\mu}$ and $\mathbb{V}[\boldsymbol{\xi}_i] = \sigma_\mu^2 \mathbb{I}_{d_\mu}$. Using the definition of the DEF, and assuming that $C(\gamma_i, \theta_i) = 1$,

$$\begin{aligned} \ell_i(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{z}_i^T\boldsymbol{\alpha} + \exp(\boldsymbol{z}_i^T\boldsymbol{\alpha})\frac{\{y_i \boldsymbol{x}_i^T\boldsymbol{\beta} - b(\boldsymbol{x}_i^T\boldsymbol{\beta})\}}{\phi/\nu_i} \\ + (1 - \exp(\boldsymbol{z}_i^T\boldsymbol{\alpha}))\frac{\{y_i\theta(y_i) - b(\theta(y_i))\}}{\phi/\nu_i}. \end{aligned} \tag{5}$$

Appendix A.1 shows that (4) is approximately equal to

$$\min_{\boldsymbol{\beta}, \boldsymbol{\alpha}} \left\{ -\sum_{i=1}^n \ell_i(\boldsymbol{\beta}, \boldsymbol{\alpha}) + \frac{1}{2}\sigma_\mu^2 \|\boldsymbol{\Theta}\boldsymbol{\beta}\|_2^2 \right\}, \tag{6}$$

where $\boldsymbol{\Theta} = \mathrm{diag}(\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{1/2}$ is a penalty matrix, $\boldsymbol{X} \in \mathbb{R}^{n,d_\mu}$ is the design matrix with ith row $\boldsymbol{x}_i$, and the diagonal weight matrix $\boldsymbol{W}$ depends on both $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$,

$$\boldsymbol{W} := \mathrm{diag}\left(\frac{\gamma_1 b''(\boldsymbol{x}_1^T\boldsymbol{\beta})}{\phi/\nu_1}, \dots, \frac{\gamma_n b''(\boldsymbol{x}_n^T\boldsymbol{\beta})}{\phi/\nu_n} \right) \in \mathbb{R}^{n,n}. \tag{7}$$

The L2 penalty shrinks the normalized vector $\boldsymbol{\Theta}\boldsymbol{\beta}$ towards the origin. As a result, the estimated weights $\hat{\beta}_j$ of some features can be close to zero, effectively removing the feature. The hyperparameter σ_μ^2 controls the degree of shrinkage.

The form of $\boldsymbol{\Theta}$ allows us to understand which features will experience little penalty. The diagonal entries are

$$\Theta_{jj} = \left(\sum_{i=1}^n (\gamma_i\nu_i/\phi)^2 \mathbb{V}[Y_i] x_{ij}^2 \right)^{1/2} = \left(\sum_{i=1}^n (\nu_i/\phi)^2 \gamma_i \varphi_i x_{ij}^2 \right)^{1/2}, \tag{8}$$

where $\mathbb{V}[Y_i] := \phi b''(\boldsymbol{x}_i^T\boldsymbol{\beta})/(\nu_i\gamma_i) = \varphi_i/\gamma_i$ is the variance of the dependent variable Y_i according to the model and $\varphi_i := \phi b''(\boldsymbol{x}_i^T\boldsymbol{\beta})/\nu_i$ is the "baseline" variance when there is no over- or under-dispersion ($\gamma_i = 1$). The expression inside the brackets in (8) is a rescaled second moment of the jth feature. It will be small if x_{ij} is small – or even zero – for most i, or if $(\gamma_i\nu_i/\phi)^2\mathbb{V}[Y_i] = (\nu_i/\phi)^2\gamma_i\varphi_i$ is small enough when x_{ij} is large. Thus, "rare" features which are close to zero for most samples, and which are associated with small baseline variances φ_i and large overdispersions γ_i when the feature value is large, will experience little penalty.

Our general perspective includes the special case of GLMs where $\gamma_i = 1$ for $i = 1, \dots, n$. Logistic regression was discussed in detail by [23] and they point out that little penalty is exerted on rare features which produce confident predictions. In addition, [23] state that $(1/n)\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X}$ is the observed Fisher information with respect to $\boldsymbol{\beta}$. This adds a geometric perspective to dropout regularization: the normalization of $\boldsymbol{\beta}$ by $\boldsymbol{\Theta}$ ensures that penalization is performed in accordance with the curvature of ℓ around the true parameter value. $\boldsymbol{\Theta}\boldsymbol{\beta}$ represents $\boldsymbol{\beta}$ in another basis such that the level sets of ℓ parameterized in $\boldsymbol{\Theta}\boldsymbol{\beta}$ are spherical. We derive this Fisher information matrix in Appendix A.2.

3.3 Dropout Regularization for the Mean and Dispersion Parameter

Next we extend to the case where dropout is performed in both the mean and dispersion models. As before, $\theta_i = \boldsymbol{x}_i^T\boldsymbol{\beta}$ and $\log(\gamma_i) = \boldsymbol{z}_i^T\boldsymbol{\alpha}$. Dropout regularization leads to the minimization problem

$$\min_{\boldsymbol{\beta},\boldsymbol{\alpha}} \sum_{i=1}^{n} -\mathbb{E}[\ell_i(\boldsymbol{\beta}\odot\boldsymbol{\xi}_i, \boldsymbol{\alpha}\odot\boldsymbol{\zeta}_i)], \tag{9}$$

where the expectation is taken with respect to the dropout noise vectors $\boldsymbol{\xi}_i$ and $\boldsymbol{\zeta}_i$, which are independent from each other. It is assumed that $\mathbb{E}[\xi_i] = \mathbf{1}_{d_\mu}$, $\mathbb{V}[\xi_i] = \sigma_\mu^2\mathbb{I}_{d_\mu}$, $\mathbb{E}[\zeta_i] = \mathbf{1}_{d_\gamma}$ and $\mathbb{V}[\zeta_i] = \sigma_\gamma^2\mathbb{I}_{d_\gamma}$. Write $\widetilde{\theta}_i := \boldsymbol{x}_i^T(\boldsymbol{\beta}\odot\boldsymbol{\xi}_i)$ and $\log(\widetilde{\gamma}_i) := \boldsymbol{z}_i^T(\boldsymbol{\alpha}\odot\boldsymbol{\zeta}_i)$. Our assumptions on the dropout noise imply that $\mathbb{E}[\widetilde{\theta}_i] = \theta_i$ and $\mathbb{E}[\log\widetilde{\gamma}_i] = \log\gamma_i$.

In order to get a better understanding of dropout regularization in the dispersion, we aim to find an approximation of (9) similar to the one in (6). We make a normality assumption, $\boldsymbol{\zeta}_i \sim \mathcal{N}(\mathbf{1}_{d_\gamma}, \sigma_\gamma^2\mathbb{I}_{d_\gamma})$ so that

$$\log(\widetilde{\gamma}_i) = \boldsymbol{z}_i^T(\boldsymbol{\alpha}\odot\boldsymbol{\zeta}_i) \sim \mathcal{N}\left(\boldsymbol{z}_i^T\boldsymbol{\alpha}, \sigma_\gamma^2\sum_{j=1}^{d_\gamma} z_{ij}^2\alpha_j^2\right). \tag{10}$$

Although this assumption may seem strong, (10) will often be a good approximation for non-Gaussian dropout noise via a central limit argument. Since $\widetilde{\gamma}_i$ is lognormal, its expectation is

$$\mathbb{E}[\widetilde{\gamma}_i] = \exp\left(\boldsymbol{z}_i^T\boldsymbol{\alpha} + \frac{1}{2}\sigma_\gamma^2\sum_{j=1}^{d_\gamma} z_{ij}^2\alpha_j^2\right) = \gamma_i\exp\left(\frac{1}{2}\sigma_\gamma^2\|\boldsymbol{z}_i\odot\boldsymbol{\alpha}\|_2^2\right). \tag{11}$$

Using (5), we obtain

$$\begin{aligned}\mathbb{E}[\ell_i(\boldsymbol{\beta}\odot\boldsymbol{\xi}_i, \boldsymbol{\alpha}\odot\boldsymbol{\zeta}_i] = \frac{1}{2}\boldsymbol{z}_i^T\boldsymbol{\alpha} + \mathbb{E}[\widetilde{\gamma}_i]\frac{\{y_i\boldsymbol{x}_i^T\boldsymbol{\beta} - \mathbb{E}[b(\boldsymbol{x}_i^T(\boldsymbol{\beta}\odot\boldsymbol{\xi}_i))]\}}{\phi/\nu_i}\\ + (1-\mathbb{E}[\widetilde{\gamma}_i])\frac{\{y_i\theta(y_i) - b(\theta(y_i))\}}{\phi/\nu_i}.\end{aligned}$$

Writing $\boldsymbol{z}_i^T\boldsymbol{\alpha} = \log\mathbb{E}[\widetilde{\gamma}_i] - 1/2\sigma_\gamma^2\|\boldsymbol{z}_i\odot\boldsymbol{\alpha}\|^2$, and

$$\begin{aligned}\widetilde{\ell}_i(\boldsymbol{\beta},\boldsymbol{\alpha}) = \frac{1}{2}\log\mathbb{E}[\widetilde{\gamma}_i] + \mathbb{E}[\widetilde{\gamma}_i]\frac{\{y_i\boldsymbol{x}_i^T\boldsymbol{\beta} - b(\boldsymbol{x}_i^T\boldsymbol{\beta})\}}{\phi/\nu_i}\\ + (1-\mathbb{E}[\widetilde{\gamma}_i])\frac{\{y_i\theta(y_i) - b(\theta(y_i))\}}{\phi/\nu_i},\end{aligned}$$

and using a second-order Taylor series approximation of $\mathbb{E}[b(\boldsymbol{x}_i^T(\boldsymbol{\beta}\odot\boldsymbol{\xi}_i))]$ (see Appendix A.3 for a derivation) gives the following approximation of (9):

$$\min_{\boldsymbol{\beta},\boldsymbol{\alpha}}\left\{\sum_{i=1}^{n} -\widetilde{\ell}_i(\boldsymbol{\beta},\boldsymbol{\alpha}) + \frac{1}{2}\sigma_\mu^2\|\widetilde{\boldsymbol{\Theta}}\boldsymbol{\beta}\|_2^2 + \frac{1}{4}\sigma_\gamma^2\|\boldsymbol{\Gamma}\boldsymbol{\alpha}\|_2^2\right\}, \tag{12}$$

with the penalty matrices $\widetilde{\boldsymbol{\Theta}} = \operatorname{diag}(\boldsymbol{X}^T \widetilde{\boldsymbol{W}} \boldsymbol{X})^{1/2}$ and $\boldsymbol{\Gamma} = \operatorname{diag}(\boldsymbol{Z}^T \boldsymbol{Z})^{1/2}$ and the weight matrix

$$\widetilde{\boldsymbol{W}} := \operatorname{diag}\left(\frac{\mathbb{E}[\widetilde{\gamma}_1] b''(\boldsymbol{x}_1^T \boldsymbol{\beta})}{\phi/\nu_1}, \ldots, \frac{\mathbb{E}[\widetilde{\gamma}_n] b''(\boldsymbol{x}_n^T \boldsymbol{\beta})}{\phi/\nu_n}\right). \tag{13}$$

We will now address the behavior of all three terms in the approximation (12).

Misspecified Log-Likelihood. The first term in (12) is the negative log-likelihood of a DEF model in which the ith observation has location parameter $\theta_i = \boldsymbol{x}_i^T \boldsymbol{\beta}$ and dispersion parameter

$$\mathbb{E}[\widetilde{\gamma}_i] = \gamma_i \exp\left(\frac{1}{2}\sigma_\gamma^2 \|\boldsymbol{z}_i \odot \boldsymbol{\alpha}\|_2^2\right).$$

If the originally specified model was correct, $\widetilde{\ell}_i(\boldsymbol{\beta}, \boldsymbol{\alpha})$ is a misspecified log-likelihood term where γ_i is multiplied by a multiplicative factor $\exp(\frac{1}{2}\sigma_\gamma^2 \|\boldsymbol{z}_i \odot \boldsymbol{\alpha}\|_2^2)$ which is larger than 1. Hence, for the misspecified log-likelihood to achieve a similar fit to the correctly specified case, $\|\boldsymbol{z}_i \odot \boldsymbol{\alpha}\|_2^2$ has to be small favoring rare and important features. As $\boldsymbol{z}_i^T \boldsymbol{\alpha} \to -\infty$ implies $\gamma_i = \exp(\boldsymbol{z}_i^T \boldsymbol{\alpha}) \to 0$, this misspecification penalty favors overdispersion.

Penalty for the Mean Parameter. The second term in (12) is a Tikhonov penalty on $\boldsymbol{\beta}$, where the penalty matrix $\widetilde{\boldsymbol{\Theta}} = \operatorname{diag}(\boldsymbol{X}^T \widetilde{\boldsymbol{W}} \boldsymbol{X})^{1/2}$ is affected by the dropout noise in the dispersion model. Writing $\boldsymbol{\Gamma}_i := \operatorname{diag}(z_{i1}, \ldots, z_{id_\gamma})$, we define

$$\boldsymbol{\Lambda} := \operatorname{diag}\left(\exp(\sigma_\gamma^2 \|\boldsymbol{\Gamma}_1 \boldsymbol{\alpha}\|_2^2/2), \ldots, \exp(\sigma_\gamma^2 \|\boldsymbol{\Gamma}_n \boldsymbol{\alpha}\|_2^2/2)\right)$$

and then $\widetilde{\boldsymbol{W}} = \boldsymbol{\Lambda} \boldsymbol{W}$ due to $\mathbb{E}[\widetilde{\gamma}_i] = \gamma_i \exp((1/2)\sigma_\gamma^2 \|\boldsymbol{\Gamma}_i \boldsymbol{\alpha}\|_2^2)$. The weight matrix in (13) is therefore a rescaled version of the weight matrix in (7). Furthermore, $\sigma_\gamma^2 > 0$ and $\|\boldsymbol{\Gamma}_i \boldsymbol{\alpha}\|_2^2 > 0$ imply $\exp((1/2)\sigma_\gamma^2 \|\boldsymbol{\Gamma}_i \boldsymbol{\alpha}\|_2^2) > 1$, which yields the property $\widetilde{W}_{jj} > W_{jj} > 0$. This carries over to the entries

$$\widetilde{\Theta}_{jj} = \left(\sum_{i=1}^n \exp\left((1/2)\sigma_\gamma^2 \|\boldsymbol{\Gamma}_i \boldsymbol{\alpha}\|_2^2\right) (\nu_i/\phi)^2 \gamma_i \varphi_i x_{ij}^2\right)^{1/2}$$

of $\widetilde{\boldsymbol{\Theta}}$, which then fulfill $\widetilde{\Theta}_{jj} > \Theta_{jj} > 0$. The observations regarding Θ_{jj} from Sect. 3.2 apply to $\widetilde{\Theta}_{jj}$ as well, i.e. rare but important features are favored and overdispersion can attenuate the penalty, such that locally the mean might be modelled by features which are not that rare. $\widetilde{\Theta}_{jj}$ contains the additional exponential term $\exp((1/2)\sigma_\gamma^2 \|\boldsymbol{\Gamma}_i \boldsymbol{\alpha}\|_2^2)$ compared to Θ_{jj} increasing the penalty. This increase will be minimal however, if the terms $\|\boldsymbol{\Gamma}_i \boldsymbol{\alpha}\|_2^2 = \|\boldsymbol{z}_i \odot \boldsymbol{\alpha}\|_2^2$ are negligible in size.

Penalty for the Dispersion Parameter. The third term in (12) is also a Tikhonov penalty, but on $\boldsymbol{\alpha}$ and with the penalty matrix $\boldsymbol{\Gamma} = \text{diag}(\boldsymbol{Z}^T\boldsymbol{Z})^{1/2}$. It shrinks $\boldsymbol{\alpha}$ towards the origin, i.e. $\gamma_i = \exp(\boldsymbol{z}_i^T\boldsymbol{\alpha}) \to 1$ and therefore no overdispersion. For rare features the diagonal entries $\Gamma_{jj} = (\sum_{i=1}^n z_{ij}^2)^{1/2}$ are small, and then α_j can still be large. Similar to the case for $\boldsymbol{\beta}$, the penalty for $\boldsymbol{\alpha}$ has an interpretation in terms of the Fisher information, with the observed Fisher information with respect to $\boldsymbol{\alpha}$ being approximately $(1/n)\boldsymbol{Z}^T\boldsymbol{Z}$; see Appendix A.2 for further details.

In summary, simultaneous dropout regularization on the mean and the dispersion parameter favors rare but important features both in the mean and the dispersion model. Over- or underdispersion can attenuate or strengthen the level of regularization in the mean. Further, the dropout noise in the dispersion model imposes an additional penalty in the mean, if deviations from the base variance cannot be modeled using rare features. While the degree of regularization in the dispersion is controlled by σ_γ^2 alone, it is regulated by σ_μ^2 and σ_γ^2 together in the mean. Lastly, overdispersion appears to be favored relative to underdispersion for two reasons. First, if the features in the dispersion are not that rare, the large misspecification term will encourage overdispersion. Second, the increased penalization due to underdispersion could be large, such that modeling underdispersion might be avoided altogether.

Based on our analysis, an "ideal scenario" in which dropout regularization will be most successful is characterized by true parameter vectors $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ which are sparse relating to a feature vector with only a few components having a significant impact. These significant features should be rare themselves, meaning they should be large only for a relatively small fraction of the data. In addition, we find that overdispersion, i.e. the true dispersion function fulfills $\gamma(z) < 1$ for all z, can be better handled by dropout compared to underdispersion. Further insights on the finite sample properties of employed approximations used can be found in Section A.3 of the Appendix.

4 Application to Adaptive Smoothing with B-Splines

4.1 Simulations

We consider the non-linear regression model

$$y_i \mid x_i \sim \text{DEF}\left(f(x_i), g(x_i)\right), \qquad i = 1, \dots, n, \tag{14}$$

with $x_i \in [0,1]$. The functional effects $f(\cdot)$ and $g(\cdot)$ are modelled using a B-spline basis expansion after a transformation by appropriate link functions. We use B-splines with a relatively large number of knots, so that each basis functions is supported on only a small compact subset of the domain $[0,1]$. If the true effects are mostly flat, but vary rapidly on small sub-intervals of $[0,1]$, then the B-spline basis functions are rare but important features.

Estimates are obtained using SGD, and the choice of optimal hyperparameters $\left(\sigma_\mu^*, \sigma_\gamma^*\right)$ for the dropout noise is performed via random search cross-validation (CV) [2]. A detailed description of the algorithm is given in Appendix

A.4. We consider Bernoulli dropout as well as Gaussian dropout and compare the performance to the PMLE approach discussed in [9].

We simulate data according to (14) for the double Gaussian, double Poisson and double binomial distributions. For both dropout regularization and PMLE the mean is estimated well, but estimation of the dispersion function reveals differences in performance. Therefore, for each distribution we chose a similar mean function and paired it with three different dispersion functions in order to illustrate the conjectures from Sect. 3. A detailed description of the simulation design can be found in Appendix A.5. The three dispersion testfunctions depicted in Fig. 1 coincide with the subsequently mentioned Scenarios 1, 2 and 3. Additional results not presented in the main paper are given in Appendix A.6. The three different scenarios considered are:

Scenario 1: The mean and dispersion functions are mostly flat, with rapid variation over some small intervals, and only overdispersion is present. This is a setting where dropout should perform well according to Subsect. 3.3.
Scenario 2: Scenario 2 is similar to Scenario 1, but there is both over- and underdispersion.
Scenario 3: The mean function is mostly flat, with rapid variation over some small intervals, but the dispersion changes slowly with the feature values, and only overdispersion is present.

For each distribution and each Scenario we simulate $R = 100$ replicate data sets with $n = 250, 500, 1000$. To evaluate and compare different regularization methods we compute the root mean squared error (RMSE) over a fine equidistant grid on $[0, 1]$.

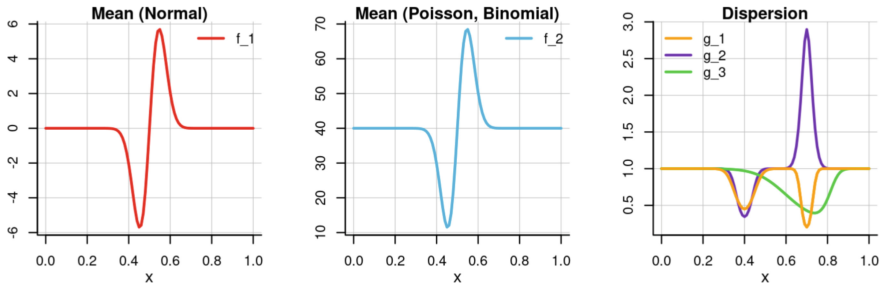


Fig. 1. Testfunctions for the mean and dispersion.

Figure 2 presents boxplots for the RMSEs of the dispersion estimates for the three distributions and across all different scenarios. Figure 7 in Appendix A.6 contains the respective boxplots for the mean estimates. All three methods estimate the mean very well. Figure 2 reveals that dropout regularization is competitive with PMLE in all scenarios.

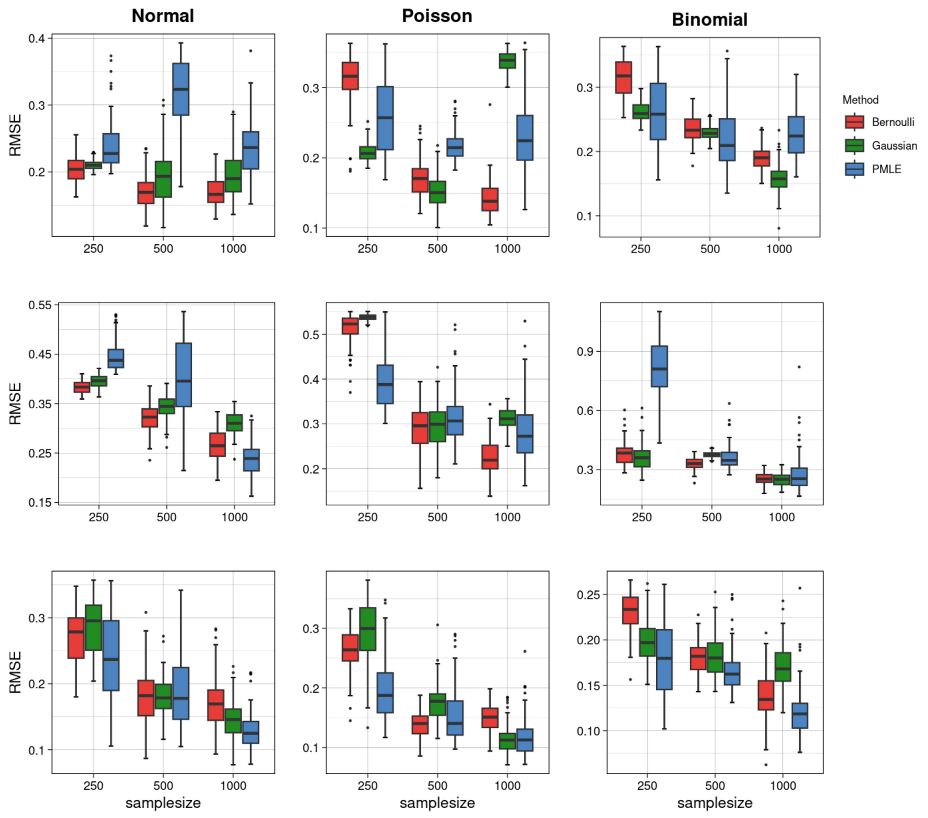


Fig. 2. Boxplots of RMSEs for dispersion parameters of Gaussian data (left column), Poisson data (middle column) and binomial data (right column) across Scenarios 1 (top row), 2 (middle row) and 3 (bottom row). Outliers in the dispersion were cut at the 95-th percentile.

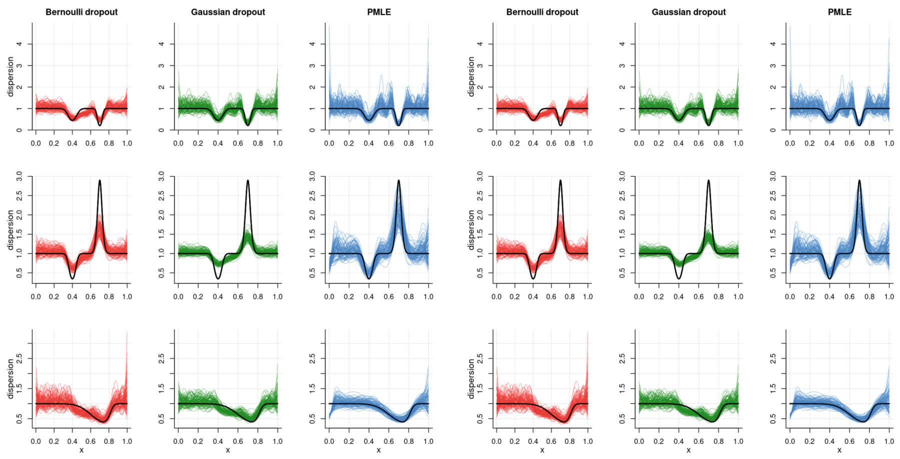
In Scenario 1 with Gaussian data, dropout outperforms PMLE across all sample sizes, and Bernoulli dropout performs slightly better than Gaussian dropout. With count data the difference between dropout and PMLE is less pronounced. Performance of Gaussian dropout is poor for Poisson data in Scenarios 1 and 2 for the largest sample size, $n = 1000$. Figure 10 gives evidence that this poor performance is associated with large bias in estimation of the dispersion function. However, Gaussian dropout performs slightly better than Bernoulli dropout in most cases involving count data. In the binomial case, dropout shows improved performance relative to PMLE as the sample size increases.

Scenario 2 is based on rare but important features as well, but now for some values of the covariate there is strong underdispersion. We find that in small sample sizes dropout performs well compared to dropout for Gaussian and binomial data, but poorly for Poisson data. In other cases the differences are minor.

For Gaussian data, Bernoulli dropout is slightly better than Gaussian dropout, similar to Scenario 1.

Since the true dispersion function for Scenario 3 cannot be modeled through rare features, dropout performs slightly worse than PMLE across all distributions and sample sizes for this case. The findings with respect to the three scenarios match our theoretical analysis from Sect. 3.

Figure 3a depicts the estimated dispersion effects from Gaussian data for all three scenarios and $n = 1,000$. Figure 8 in Appendix A.6 contains the same plots for the mean estimates. The estimation of the mean function is quite accurate across the three methods. The estimated dispersion functions are less accurate since estimation is more difficult [9]. However, in most cases the dispersion estimates reproduced the correct shape. The PMLE estimates suffer from oscillations at the boundaries. For Scenario 2, the rapid changes are not captured very well by Gaussian dropout, particularly where they correspond to regions of underdispersion. The problem also occurs with Bernoulli dropout but is less pronounced. For Scenario 3 the dropout estimates of the dispersion function are noticeably less smooth than the estimates obtained via PMLE. This is consistent with the boxplots at the bottom of the first column of Fig. 2.



(a) Gaussian model (b) Binomial model

Fig. 3. Estimated dispersion effects in the (a) Gaussian model and (b) binomial model for Bernoulli dropout (left), Gaussian dropout (middle) and PMLE (right) in Scenario 1 (upper row), Scenario 2 (middle row) and Scenario 3 (bottom row) for $R = 100$ replicates and $n = 1,000$. The true effects are given by the black lines.

Figure 3b presents the estimated dispersion effects obtained from binomial data. Previous observations with respect to Gaussian data apply as well. Additionally, we can observe a notable local minimum in the dispersion estimates in the vicinity of $x \approx 0.5$. This is even more pronounced in case of smaller sample sizes (not shown). It confirms the observation from Sect. 3, that there is an incentive to compensate a large variance caused by the mean with overdispersion. The variance function of the binomial distribution is given by $V(\mu) = \mu(N - \mu)/N$, i.e. its maximizer is $N/2$. Thus, one can expect this behavior to take place at $x \approx 0.5$ for the mean function depicted in Fig. 1. For the Poisson distribution, the variance function is given by $\mathbb{V}(\mu) = \mu$, and again referring to Fig. 1, we expect the regularization to favour overdispersion around $x \approx 0.55$. This could be the reason for the reduced competitiveness of dropout regularization in the case of count data. The PMLE estimates are less smooth than the dropout estimates in Scenarios 1 and 2 and they oversmooth the true function in Scenario 3.

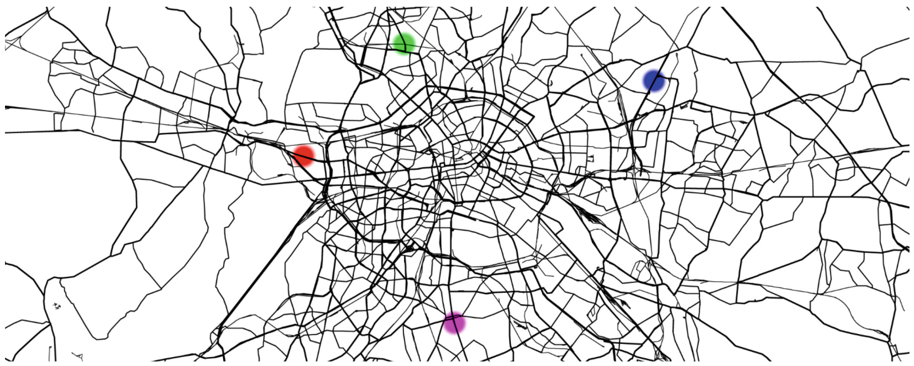
4.2 Traffic Detection Data

Several hundred sensors monitor the traffic along main roads in the German capital Berlin. These sensors provide hourly aggregated data on the number of cars passing the sensors that is publicly available through the Senatsverwaltung für Umwelt, Mobilität, Verbraucher- und Klimaschutz and Verkehrsinformationszentrale Berlin[2]. The Berlin traffic detection data was recently analysed in a regression context in [13]. Here, we consider data collected during summer of 2019 (June, July, and August) from four distinct locations on the outskirts of the city center (see Fig. 4a), leading to a total of $10,314$ data points. The sensors are located at streets connecting outskirt neighbourhoods with the city center. Hence, as one might expect, the number of cars peaks during rush hour (see Figs. 4b–4e), when people are commuting between home and work making this data set especially suitable for an analysis with a dropout DEF-GLM.

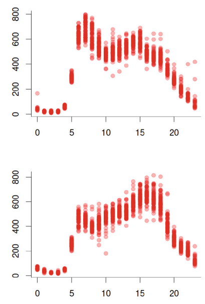
Denote the number of cars passing the sensor at time-point t as y_t. We assume a double Poisson model $y_t \mid t \sim \mathrm{DPoi}\big(f(t), g(t)\big)$, where the functional effects $f(t)$ and $g(t)$ are effects of the day time t in the mean and disperson modelled using cyclic B-spline basis expansions. The dropout rates (p_μ, p_γ and σ_μ, σ_γ for Bernoulli and Gaussian dropout, respectively) are chosen via random search cross-validation by sampling 5,000 times from the sets $\mathcal{H}_{\text{Ber}} = [0, 1]^2$ and $\mathcal{H}_{\mathcal{N}} = [0, 2]^2$ (see Fig. 6 in the Appendix).

Estimates based on the optimal dropout rates are shown in Fig. 5. We find a similar behaviour across all four sensors. The mean effects are bimodal with one peak in the morning and a second peak in the evening. For the inbound traffic the peaks in the morning are more pronounced than the peaks in the afternoon and the outbound traffic mirrors this behaviour having a stronger peak in the evening. This corresponds well with an expected rush hour effect as people drive into the city for work in the morning and come back in the evening. These findings indicate that the dropout estimators are well suited to capture the peaks during morning and evening rush hours for inbound and outbound traffic respectively. Moreover, time-dependent overdispersion ($\gamma < 1$) with narrow peaks in the early morning justifies the use of the extended GLM. The mean estimates obtained from Bernoulli noise are virtually indistinguishable from the ones based

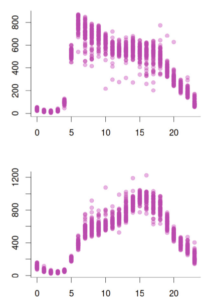
[2] https://viz.berlin.de.



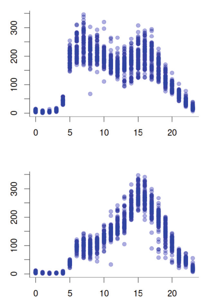
(a) Positioning of sensors



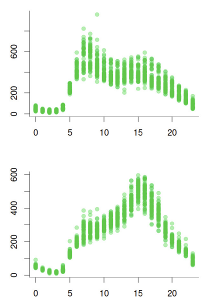
(b) West



(c) South



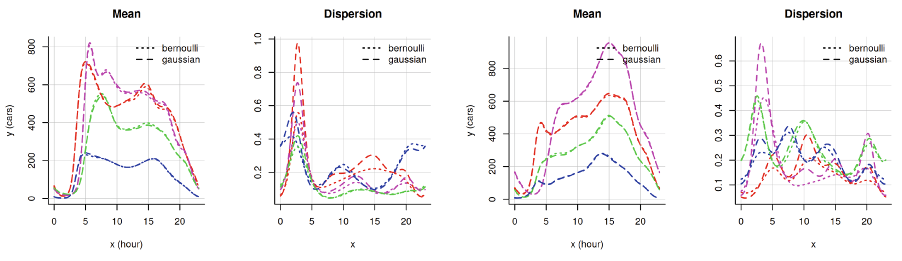
(d) East



(e) North

Fig. 4. Traffic detection data with the (a) positioning of the four sensors in the (b) West, (c) South, (d) East and (e) North of the Berlin city center. For (b)–(e) the upper panels depict the counts (y-axis) for inbound traffic for each hour from 0am (=0) to 11pm (=23) (x-axis). The bottom panels show the corresponding outbound traffic.

on Gaussian noise, which is most likely due to the small degree of regularization in the mean across all data sets (see Fig. 6 in the Appendix). Differences are more pronounced between the dispersion estimates. However this is to be expected, since the dispersion is more difficult to estimate.



(a) Inbound (b) Outbound

Fig. 5. Cross-validated estimates for the traffic detection data of the four sensors in the (●) West, (●) South, (●) East and (●) North of the Berlin city center. (Color figure online)

5 Conclusion

We studied dropout regularization in the context of flexible GLMs based on the DEF. Our theoretical analysis shows that dropout favors rare but important features in the mean and dispersion parameters, and overdispersion relative to underdispersion. Overdispersion alleviates the penalization on the mean provided the dispersion can be modelled by rare but important features itself. These findings are further justified by an empirical application to adaptive smoothing with B-splines and an application to real data on traffic detection in Berlin. Our experiments confirm that dropout regularization outperforms PMLE under ideal conditions. Deviations from the ideal scenario, such as the presence of underdispersion or a mean or dispersion function which cannot be modelled by rare but important features, leads to a decrease in performance of dropout regularization relative to PMLE. Thus, our findings extend previous work on dropout regularization for GLMs and add to the theoretical understanding of dropout methods in general.

However, the presented results can surely be broadened in a number of ways. Among them could be an extension of our findings to generalized additive models and quasi-likelihood estimation. Considering a greater number of real-world datasets certainly presents a challenge, given that the requirements on such data are quite particular, but it would unquestionably generalize our work. Lastly, the interpretable approximations used in our work to gain understanding of dropout regularization in extended GLMs are useful for predicting what we observe in the numerical experiments. Yet it must be acknowledged that we use approximations, and that there may be some situations where these approximations are inadequate. It could be interesting to investigate this more systematically in future work.

Acknowledgments. Nadja Klein acknowledges support through the Emmy Noether grant KL 3037/1-1 of the German research foundation (DFG).

References

1. Baldi, P., Sadowski, P.J.: Understanding dropout. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
2. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**, 281–305 (2012)
3. Bishop, C.M.: Training with noise is equivalent to Tikhonov regularization. Neural Comput. **7**(1), 108–116 (1995)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
5. Burges, C.J.C., Schölkopf, B.: Improving the accuracy and speed of support vector machines. In: Advances in Neural Information Processing Systems, vol. 9 (1996)
6. Dunn, P.K., Smyth, G.K.: Generalized Linear Models With Examples in R. Springer Texts in Statistics, 1st edn. Springer, New York (2018). https://doi.org/10.1007/978-1-4419-0118-7
7. Efron, B.: Double exponential families and their use in generalized linear regression. J. Am. Stat. Assoc. **81**(395), 709–721 (1986)
8. Eilers, P.H.C., Marx, B.D.: Flexible smoothing with b-splines and penalties. Stat. Sci. **11**(2), 89–121 (1996)
9. Gijbels, I., Prosdocimi, I., Claeskens, G.: Nonparametric estimation of mean and dispersion functions in extended generalized linear models. TEST **19**(3), 580–608 (2010)
10. Hannah, L.A., Blei, D.M., Powell, W.B.: Dirichlet process mixtures of generalized linear models. J. Mach. Learn. Res. **12**(6) (2011)
11. Helmbold, D.P., Long, P.M.: On the inductive bias of dropout. J. Mach. Learn. Res. **16**(105), 3403–3454 (2015)
12. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv: 1207.0580 (2012)
13. Kock, L., Klein, N.: Truly multivariate structured additive distributional regression. arXiv:2306.02711 (2023)
14. Lee, Y., Nelder, J.A.: The relationship between double-exponential families and extended quasi-likelihood families, with application to modelling Geissler's human sex ratio data. J. Royal Stat. Soc. Ser. C **49**(3), 413–419 (2000)
15. Lehmann, E.L., Casella, G.: Theory of Point Estimation. Springer Texts in Statistics, 2nd edn. Springer, New York (1998). https://doi.org/10.1007/b98854
16. Maaten, L., Chen, M., Tyree, S., Weinberger, K.: Learning with marginalized corrupted features. In: Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, vol. 28, pp. 410–418 (2013)
17. McCullagh, P., Nelder, J.A.: Generalized Linear Models. No. 37 in Monographs on Statistics and Applied Probability, 2nd edn. Chapman & Hall/CRC, London (1998)
18. Merity, S., Keskar, N.S., Socher, R.: Regularizing and optimizing LSTM language models. arXiv:1708.02182 (2017)
19. Shao, J.: Mathematical Statistics. Springer Texts in Statistics, 2nd edn. Springer, New York (2003). https://doi.org/10.1007/b97553
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(56), 1929–1958 (2014)

21. Tan, M., Le, Q.: EfficientNet: rethinking model scaling for convolutional neural networks. In: Proceedings of the 36th International Conference on Machine Learning, vol. 97, pp. 6105–6114 (2019)
22. Van Erven, T., Kotłowski, W., Warmuth, M.K.: Follow the leader with dropout perturbations. In: Proceedings of The 27th Conference on Learning Theory, Barcelona, Spain, vol. 35, pp. 949–974 (2014)
23. Wager, S., Wang, S., Liang, P.S.: Dropout training as adaptive regularization. In: Advances in Neural Information Processing Systems, vol. 26 (2013)
24. Wang, S., Manning, C.: Fast dropout training. In: Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, vol. 28, pp. 118–126 (2013)
25. Wei, C., Kakade, S., Ma, T.: The implicit and explicit regularization effects of dropout. In: Proceedings of the 37th International Conference on Machine Learning, vol. 119, pp. 10181–10192 (2020)
26. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv:1212.5701 (2012)

GLADformer: A Mixed Perspective for Graph-Level Anomaly Detection

Fan Xu[2], Nan Wang[1](✉), Hao Wu[2], Xuezhi Wen[1], Dalin Zhang[1], Siyang Lu[1], Binyong Li[3], Wei Gong[2], Hai Wan[4], and Xibin Zhao[4](✉)

[1] School of Software Engineering, Beijing Jiaotong University, Beijing, China
{wangnanbjtu,22126399,dalin,9450}@bjtu.edu.cn

[2] University of Science and Technology of China, Hefei, China
{markxu,wuhao2022}@mail.ustc.edu.cn, weigong@ustc.edu.cn

[3] Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu, China
lby@cuit.edu.cn

[4] BNRist, KLISS, School of Software, Tsinghua University, Beijing, China
{wanhai,zxb}@tsinghua.edu.cn

Abstract. Graph-Level Anomaly Detection (GLAD) aims to distinguish anomalous graphs within a graph data set. However, current methods are constrained by their receptive fields, struggling to learn global features within the graphs. Moreover, most these methods are based on spatial domain and lack exploration of spectral characteristics. In this paper, we propose a multi-perspective hybrid graph-level anomaly detector named GLADformer, consisting of two key modules. Specifically, we first design a Graph Transformer module with global spectrum enhancement, which ensures balanced and resilient parameter distributions by fusing global features and spectral distribution characteristics. Furthermore, to explore local anomalous attributes, we customize a band-pass spectral GNN message passing module that enhances the model's generalization capability. Through comprehensive experiments on ten real-world datasets from multiple domains, we validate the effectiveness and robustness of GLADformer. This demonstrates that GLADformer outperforms current state-of-the-art graph-level anomaly detection methods, particularly in effectively capturing global anomaly representations and spectral characteristics.

Keywords: Graph-level Anomaly Detection · Graph Transformer · Spectral Graph Neural Network

1 Introduction

Graphs are profoundly employed to model the intricate relationships between data instances across various domains, spanning bioinformatics [48], chemistry [29], transportation [13], and social networks [13], *etc.* Among the downstream tasks for graph data, graph-level anomaly detection [50] emerges as a

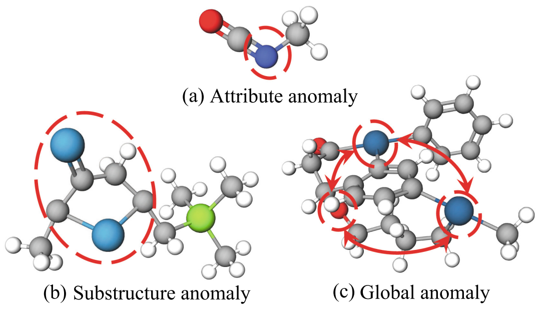


Fig. 1. Examples for different categories of graph-level anomalies.

significant challenge and encompasses a wide array of application scenarios, such as cancer drug discovery [10] and the identification of toxic molecules [35].

In recent years, Graph Neural Networks (GNNs) have achieved significant advancements in graph representation learning, like node classification [18], link prediction [5], and graph classification [41], *etc.* Specifically, GNNs encode intricate structure and attribute information of graphs into vectors for learning within the latent representation space. To date, a multitude of GNN-based models have been proposed for anomaly detection in graph-structured data [9], while they predominantly focus on detecting anomalous nodes or edges [40] within a single large graph. In contrast, the domain of graph-level anomaly detection is yet to be extensively explored.

Anomalous graphs manifest as outliers and may arise in various scenarios, including local attribute anomalies, substructure anomalies, global interaction anomalies [28,32], *etc.* Taking biochemistry molecular data [31] as an example, Fig. 1 (a) illustrates the toxic molecule of Methyl Isocyanate (CH_3NCO), where the nitrogen atom serves as a critical toxic factor. Figure 1 (b) depicts a type of muscarinic molecule, featuring a cyclic peptide structure that distinguishes it from other compounds. Figure 1 (c) displays the main active ingredient in Nux vomica, which is Strychnine ($C_{21}H_{22}N_2O_2$), a toxic ketone alkaloid. However, as Strychnine lacks distinctive elements or specific substructures, researchers need to discern it from the complete molecular structure. Therefore, for a more comprehensive identification of various anomalous graphs, it is imperative to design models that consider both local attributes and structural information while incorporating global interactions within the graph.

The mainstream GLAD models primarily employs GNNs to jointly encode node attributes and topological structural features for acquiring node representations [50]. Additionally, they design various graph pooling functions [22] to generate graph-level representations for identification of anomalous graphs. However, these methods still face several challenges. On the one hand, traditional GNNs exhibit restricted receptive fields [21], focusing solely on local neighbors or subgraph information of the current node, thereby lacking the ability to capture long-distance information interactions and global features. Although some methods attempt to learn intra-graph and inter-graph knowledge by maintaining a repository of anomalous node or graph candidates [28], they still fall short in capturing the global information within the graphs. On the other hand, when learning local features, spatial-domain GNNs tend to overlook essential high-frequency information and underlying semantic details due to their low-pass filtering characteristics [4]. Moreover, the current graph-level pooling mechanisms exhibit constrained generalization capabilities, leading to erratic performances across disparate datasets.

To address the aforementioned challenges, we propose a novel graph-level anomaly detection model, which combines spectral-enhanced global perception and local multi-frequency information guidance. Our method, GLADformer, primarily consists of two key modules: the Spectrum-Enhanced Graph Transformer module and the Local Spectral Message Passing module. Specifically, we first introduce a Graph Transformer operator in the spatial domain, where effective graph-induced biases such as node degrees and structural information are jointly inputted, incorporating explicit spectral distribution deviations to capture anomaly information from a global perspective. Subsequently, to better explore local features and mitigate the limitations of spatial GNNs, we design a novel wavelet spectral GNN to learn discriminative local attributes and structural features in a complex spectral domain. Finally, to overcome class imbalance issues and excessive confidence of traditional cross-entropy loss, we propose an improved variation-optimized cross-entropy loss function. Our main contributions can be summarized as follows:

- Our approach not only incorporates a Graph Transformer module designed in the spatial domain but also integrates spectral energy distribution deviations to enhance global perception. Additionally, we design a spectral GNN with multi-frequency message passing characteristics to guide the extraction of local anomaly features.
- To better alleviate the issue of class imbalance and overcome the limitations of using cross-entropy as a measure for anomaly detdection, we propose a weighted variation-optimized cross-entropy loss function.
- Comprehensive experiments on a variety of datasets across ten baselines demonstrate that GALDformer exhibits competitive performance in terms of both effectiveness and robustness.

2 Related Work

2.1 Graph-Level Anomaly Detection

In recent years, graph anomaly detection has garnered extensive attention across various domains. However, most existing approaches focus on detecting anomalous nodes or edges within an individual graph [44,45], while graph-level anomaly detection remains largely unexplored.

GLAD aims to differentiate deviant structures or abnormal properties within a single graph to identify anomalous graphs that exhibit substantial differences compared to the majority in a collection. State-of-the-art end-to-end methods leverage powerful GNN backbones and incorporate advanced strategies to learn graph representations suitable for anomaly detection. For instance, GLocalKD [27] and OCGTL [33] respectively combine GNN with knowledge distillation and one-class classification to detect anomalous graphs. iGAD [50] introduces an abnormal substructure-aware deep random walk kernel and a node-aware kernel to capture both topology information and node features. To better explore inter-graph information, GmapAD [28] maps individual graphs to a representation space by computing similarity between graphs and inter-graph candidate nodes, achieving high discriminability between abnormal and normal graphs. Some recent works focus on interpretable analysis of graph-level anomaly detection and have achieved promising results. For instance, SIGNET [24] measures the anomaly degree of each graph based on cross-view mutual information [11] and extracts bottleneck subgraphs in a self-supervised manner to provide explanations for anomaly discrimination. However, existing methods mostly originate from a spatial perspective, and there is still a lack of exploration regarding the influence of graph-level spectrum energy information.

2.2 Graph Transformer

Transformer [38,42] has achieved overwhelming advantages in the NLP [6,43] and CV [8,25] domains, and recently many researchers have been devoted to extending Transformer to the study of graph-structured data [49]. One of the strengths of Transformer is its ability to capture global receptive fields, but it lacks the capability to capture positional information, which poses significant limitations in graph data [37]. Recently, researchers investigate the use of Position Encoding (PE) and Structure Encoding (SE) [23,51] within the graph domain to capture various types of graph structure features, leveraging techniques such as shortest path proximity [26] or spectral information to enhance inductive bias. For example, Graphformer [49] designs novel structural position encoding that outperforms popular GNN models in a wide range of graph prediction tasks. Further, SAN [19] employs both sparse and global attention mechanisms at each layer and introduces learnable Position Encoding (LapPE) to replace static Laplacian eigenvectors. Exphormer [34] explores sparse attention mechanisms with virtual global nodes and extended graphs, showcasing linear complexity and desirable theoretical properties.

3 Preliminary

3.1 Problem Definition

Given a graph set $\hat{\mathcal{G}} = \{G_1, G_2, ..., G_N\}$, each graph is denoted as $G_i = (\mathcal{V}_i, \mathcal{E}_i)$, where $\mathcal{V}_i$ is the set of nodes and $\mathcal{E}_i$ is the set of edges. The node features can be represented as $X_i \in R^{N_i \times d}$, and the edge information can be denoted as an adjacency matrix $A_i \in [0,1]^{N_i \times N_i}$. In this paper, we concentrate on supervised graph-level anomaly detection, thus we aim to learn an anomaly labeling function based on the given training graphs and their graph labels $y_i = \{0, 1\}$. Then we hope to assign a high anomaly score to a graph G in the testing set if it significantly deviates from the majority.

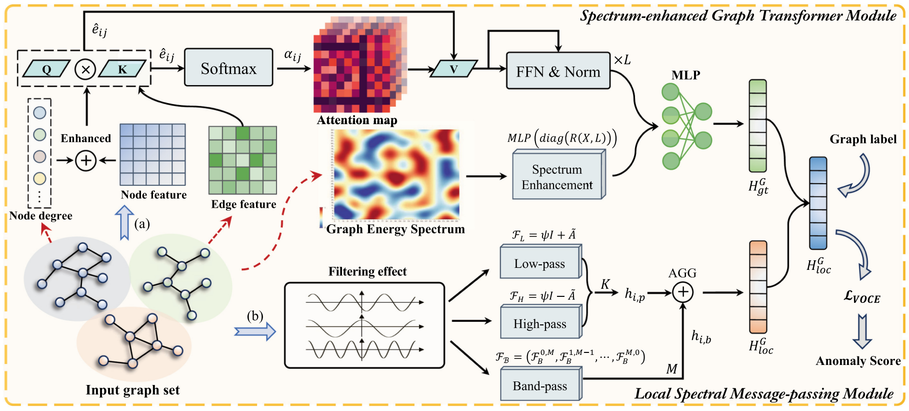


Fig. 2. The above image presents the overview of our model GLADformer, where (a) and (b) demonstrate the the spectrum-enhanced graph-transformer module and the local spectral message passing module respectively.

3.2 Graph Spectrum and Rayleigh Quotient

For each graph G in the graph set, the adjacency matrix is denoted by A. The diagonal degree matrix D is defined as $(D)_{ii} = \sum_j (A)_{ij}$, and the normalized laplacian matrix L is defined as $I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$. The laplacian L can be decomposed into its eigenvectors and eigenvalues as $U \Lambda U^T$. The diagonal elements of Λ are composed of its eigenvalues: $\Lambda = \text{diag}([\lambda_1, \lambda_2, \ldots, \lambda_N])$, and the eigenvalues satisfy $0 \leq \lambda_1 \leq \cdots \leq \lambda_N \leq 2$. Let $X = (x_1, x_2, ..., x_N)$ be a signal on graph G, then $U^T X$ is the graph Fourier transformation of X. Then we introduce Rayleigh quotient to demonstrate the standardized variance fraction of signal X.

$$R(L, X) = \frac{X^T L X}{X^T X} = \frac{\sum_{i=1}^{N} \lambda_i \tilde{x}_i^2}{\sum_{i=1}^{N} \tilde{x}_i^2} = \lambda_N * f(t)_{max} - S_{spec}, \tag{1}$$

where S_{spec} denotes the the integration of signal energy in the frequency domain (with respect to eigenvalues) [36]. Thus Rayleigh quotient can also denote the contribution of high-frequency region.

4 Method

In this section, we will systematically describe the technical details of the GLADformer framework (view Fig. 2 for demonstration), which consists of three core components. Specifically, we first detail the design of our spectrum-enhanced of Graph Transformer from a global perspective (Sect. 4.1). Subsequently, we explore the discriminative local attribute and structure features through a multi-frequency spectral GNN (Sect. 4.2). Finally, we present the meticulously designed variation-optimize cross-entropy loss function (Sect. 4.3).

4.1 Spectrum-Enhanced Graph Transformer Module

Initially, we establish a super-node in each graph, which forms connections with all remaining nodes within that graph. Further, to address the issue of the Transformer architecture's lack of strong inductive biases and to leverage the advantages of message passing strategy in GNNs, we incorporate appropriate graph structures and relative position encoding. We first employ a single-layer MLP to obtain the initial node representations.

$$h_i = \phi(Wx_i + b). \tag{2}$$

Subsequently, we enhance the node features by introducing an adaptive degree-scaler to maintain the degree information.

$$\mathrm{g}_i = h_i \cdot \theta_1 + (\log(1 + deg_i) \cdot h_i \cdot \theta_2), \tag{3}$$

where $\theta_1, \theta_2 \in R^d$ are learnable weights, and deg_i is the degree of node i. Then we utilize Relative Random Walk Probabilities (RRWP) [16] to initialize the relative position encoding, thus capturing intriguing graph structure information. According to [26], RRWP has been proved to be more expressive than shortest path distances (SPD) [1] through recently proposed Weisfeiler-Leman-like graph isomorphism tests [30]. Let A, D be the adjacency matrix and degree matrix of a graph, we can define $R := AD^{-1}$, and R_{ij} denotes the possibility of node i transitioning to node j in one step.

Then the RRWP initial positional encoding for each pair of nodes is illustrated as $R_{i,j} = [I, R, R^2, \ldots, R^{T-1}]_{i,j} \in R^T$. To make full use of the structural information, we utilize the position encoding R_{ij} as the edge feature e_{ij}. Further, we cleverly combine Transformer's self-attention mechanism and customized edge features to design a novel Graph Transformer architecture. Specifically, in the stage of calculating attention score, we replace the scalar product of Query and Key with the vector product [47], and inject the initialized edge features to obtain learnable edge features [12]. This learnable edge feature is subsequently utilized in the calculation of attention score and output encoding. The specific formula is as follows:

$$\begin{aligned}
\hat{e}_{ij} &= \sigma((\mathrm{g}_i W_Q \otimes \mathrm{g}_j W_K) + e_{ij} W_E), \\
\alpha_{ij} &= Softmax(\hat{e}_{ij} \cdot W_\alpha), \\
\hat{\mathrm{g}}_i &= \sum_{j \in V} \alpha_{ij} \left(\mathrm{g}_j W_V + \hat{e}_{ij}\right),
\end{aligned} \tag{4}$$

where $W_Q, W_K, W_V \in R^{d \times d'}$, $W_E \in R^{K \times d'}$ and $W_\alpha \in R^{d' \times 1}$ are learnable weight matrices; and $\otimes$ denotes element-wise multiplication of vectors.

Similar to other Transformer architectures, we can easily convert the aforementioned attention mechanism into multi-head attention. Then we follow [26] to obtain the output of the l-th layer, denoted as $\mathcal{G}^{(l)}$:

$$\begin{aligned}\mathcal{G}'^{(l)} &= \text{Norm}(\text{RRWP-MHA}(\mathcal{G}^{(l-1)}) + \mathcal{G}^{(l-1)}),\\ \mathcal{G}^{(l)} &= \text{Norm}(\text{FFN}(\mathcal{G}'^{(l)})) + \mathcal{G}'^{(l)},\end{aligned} \tag{5}$$

where Norm$(\cdot)$ indicates layer-norm function, RRWP-MHA is the multi-head attention mechanism designed before, and FFN [38] is a feed-forward network. Afterward, we merge the encoding representations from each layer of the super-node to obtain the graph-level representations.

$$H^G_{sup} = f_{\text{COMBINE}}\left(\mathcal{G}^{(l)}_{sup} \mid l = 1, ..., L\right). \tag{6}$$

Spectrum Enhancement. Subsequently, as discussed in Sect. 3.2, we have already known the positive correlation between Rayleigh quotient and high-frequency components. According to [7], the spectral energy distributions of normal and anomalous graphs are distinct in the graph spectral domain. Therefore, it is crucial to incorporate spectral energy distributions into the GLAD task. However, accurately calculating the ratios of spectral energies requires the eigenvalue-eigenvector decomposition of the graph Laplacian matrix, which has a complexity of $O(n^3)$ and is computationally expensive for large-scale graphs. To avoid high time complexity, we employ the Rayleigh quotient $R(L, X) = \frac{X^T LX}{X^T X}$ as a substitute. For signal $X \in R^{N\times d}$, it can be proven that the diagonal elements of the Rayleigh quotient $R(L, X)$ correspond to the spectral characteristics of the signal X in each feature dimension on the graph.

Given the potential sparsity of graph adjacency matrices and graph signals, as well as the possibility of high discretization levels, we utilize a single-layer MLP to derive the Rayleigh quotient vector for each graph. This vector denotes the explicit Rayleigh quotient feature and it primarily focuses on the global spectral distribution characteristics of the graph. Then, we utilize this to enhance the previously obtained graph-level representation.

$$\begin{aligned}H^G_{rq} &= \text{MLP}\left(diag\left(R(X, L)\right)\right),\\ H^G_{gt} &= \text{MLP}\left(\text{CONCAT}\left(H^G_{sup}, H^G_{rq}\right)\right).\end{aligned} \tag{7}$$

4.2 Local Spectral Message-Passing Module

To devise an adaptive local message passing filter, we introduce the Beta-distributed wavelet basis [36], which conforms to the Hammond's graph wavelet theory [3] and is band-pass in nature. The underlying probability distribution function is $f_{Beta}(w) = w^\alpha(1 - w)^\beta/B(\alpha + 1, \beta + 1)$, where $w \in [0, 1]$, and $B(\alpha + 1, \beta + 1) = \alpha!\beta!/(\alpha + \beta + 1)!$ is a constant. According to [36], since the eigenvalues of the normalized Laplacian matrix satisfy $\lambda \in [0, 2]$, we utilize a variation like $f^*_{Beta}(\lambda) = \frac{1}{2} f_{Beta}(\frac{\lambda}{2})$. The obtained band-pass filter is expressed below:

$$\mathcal{F}^{\alpha,\beta}_B = U f^*_{Beta}(\Lambda)U^T = \frac{(\frac{L}{2})^\alpha(I - \frac{L}{2})^\beta}{2B(\alpha + 1, \beta + 1)}. \tag{8}$$

Mapping this band-pass wavelet kernel into the graph spectral domain, $M = \alpha + \beta$ denotes the neighboring receptive field order of this filter. In turn, we obtain the spectral domain Beta wavelet transform group $\mathcal{F}_{\mathcal{B}}$:

$$\mathcal{F}_{\mathcal{B}} = (\mathcal{F}_B^{0,M}, \mathcal{F}_B^{1,M-1}, \cdots, \mathcal{F}_B^{M,0}). \tag{9}$$

Contrasting with conventional graph neural network message passing mechanisms, this work employs parallel wavelet kernels for message propagation. Subsequently, it combines the corresponding filtered information.

$$h_{i,b} = f_{\text{COMBINE}}\left(\mathcal{F}_B^{m,M-m} \cdot h_i^{(0)} \mid m = 1, ..., M\right), \tag{10}$$

where $h_{i,b}$ denotes the node representation of v_i after band-pass filtering.

Upon meticulous analysis, it is ascertained that each constituent of the Beta band-pass wavelet kernel manifests as an amalgam of elevated powers of the adjacency matrix A and the Laplacian matrix L. This architecture conspicuously lacks specialized low-pass and high-pass filtering kernels imperative for mitigating lower and higher frequencies, potentially precipitating the omission of pivotal information. To rectify this shortfall, we propose the adoption of the generalized Laplacian low-pass and high-pass filters:

$$\begin{aligned} \mathcal{F}_L &= (\psi + 1)I - L = U[(\psi + 1)I - \Lambda]U^\top, \\ \mathcal{F}_H &= (\psi - 1)I + L = U[(\psi - 1)I + \Lambda]U^\top, \end{aligned} \tag{11}$$

where $\psi \in [0, 1]$ plays a pivotal role in modulating the characteristics of both low-pass and high-pass filter kernels. Subsequently, the dual filter kernels are concurrently utilized to discern and assimilate the pure low-frequency and high-frequency signals aggregating from neighbors, thereby enhancing the model's fidelity to local structures.

$$\begin{aligned} h_{i,p}^{(l)} &= \text{MLP}\left(\text{AGG}\left(\mathcal{F}_L \cdot h_i^{(l-1)}, \mathcal{F}_H \cdot h_i^{(l-1)}\right)\right), \\ h_{i,p} &= f_{\text{COMBINE}}\left(h_{i,p}^{(l)} \mid l = 1, ..., K\right). \end{aligned} \tag{12}$$

The features from each layer are sequentially concatenated and subsequently amalgamated to obtain the corresponding aggregated low-order neighbor attributes. Ultimately, the feature is amalgamated with the band-pass filtering outcomes, procuring the definitive local graph-level representation.

$$H_{loc}^G = \text{READOUT}\left(\text{AGG}\left(h_{i,b}, h_{i,p}\right) \mid v_i \in G\right), \tag{13}$$

where READOUT function can be achieved by permutation invariant graph pooling functions like summation or mean [28]. Finally, we fuse the features obtained from two modules to obtain the final graph-level representation:

$$H^G = \text{MLP}\left(\text{CONCAT}\left(H_{gt}^G, H_{loc}^G\right)\right). \tag{14}$$

4.3 Variation-Optimize Cross-Entropy Loss Function

The cross-entropy loss is widely used in various classification tasks and is often employed as the loss function for designing anomaly detection. However, cross-entropy suffers from phenomena such as overconfidence and struggles to adapt to scenarios like data imbalance. It is well known that the conventional evaluation metric for classification problems is accuracy, but cross-entropy is not a smooth approximation of accuracy, which significantly impacts the precision of model predictions. For instance, when the predicted probability of training samples is very low, cross-entropy tends to yield a tremendously large loss, even though these data points are likely to be noises. This phenomenon is particularly pronounced in anomaly detection data, leading to overfitting of the model to noise data. To address these issues and cater to the graph anomaly detection scenarios, we propose a novel cross-entropy loss function.

Approaching from a gradient-based perspective, if accuracy is employed as the evaluation metric, the gradient with respect to $p_\theta(y|x)$ is $-\nabla_\theta p_\theta(y|x)$. Conversely, for the cross-entropy $-\log p_\theta(y|x)$, the gradient is $-\frac{1}{p_\theta(y|x)}\nabla_\theta p_\theta(y|x)$ (y_{true} is not involved in the gradient computation and has been omitted). Now, we construct a new gradient based on these two equations:

$$-\frac{1}{\kappa+(1-\kappa)p_\theta(y|x)} \quad \kappa \in [0,1]\,. \tag{15}$$

This new gradient retains the advantages of cross-entropy while synchronizing better with changes in accuracy, thus overcoming the problem of overfitting. Subsequently, we seek the original function based on this differential gradient.

$$-\frac{\nabla_\theta p_\theta(y|x)}{\kappa+(1-\kappa)p_\theta(y|x)} = \nabla_\theta\left(-\frac{\log\left[\kappa+(1-\kappa)p_\theta(y|x)\right]}{1-\kappa}\right). \tag{16}$$

By incorporating the original function into the GLAD task, we can obtain the variation-optimize cross-entropy loss function $\mathcal{L}_{VOCE}$.

$$\mathcal{L}_{VOCE} = -\partial y\frac{\log\left[\kappa+(1-\kappa)p\right]}{1-\kappa} - (1-y)\frac{\log\left[\kappa+(1-\kappa)(1-p)\right]}{1-\kappa}, \tag{17}$$

where the hyperparameter κ is utilized to modulate the balance between accuracy and cross-entropy within the loss function (κ is set to 0.2 in experiments), and ∂ is logarithm of the ratio between normal and anomalous samples.

5 Experiment

In this section, we perform thorough experiments to validate the effectiveness of our proposed GLADformer method against nine baselines on ten real-world datasets. Furthermore, we conduct comprehensive ablation tests on GLADformer and perform visual analysis of key components.

5.1 Experiment Settings

Datasets. We conduct experiments on 10 real-world graph datasets from two popular application domains: (i) BZR, AIDS, COX2, NCI1, ENZYMES, and PROTEINS are collected from biochemistry [31], samples from minority or truly abnormal classes are considered as anomalies. Following [28], the selected anomaly samples are downsampled, retaining only 10% of the samples. (ii) MCF-7, MOLT-4, SW-620, and PC-3 are collected from PubChem [17], and they reflect the anti-cancer activity test results of a large number of compounds on cancer cell lines. Chemical compounds that exhibit antibody activity against cancer are labeled as abnormal graphs, while others are labeled as normal graphs. The statistics of these graph datasets are summarized in Table 1.

Table 1. The statistics of the 10 datasets.

Dataset	$N.G$	$N.A$	$Ratio\%$	$AVG.n$	$AVG.e$	$Attr$
AIDS	2000	400	20.00	15.69	16.20	4
BZR	405	86	21.23	35.75	38.36	3
COX2	467	102	21.84	41.22	43.45	3
NCI1	4110	2053	49.95	29.87	32.30	37
ENZYMES	600	100	16.67	32.63	62.14	18
PROTEINS	1113	450	40.43	39.06	72.82	29
MCF-7	25476	2294	8.26	26.39	28.52	46
MOLT-4	36625	3140	7.90	26.07	28.13	64
SW-620	38122	2410	5.95	26.05	28.08	65
PC-3	25941	1568	5.70	26.35	28.49	45

Comparison Method. To illustrate the effectiveness of our proposed model, we conduct extensive experiments between GLADformer and 9 competitive methods, which can be classified into three groups. (i) Spatial GNNs with average pooling function: GCN [18], GAT [39] and GIN [46]. (ii) Graph classification methods: SAGPool [20] and GMT [2]. (iii) State-of-the-art deep GLAD methods: GLocalKD [27], iGAD [50], HimNet [32], and GmapAD [28].

Evaluation Metrics. We evaluate methods using popular anomaly detection metrics, AUC values and Macro-F1 scores [15]. The results are reported by performing 5-fold cross-validation for all datasets.

Experimental Settings. For our model, the dimensions of the hidden layers and output features for all three modules are set to 128 and 32 respectively. During training, the hyperparameters T, L, M, K are set to 4, 6, 3 and 4 separately, and the batch size is 128 for all datasets. We use Adam as the optimizer with a learning rate of 0.001 and utilize cross-entropy loss as the basic loss function. Each dataset is randomly shuffled and split for training, validation, and testing with ratios of 70%, 15%, and 15%. For GCN, GAT and GIN, we utilize 2 layers and apply average pooling function to obtain graph-level representations. For other baselines, we use their published settings unless the parameters are specially identified in the original paper. All experiments in this work are conducted on an NVIDIA A100-PCIE-40GB.

Table 2. The performance comparison in terms of AUC value (in percent, mean value). Best result in bold, second best underlined.

Datasets	GCN	GAT	GIN	SAGPool	GMT	GLocalKD	iGAD	HimNet	GmapAD	Ours
BZR	55.87	57.37	58.92	61.78	65.91	67.95	69.37	70.38	<u>72.58</u>	**77.25**
AIDS	86.37	88.29	87.62	90.48	92.16	93.24	97.62	98.71	<u>99.06</u>	**99.62**
COX2	50.21	52.77	52.08	54.38	53.58	58.93	61.48	<u>63.76</u>	62.73	**68.47**
ENZYMES	53.94	55.61	54.75	56.90	58.75	<u>63.27</u>	60.92	58.94	57.62	**63.86**
PROTEINS	68.70	69.53	70.05	72.58	74.94	76.41	75.93	77.28	**78.35**	<u>77.68</u>
NCI1	62.34	64.28	63.98	66.32	<u>71.98</u>	68.38	70.45	68.63	71.52	**76.83**
MCF-7	64.48	65.26	65.62	71.64	77.06	63.63	<u>81.46</u>	63.69	71.28	**83.58**
PC-3	66.97	67.36	67.95	69.37	78.96	67.27	**85.63**	67.03	74.26	<u>84.19</u>
MOLT-4	63.74	65.21	64.82	65.11	76.06	66.31	<u>82.79</u>	66.33	69.82	**83.24**
SW-620	59.87	62.31	61.30	72.51	74.67	65.42	<u>84.86</u>	65.44	72.97	**85.73**

Table 3. The performance comparison in terms of F1 score (in percent, mean value). Best result in bold, second best underlined.

Datasets	GCN	GAT	GIN	SAGPool	GMT	GLocalKD	iGAD	HimNet	GmapAD	Ours
BZR	51.27	52.31	52.97	55.93	54.35	56.12	56.74	58.14	<u>59.82</u>	**61.87**
AIDS	68.10	70.83	71.64	74.28	74.32	76.63	77.93	78.92	<u>80.17</u>	**82.47**
COX2	43.28	43.75	45.15	47.33	48.85	50.82	51.35	<u>52.18</u>	51.38	**57.05**
ENZYMES	44.73	43.95	44.16	47.18	46.94	<u>50.23</u>	48.02	48.85	48.53	**53.28**
PROTEINS	49.82	49.26	50.08	55.74	58.75	61.03	60.61	52.47	**62.84**	<u>61.74</u>
NCI1	51.17	50.84	51.20	59.03	<u>61.27</u>	57.92	58.09	56.93	58.29	**64.29**
MCF-7	48.83	47.97	48.05	58.83	62.12	60.31	<u>64.68</u>	57.58	60.47	**65.84**
PC-3	48.79	47.98	48.56	59.65	<u>63.87</u>	59.93	67.10	61.25	63.31	**67.37**
MOLT-4	49.87	50.32	50.02	59.37	62.07	61.74	<u>66.81</u>	60.53	62.84	**68.06**
SW-620	48.65	48.84	48.74	58.60	61.28	60.86	<u>66.23</u>	59.46	61.73	**68.58**

5.2 Main Results

We first compare GLADformer with the aforementioned baselines of different categories. The overall performances of all methods with respect to AUC value and F1 score against ten datasets are illustrated in Table 2 and Table 3. Among all the ten datasets, GLADformer achieves the highest AUC values in eight and the highest F1 scores in nine. Specifically, we have the following observations:

- When compared with the spatial GNNs combined with average graph pooling function, GLADformer demonstrates significant advantages on ten real world datasets in terms of AUC value and F1 score. The performance of conventional GCN, GAT, and GIN models falls short of expectations, this may be because they can only capture local low-frequency features, and the average pooling function further exacerbates the over-smoothing issue.
- Subsequently, when compared with the two graph classification methods, SGAPool and GMT obtain some performance improvement compared to traditional methods, which can be attributed to their specially designed pooling strategies. However, they are not specifically tailored for detecting anomalous graphs, and GLADformer explores more comprehensive global and local information and fully considers spectral characteristics, thus there still exists a considerable gap between them and GLADformer.
- Finally, in comparison with the state-of-the-art GLAD methods, our approach achieves superior performance on nearly all datasets in terms of AUC value and F1 score. Specifically, the performance improvement over OCGTL and GLocalKD demonstrates the effectiveness of considering global and local features in intra-graph collaboration and exploring spectral characteristics within and between graphs for graph-level anomaly detection. Moreover, the HimNet and GmapAD methods achieve the modest performance on the first six biochemistry datasets, indicating the effectiveness of maintaining and updating a pool of anomalous nodes or subgraphs on certain datasets with attribute anomalies. It is noteworthy that iGAD achieves commendable performance on the four Pubchem datasets. We hypothesize that this may be because the iGAD method encodes rich local substructure information into the graph-level embeddings, and these four datasets contain a multitude of substructure anomalies. The aforementioned results validate the superiority and effectiveness of GLADformer in graph-level anomaly detection tasks.

5.3 Ablation Study

To evaluate the performance impact of different components on our proposed model GLADformer, we conduct comprehensive ablation study experiments. Specifically, for the two key modules: spectrum-enhance graph-transformer module (**GT**) and local spectral message-passing module (**LS**), we construct a total of five variant models, and the overall results are demonstrated in Table 4. From top to bottom, the five variant models represent the following: (1) Elimination of the GT module, (2) Elimination of the spectral enhancement component within the GT module, (3) Elimination of the LS module, (4) Replacement of the LS module with GIN, and (5) Replacement of the LS module with BernNet [14].

As illustrated in Table 4, we observe that removing either the GT module or the LS module resulted in a significant decline in model performance, with

Table 4. Experimental results of ablation with different model variants on four datasets with respect to AUC value. Best result in bold, second best underlined.

Model	BZR	NCI1	MCF-7	MOLT-4
GLADformer w/o GT	72.56	71.30	77.64	78.47
GLADformer (GT w/o SEC)	<u>76.40</u>	<u>76.18</u>	<u>82.37</u>	81.92
GLADformer w/o LS	73.68	72.20	78.19	78.70
GLADformer w/o LS + GIN	73.73	72.56	78.23	79.18
GLADformer w/o LS + BernNet	75.68	74.79	80.85	<u>82.03</u>
GLADformer	**77.25**	**76.83**	**83.58**	**83.24**

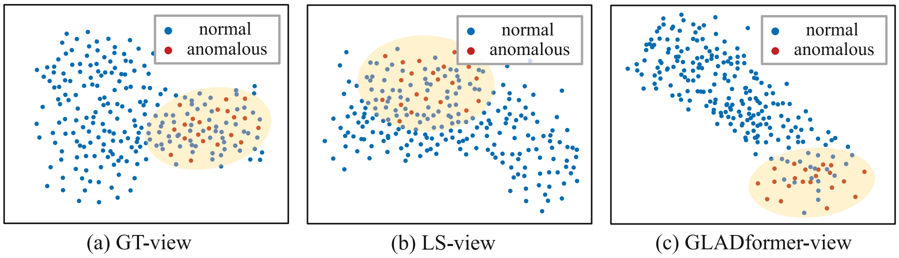


Fig. 3. Visualization analysis on NCI1 dataset.

the performance decrease being more pronounced when the GT module was eliminated. Furthermore, when the spectrum enhancement component (SEC) within the GT module is removed, we notice a slight performance degradation. This indicates that the distribution of spectral energy indeed plays a guiding role in determining the anomalous properties of the graph. Additionally, when replacing the LS module with different local GNNs like GIN and BernNet, we find the model performs better with the use of BernNet, which possesses spectral filtering characteristics. This suggests that the low-pass filtering effect smooths out the local feature, resulting in the losses of local anomaly information.

5.4 Visualization Analysis

To further demonstrate the performance of our model and validate the effectiveness of each module, we conduct extensive visual analysis in this subsection. We firstly load the parameters of the last two models that only retain a single module in the ablation experiment, as well as the complete GLADformer model. Specifically, we employ t-SNE to visualize the embeddings generated by these three models. Due to space constraints, we present visualization results for the NCI1 dataset only in this section, the visual results are presented in Fig. 3.

As observed in Fig. 3, the three figures respectively represent the embeddings output of individual GT module, LS module, as well as the entire GLADformer model, visualized in a two-dimensional space. We observe that each of the first two figures demonstrates a certain level of discriminability between anomalous and normal graphs, verifying the meaningfulness of the two core modules. Then in the third figure, the anomalous graphs are well separated from the normal graphs, and they exhibit distinct distributions. This observation highlights the successful integration of the two modules we designed, further confirming their synergistic effectiveness.

6 Conclusion

In this paper, we rethink the task of graph-level anomaly detection from a multi-perspective view and propose a meticulously designed GLAD model, namely GLADformer. Firstly, we introduce Graph Transformer into the GLAD task to incorporate the inductive bias of graph structures and leverage the differences in spectral energy distribution across graphs. This helps capture global implicit attributes and structural features. Subsequently, departing from the conventional spatial GNNs, we design a novel wavelet spectral GNN for local feature extraction. Further, we propose an improved optimization for the cross-entropy loss function, alleviating the issue of overfitting during training.

Acknowledgement. This work was supported in part by the Supported by the Fundamental Research Funds for the Central Universities under Grant 2024JBM- C031; in part by the CCF-NSFOCUS Open Fund; in part by the NSFC Program under Grant 62202042, Grant U20A6003, Grant 62076146, Grant 62021002, Grant U19A2062, Grant 62127803, Grant U1911401 and Grant 6212780016; in part by Industrial Technology Infrastructure Public Service Platform Project "Public Service Platform for Urban Rail Transit Equipment Signal System Testing and Safety Evaluation" (No. 2022-233-225), Ministry of Industry and Information Technology of China.

References

1. Abraham, I., Filtser, A., Gupta, A., Neiman, O.: Metric embedding via shortest path decompositions. In: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, pp. 952–963 (2018)
2. Baek, J., Kang, M., Hwang, S.J.: Accurate learning of graph representations with graph multiset pooling. In: International Conference on Learning Representations (2021)
3. Bastos, A., Nadgeri, A., Singh, K., Suzumura, T., Singh, M.: Learnable spectral wavelets on dynamic graphs to capture global interactions. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 6779–6787 (2023)
4. Bo, D., Wang, X., Shi, C., Shen, H.: Beyond low-frequency information in graph convolutional networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 3950–3957 (2021)
5. Cai, L., Li, J., Wang, J., Ji, S.: Line graph neural networks for link prediction. IEEE Trans. Pattern Anal. Mach. Intell. **44**(9), 5103–5113 (2021)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)

7. Dong, X., Zhang, X., Wang, S.: Rayleigh quotient graph neural networks for graph-level anomaly detection. arXiv preprint arXiv:2310.02861 (2023)
8. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. In: International Conference on Learning Representations (2021)
9. Duan, J., et al.: Graph anomaly detection via multi-scale contrastive learning networks with augmented view. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 7459–7467 (2023)
10. Gaudelet, T., et al.: Utilizing graph machine learning within drug discovery and development. Brief. Bioinf. **22**(6), bbab159 (2021)
11. Guan, R., Li, Z., Li, X., Tang, C.: Pixel-superpixel contrastive learning and pseudo-label correction for hyperspectral image clustering. In: ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6795–6799 (2024)
12. Guan, R., et al.: Contrastive multi-view subspace clustering of hyperspectral images based on graph convolutional networks. IEEE Trans. Geosci. Remote Sens. **62**, 1–14 (2024)
13. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)
14. He, M., Wei, Z., Xu, H., et al.: Bernnet: learning arbitrary graph spectral filters via bernstein approximation. Adv. Neural. Inf. Process. Syst. **34**, 14239–14251 (2021)
15. Huang, J., Ling, C.X.: Using AUC and accuracy in evaluating learning algorithms. IEEE Trans. Knowl. Data Eng. **17**(3), 299–310 (2005)
16. Huang, Z., Silva, A., Singh, A.: A broader picture of random-walk based graph embedding. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 685–695 (2021)
17. Kim, S., et al.: Pubchem substance and compound databases. Nucleic Acids Res. **44**(D1), D1202–D1213 (2016)
18. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
19. Kreuzer, D., Beaini, D., Hamilton, W., Létourneau, V., Tossou, P.: Rethinking graph transformers with spectral attention. Adv. Neural. Inf. Process. Syst. **34**, 21618–21629 (2021)
20. Lee, J., Lee, I., Kang, J.: Self-attention graph pooling. In: International Conference on Machine Learning, pp. 3734–3743. PMLR (2019)
21. Li, G., et al.: Deepgcns: making gcns go as deep as cnns. IEEE Trans. Pattern Anal. Mach. Intell. (2021)
22. Li, J., Xing, Q., Wang, Q., Chang, Y.: Cvtgad: simplified transformer with cross-view attention for unsupervised graph-level anomaly detection. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 185–200. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-43412-9_11
23. Liu, C., et al.: Gapformer: graph transformer with graph pooling for node classification. In: International Joint Conference on Artificial Intelligence, pp. 2196–2205 (2023)
24. Liu, Y., Ding, K., Lu, Q., Li, F., Zhang, L.Y., Pan, S.: Towards self-interpretable graph-level anomaly detection. Adv. Neural Inf. Process. Syst. (2023)
25. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10012–10022 (2021)

26. Ma, L., et al.: Graph inductive biases in transformers without message passing. In: International Conference on Machine Learning (2023)
27. Ma, R., Pang, G., Chen, L., van den Hengel, A.: Deep graph-level anomaly detection by glocal knowledge distillation. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, pp. 704–714 (2022)
28. Ma, X., Wu, J., Yang, J., Sheng, Q.Z.: Towards graph-level anomaly detection via deep evolutionary mapping. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1631–1642 (2023)
29. Manzoor, E., Milajerdi, S.M., Akoglu, L.: Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1035–1044 (2016)
30. Maron, H., Ben-Hamu, H., Serviansky, H., Lipman, Y.: Provably powerful graph networks. Adv. Neural Inf. Process. Syst. **32** (2019)
31. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: Tudataset: a collection of benchmark datasets for learning with graphs. In: International Conference on Machine Learning (2020)
32. Niu, C., Pang, G., Chen, L.: Graph-level anomaly detection via hierarchical memory networks. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 201–218. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-43412-9_12
33. : Qiu, C., Kloft, M., Mandt, S., Rudolph, M.: Raising the bar in graph-level anomaly detection. In: International Joint Conference on Artificial Intelligence (2022)
34. Shirzad, H., Velingker, A., Venkatachalam, B., Sutherland, D.J., Sinop, A.K.: Exphormer: sparse transformers for graphs. In: International Conference on Machine Learning (2023)
35. Song, Y., Zheng, S., Niu, Z., Fu, Z.H., Lu, Y., Yang, Y.: Communicative representation learning on attributed molecular graphs. In: International Joint Conference on Artificial Intelligence, vol. 2020, pp. 2831–2838 (2020)
36. Tang, J., Li, J., Gao, Z., Li, J.: Rethinking graph neural networks for anomaly detection. In: International Conference on Machine Learning, pp. 21076–21089. PMLR (2022)
37. Tu, W., et al.: Attribute-missing graph clustering network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 15392–15401 (2024)
38. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017)
39. Veli**v**ković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018)
40. Wang, Y., Zhang, J., Guo, S., Yin, H., Li, C., Chen, H.: Decoupling representation learning and classification for gnn-based anomaly detection. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1239–1248 (2021)
41. Wang, Y., Wang, W., Liang, Y., Cai, Y., Hooi, B.: Mixup for node and graph classification. In: Proceedings of the Web Conference 2021, pp. 3663–3674 (2021)
42. Wu, H., et al.: Pastnet: introducing physical inductive biases for spatio-temporal video prediction. arXiv preprint arXiv:2305.11421 (2023)
43. Wu, H., Xu, F.: Slfnet: generating semantic logic forms from natural language using semantic probability graphs. arXiv preprint arXiv:2403.19936 (2024)

44. Xu, F., Wang, N., Wen, X., Gao, M., Guo, C., Zhao, X.: Few-shot message-enhanced contrastive learning for graph anomaly detection. In: 2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS), pp. 288–295 (2023)
45. Xu, F., Wang, N., Wu, H., Wen, X., Zhao, X., Wan, H.: Revisiting graph-based fraud detection in sight of heterophily and spectrum. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 9214–9222 (2024)
46. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: International Conference on Learning Representations (2019)
47. Yan, F., Yang, G., Li, Y., Liu, A., Chen, X.: Dual graph attention based disentanglement multiple instance learning for brain age estimation. arXiv preprint arXiv:2403.01246 (2024)
48. Yi, H.C., You, Z.H., Huang, D.S., Kwoh, C.K.: Graph representation learning in bioinformatics: trends, methods and applications. Brief. Bioinf. **23**(1), bbab340 (2022)
49. Ying, C., et al.: Do transformers really perform badly for graph representation? Adv. Neural. Inf. Process. Syst. **34**, 28877–28888 (2021)
50. Zhang, G., et al.: Dual-discriminative graph neural network for imbalanced graph-level anomaly detection. Adv. Neural. Inf. Process. Syst. **35**, 24144–24157 (2022)
51. Zhang, Z., Liu, Q., Hu, Q., Lee, C.K.: Hierarchical graph transformer with adaptive node sampling. Adv. Neural. Inf. Process. Syst. **35**, 21171–21183 (2022)

HiGraphDTI: Hierarchical Graph Representation Learning for Drug-Target Interaction Prediction

Bin Liu, Siqi Wu, Jin Wang(✉), Xin Deng, and Ao Zhou

Key Laboratory of Data Engineering and Visual Computing, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
{liubin,wangjin,dengxin}@cqupt.edu.cn, yunning996@gmail.com, zacqupt@gmail.com

Abstract. The discovery of drug-target interactions (DTIs) plays a crucial role in pharmaceutical development. The deep learning model achieves more accurate results in DTI prediction due to its ability to extract robust and expressive features from drug and target chemical structures. However, existing deep learning methods typically generate drug features via aggregating molecular atom representations, ignoring the chemical properties carried by motifs, i.e., substructures of the molecular graph. The atom-drug double-level molecular representation learning can not fully exploit structure information and fails to interpret the DTI mechanism from the motif perspective. In addition, sequential model-based target feature extraction either fuses limited contextual information or requires expensive computational resources. To tackle the above issues, we propose a hierarchical graph representation learning-based DTI prediction method (HiGraphDTI). Specifically, HiGraphDTI learns hierarchical drug representations from triple-level molecular graphs to thoroughly exploit chemical information embedded in atoms, motifs, and molecules. Then, an attentional feature fusion module incorporates information from different receptive fields to extract expressive target features. Last, the hierarchical attention mechanism identifies crucial molecular segments, which offers complementary views for interpreting interaction mechanisms. The experiment results not only demonstrate the superiority of HiGraphDTI to the state-of-the-art methods, but also confirm the practical ability of our model in interaction interpretation and new DTI discovery.

Keywords: Drug-target interaction prediction · Hierarchical graph representation learning · Feature fusion · Attention mechanism

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_21.

1 Introduction

Nowadays, pharmaceutical scientists still rely on existing drug-target interactions (DTIs) to develop novel drugs [3]. Therefore, there is a pressing need to accurately and efficiently discover new DTIs. Although traditional *in vitro* wet-lab verification can obtain reliable DTIs, the complex experimental process consumes considerable time and labor, making it challenging to screen through a large number of candidates rapidly [30]. The computational methods receive considerable focus, since they can significantly diminish the resources for screening by predicting reliable DTI candidates [1]. Deep learning models have achieved superior performances in DTI prediction due to their ability to extract robust and high-quality features from abundant drug and target structure information [3,23]. Deep learning DTI prediction methods typically extract drug and target features from their chemical structures and integrate them to infer unseen interactions [28].

Drugs are chemical molecules, represented by either the Simplified Molecular Input Line Entry System (SMILES) strings [2] or molecular graphs [20]. Convolutional Neural network (CNN) [29] and Transformer [13,27] are utilized to generate drug embeddings via encoding sequential molecular information in SMILES strings. On the other hand, the molecular graphs explicitly depict atom relations in 2-dimensional geometric space, enabling Graph Neural Networks (GNNs) to extract more informative drug representations [12,16]. Motifs, molecular subgraphs composed of part of atoms and their bonds, usually carry indicative information about the important molecular properties and functions [26]. Nevertheless, existing GNN-based approaches typically learn atom node embeddings and aggregate them via readout or attention-weighted summation to derive molecular representations, ignoring important functional characteristics expressed by motifs. Furthermore, current DTI prediction methods focus on identifying the contribution of each atom to DTIs, failing to investigate the biological interpretation of interactions from a motif perspective.

For protein targets, DTI prediction methods use sequential models, such as CNN [12,24], RNN [11] and Transformer [7] to extract high-level features from their amino acid sequences. CNNs typically have limited convolution kernel size and number of layers, and RNNs suffer from the issue of vanishing gradients when dealing with long sequences. Consequently, they learn target features that lack a broad receptive field, i.e., failing to capture distant amino acid relationships within the target sequence [11,12,24]. Although Transformer-based target features fuse every amino acid embeddings, they suffer expensive computational costs [7].

In this study, we propose a hierarchical graph representation learning-based DTI prediction method (HiGraphDTI) to enrich the information involved in drug and target features and enhance the interpretation of DTI mechanisms. First, we employ hierarchical molecular graph representations to extract atom, motif, and global-level embeddings, enabling atomic information aggregation more orderly and reliable while incorporating more chemical properties. Then, we develop an attentional target feature fusion module, which extends receptive fields to improve the expressive ability of protein representations. Finally, we design a hierarchical attention mechanism to capture the various level correlations between drugs and targets, providing comprehensive interpretations of DTIs from multiple perspectives. Experimental results on four benchmark DTI datasets verify that HiGraphDTI surpasses six state-of-the-art methods. The effectiveness of our method in providing valuable biological insights is confirmed via case studies on multi-level attention weight visualizations.

2 Related Work

Predicting drug-target interactions (DTIs) is a crucial area of research in drug development. In recent years, predominant computational approaches comprise two categories: traditional machine learning and deep learning.

Traditional machine learning DTI prediction methods typically rely on manually crafted features, e.g., molecular descriptors for drugs and structural and physicochemical property-based protein features [22]. The SVM classifier utilizes different kernel functions to determine the similarity of compounds and proteins, and combines chemical and genomic spaces via tensor products [14]. WkNNIR is a neighborhood method that recovers potential missing interactions and predicts new DTIs based on the interaction information of proximate known drugs and targets [18]. EBiCTR [21] is an ensemble of bi-clustering trees trained on the reconstructed output space and dyadic (drug and target) feature space. MDMF [17] is a DeepWalk-based Matrix Factorization model that explores potential interactions between drugs and targets embedded in multiplex heterogeneous networks.

Compared with machine learning methods, deep learning-based DTI prediction approaches directly learn high-level features from drug and target structures. DeepDTA [31] leverages the sequence information of drugs and targets to predict drug-target binding affinity. DeepConv-DTI [15] employs convolution on amino acid subsequences of varying lengths to capture local residue patterns in proteins, enriching the information of target features. TransformerCPI [6] let target features serve as the output of the Transformer encoder and the drug features serve as the input to the Transformer decoder to catch the interactions between drugs and targets. MolTrans [13] introduces a Frequent Consecutive Sub-sequence (FCS) mining algorithm, which utilizes unlabeled data to learn contextual semantic information in SMILES strings and amino acid sequences. The FCS algorithm enhances the expressive power of the model and makes progress in exploiting other information. However, it merely identifies patterns in SMILES strings, which may not correspond to the structural characteristics of drugs. IIFDTI [8] comprises four feature components: target features extracted by convolutional neural networks, drug features extracted by graph attention networks, and two interaction features obtained from the Transformer. Similar to MolTrans, it also incorporates semantic information of SMILES and amino acid sequences. DrugBAN [4] utilizes a bilinear attention network module to capture local interactions between drugs and targets for DTI prediction.

Although the aforementioned methods have achieved good performance, they still encounter three issues: the inadequate exploration of triple-level structural information in drug molecules, the unordered information aggregation via simple summation or averaging, and the absence of multi-level biological interpretation.

3 Method

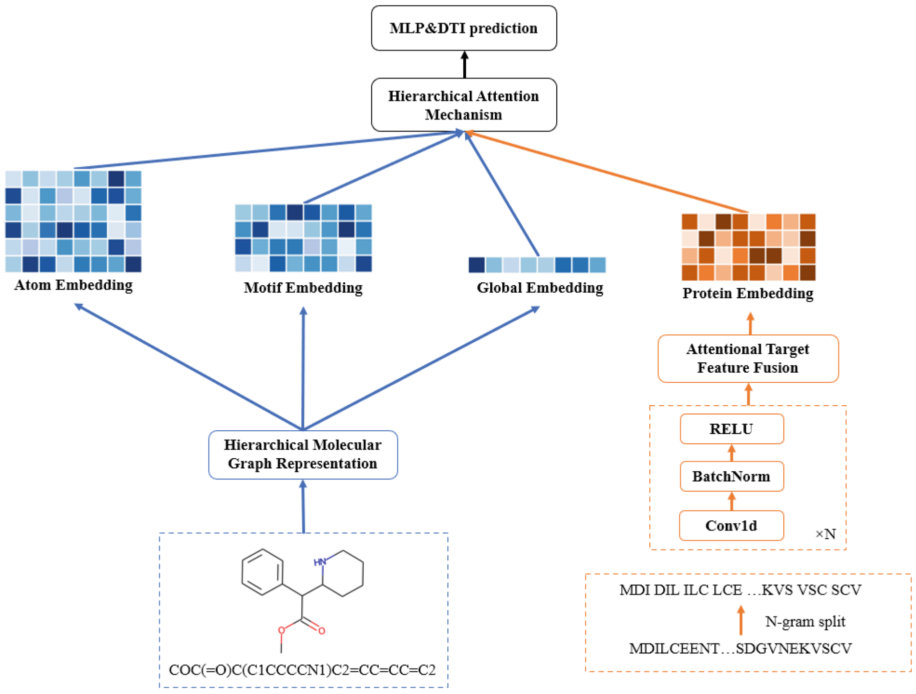

MLP&DTI prediction
Hierarchical Attention Mechanism
Atom Embedding
Motif Embedding
Global Embedding
Protein Embedding
Attentional Target Feature Fusion
RELU
BatchNorm
Conv1d
×N
Hierarchical Molecular Graph Representation
HN
O
O
COC(=O)C(C1CCCCN1)C2=CC=CC=C2
MDI DIL ILC LCE …KVS VSC SCV
N-gram split
MDILCEENT…SDGVNEKVSCV

Fig. 1. The overview architecture of HiGraphDTI.

In this section, we illustrate the proposed HiGraphDTI model that predicts interactions between drugs and targets via hierarchical graph representation learning. Figure 1 outlines the architecture of HiGraphDTI, which consists of three main modules:

– Hierarchical molecular graph representation that extracts drug features, enriching the chemical structure properties and characteristics exploitation and making the information aggregation process more orderly and reliable.
– Attentional target feature fusion that adopts a broader receptive field to protein sequence representation extraction.
– Hierarchical attention mechanism that captures the correlations between drug and target features from various perspectives, providing comprehensive explanations for DTI mechanisms.

3.1 Hierarchical Molecular Graph Representation

Hierarchical graph representation for drugs contains two steps, namely hierarchical graph construction and message-passing.

The molecular graph partition process is illustrated in Fig. 2. First, we transform the original drug molecules into a graph $G = (V, E)$, where each atom corresponds to a node $v \in V$, and the bonds between atoms correspond to bidirectional edges in E. G is the atom layer of the molecular graph. Then, we divide the drug molecules into multiple functional fragments using the Breaking of Retrosynthetically Interesting Chemical Substructures (BRICS) algorithm, which defines 16 rules and breaks strategic bonds in a molecule that match a set of chemical reactions [10]. Following the work [26], we supplement an additional partition rule, i.e., disconnecting cycles and branches around minimum rings, to BRICS algorithm to get rid of excessively large fragments. These obtained fragments, referred to as motifs, construct the second level of the molecular graph. We create a node for each motif, and the collection of nodes is defined as V_m. We connect each motif node with its involved atoms in the atom layer, and the collection of these edges is defined as E_m. To avoid the over-smoothing issue in graph neural networks and make message aggregation more reasonable, these edges are unidirectional, pointing from the atom layer to the motif layer. Finally, to aggregate the global information of drug molecules, we construct a global node V_g, which is the graph-layer. We establish connections between it and all motif nodes, and the collection of these edges is referred to as E_g. These edges are also unidirectional, pointing from motif nodes to the global node V_g. The final hierarchical graph is constructed as follows:

$$\bar{G} = (\bar{V}, \bar{E}), \bar{V} = (V, V_m, V_g), \bar{E} = (E, E_m, E_g) \tag{1}$$

Given the triple-layer molecular graph, we employ Graph Isomorphism Network (GIN) to propagate messages and learn node embeddings due to its superior expressive power demonstrated by Weisfeiler-Lehman (WL) test [25]. Specifically, the message-passing formula of GIN is defined as:

$$\mathbf{h}_v^s = MLP^s(\mathbf{h}_v^{s-1} + \sum_{u \in \mathcal{N}(v)} (\mathbf{h}_u^{s-1} + \mathbf{W}^s \mathbf{X}_{uv})) \tag{2}$$

where MLP^s represents a multi-layer perceptron (MLP) at the s-th layer that contains an input linear layer, a Relu activation function, and an output linear layer, $\mathbf{X}_{uv}$ represents the edge embeddings between nodes u and v, $\mathbf{W}^s$ represents the embedding parameters of $\mathbf{X}_{uv}$ for s-th layer, and $\mathbf{h}_v^0 = \mathbf{X}_v$ is the input node feature of $v \in \bar{V}$, $\mathbf{h}_v^s$ represents the hidden state for node at the s-th layer. After multiple iterations of updates, we obtain the final embeddings of atom, motif, and global nodes, denoted as $\mathbf{H}_a \in \mathcal{R}^{|a| \times d}$, $\mathbf{H}_m \in \mathcal{R}^{|m| \times d}$ and $\mathbf{H}_g \in \mathcal{R}^{1 \times d}$ respectively, where $|a|$ is the number of atoms, $|m|$ is the number of motifs. We adopt $\mathbf{H}_g$ as the representation of the whole drug molecule.

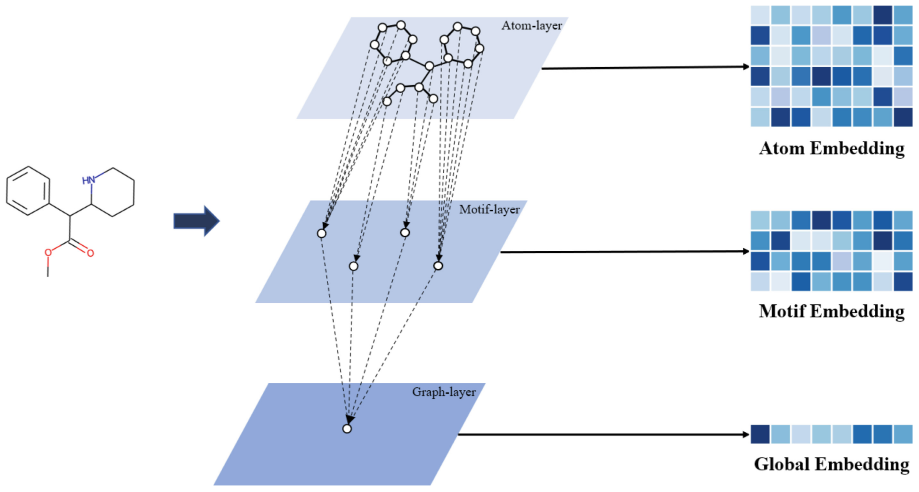


Fig. 2. Hierarchical graph representation construction, where solid lines represent bidirectional edges, and dashed lines represent unidirectional edges.

3.2 Attentional Target Feature Fusion

Following previous work [24], we partition the target sequence into 3-gram amino acids to obtain the initial vector, denoted as $\mathbf{X}_P = \{x_1, x_2, ..., x_l\}$, where $x_i \in \mathcal{R}^d$ represents the embedding of the i-th segment, l is the number of the partitioned sequences, and d is the embedding dimension. To better aggregate critical features in the protein vector representation, We design a one-dimensional (1D) convolutional neural network with layer-wise decreasing channels, and the formula for each layer is as follows:

$$\mathbf{X}_i = Relu(BN_i(Conv1D_i(\mathbf{X}_{i-1}))) \tag{3}$$

where $\mathbf{X}_i$ represents the feature representation for the i-th layer, and $\mathbf{X}_0 = \mathbf{X}_P$, $Conv1D_i$ represents the 1D convolution in the i-th layer with the kernel size of 15 and the output channels reduced by half, $Relu$ represents the ReLu nonlinear activation function, BN_i represents the batch normalization in the i-th layer.

We obtain target feature representations $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$ at three different convolutional layers. To aggregate target information, we adapt the attentional feature fusion (AFF) module [9] tailored to amino acid sequences. The process is depicted in Fig. 3. We perform transposed convolution on $\mathbf{X}_3$ to map it to the same dimension as $\mathbf{X}_2$ and then put it into the AFF module. Next, we map the result to the same dimension as $\mathbf{X}_1$ and put it into the AFF module to obtain the outcome, denoted as $\mathbf{H}_P \in \mathcal{R}^{l \times d}$, of which the size is the same with initial target features $\mathbf{X}_P$.

The detailed illustration of AFF module are shown in Fig. 4. AFF module receives two inputs, $\mathbf{I}_1$ and $\mathbf{I}_2$, where $\mathbf{I}_1$ is the high-level feature after transposed convolution, and $\mathbf{I}_2$ is the low-level feature. We combine the information

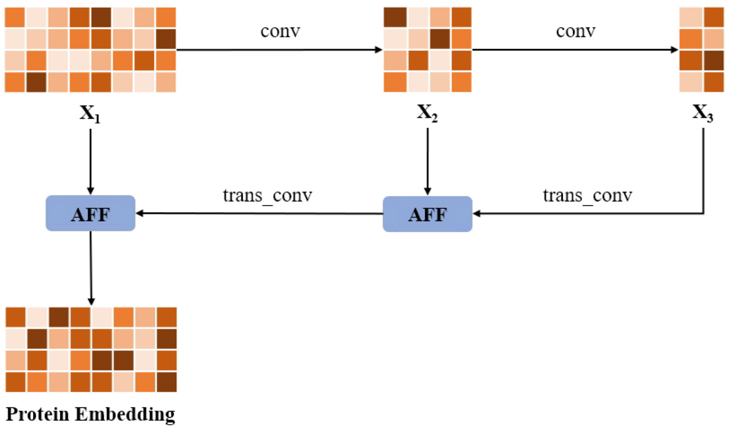


Fig. 3. The overview architecture of feature fusion module for protein. The high-level feature is mapped to the same dimension as the low-level using transposed convolution and then input into the AFF module for fusion. Taking the result as high-level features, repeat the operation.

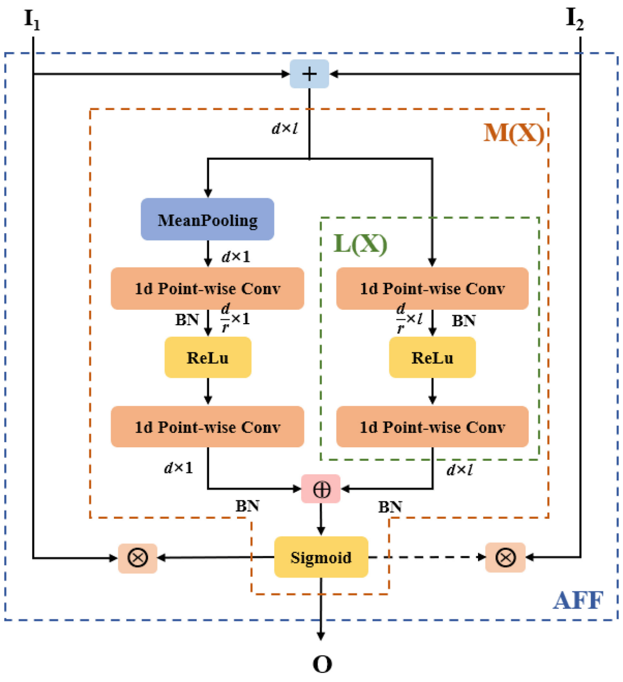


Fig. 4. The architecture of AFF module, which utilizes operation M to compute the attention matrix for weighted aggregation of inputs $\mathbf{I}_1$ and $\mathbf{I}_2$.

from those through element-wise summation as $\mathbf{I}$ and feed the result into module M to extract additional information. Module M achieves channel attention across multiple scales by changing the spatial pooling size. It mainly consists of two parts: one for extracting global features and the other for extracting local features, as illustrated in Eq. (4).

$$M(\mathbf{I}) = \sigma(L(\mathbf{I}) \oplus L(MeanPooling(\mathbf{I}))) \tag{4}$$

where σ is the Sigmoid function, $MeanPooling(\mathbf{I}) = \frac{1}{l}\sum_{i=1}^{l} \mathbf{I}[i,:]$ is the average pooling along columns, $\oplus$ refers to the broadcasting addition and $L(\mathbf{I})$ is defined as:

$$L(\mathbf{I}) = BN(PWConv_2(Relu(BN(PWConv_1(\mathbf{I}))))) \tag{5}$$

In Eq. (5), $PWConv_1$ and $PWConv_2$ refer to two point-wise 1D convolutions to capture information from diverse channels. To maintain the model as lightweight as possible., we adopt a bottleneck structure. Specifically, $PWConv_1$ is used to reduce the channels by a factor of r, and $PWConv_2$ is employed to restore them. After applying the module M, We obtain the attention matrix. To get the final output, we perform the following operations:

$$\mathbf{O} = M(\mathbf{I}) \otimes \mathbf{I}_1 + (1 - M(\mathbf{I})) \otimes \mathbf{I}_2 \tag{6}$$

where $\otimes$ denotes the element-wise multiplication. In Fig. 4, the black dashed line denotes $(1 - M(\mathbf{I}))$. $M(\mathbf{I})$ and $1 - M(\mathbf{I})$ are real arrays in the range of 0 to 1, facilitating a weighted sum of $\mathbf{I}_1$ and $\mathbf{I}_2$.

3.3 Hierarchical Attention Mechanism

We design a hierarchical attention mechanism to capture the correlation between triple-level drug features ($\mathbf{H}_a$, $\mathbf{H}_m$, $\mathbf{H_g}$) and target features ($\mathbf{H}_P$). Graph level drug feature vector $\mathbf{H}_g$ aggregates the global molecular information via cross-level message-passing. Nevertheless, incorporating excessive information into $\mathbf{H}_g$ could significantly decrease its express capacity. Therefore, we incorporate drug features into the target embedding $\mathbf{H}_P$, and calculate the attention between targets and different levels of drugs as follows:

$$\mathbf{Attn}_a = Relu(\mathbf{H}_P^\top \mathbf{W}_a \mathbf{H}_a) \tag{7}$$

$$\mathbf{Attn}_m = Relu(\mathbf{H}_P^\top \mathbf{W}_m \mathbf{H}_m) \tag{8}$$

$$\mathbf{Attn}_g = Relu(\mathbf{H}_P^\top \mathbf{W}_g \mathbf{H}_g) \tag{9}$$

where $\mathbf{Attn}_a \in \mathcal{R}^{l\times|a|}$, $\mathbf{Attn}_m \in \mathcal{R}^{l\times|m|}$, $\mathbf{Attn}_g \in \mathcal{R}^{l\times 1}$ represent attention matrices between protein partitioned sequence and different levels (atom, motif, and global) of the drug molecule. Next, we calculate the mean along the rows for each attention matrix, resulting in three attention vectors $\mathbf{A}_a$, $\mathbf{A}_m$, $\mathbf{A}_g$. The summation of these vectors utilized as weights for updating $\mathbf{H}_P$ yields the protein representation enriched with drug information:

$$\mathbf{F}_P = \mathbf{H}_P \cdot (SF(\mathbf{A}_a) + SF(\mathbf{A}_m) + SF(\mathbf{A}_g)) \tag{10}$$

where SF is the softmax function.

Finally, we concatenate $\mathbf{H}_g$ and $\mathbf{F}_P$ and then feed them into a multi-layer perceptron (MLP) model to derive the probability of drug-target interaction $\hat{\mathbf{Y}}$. Our model The binary cross-entropy loss, defined in Eq. (11), is utilized for training our model.

$$\mathcal{L} = -\frac{1}{N}\sum_{i}^{N} y_i \cdot log(\hat{y_i}) + (1 - y_i) \cdot log(1 - \hat{y_i}) \tag{11}$$

In Eq. (11), y_i and $\hat{y_i}$ are the true and predicted label of i-th drug-target pair (samples) respectively, N is the number of training drug-target pairs.

4 Experiments

4.1 Experimental Setup

We select four benchmark datasets in the DTI field to evaluate our model, including Human [19], *C.elegans* (*Caenorhabditis elegans*) [19], BindingDB [11], GPCR [6]. Human and *C.elegans* datasets are created using a systematic screening framework to obtain highly credible negative samples [19]. GPCR dataset is constructed from the GLASS database [5], which uses scores to describe the drug-target affinity(DTA). To obtain samples for DTIs, GPCR uses a threshold of 6.0 to categorize positive and negative samples. The BindingDB dataset [11] is created from the BindingDB database, which is a public, web-accessible database of measured binding affinities, focusing on the interactions of small molecules. BindingDB dataset employs IC50 to filter samples, i.e., samples with IC50 below 100nM is labeled as positive, while those with IC50 above 10000nM is labeled as negative. Table 1 presents the statistics of the four datasets.

Table 1. Statistics of datasets.

Datasets	Targets	Drugs	Interactions	Positive	Negative
Human	852	1052	6738	3369	3369
C.elegans	2504	1434	8000	4000	4000
BindingDB	812	49745	61258	33772	27486
GPCR	356	5359	15343	7989	7354

We employ a five-fold cross-validation approach for the Human and *C.elegans* datasets, dividing the whole datasets into training, validation and test sets according to the ratio of 8:1:1. For the BindingDB dataset, we employ the same data partition strategy used by [11], i.e., 28237 samples in the training set, 2830 samples in the validation set, and 2705 samples in the test set. In regard to GPCR dataset, we follow the data partition strategy described in [6], and further randomly select the 20% of the training set as the validation set. The size of the training, validation and test sets are 11,045, 2761 and 15377, respectively.

We select six state-of-the-art DTI prediction methods for comparison: DeepDTA [31], DeepConv-DTI [15], MolTrans [13], TransformerCPI [6], IIFDTI [8], and DrugBAN [4]. A brief introduction to the methods mentioned above is provided in the Supplementary Materials. To adapt DeepDTA, a drug-target affinity prediction model, to the DTI prediction task, we replace the loss function in its last layer with binary cross-entropy loss.

We choose four metrics for evaluating our models: AUC (the area under the receiver operating characteristic curve), AUPR (the area under the precision-recall curve), Precision, and Recall. We execute all models on all datasets ten times using different random seeds, calculating their averages and standard deviation (std) for performance comparison.

Our experiments were conducted on a machine with a 32-core 2.30 GHz Intel i9-10900J processor, 512G of RAM, and a single V100 GPU. We utilize the Pytorch Framework to implement the proposed model. Regarding the experiment process, we save the model parameters that achieve the highest AUC on the validation set. Then, we evaluate its performance on the test set to obtain results. Details regarding model hyperparameter settings are available in Table 2. The codes of our model are available at https://github.com/547-own/HiGraphDTI.

Table 2. Summary of model hyperparameters

Hyperparameter	Value
Number of GNN layers	5 for *C.elegans* 3 for the other three datasets
Number of CNN layers	3
Epochs	100
Weight decay	1e–6
Learning rate	1e–4
Embedding dimension (d)	256
Batch size	32
Dropout rate	0.1
N-gram	3

4.2 Comparison Results

As shown in Tables 3 and 4, HiGraphDTI outperforms the six baselines in terms of AUC and AUPR on all datasets. We attribute the excellent performance to its three merits. First, HiGraphDTI learns hierarchical drug graph representation to aggregate chemical structure information across different levels, enriching the molecular structure representation. Second, employing feature fusion modules enables targets to capture information from different receptive fields, enhancing the protein sequence representation. Third, the hierarchical attention mechanism computes interactive attention between different levels of drugs and targets, augmenting the interaction information between drugs and targets.

Table 3. Experiment results in terms of AUC, where the best and runner-up results are highlighted in bold and underlined, respectively.

Model	Dataset			
	Human	*C.elegans*	BindingDB	GPCR
DeepDTA	0.972 (0.001)	0.983 (0.001)	0.934 (0.007)	0.776 (0.006)
DeepConv-DTI	0.967 (0.002)	0.983 (0.002)	0.922 (0.003)	0.752 (0.011)
MolTrans	0.974 (0.002)	0.982 (0.003)	0.899 (0.006)	0.807 (0.004)
TransformerCPI	0.970 (0.006)	0.984 (0.002)	0.933 (0.011)	0.842 (0.007)
IIFDTI	<u>0.984 (0.003)</u>	<u>0.991 (0.002)</u>	0.944 (0.003)	<u>0.845 (0.008)</u>
DrugBAN	0.984 (0.001)	0.989 (0.001)	<u>0.945 (0.007)</u>	0.837 (0.010)
Ours	**0.985 (0.001)**	**0.993 (0.001)**	**0.954 (0.003)**	**0.858 (0.004)**

Table 4. Experiment results in terms of AUPR, where the best and runner-up results are highlighted in bold and underlined, respectively.

Model	Dataset			
	Human	*C.elegans*	BindingDB	GPCR
DeepDTA	0.973 (0.002)	0.984 (0.007)	0.934 (0.008)	0.762 (0.015)
DeepConv-DTI	0.964 (0.004)	0.985 (0.001)	0.921 (0.004)	0.685 (0.010)
MolTrans	0.976 (0.003)	0.982 (0.003)	0.897 (0.010)	0.788 (0.009)
TransformerCPI	0.974 (0.005)	0.983 (0.003)	0.934 (0.015)	0.837 (0.010)
IIFDTI	<u>0.985 (0.003)</u>	<u>0.992 (0.003)</u>	<u>0.945 (0.004)</u>	<u>0.842 (0.007)</u>
DrugBAN	0.981 (0.001)	0.990 (0.002)	0.944 (0.005)	0.823 (0.013)
Ours	**0.988 (0.001)**	**0.993 (0.001)**	**0.955 (0.003)**	**0.850 (0.003)**

IIFDTI ranks second on the Human, *C. elegans* and GPCR datasets. It leverages Word2Vec to extract drug features from textual information encoded in SMILES string, while HiGraphDTI enriches hierarchical information in molecular graph representations. Compared to compressed textual information, hierarchically aggregated information based on molecular chemical properties is more expressive. At the same time, after hierarchical partitioning, our method can calculate attention scores between different levels and the target. That enriches the information of interaction features and allows for diverse biological interpretations at different levels of DTI. HiGraphDTI surpasses IIFDTI in AUC and AUPR, especially on the GPCR dataset, with improvements of 1.3% and 0.8%, respectively.

For the larger dataset BindingDB, DrugBAN is the second-best method in terms of AUC. DrugBAN utilizes GNN and CNN to extract feature representations for drugs and targets and employs Bilinear Attention Network to obtain interaction features. However, it does not incorporate additional information to enrich its feature representations, resulting in its inferiority to HiGraphDTI. Furthermore, HiGraphDTI also exhibits advantages over IIFDTI on the BindingDB dataset, achieving improvements of 0.9% in AUC and 1.1% in AUPR. The results for precision and recall are presented in the Supplementary Materials.

4.3 Ablation Experiment

To validate the effectiveness of each module in HiGraph, we design the following ablation experiments:

- HiGraphDTIw/oFF: We remove the target feature fusion module and retain the last output convolutional layer as target representation.
- HiGraphDTIw/oHI: We remove all attentions between drugs and targets. We only concatenate the global-level features of drugs and the mean of target features for prediction.
- HiGraphDTIw/oHC: We remove the hierarchical structure from the graph representation and only use atom-level embeddings to construct drug features.
- HiGraphDTIw/oML: We remove the motif-level nodes from the hierarchical molecular graph and only utilize atom and global nodes to construct drug features.

The experimental results on the GPCR dataset are shown in Fig. 5. The results of HiGraphDTIw/oFF validates the importance of the feature fusion module in constructing target features. Losing multiple receptive fields leads to a decrease in model performance. The results of HiGraphDTIw/oHI demonstrate the validity of the hierarchical attention mechanism, which comprehends the interaction between drugs and targets from different perspectives, enhancing the understanding and predictive capability of the model. Finally, the comparison between HiGraphDTIw/oHC and HiGraphDTIw/oML confirms the superiority of hierarchical graph representation learning methods in drug feature extraction. The multi-layered structure enriches the expression of drug features.

4.4 Attention Interpretation

The hierarchical attention mechanism not only enhances model performance but also assists us in understanding DTIs from various insights. In this part, we utilize the attention weights to interpret the effectiveness of the hierarchical attention mechanism. Furthermore, we illustrate the drug-target interaction from the atom and motif levels to offer valuable interpretations for drug discovery.

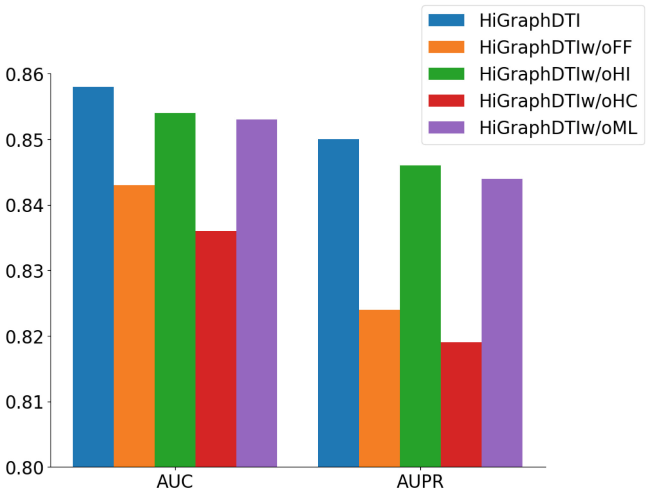
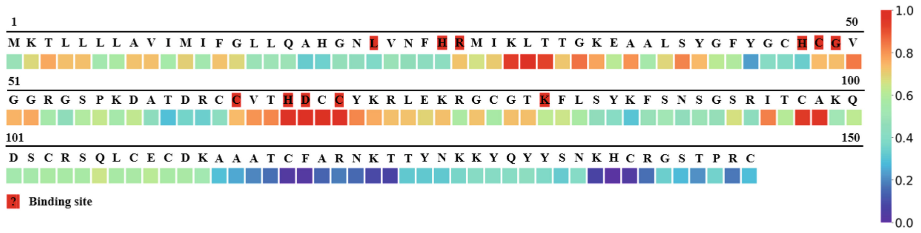


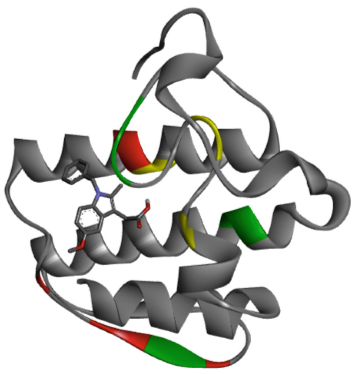
Fig. 5. Ablation experiment results on the GPCR dataset

We choose target PDB: 1N28 and drug (ligands) PDB: I3N ($C_{19}H_{19}NO_3$) as a case study for attention interpretation. We use the hierarchical attention mechanism to calculate the attention vector $\mathbf{B}_P = SF(\mathbf{A}_a)+SF(\mathbf{A}_m)+SF(\mathbf{A}_g) \in \mathcal{R}^l$, which demonstrates the distribution of amino acid attention weights. The values in $\mathbf{B}_P$ are all within the range of 0 to 1. The attention weights for each amino acid of PDB:1N28 are shown in Fig. 6(a), where different colors represent varying attention weights. The actual binding sites are represented by amino acid letters with a red background. From Fig. 6(a), we can observe that the model gives high attention to six among the total eleven binding sites. In addition, the model provides seven other positions (located at 30, 31, 32, 69, 70, 97, 98) with high attention weights, which could serve as potential binding sites for future chemical experiments. Figure 6(b) depicts the 3D visualization of the docking interaction of PDB: I3N and PDB: 1N28, where red regions represent binding sites with high attention weights, yellow segments indicate the binding site with low attention weights, green regions represent the high attention weighted amino acids that have not been recognized as binding sites.

In the process of computing $\mathbf{B}_P$, we obtain three attention matrices: $\mathbf{Attn}_a \in \mathcal{R}^{l\times|a|}$, $\mathbf{Attn}_m \in \mathcal{R}^{l\times|m|}$, and $\mathbf{Attn}_g \in \mathcal{R}^{l\times 1}$. We further average every column of $\mathbf{Attn}_a$, $\mathbf{Attn}_m$ to obtain the attention vector $\mathbf{B}_a \in \mathcal{R}^{|a|}$, $\mathbf{B}_m \in \mathcal{R}^{|m|}$, where each element illustrates the importance of each atom and motif nodes to the DTI. The visualization of drug attention weights for the interaction of PDB: I3N and PDB: 1N28 is shown in Fig. 7, where dashed lines of the same color connect motif and its composed atoms. There are fifteen atoms interacting with at least one amino acid, where 2/3 attention weights exceed 0.6. The corresponding motif nodes also exhibit high attention weights. It can be observed that the 11-th atom (C) is an active node in the docking simulation. While its atom attention weight is not



(a) Attention Weights for each amino acid of PDB: 1N28



(b) 3D visualization of docking interaction of PDB: I3N with PDB: 1N28

Fig. 6. Visualization of target attention weights for interaction of PDB: I3N and PDB: 1N28

high, the 3-rd motif node containing it has a high attention weight, serving as a powerful supplement. This validates that the hierarchical graph representation approach to constructing drug features permits the model to better discern the importance of atoms and motifs and ensures crucial elements are not overlooked during the drug development process.

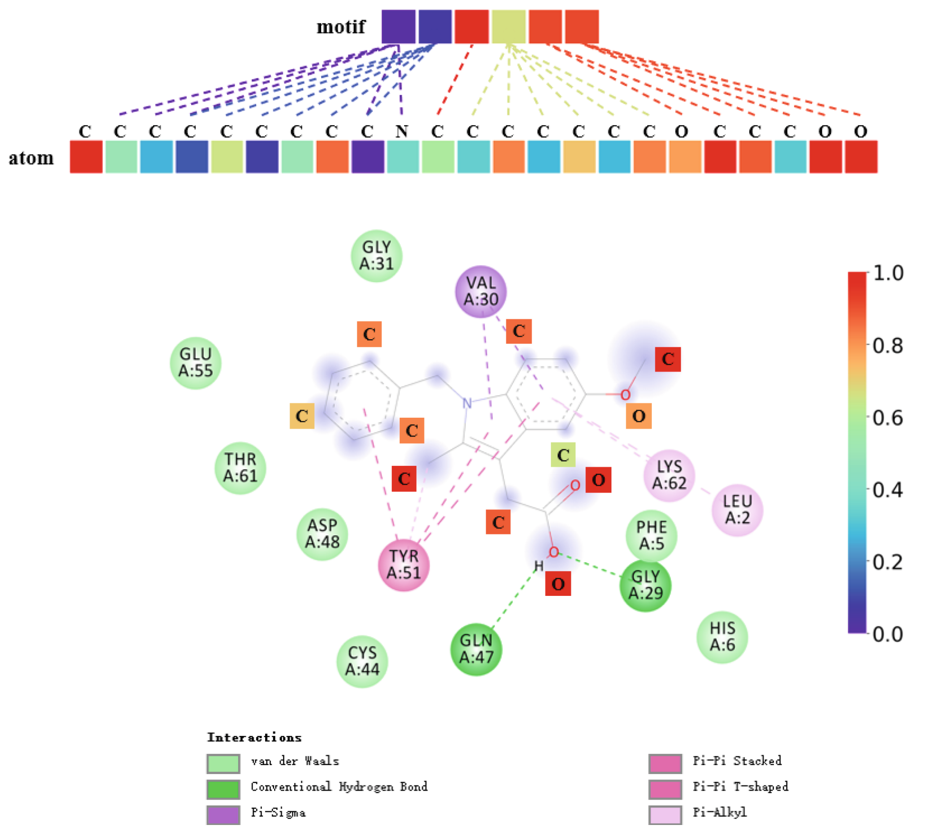


Fig. 7. Visualization of drug attention weights for the interaction of PDB: I3N and PDB:1N28

5 Conclusion

In this paper, we propose HiGraphDTI, a novel deep learning model designed for predicting DTIs. HiGraphDTI has three key advantages. First, it employs a hierarchical molecular graph representation to construct drug features, enhancing the incorporation of chemical structure information and facilitating more effective message conveyance. Second, to augment the receptive field of target features, HiGraphDTI integrates an attentional target feature fusion strategy, resulting in more informative protein representations. In addition, the model incorporates a hierarchical attention mechanism to capture interactive information between drugs and targets from multiple perspectives. To validate the effectiveness of the proposed model, we compare it with six state-of-the-art baselines on four datasets. The experimental results indicate that our model outperforms comparing baselines in terms of AUC and AUPR metrics. Finally, visualizations of attention weights confirm the interpretation ability of HiGraphDTI to support new drug discovery.

Acknowledgements. This work was supported by the National Natural Science Foundation of China (62302074) and the Science and Technology Research Program of Chongqing Municipal Education Commission (KJQN202300631).

References

1. Abbasi, K., Razzaghi, P., Poso, A., Ghanbari-Ara, S., Masoudi-Nejad, A.: Deep learning in drug target interaction prediction: current and future perspectives. Curr. Med. Chem. **28**(11), 2100–2113 (2021)
2. Anderson, E., Veith, G.D., Weininger, D.: Smiles: a line notation and computerized interpreter for chemical structures. J. Chem. Inf. Comput. Sci. **28**(1), 31–36 (1987)
3. Bagherian, M., Sabeti, E., Wang, K., Sartor, M.A., Nikolovska-Coleska, Z., Najarian, K.: Machine learning approaches and databases for prediction of drug-target interaction: a survey paper. Brief. Bioinf. **22**(1), 247–269 (2021)
4. Bai, P., Miljković, F., John, B., Lu, H.: Interpretable bilinear attention network with domain adaptation improves drug-target prediction. Nat. Mach. Intell. **5**(2), 126–136 (2023)
5. Chan, W.K.B., et al.: Glass: a comprehensive database for experimentally validated GPCR-ligand associations. Bioinformatics **31**(18), 3035–3042 (2015)
6. Chen, L., et al.: Transformercpi: improving compound-protein interaction prediction by sequence-based deep learning with self-attention mechanism and la- bel reversal experiments. Bioinformatics **36**(16), 4406–4414 (2020)
7. Cheng, Z., Yan, C., Wu, F.X., Wang, J.: Drug-target interaction prediction using multi-head self-attention and graph attention network. IEEE/ACM Trans. Comput. Biol. Bioinf. **19**(4), 2208–2218 (2022)
8. Cheng, Z., Zhao, Q., Li, Y., Wang, J.: IIFDTI: predicting drug-target interactions through interactive and independent features based on attention mechanism. Bioinformatics **38**(17), 4153–4161 (2022)
9. Dai, Y., Gieseke, F., Oehmcke, S., Wu, Y., Barnard, K.: Attentional feature fusion. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3560-3569 (2021)
10. Degen, J., Wegscheid-Gerlach, C., Zaliani, A., Rarey, M.: On the art of compiling and using "drug-like" chemical fragment spaces. ChemMedChem **3**(10), 1503 (2008)
11. Gao, K.Y., Fokoue, A., Luo, H., Iyengar, A., Dey, S., Zhang, P.: Interpretable drug target prediction using deep neural representation. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, pp. 3371–3377 (2018)
12. Hua, Y., Song, X.N., Feng, Z., Wu, X.J., Kittler, J., Yu, D.J.: Cpinformer for efficient and robust compound-protein interaction prediction. IEEE/ACM Trans. Comput. Biol. Bioinf. **20**(1), 285–296 (2022)
13. Huang, K., Xiao, C., Glass, L.M., Sun, J.: Moltrans: molecular interaction transformer for drug target interaction prediction. Bioinformatics **37**(6), 830–836 (2021)
14. Jacob, L., Vert, J.P.: Protein-ligand interaction prediction: an improved chemogenomics approach. Bioinformatics **24**(19), 2149–2156 (2008)
15. Lee, I., Keum, J., Nam, H.: Deepconv-dti: prediction of drug-target interactions via deep learning with convolution on protein sequences. PLoS Comput. Biol. **15**(6), e1007129 (2019)
16. Li, F., Zhang, Z., Guan, J., Zhou, S.: Effective drug-target interaction prediction with mutual interaction neural network. Bioinformatics **38**(14), 3582–3589 (2022)

17. Liu, B., Papadopoulos, D., Malliaros, F., Tsoumakas, G., Papadopoulos, A.: Multiple similarity drug-target interaction prediction with random walks and matrix factorization. Brief. Bioinform. **23**(5), 1–9 (2022)
18. Liu, B., Pliakos, K., Vens, C., Tsoumakas, G.: Drug-target interaction prediction via an ensemble of weighted nearest neighbors with interaction recovery. Appl. Intell. **52**(4), 3705–3727 (2022)
19. Liu, H., Sun, J., Guan, J., Zheng, J., Zhou, S.: Improving compound-protein interaction prediction by building up highly credible negative samples. Bioinformatics **31**(12), i221–i229 (2015)
20. Nguyen, T., Le, H., Quinn, T.P., Nguyen, T., Le, T.D., Venkatesh, S.: GraphDTA: predicting drug-target binding affinity with graph neural networks. Bioinformatics **37**(8), 1140–1147 (2020)
21. Pliakos, K., Vens, C.: Drug-target interaction prediction with tree-ensemble learning and output space reconstruction. BMC Bioinf. **21**(49), 1–11 (2020)
22. Sachdev, K., Gupta, M.K.: A comprehensive review of feature based methods for drug target interaction prediction. J. Biomed. Inf. **93**, 103159 (2019)
23. Sun, M., Zhao, S., Gilvary, C., Elemento, O., Zhou, J., Wang, F.: Graph convolutional networks for computational drug development and discovery. Brief. Bioinf. **21**(3), 919–935 (2020)
24. Tsubaki, M., Tomii, K., Sese, J.: Compound-protein interaction prediction with end-to-end learning of neural networks for graphs and sequences. Bioinformatics **35**(2), 309–318 (2019)
25. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? CoRR arxiv:1810.00826 (2018)
26. Zhang, Z., Liu, Q., Wang, H., Lu, C., Lee, C.K.: Motif-based graph self-supervised learning for molecular property prediction. Adv. Neural Inf. Process. Syst. 15870–15882 (2021)
27. Zhao, Q., Duan, G., Zhao, H., Zheng, K., Li, Y., Wang, J.: Gifdti: prediction of drug-target interactions based on global molecular and intermolecular interaction representation learning. IEEE/ACM Trans. Comput. Biol. Bioinf. **20**(3), 1943–1952 (2023)
28. Zhao, Q., Yang, M., Cheng, Z., Li, Y., Wang, J.: Biomedical data and deep learning computational models for predicting compound-protein relations. IEEE/ACM Trans. Comput. Biol. Bioinf. **19**(4), 2092–2110 (2022)
29. Zhao, Q., Zhao, H., Zheng, K., Wang, J.: HyperAttentionDTI: improving drug-protein interaction prediction by sequence-based deep learning with attention mechanism. Bioinformatics **38**(3), 655–662 (2021)
30. Zitnik, M., Nguyen, F., Wang, B., Leskovec, J., Goldenberg, A., Hoffman, M.M.: Machine learning for integrating data in biology and medicine: principles, practice, and opportunities. Inf. Fusion **50**, 71–91 (2019)
31. Öztürk, H., Özgür, A., Ozkirimli, E.: DeepDTA: deep drug-target binding affinity prediction. Bioinformatics **34**(17), i821–i829 (2018)

On the Two Sides of Redundancy in Graph Neural Networks

Franka Bause[1,2](✉), Samir Moustafa[1,2], Johannes Langguth[4], Wilfried N. Gansterer[1], and Nils M. Kriege[1,3]

[1] Faculty of Computer Science, University of Vienna, Vienna, Austria
{franka.bause,samir.moustafa,wilfried.gansterer,nils.kriege}@univie.ac.at
[2] UniVie Doctoral School Computer Science, University of Vienna, Vienna, Austria
[3] Research Network Data Science, University of Vienna, Vienna, Austria
[4] Simula Research Laboratory, Oslo, Norway
langguth@simula.no

Abstract. Message passing neural networks iteratively generate node embeddings by aggregating information from neighboring nodes. With increasing depth, information from more distant nodes is included. However, node embeddings may be unable to represent the growing node neighborhoods accurately and the influence of distant nodes may vanish, a problem referred to as oversquashing. Information redundancy in message passing, i.e., the repetitive exchange and encoding of identical information amplifies oversquashing. We develop a novel aggregation scheme based on neighborhood trees, which allows for controlling redundancy by pruning redundant branches of unfolding trees underlying standard message passing. While the regular structure of unfolding trees allows the reuse of intermediate results in a straightforward way, the use of neighborhood trees poses computational challenges. We propose compact representations of neighborhood trees and merge them, exploiting computational redundancy by identifying isomorphic subtrees. From this, node and graph embeddings are computed via a neural architecture inspired by tree canonization techniques. Our method is less susceptible to oversquashing than traditional message passing neural networks and can improve the accuracy on widely used benchmark datasets.

Keywords: Graph neural networks · Oversquashing · Non-redundant message passing

1 Introduction

Graph neural networks (GNNs) emerged as the dominant approach for machine learning on graph data, with the class of message passing neural networks (MPNNs) [13] being widely used. These networks update node embeddings layer wise by combining the current embedding of a node with those of its neighbors, involving learnable parameters. Suitable neural architectures, which admit a parametrization such that each layer represents an injective function uniquely encoding the input, have the same expressive power as the Weisfeiler-Leman algorithm [36]. The Weisfeiler-Leman algorithm distinguishes two nodes if and only if the unfolding trees representing their neighborhoods are non-isomorphic. These unfolding trees correspond to the computational trees of MPNNs [16,31]. Hence, nodes with isomorphic unfolding trees will obtain the same embedding, while for nodes with non-isomorphic unfolding trees, there exist parameters such that their embeddings differ. This implies that deeper unfolding trees lead to more expressive methods. Despite this theoretical connection, shallow MPNNs are often favored in practice. Challenges arise from the observed convergence of node embeddings for deep architectures, referred to as *oversmoothing* [21,22], and the issue of *oversquashing* [5], where the node neighborhood grows exponentially with the number of layers, and therefore, cannot be supposed to be accurately represented by a fixed-sized embedding. Recently, oversquashing has been investigated by analyzing the sensitivity of node embeddings to the initial features of distant nodes, relating the phenomenon to the *graph curvature* [34], the *effective resistance* [8] and the *commute time* [14]. On this basis several graph rewiring strategies have been proposed to mitigate oversquashing [8,34].

We address the problem of oversquashing by modifying the message passing scheme for eliminating the encoding of repeated information. For example, in an undirected graph, when a node sends information to its neighbour, future messages sent back via the same edge will contain the exact information previously sent, leading to redundancy. In the context of walk-based graph learning this problem is well-known and referred to as *tottering* [23]. Recent work by Chen et al. [9] established a first result formalizing the relation between redundancy and oversquashing by sensitivity analysis. Several recent GNNs replace the walk-based aggregation by mechanisms based on simple or shortest paths reporting promising results [2,17,27]. PathNNs [27] and RFGNN [17] are closely related approaches, defining path-based trees for nodes and employing custom aggregation schemes. However, these methods suffer from high computational costs compared to standard MPNNs and often have an exponential time complexity. The crucial advantage of MPNNs is the regular structure of aggregations applied through all layers, while reducing information redundancy leads to a less regular structure, rendering it challenging to exploit computational redundancy.

Our Contribution. We systematically explore the issue of information redundancy within MPNNs and introduce principled techniques to eliminate superfluous messages. Our investigation is based on the implicit tree representation used by both MPNNs and the Weisfeiler-Leman algorithm. We first develop a neural tree canonization approach that systematically processes trees in a bottom-up fashion and extend it to directed acyclic graphs (DAGs). To exploit computational redundancy, we merge trees representing node neighborhoods into a DAG, identifying isomorphic subtrees. Our approach, termed DAG-MLP, recovers the computational graph of MPNNs for unfolding trees, while avoiding redundant computations in the presence of symmetries. We employ the canonization technique on *neighborhood trees*, which are derived from unfolding trees by eliminating redundant nodes. We show that neighborhood trees allow distinguishing

Table 1. Time complexity of preprocessing, size of computation graph and expressivity of our method compared to related work. n: number of nodes, m: number of edges, b: maximum node degree, K: path length, h: tree height, L: number of layers, and $m_2 = 0.5 \sum_{v \in V} |N_2(v)|$.

Method	Preprocessing	Size Comp. Graph/Runtime	Expressivity
PathNet	$O(mb)$	$O(2^L(m + m_2))$	n/a
PathNN-SP	$O(nb^K)$	$O(nbK)$	incomparable
PathNN-SP+	$O(nb^K)$	$O(nbK)$	$>$ 1-WL
RFGNN	$O(^{n!}/_{(n-h-1)!})$	$O(^{n!}/_{(n-h-1)!})$	incomparable
DAG-MLP (0-/1-NTs)	$O(nm)$	$O(nm)$	incomparable

nodes that are indistinguishable by the Weisfeiler-Leman algorithm. The DAGs derived from neighborhood trees have size at most $O(nm)$ for input graphs with n nodes and m edges making the approach computational feasible. We formally show by sensitivity analysis that our approach reduces oversquashing. Our approach achieves high accuracy across various node and graph classification tasks.

2 Related Work

The graph isomorphism network (GIN) [36] is an MPNN that generalizes the Weisfeiler-Leman algorithm, achieving its expressive power. The limited expressivity of simple MPNNs has led to an increased interest in researching more powerful architectures, such as encoding graph structure as additional features or modifying the message passing procedure. Shortest Path Networks [2] use multiple aggregation functions for different shortest path lengths, allowing direct communication with distant nodes. While this might help mitigate oversquashing, information about the structure of the neighborhood still cannot be represented adequately and the gain in expressivity is limited. Distance Encoding GNNs [20] encode the distances of nodes to a set of target nodes. While being provably more expressive than the standard WL algorithm, the approach is limited to solving node-level tasks, as the encoding depends on a fixed set of target nodes and has not been employed for graph-level tasks. MixHop [3] concatenates results from activation functions for each neighborhood, but in contrast to Shortest Path Networks [2], the aggregation is based on normalized powers of the adjacency matrix, not shortest paths, which fails to solve the problem of redundant messages. SPAGAN [38] proposes a path-based attention mechanism, sampling shortest paths and using them as features. However, a theoretical investigation is lacking and the approach utilizes only one layer. Short-rooted random walks in [33] capture long-range dependencies, but have notable limitations due to sampling paths. The evaluation is restricted to node classification datasets and an extensive study of their expressive power is lacking. IDGNN [39] tracks the identity of the root node in unfolding trees, achieving higher expressivity than 1-WL, but failing to reduce redundant information aggregation. PathNNs [27]

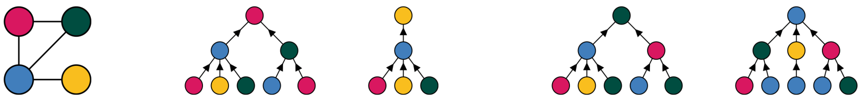


Fig. 1. Graph G and its unfolding trees F_2^v for all $v \in V(G)$.

define path-based trees and a custom aggregation scheme, but overlook exploiting computational redundancy. RFGNNs [9] aim to reduce redundancy by altering the message flow to only include each node (except for the root node) at most once in each path of the computational tree. While this reduces redundancy to some extent, nodes and even the same subpaths may repeatedly occur in the computational trees. The size and running time complexity of RFGNN are very restrictive. While the term breadth-first search used in the definition of the underlying TPTs suggests linear running time, the breadth-first search has to be modified, leading to exponential running time. Additionally, redundancy in computation is not addressed resulting in a highly inefficient preprocessing and computation, limiting the method to a maximum of 3 layers in the experiments.

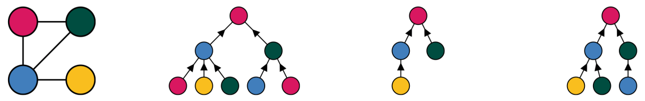
These architectures lack thorough investigation of their expressivity and connections to other approaches. Importantly, they do not explicitly investigate both types of redundancy in MPNNs – redundancy in the information flow and in computation. We compare the time complexity, as well as the expressivity of our method DAG-MLP and other relevant methods in Table 1 and further discuss it in Sect. 4.5.

3 Preliminaries

Graph Theory. A *graph* $G = (V, E, \mu, \nu)$ consists of a set of vertices V, a set of edges $E \subseteq V \times V$ between them, and functions $\mu\colon V \to X$ and $\nu\colon E \to X$ assigning arbitrary attributes to the vertices and edges, respectively.[1] An edge from u to v is denoted by uv, and in undirected graphs $uv = vu$. The vertices and edges of a graph G are denoted by $V(G)$ and $E(G)$, respectively. The *neighbors* (or in-neighbors) of a vertex $u \in V$ are denoted by $N(u) = \{v \mid vu \in E\}$, and the *out-neighbors* of a vertex $u \in V$ are denoted by $N_o(u) = \{v \mid uv \in E\}$. A *multigraph* is a graph, where E is a multiset, allowing multiple edges between a pair of vertices. Two graphs G and H are isomorphic, denoted by $G \simeq H$, if there exists a bijection $\phi\colon V(G) \to V(H)$, such that $\forall u, v \in V(G)\colon \mu(v) = \mu(\phi(v)) \wedge uv \in E(G) \Leftrightarrow \phi(u)\phi(v) \in E(H) \wedge \forall uv \in E(G)\colon \nu(uv) = \nu(\phi(u)\phi(v))$. We refer to ϕ as an *isomorphism* between G and H.

An *in-tree* T is a connected, directed, acyclic graph with a distinct vertex $r \in V(T)$ with no outgoing edges, referred to as *root* ($r(T)$), in which $\forall v \in V(T) \backslash r(T) : |N_o(v)| = 1$. For $v \in V(T) \backslash r(T)$ the *parent* $p(v)$ is the unique vertex $u \in N_o(v)$, and $\forall v \in V(T)$ the *children* are defined as $\mathrm{chi}(v) = N(v)$.

[1] Edge attributes are not considered in the following for clarity of presentation, though the proposed methods can be extended to incorporate them.



(a) Graph G (b) F_2^v (c) $T_{2,0}^v$ (d) $T_{2,1}^v$

Fig. 2. Graph G and the unfolding, 0- and 1-redundant neighborhood trees of height 2 of vertex v (vertex in the upper left of G).

We refer to vertices without incoming edges as *leaves*, denoted by $l(T) = \{v \in V(T) \mid \text{chi}(v) = \emptyset\}$. Conceptually an in-tree is a directed tree, in which there is a unique directed path from each vertex to the root [26]. In our paper, we only consider in-trees and will therefore refer to them simply as *trees*. In-trees are generalized by directed, acyclic graphs (DAGs). The *leaves* of a DAG D and the *children* of a vertex are defined as in trees. However, there can be multiple roots, and a vertex may have more than one parent. We refer to all vertices in D without outgoing edges as *roots*, denoted by $r(D) = \{v \in V(D) \mid N_o(v) = \emptyset\}$, and define the *parents* $p(v)$ of a vertex v as $p(v) = N_o(v)$. The height hgt of a node v is the length of the longest path from any leaf to v: $\text{hgt}(v) = 0$, if $v \in l(D)$ and $\text{hgt}(v) = \max_{c \in \text{chi}(v)} \text{hgt}(c) + 1$, otherwise. The height of a DAG D is defined as $\text{hgt}(D) = \max_{v \in V(D)} \text{hgt}(v)$. For clarity we refer to the vertices of a DAG as nodes to distinguish them from the graphs that are the input of a graph neural network.

Weisfeiler-Leman and Message Passing Neural Networks. The 1-dim. Weisfeiler-Leman (WL) algorithm, also known as *color refinement*, starts with vertices having a color corresponding to their label (or a uniform coloring for unlabeled vertices). In each iteration the vertex color is updated based on the multiset of colors of its neighbors according to

$$c_{\mathsf{wl}}^{(i+1)}(v) = h\left(c_{\mathsf{wl}}^{(i)}(v), \{\!\{c_{\mathsf{wl}}^{(i)}(u) \mid u \in N(v)\}\!\}\right) \quad \forall v \in V(G),$$

where h is an injective function, typically using integers to represent colors.

The color of a vertex encodes its neighborhood through a tree T, which may contain multiple representatives of each vertex. Let $\phi\colon V(T) \to V(G)$ be a mapping such that $\phi(n) = v$ if the node n in $V(T)$ represents the vertex v in $V(G)$. The *unfolding tree* F_i^v with height i of the vertex $v \in V(G)$ consists of a root n_v with $\phi(n_v) = v$ and child subtrees F_{i-1}^u for all $u \in N(v)$, where $F_0^v = (\{n_v\}, \emptyset)$. The attributes of the original graph are preserved, as illustrated in Fig. 1. The unfolding trees F_i^v and F_i^w of two vertices v and w are isomorphic if and only if $c_{\mathsf{wl}}^{(i)}(v) = c_{\mathsf{wl}}^{(i)}(w)$.

Message passing neural networks such as GIN [36] can be seen as a neural variant of the Weisfeiler-Leman algorithm. The embedding of a vertex v in layer i of GIN is defined as

$$x_i(v) = \mathrm{MLP}_i \left((1 + \epsilon_i) \cdot x_{i-1}(v) + \sum_{u \in N(v)} x_{i-1}(u) \right), \tag{1}$$

where the initial features $x_0(v)$ are usually acquired by applying a multi-layer perceptron (MLP) to the vertex features.

4 Non-redundant Graph Neural Networks

We propose to restrict the information flow in message passing to regulate redundancy through the use of k-redundant neighborhood trees. We first develop a neural tree canonization technique, and obtain an MPNN via its application to unfolding trees. Subsequently, we explore computational methods on graph level, reusing information computed for subtrees and derive a customized GNN architecture. Finally, we prove that k-redundant neighborhood trees and unfolding trees are incomparable regarding their expressivity on node-level.

4.1 Removing Information Redundancy

As previously discussed, two vertices obtain the same WL color if and only if their unfolding trees are isomorphic. This concept directly carries over to message passing neural networks and their computational tree [16,31]. However, unfolding trees were mainly used as tools in expressivity analysis and as a conceptual framework for explaining mathematical properties in graph learning [19,30]. We propose a novel perspective on MPNNs through tree canonization. From this perspective, we derive a non-redundant GNN architecture based on neighborhood trees.

In their classical textbook, Aho, Hopcroft, and Ullman [4, Section 3.2] describe a linear time isomorphism test for rooted unordered trees. We give a high-level description to establish the foundation for our neural variant without focusing on the running time. The algorithm proceeds in a bottom-up manner, assigning integers $\mathsf{c}_{\mathsf{ahu}}(v)$ to each node v in the tree. The function f injectively maps a pair consisting of an integer and a multiset of integers to a new, unused integer. Initially, all leaves v are assigned integers $\mathsf{c}_{\mathsf{ahu}}(v) = f(\mu(v), \emptyset)$ based on their label $\mu(v)$. Then internal nodes are processed level-wise in a bottom-up manner, ensuring that whenever a node is processed, all its children have been considered. Hence, the algorithm computes for all nodes v of the tree

$$\mathsf{c}_{\mathsf{ahu}}(v) = f(\mu(v), \{\!\!\{\mathsf{c}_{\mathsf{ahu}}(u) \mid u \in \mathrm{chi}(v)\}\!\!\}). \tag{2}$$

This process ensures the unique representation of non-isomorphic trees, serving as the foundation of our neural tree canonization technique.

GNNs via Unfolding Tree Canonization. Combining Eq. (2) and unfolding trees, denoting the root of an unfolding tree of a vertex v of height i by n_v^i, yields

$$\begin{aligned} \mathsf{c}_{\mathsf{ahu}}(n_v^i) &= f(\mu(n_v^i), \{\!\{\mathsf{c}_{\mathsf{ahu}}(n_u^{i-1}) \mid n_u^{i-1} \in \mathrm{chi}(n_v^i)\}\!\}) \\ &= f(\mu(v), \{\!\{\mathsf{c}_{\mathsf{ahu}}(n_u^{i-1}) \mid u \in N(v)\}\!\}). \end{aligned} \tag{3}$$

By implementing the function f using a suitable neural architecture and replacing its codomain with embeddings in $\mathbb{R}^d$, we readily obtain a GNN based on our canonization approach. The key difference to standard GNNs is that the first component of the pair in Eq. (3) is the initial vertex feature instead of the embedding from the previous iteration. Utilizing the technique proposed by Xu et al. [36] and replacing the first addend in Eq. (1) with the initial embedding, we formulate the *unfolding tree canonization GNN*

$$x_i(v) = \mathrm{MLP}_i\left((1+\epsilon_i)\cdot x_0(v) + \sum_{u\in N(v)} x_{i-1}(u)\right). \tag{4}$$

It is established that MPNNs cannot distinguish two vertices with the same WL color or unfolding tree. Given that the function $\mathsf{c}_{\mathsf{ahu}}(n_v^i)$ uniquely represents the unfolding tree for an injective function f, realizable by Eq. (4) [36], we infer the following proposition.

Proposition 1. *Unfolding tree canonization GNNs, as defined in Eq.* (4)*, are as expressive as GIN, as defined in Eq.* (1)*.*

Despite the equivalence in expressivity, the canonization-based approach avoids redundancy since $x_{i-1}(v)$ represents the entire unfolding tree rooted at v of height $i-1$, while using the initial vertex features $x_0(v)$ is sufficient. We proceed by investigating redundancy within unfolding trees themselves.

GNNs via Neighborhood Tree Canonization. We leverage the concept of neighborhood trees to manage redundancy in unfolding trees.[2] A k-redundant neighborhood tree (k-NT) $T_{i,k}^v$ is derived from the unfolding tree F_i^v by removing all subtrees with roots that occurred more than k levels before (seen from root to leaves). Here, $\mathrm{depth}(v)$ denotes the length of the path from v to the root, and $\phi(v)$ denotes the original vertex in $V(G)$ represented by v in the tree.

Definition 1 (k-redundant Neighborhood Tree). *For $k \geq 0$, the k-*redundant neighborhood tree *(k-NT) of a vertex $v \in V(G)$ with height i, denoted by $T_{i,k}^v$, is defined as the subtree of the unfolding tree F_i^v induced by the nodes $u \in V(F_i^v)$ satisfying*

$$\forall w \in V(F_i^v)\colon \phi(u) = \phi(w) \Rightarrow \mathrm{depth}(u) \leq \mathrm{depth}(w) + k.$$

Figures 2 and 3 provide examples of unfolding and neighborhood trees. It is worth noting that for $k \geq i$ the k-NT is equivalent to the WL unfolding tree.

[2] In a parallel work, neighborhood trees were investigated for approximating the graph edit distance [7].

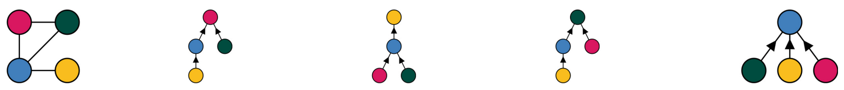
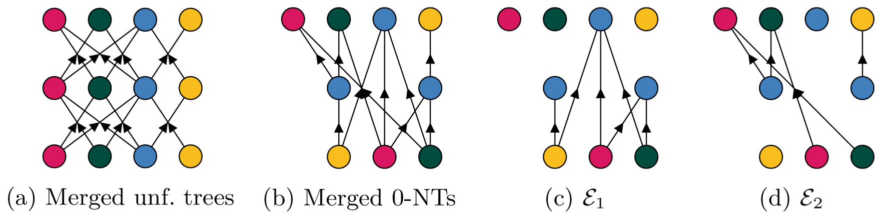


Fig. 3. Graph G and its 0-NTs $T^v_{2,0}$ for all $v \in V(G)$.



(a) Merged unf. trees (b) Merged 0-NTs (c) $\mathcal{E}_1$ (d) $\mathcal{E}_2$

Fig. 4. Computation DAGs for unfolding (a) and 0-NTs (b) of height 2 of graph G. And edges in the different layers of the merge DAG of 0-NTs (c), (d).

We can directly apply the neural tree canonization technique to neighborhood trees. However, a simplifying expression based on the neighbors in the input graph, as given by Eq. (3) for unfolding trees, is not possible for neighborhood trees. Therefore, we explore techniques to systematically exploit computational redundancy.

4.2 Removing Computational Redundancy

The computation DAG of an MPNN involves the embedding of a set of trees representing the vertex neighborhoods of a single or multiple graphs. Results computed for one tree can be reused for others by identifying isomorphic substructures, thereby minimizing computational redundancy. We first describe how to merge trees in a general context and then discuss its application to unfolding and neighborhood trees.

Merging Trees Into a DAG. The neural tree canonization approach developed in the last section can be directly applied to DAGs. Given a DAG D, it computes an embedding for each node n in D that represents the tree F_n obtained by recursively following its children, similar as in unfolding trees, cf. Sect. 3. Since D is acyclic, the height of F_n is bounded. A detailed description of a neural architecture is postponed to Sect. 4.3.

Given a set of trees $\mathcal{T} = \{T_1, \ldots, T_n\}$, a *merge DAG* of $\mathcal{T}$ is a pair (D, ξ), where D is a DAG, $\xi \colon \{1, \ldots n\} \to V(D)$ is a mapping, and for all $i \in \{1, \ldots, n\}$ we have $T_i \simeq F_{\xi(i)}$. The definition guarantees that the neural tree canonization approach applied to the merge DAG produces the same result for the nodes in the DAG as for the nodes in the original trees. A trivial merge DAG is the disjoint union of the trees with $\xi(i) = r(T_i)$. However, depending on the structure of the given trees, we can identify the subtrees they have in common and represent them only once, such that two nodes of different trees share the same child, resulting in a DAG instead of a forest.

We propose an algorithm that builds a merge DAG by successively adding trees to an initially empty DAG, creating new nodes only when necessary. Our approach maintains a canonical labeling for each node of the DAG and computes a canonical labeling for each node of the tree to be added using the AHU algorithm. Then, the tree is processed starting at the root. If the canonical labeling of the root is present in the DAG, then the algorithm terminates. Otherwise, the subtrees rooted at its children are inserted into the DAG by recursive calls. Finally, the root is created and connected to the representatives of its children in the DAG. We introduce a node labeling $L\colon V_T \rightarrow \mathcal{O}$ used for tree canonization, where $V_T = \bigcup_{i=1}^{n} V(T_i)$ and $\mathcal{O}$ an arbitrary set of labels, refining the original node attributes, i.e., $L(u) = L(v) \Rightarrow \mu(u) = \mu(v)$ for all u, v in V_T. When $\mathcal{O}$ consists of integers from the range 1 to $|V_T|$, the algorithm runs in $O(|V_T|)$ time. When two siblings that are the roots of isomorphic subtrees are merged, this leads to parallel edges in the DAG. Parallel edges can be avoided by using a labeling satisfying $L(u) = L(v) \Rightarrow \mu(u) = \mu(v) \wedge p(u) \neq p(v)$ for all u, v in V_T.

Unfolding trees and k-NTs can grow exponentially in size with increasing height. However, this is not case for merge DAGs obtained by the algorithm described above, as we will show below. Moreover, we can directly generate DAGs of size $O(m \cdot (k + 1))$ representing individual k-NTs with unbounded height in a graph with m edges (see Bause et al. [7] for details on generating compact DAGs).

Merging Unfolding Trees. Merging the unfolding trees of a graph with the labeling $L = \phi$ leads to the computation DAG of GNNs. Figure 4a shows the computation DAG for the graph from Fig. 1. The roots in this DAG correspond to the representation of the vertices after aggregating information from the lower layers. Each vertex occurs once at every layer of the DAG, and the links between any two consecutive layers are given by the adjacency matrix of the original graph. While this allows computation based on the adjacency matrix widely-used for MPNNs, it involves the encoding of redundant information. Our method has the potential to compress the computation DAG further by using the less restrictive labeling $L = \mu$, leading to a DAG where at layer i all vertices u, v with $c_{\mathsf{wl}}^{(i)}(u) = c_{\mathsf{wl}}^{(i)}(v)$ are represented by the same node. This compression appears particularly promising for graphs with symmetries.

Merging Neighborhood Trees. When merging k-redundant neighborhood trees in the same way using the labeling $L = \mu$ (or $L = \phi$ to avoid parallel edges), it results in a computation DAG with a more irregular structure, as illustrated in Fig. 4b. Firstly, there might be multiple nodes at the same level representing the same original vertex. Secondly, the adjacency matrix of the original graph cannot be used to propagate the information. A straightforward upper bound on the size of the merge DAG for a graph with n nodes and m edges is $O(nmk + nm)$.

We apply the neural tree canonization approach to the merge DAG in a bottom-up fashion, starting from the leaves and progressing to the roots. Each edge is used exactly once in this computation. Let $D = (\mathcal{V}, \mathcal{E})$ be a merge DAG. The nodes can be partitioned based on their height, leading to $\mathcal{L}_i = \{v \in \mathcal{V} \mid \text{hgt}(v) = i\}$. This induces an edge partition $\mathcal{E}_i = \{uv \in \mathcal{E} \mid v \in \mathcal{L}_i\}$, where all edges with the same end node v are in the same layer, and all incoming edges of children of v belong to a previous layer. Note that, since $\mathcal{L}_0$ contains all leaves of the DAG, there is no $\mathcal{E}_0$. Figures 4c and 4d depict the edge sets $\mathcal{E}_1$ and $\mathcal{E}_2$ for the example merge DAG illustrated in Fig. 4b.

4.3 Non-redundant Neural Architecture (DAG-MLP)

We propose a neural architecture to compute embeddings for nodes in a merge DAG, allowing to retrieve embeddings of the contained trees from their roots. The process involves a preprocessing step that transforms the node labels, using MLP_0 to map them to an embedding space of fixed dimensions. Subsequently, an MLP_i is used to process nodes at each layer $\mathcal{L}_i$.

$$\begin{aligned}
\mu'(v) &= \text{MLP}_0\left(\mu(v)\right), && \forall v \in \mathcal{V}\\
x(v) &= \mu'(v), && \forall v \in \mathcal{L}_0\\
x(v) &= \text{MLP}_i\left((1+\epsilon_i)\cdot \mu'(v) + \sum_{\forall u:\, uv \in \mathcal{E}_i} x(u)\right), && \forall v \in \mathcal{L}_i, i \in \{1,\dots,n\}
\end{aligned}$$

The DAG-MLP can be computed through iterated matrix-vector multiplication analogous to standard GNNs. Let $\mathbf{L}_i$ be a square matrix with ones on the diagonal at position j if $v_j \in \mathcal{L}_i$, and zeros elsewhere. Let $\mathbf{E}_i$ represent the adjacency matrix of $(\mathcal{V}, \mathcal{E}_i)$, and let $\mathbf{F}$ denote the node features of $\mathcal{V}$, corresponding to the initial node labels. The transformed features $\mathbf{F}'$ are obtained through MLP_0, and $\mathbf{X}^{[i]}$ represents the updated embeddings at layer i of the DAG.

$$\begin{aligned}
\mathbf{F}' = \text{MLP}_0\left(\mathbf{F}\right), \quad \mathbf{X}^{[0]} &= \mathbf{L}_0\mathbf{F}',\\
\mathbf{X}^{[i]} = \text{MLP}_i\left((1+\epsilon_i)\cdot \mathbf{L}_i\mathbf{F}' + \mathbf{E}_i\mathbf{X}^{[i-1]}\right) &+ \mathbf{X}^{[i-1]}
\end{aligned}$$

In the above equation, MLP_i is applied to the rows associated with nodes in $\mathcal{L}_i$. The embeddings $\mathbf{X}^{[i]}$ are initialized to zero for inner nodes and computed level-wise. To preserve embeddings from all previous layers, we add $\mathbf{X}^{[i-1]}$ during the computation of $\mathbf{X}^{[i]}$. Suppose the merge DAG (D, ξ) contains the trees $\{T_1, \dots, T_n\}$. We obtain a node embedding $\mathbf{X}^{[n]}_{\xi(i)}$ for each tree T_i with $i \in \{1, \dots, n\}$. This approach allows for obtaining the final embedding for a vertex by using a single tree (Fixed Single-Height) or combining trees of different heights, for example all NTs of size up to a certain maximum (Combine Heights).

Figure 5 shows the DAG-MLP architecture for graph-level tasks. Vertex features are transformed using MLP_0. Messages propagate through the DAG according to $\mathcal{E}_i$. After n layers, all node embeddings have been computed and stored in $\mathbf{X}^{[n]}$. Readouts extract embeddings from k-NTs of varying heights. Pooling operations are applied to each layer's output, and the pooled outputs are averaged and processed by a final MLP for prediction.

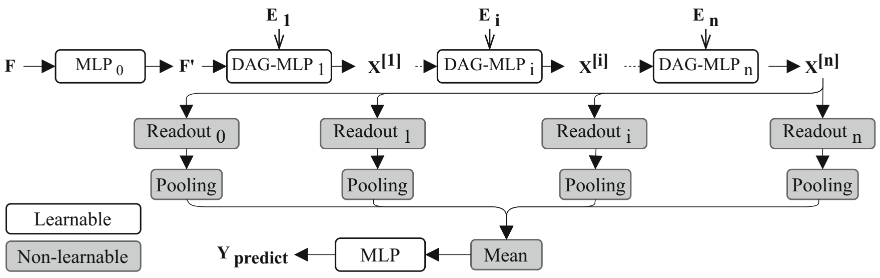


Fig. 5. DAG-MLP architecture for graph-level tasks with n layers. Each layer includes a DAG-MLP, readout, and pooling operations, processing the input $\mathbf{F}$.

4.4 Expressivity of k-NTs

We investigate how expressive k-NTs are compared to unfolding trees. While it is evident that k-NTs are a node invariant, providing the same result for nodes that can be mapped to each other by an isomophism or automorphism, they might also produce the same results for nodes that cannot. This means that, similar to unfolding trees, they are not a complete node invariant.

We show that there are vertices that 1-WL cannot distinguish, but k-NTs can, and vice versa, proving that both methods are incomparable regarding their expressivity on node level. We further conjecture that, while k-NTs cannot distinguish certain vertices that 1-WL can, they can still distinguish the graphs containing such vertices, making k-NTs more expressive on the graph level.

Theorem 1. *The expressivity of k-NT and unfolding trees is incomparable, i.e.,*

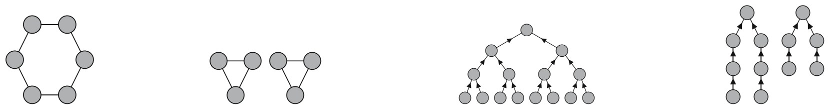
1. $\exists u, v\colon F^u_\infty = F^v_\infty \wedge T^u_{\infty,k} \neq T^v_{\infty,k}$
2. $\exists u, v\colon F^u_\infty \neq F^v_\infty \wedge T^u_{\infty,k} = T^v_{\infty,k}$

Proof. We prove the statement by giving concrete examples. Figure 6 gives an example for 1.: the Weisfeiler-Leman algorithm is unable to distinguish the two graphs, indicating identical unfolding trees of the vertices. However, for any k the k-NT of the vertices will differ for $h \geq k + 2$. Figure 7 gives an example for 2. (for 1-NTs): the unfolding trees of the two marked vertices differ, while the 1-NTs are identical.

We have shown that on node level, the expressivity of k-NTs and unfolding trees is incomparable. However, the examples, where k-NTs fail to distinguish nodes that 1-WL can, can actually be distinguished on the graph level. This arises from the fact, that the graphs have a different number of vertices and the k-NTs of the other nodes differ.

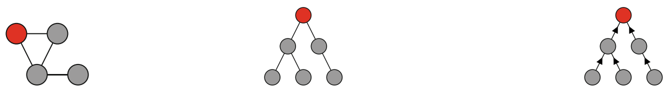
4.5 Theoretical Analysis and Comparison to Related Work

We provide a detailed analysis of the computational complexity and expressivity of our approach compared to related work and formally prove, that our method can mitigate oversquashing better than comparable approaches.



(a) Hexagon (b) Two triangles (c) Unfolding trees (d) 1-NTs

Fig. 6. Two graphs (a), (b) that cannot be distinguished by unfolding trees, but by k-NTs. Figure (c) shows the unfolding tree F_3, which is the same for all vertices of both graphs, while (d) shows the 1-NTs of the vertices in the hexagon (left) and the triangle (right).



(a) G_1 (b) G_2 (c) 1-NTs of red vertices

Fig. 7. Two graphs (a), (b) in which the red vertices can be distinguished by unfolding trees, but not by k-NTs. The 1-NTs of the red vertices (c) are the same. However, G_1 and G_2 can be distinguished by their multisets of 1-NTs. (Color figure online)

Computational Complexity and Expressivity. The DAG representing the k-NT of a single vertex has a size in $O(mk+m)$, where m is the number of edges in the graph. The lexicographic encoding and merging of k-NTs to generate the DAG can be done in time linear in its size. A trivial upper bound on the size of the merge DAG of a graph with n nodes and m edges is $O(nmk+nm)$. Overall, this means that preprocessing can be done in $O(nmk)$ time, where k can be considered constant. For 0- and 1-NTs, we obtain time $O(nm)$.

Table 1 compares the complexity and expressivity of our method to related work. Understanding and relating the expressivity of different approaches is non-trivial. In the case of PathNet, its expressivity concerning the WL-hierarchy remains unexplored. PathNN-SP+ has been shown to be more expressive than 1-WL. While it is claimed that RFGNNs are maximally expressive, the proof claiming higher expressivity on node level as presented in Chen et al. [9, Lemma 7] is not correct (rather it is incomparable on node level), which was shown in Bause et al. [6]. Consequently, it remains uncertain, whether RFGNN is strictly more expressive on graph level. PathNN-SP [27] states that it can only disambiguate graphs at least as well as 1-WL and is not strictly more powerful. Due to sampling, isomorphic graphs could be mapped to different representations, indicating that it is not a graph invariant.

RFGNN and PathNN-SP+ involve enumerating all possible paths and all shortest paths, respectively. Our approach avoids this redundancy by not explicitly building NTs, but instead generate DAGs, resulting in a much more compact representation (refer to Bause et al. [7] for an algorithm on how to generate compact DAGs). We investigate the theoretical connection of our method and RFGNN in terms of relative influence, and show that 0- and 1-NTs can address the oversquashing problem more effectively.

Oversquashing. Several authors [8,9,14,34,37] developed and refined techniques to measure the influence of a vertex v with initial vertex feature $\mathbf{x}_v$ on the output $\mathbf{h}_u^{(l)}$ of a vertex u after layer l by the Jacobian $\partial\mathbf{h}_u^{(l)}/\partial\mathbf{x}_v$. Following Chen et al. [9, Lemma 3], the relative influence of a vertex v on a vertex u in an MPNN is

$$I(v,u) = \mathbb{E}\left(\frac{\partial\mathbf{h}_u^{(l)}/\partial\mathbf{x}_v}{\sum_{w\in V}\partial\mathbf{h}_u^{(l)}/\partial\mathbf{x}_w}\right) = \frac{[\hat{A}^l]_{u,v}}{\sum_{w\in V}[\hat{A}^l]_{u,w}},$$

where $\hat{A} = A + I$ is the adjacency matrix of G with added self-loops, and $[\hat{A}^l]_{u,v}$ is the number of walks of length l from u to v (and vice versa) in G with added self-loops. Oversquashing occurs when $I(v,u)$ becomes small, indicating that only a small fraction of walks of length up to l ending at u start at v.

This idea can easily be linked to the trees underlying our work. Consider the unfolding tree F_l^u of u with height l. It follows from its construction that there is a bijection between walks of length at most l ending at u in G and paths in F_l^u from some node to the root (see [18] for details on unfolding trees and walks). Therefore, pruning the unfolding tree has an effect on walk counts and, thus, on the relative influence. Consider the example in Fig. 1 and let u be the red vertex (upper left) and v the yellow vertex (lower right). We obtain a relative influence of $I_{\mathrm{MPNN}}(v,u) = \frac{1}{8}$ for unfolding trees, and $I_{0\mathrm{NT}}(v,u) = \frac{1}{4}$ for 0-NTs, showing that NTs have the potential to reduce oversquashing.

We formally show that our method is less susceptible to oversquashing than MPNNs and RFGNN [9]. Consider a vertex v and a vertex u with shortest-path distance of l. To pass information from v to u, at least l layers are required. Comparing the unfolding tree underlying MPNNs, the 0- and 1-NT, and the TPT used in RFGNN, all of height l, reveals that the vertex v occurs in the last level only, i.e., as a leaf of the tree, and the number of occurrences of v is equal in all trees, since all walks and simple paths of length l reaching v are shortest paths. Hence, the numerator of the relative influence is equal for all methods. However, since 0- and 1-NTs are subtrees of unfolding trees, and 0-NTs/1-NTs are subtrees of TPTs (they contain only shortest paths/some simple paths, instead of all simple paths), the total number of nodes in the trees, i.e., walks contributing to the denominator of the relevant information can be compared, obtaining

$$I_{\mathrm{MPNN}}(v,u) \leq I_{\mathrm{TPT}}(v,u) \leq I_{1\mathrm{NT}}(v,u) \leq I_{0\mathrm{NT}}(v,u).$$

This theoretical analysis shows that our proposed method offers advantages in mitigating oversquashing, leveraging the formalization developed in recent papers. Additionally, it establishes a theoretical connection between the proposed approach and RFGNN [9], highlighting that our method more effectively addresses the oversquashing problem.

To summarize, our method is the first to address both types of redundancy in GNNs, informational and computational redundancy, with polynomial running time, and can mitigate oversquashing better than comparable approaches.

5 Experimental Evaluation

We assess the performance of DAG-MLP with k-NTs on a range of synthetic [1,29] and real-world datasets [10,12,25,28,32].[3]

Experimental Setup. For synthetic datasets, we determine the number of layers in DAG-MLP based on the average graph diameter, ensuring effective aggregation during message propagation. The embeddings at each layer are obtained using readouts, concatenated, and then fed through two learnable linear layers for prediction. For TUDataset, we use the 10-fold splits proposed by Errica et al. [11], allowing a grid search for optimal hyper-parameters. The architecture for combined heights involves using each "readout$_i$" to extract the embeddings for each layer, with mean of the average-pooled embeddings being passed to a final MLP layer responsible for prediction. For the fixed single-height architecture, only the last readout is used, pooled, and passed to the final MLP layer.

Graph Classification. Table 2 presents the results on synthetic expressivity datasets. Our hypothesis that NTs are more expressive than GIN on graph level is experimentally validated, but a theoretical proof remains future work.

In Table 3, we investigate the impact of the parameter k and the number of layers l on the accuracy for the EXP-Class dataset. Cases where $k > l$ can be disregarded, as the computation for NTs remains the same as when $k = l$. Empirically, 0- and 1-NTs yield the highest accuracy. This observation aligns with our discussions on expressivity in Sect. 4.4. The decrease in accuracy with increasing k indicates that information redundancy leads to oversquashing. We investigate this theoretically in Sect. 4.5.

For TUDataset, we report the accuracy compared to related work in Table 4. We report only the best results for the different parameter combinations reported in Michel et al. [27], and the best result for our different combine methods. We group the methods by their time complexity. Note that, while PathNN performs well on ENZYMES and PROTEINS, the time complexity of this method is exponential. Therefore, we also highlight the best method with polynomial time complexity. For IMDB-B and IMDB-M, which have small diameters, k-NTs outperform all other methods. For ENZYMES a variant of our approach achieves the best result among the approaches with non-exponential time complexity, and k-NTs lead to a significant improvement over GIN.

Node Classification. We investigate the performance of our approach on node classification datasets. These datasets differ regarding their homophily ratio, i.e., the fraction of edges in a graph that connect vertices with the same class label [40]. Heterophily tasks are particularly challenging for standard GNNs [40] as they require capturing the structure of neighborhoods instead of "averaging" over the neighboring features. In Table 5 we present results from Giraldo et

[3] The implementation is available at github.com/samirmoustafa/k-redundancyGNNs.

Table 2. Average classification accuracy for EXP-Class and CSL across k-folds (4-folds and 5-folds), and the number of indistinguishable pairs of graphs in EXP-Iso. Best results are highlighted in gray, best results from methods with polynomial time complexity are highlighted in **bold**.

Model	**EXP-Class** ↑	**EXP-Iso** ↓	**CSL** ↑
GIN [36]	50.0 ± 0.0	600	10.0 ± 0.0
3WLGNN [24]	**100.0 ± 0.0**	**0**	97.8 ± 10.9
PathNN-$\mathcal{SP}^+$ [27]	100.0 ± 0.0	0	100.0 ± 0.0
PathNN-$\mathcal{AP}$ [27]	100.0 ± 0.0	0	100.0 ± 0.0
DAG-MLP (0-NTs)	**100.0 ± 0.0**	**0**	**100.0 ± 0.0**
DAG-MLP (1-NTs)	**100.0 ± 0.0**	**0**	**100.0 ± 0.0**

Table 3. Average accuracy for DAG-MLP using 4-fold cross-validation on EXP-Class [1], evaluated with varying number of layers.

k-NTs	1 layer	2 layers	3 layers	4 layers	5 layers	6 layers
0-NTs	51.1 ± 1.6	57.5 ± 6.6	91.7 ± 11.6	99.7 ± 0.3	**100.0 ± 0.0**	**100.0 ± 0.0**
1-NTs	50.1 ± 0.2	58.9 ± 4.6	59.4 ± 5.7	99.6 ± 0.5	99.9 ± 0.2	**100.0 ± 0.0**
2-NTs	–	52.6 ± 3.4	54.9 ± 5.3	52.4 ± 3.8	97.6 ± 1.9	**100.0 ± 0.0**
3-NTs	–	–	56.2 ± 5.7	51.1 ± 1.9	52.4 ± 4.1	87.1 ± 21.4
4-NTs	–	–	–	50.1 ± 0.2	50.6 ± 1.0	50.4 ± 0.7
5-NTs	–	–	–	–	50.4 ± 0.7	50.0 ± 0.0
6-NTs	–	–	–	–	–	53.2 ± 5.2

al. [15] including the state-of-the-art graph rewiring technique SJLR combined with SGC and GCN, which performs best in the evaluation. We also performed experiments with GIN and DAG-MLP, using the same data splits as Giraldo et al. [15] to ensure a fair comparison. We report the best results for l layers with $l \in \{2, 3, 4\}$ and four different combine methods for GIN and DAG-MLP.

DAG-MLP outperforms GIN on heterophily datasets (those with low homophily ratio), while GIN performs better on homophily ones, indicating that neighborhood trees can capture the relevant neighborhood structure more accurately than unfolding trees used by GIN. Additionally, our method outperforms SJLR on heterophily datasets Texas and Wisconsin by a large margin.

6 Conclusion

We introduce a neural tree canonization technique and combine it with neighborhood trees, which are pruned versions of unfolding trees used by standard MPNNs. By merging trees in a DAG, we create compact representations that

Table 4. Classification accuracy (± standard deviation) over 10-fold cross-validation on the datasets from TUDataset, taken from Michel et al. [27]. Best results highlighted in gray, best results from methods with polynomial time complexity highlighted in **bold**. "-": not applicable, "NA": not available.

	Model	**IMDB-B**	**IMDB-M**	**ENZYMES**	**PROTEINS**
Linear	GIN [36]	71.2 ± 3.9	48.5 ± 3.3	59.6 ± 4.5	73.3 ± 4.0
	GAT [35]	69.2 ± 4.8	48.2 ± 4.9	49.5 ± 8.9	70.9 ± 2.7
	SPN ($l = 1$) [2]	NA	NA	67.5 ± 5.5	71.0 ± 3.7
	SPN ($l = 5$) [2]	NA	NA	69.4 ± 6.2	**74.2 ± 2.7**
Exp	PathNet [33]	70.4 ± 3.8	49.1 ± 3.6	69.3 ± 5.4	70.5 ± 3.9
	PathNN-$\mathcal{P}$ [27]	72.6 ± 3.3	50.8 ± 4.5	73.0 ± 5.2	75.2 ± 3.9
	PathNN-$\mathcal{SP}^+$ [27]	-	-	70.4 ± 3.1	73.2 ± 3.3
Ours	DAG-MLP (0-NTs)	**72.9 ± 5.0**	50.2 ± 3.2	67.9 ± 5.3	70.1 ± 1.7
	DAG-MLP (1-NTs)	72.4 ± 3.8	**51.3 ± 4.4**	**70.6 ± 5.5**	70.2 ± 3.4

Table 5. Classification accuracy (± standard deviation) on node classification tasks (GCN, SJLR-GCN, SGC and SJLR-GCN taken from Giraldo et al. [15]).

Model	Texas	Wisconsin	Cornell	Cora	CiteSeer	PubMed
Homophily ratio	0.11	0.21	0.3	0.8	0.74	0.8
GCN	58.05 ± 0.9	52.10 ± 0.9	67.34 ± 1.5	81.81 ± 0.2	68.35 ± 0.3	78.25 ± 0.3
SJLR-GCN	60.13 ± 0.8	55.16 ± 0.9	71.75 ± 1.5	**81.95 ± 0.2**	**69.50 ± 0.3**	**78.60 ± 0.3**
SGC	56.69 ± 1.7	47.90 ± 1.7	53.40 ± 2.1	76.90 ± 1.3	67.45 ± 0.8	71.79 ± 2.1
SJLR-SGC	58.40 ± 1.4	55.42 ± 0.9	67.37 ± 1.6	81.24 ± 0.7	68.39 ± 0.6	76.28 ± 0.9
GIN	73.78 ± 6.0	71.76 ± 5.1	60.81 ± 8.5	76.76 ± 1.4	64.49 ± 1.5	76.46 ± 1.1
DAG-MLP (0-NTs)	**85.68 ± 4.8**	81.35 ± 4.1	79.02 ± 6.8	74.01 ± 2.0	60.55 ± 3.6	75.33 ± 1.1
DAG-MLP (1-NTs)	80.54 ± 6.0	**81.62 ± 3.4**	**79.41 ± 4.6**	74.54 ± 1.4	61.09 ± 1.5	75.53 ± 1.1

serve as the foundation for our neural architecture termed DAG-MLP. It inherits the advantageous properties of the GIN architecture, while being more expressive than 1-WL on many graphs. Notably, our method is only less expressive on node level for specific examples. Our work contributes general techniques for constructing compact computation DAGs for tree structures that encode node neighborhoods. This exploration reveals a complex interplay between information redundancy, computational redundancy, and expressivity. The delicate balance of these factors is an avenue for future work.

Acknowledgments. We would like to thank Christian Permann for his contribution to the conception of k-redundant neighborhood trees and their efficient generation. This work was supported by the Vienna Science and Technology Fund (WWTF) [10.47379/VRG19009]. The computational results presented have been achieved in part using the Vienna Scientific Cluster (VSC).

References

1. Abboud, R., Ceylan, I.I., Grohe, M., Lukasiewicz, T.: The surprising power of graph neural networks with random node initialization. In: IJCAI (2021)
2. Abboud, R., Dimitrov, R., Ceylan, İ.İ.: Shortest path networks for graph property prediction. LoG (2022)
3. Abu-El-Haija, S., Perozzi, B., et al.: MixHop: higher-order graph convolutional architectures via sparsified neighborhood mixing. In: ICML (2019)
4. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Computer Algorithms. Addison-Wesley (1974)
5. Alon, U., Yahav, E.: On the bottleneck of graph neural networks and its practical implications. In: ICLR (2021)
6. Bause, F., Moustafa, S., Langguth, J., Gansterer, W.N., Kriege, N.M.: On the two sides of redundancy in graph neural networks. CoRR abs/2310.04190 (2023)
7. Bause, F., Permann, C., Kriege, N.: Approximating the graph edit distance with compact neighborhood representations. In: ECML/PKDD (2024)
8. Black, M., Wan, Z., Nayyeri, A., Wang, Y.: Understanding oversquashing in GNNs through the lens of effective resistance. In: ICML (2023)
9. Chen, R., Zhang, S., U, L.H., Li, Y.: Redundancy-free message passing for graph neural networks. In: NeurIPS (2022)
10. Craven, M.W., et al.: Learning to extract symbolic knowledge from the world wide web. In: AAAI/IAAI (1998)
11. Errica, F., Podda, M., Bacciu, D., Micheli, A.: A fair comparison of graph neural networks for graph classification. In: ICLR (2020)
12. Giles, C.L., Bollacker, K.D., Lawrence, S.: Citeseer: an automatic citation indexing system. Digital library (1998)
13. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: ICML (2017)
14. Giovanni, F.D., Giusti, L., Barbero, F., Luise, G., Lio, P., Bronstein, M.M.: On over-squashing in message passing neural networks: The impact of width, depth, and topology. In: ICML (2023)
15. Giraldo, J., Skianis, K., Bouwmans, T., Malliaros, F.: On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In: CIKM (2023)
16. Jegelka, S.: Theory of graph neural networks: Representation and learning. CoRR abs/2204.07697 (2022)
17. Jia, Z., Lin, S., Ying, R., You, J., Leskovec, J., Aiken, A.: Redundancy-free computation for graph neural networks. In: KDD (2020)
18. Kriege, N.M.: Weisfeiler and Leman go walking: random walk kernels revisited. In: NeurIPS (2022)
19. Kriege, N.M., Giscard, P.L., Wilson, R.C.: On valid optimal assignment kernels and applications to graph classification. In: NIPS (2016)
20. Li, P., Wang, Y., Wang, H., Leskovec, J.: Distance encoding: Design provably more powerful neural networks for graph representation learning. In: NeurIPS (2020)
21. Li, Q., Han, Z., Wu, X.: Deeper insights into graph convolutional networks for semi-supervised learning. In: AAAI (2018)
22. Liu, M., Gao, H., Ji, S.: Towards deeper graph neural networks. In: KDD (2020)
23. Mahé, P., Ueda, N., Akutsu, T., Perret, J.L., Vert, J.P.: Extensions of marginalized graph kernels. In: ICML (2004)

24. Maron, H., Ben-Hamu, H., Serviansky, H., Lipman, Y.: Provably powerful graph networks. In: NeurIPS (2019)
25. McCallum, A., Nigam, K., Rennie, J.D.M., Seymore, K.: Automating the construction of internet portals with machine learning. Information Retrieval (2000)
26. Mehlhorn, K., Sanders, P.: Algorithms and Data Structures: The Basic Toolbox. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-77978-0
27. Michel, G., Nikolentzos, G., Lutzeyer, J., Vazirgiannis, M.: Path neural networks: expressive and accurate graph neural networks. In: ICML (2023)
28. Morris, C., Kriege, N.M., Bause, F., Kersting, K., Mutzel, P., Neumann, M.: TUDataset: a collection of benchmark datasets for learning with graphs. In: ICML GRL+ Workshop (2020)
29. Murphy, R., Srinivasan, B., Rao, V., Ribeiro, B.: Relational pooling for graph representations. In: ICML (2019)
30. Nikolentzos, G., Chatzianastasis, M., Vazirgiannis, M.: Weisfeiler and Leman go hyperbolic: Learning distance preserving node representations. In: AISTATS (2023)
31. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: Computational capabilities of graph neural networks. IEEE Trans. Neural Networks (2009)
32. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Mag. (2008)
33. Sun, Y., et al.: Beyond homophily: structure-aware path aggregation graph neural network. In: IJCAI (2022)
34. Topping, J., Giovanni, F.D., Chamberlain, B.P., Dong, X., Bronstein, M.M.: Understanding over-squashing and bottlenecks on graphs via curvature. In: ICLR (2022)
35. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
36. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: ICLR (2019)
37. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: ICML (2018)
38. Yang, Y., Wang, X., Song, M., Yuan, J., Tao, D.: SPAGAN: shortest path graph attention network. In: IJCAI (2019)
39. You, J., Selman, J.M.G., Ying, R., Leskovec, J.: Identity-aware graph neural networks. In: AAAI (2021)
40. Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., Koutra, D.: Beyond homophily in graph neural networks: current limitations and effective designs. In: NeurIPS (2020)

Policy Control with Delayed, Aggregate, and Anonymous Feedback

Guilherme Dinis Junior(✉), Sindri Magnússon, and Jaakko Hollmén

Stockholm University, Stockholm, Sweden
{guilherme,sindri.magnusson,jaakko.hollmen}@dsv.su.se

Abstract. Reinforcement learning algorithms have a dependency on observing rewards for actions taken. The relaxed setting of having fully observable rewards, however, can be infeasible in certain scenarios, due to either cost or the nature of the problem. Of specific interest here is the challenge of learning a policy when rewards are delayed, aggregated, and anonymous (DAAF). A problem which has been addressed in bandits literature and, to the best of our knowledge, to a lesser extent in the more general reinforcement learning (RL) setting. We introduce a novel formulation that mirrors scenarios encountered in real-world applications, characterized by intermittent and aggregated reward observations. To address these constraints, we develop four new algorithms: one employs least squares for true reward estimation; two and three adapt Q-learning and SARSA, to deal with our unique setting; and the fourth leverages a policy with options framework. Through a thorough and methodical experimental analysis, we compare these methodologies, demonstrating that three of them can approximate policies nearly as effectively as those derived from complete information scenarios, albeit with minimal performance degradation due to informational constraints. Our findings pave the way for more robust RL applications in environments with limited reward feedback.

Keywords: control · reinforcement learning · delayed feedback · aggregate feedback

1 Introduction

Sequential decision making problems exist in many domains, from manufacturing to networks, with applications ranging from small to large scale settings. One of the most common formulations of sequential decision making problems is reinforcement learning (RL). In such formulations, we have an agent making decisions in order to achieve an objective that has been encoded as a reward signal. RL has been widely adopted in industrial applications. In the energy

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_23.

sector, for instance, it has been used to control the temperature in buildings [18,30], as well as for dynamic pricing [7] and demand management of electrical grids [13]. In network systems, it has been used to dynamically manage load for both network efficiency [32] and throughput [9]. Recent advances have garnered further interest in RL, resulting in its deployment in areas such as ride ordering and car pooling [12,15,27,29], recommender systems [4,5], and to generate feeds of digital content [31,33].

While early RL algorithms for learning and evaluating policies rely on perfect observability of the interaction of a policy with the environment, many real world applications have constraints. These constraints limit the signal available to agents, which can make learning difficult, if not impossible, without compensating strategies. As a result, problems with limited feedback settings have been studied for the past few decades. In literature, the term *delay* in relation to feedback can carry different meanings. It can refer to rewards observed at a future time, though still clearly connected to a specific decision. It can also refer to feedback that is distributed in time, with partial observations at each time step. In this paper, we focus on a problem that lies between the ideal setting of reward signals available at each step and those where reward signals are only available at the end. The problem is one of delayed rewards, observed on aggregate for several actions and thus anonymous with respect to each specific action since it is unclear how much each action contributed to the observed feedback.

In industrial settings, this scenario is common for instance in decision or recommender systems that rely on third party or external signals for attribution, such as marketing and market-places. In the latter case, for instance, there is often aggregate data from sellers on sales, which can be tied to a user but not a specific impression. While these problems have been studied extensively for standard click-based optimisation algorithms, they have been less investigated for sequential decision making settings, where there is a time dependency between actions. Our work is a step in that direction.

Delayed, aggregated and anonymous feedback (DAAF) is presented in [19] for multi-armed bandits, which are a special case of RL where there is only one state. In this paper, we expand the study of DAAF in three ways. First, we formulate the problem for the more general RL setting, and study policy control. Second, we draw a connection between Non-markovian decision processes and DAAF. Third and finally, we develop new methods for learning policies more efficiently under constrained feedback, and demonstrate their efficacy through extensive empirical experiments on different environments, with varying degrees of difficulty. The main research questions we raise are **RQ1** what impact does delayed, aggregate and anonymous feedback have on policy control? and **RQ2** how might we develop methods for sample efficient policy control under DAAF?

2 Related Work

We discuss the four areas of research most related to our work: adversarial markov decision process (MDP)s, inverse reinforcement learning (IRL), credit assignment, and non-Markovian rewards. We start with adversarial MDPs [11,16], which study environments where rewards are observed based on an unknown distribution of time, set by an adversarial agent. The defining trait of these MDPs is that rewards are delayed, but observable and tied to a specific state-action. The rewards in our problem are observed with either a known or unknown delay, yet they are also aggregated, so that one cannot discern the contribution of each state-action.

For the second area, we have inverse reinforcement learning, the subject of which is deriving a reward function from observations. It is typically employed to extract knowledge from experts to then design RL agents, a task known as learning from imitation. In [20], this problem is formulated as a supervised learning task, where feature maps representing state-action pairs are used to learn rewards that can maximize the similarity of a learned policy to that of an expert policy or trajectory. Later works have extended this problem, with either novel formulations or new conditions. Both [22] and [8] sought to incorporate expert feedback on trajectories to constrain the learned reward functions and policies, the former with a binary success and failure labeling, and the latter with scoring. Others, such as [6] and [21], expanded IRL to MDPs with partially observable and hidden states, respectively, allowing for imitation learning under limited observations of state. Unlike the traditional IRL setting, we consider problems where the observation of rewards is limited, instead of fully absent, and the observed rewards are delayed, aggregated and anonymous. Our approach differs in that we seek to leverage the rewards available and their structure to constrain the reward function we estimate.

For the third area, credit assignment, we can draw a parallel between our work and that in [1]. The challenge they solve is one of single-step structural credit attribution. Given a multi-agent system, at every step all agents make a decision, and the environment gives a global signal in return. The task is then to determine how each agent contributed to the global signal. To solve that, the authors use agent-centric utility functions. Our setting has a time-domain credit attribution challenge for different state-actions, instead; and the equivalent of multi-agent utility functions in our setting are the rewards for each state-action, and that forms the basis of our reward recovery approach.

Finally, there are studies on non-Markovian rewards, which span a wide range of topics. For instance, the authors in [26] address the problem of maximizing rewards when an agent can actively choose to observe the reward for an action at a cost. Their proposed solution relies on estimating the gain of a reward observation, and pay the cost whenever the gain is deemed valuable. Our problem is distinct from theirs in that the agent in our setting cannot decide when to observe a reward, because this is controlled by a known or unknown random process. In [2], the authors address the problem of sample efficiency when learning policies in environments where rewards are only observed at the end of an episode, i.e. sparse rewards. Their solution relies on a theorem built on the idea that learning can be accelerated by re-distributing rewards to earlier time-steps, and learning a return-equivalent policy afterwards. The distinction with our work is that we focus on a different setting where there are intermittent aggregate rewards rather than sparse rewards; and furthermore, their experiments rely on function approximation with deep learning, whilst we aim to study the problem in more simplistic conditions with tabular RL and derive insights on the impact learning the true rewards. In contrast, [10] presents a closer setting of rewards observed in intervals that are sampled uniformly at random from a range of values. Though, their study is centered on policy gradient methods, whilst we address the tabular case.

Turning to theory, a framework that describes a generalization that can be leveraged in our setting is introduced in [25]. The authors present MDPs with options, which are semi-MDPs. Options are actions that last for more than one time step. The reward for an option is computed when an option ends and another starts, and this can be after a set of known or unknown steps. Since in our setting a policy can make several decisions before observing a reward, we can recast DAAF environments as trajectories of options policies, where the options are composite actions (e.g. a set of actions taken in a specific order). With this lens, we leverage existing theory on MDPs with options to study control for our setting.

Our problem is similar to that presented in [3], where delayed, aggregated and anonymous feedback (DAAF) is studied the bandit setting. For control problems, the authors do so by playing the same action repeatedly, and estimate its average based on the number of steps. To the best of our knowledge, our work is the first to extend the problem of policy control with DAAF to the tabular RL setting, where there are state transitions.

3 Problem Formulation

3.1 Preliminaries

We consider the typical reinforcement learning (RL) setting, where an agent wants to learn a policy to maximize rewards in an unknown environment. The environment is modeled as an MDP with a set of states $\mathbf{S}$, actions $\mathbf{A}$, transition dynamic $\mathbf{T}(S, A, S')$ and rewards $(S, A) \rightarrow R$. A state S_t encapsulates information about the environment at time step t. An action A_t is chosen by the agent given the state $\pi(A_t \mid S_t)$ - which is a density function indicating the probability of action A given that we are in state S - upon which the environment transitions into a new state S_{t+1} and the agent receives a reward R_{t+1}. Thus the agent follows the learning trajectory

$$S_0, A_0, R_1, S_1, A_1, R_2, ..., S_{T-1}, A_{T-1}, R_T$$

where T is the length of the episode, which is finite for episodic tasks and infinite for continuing tasks. The goal of the agent is to maximize future cumulative rewards, i.e. the sum of rewards from future states $\sum_{t=0}^{T} \gamma^t R_{t+1}$, where $\gamma \in (0, 1]$ is a discount factor that can be chosen to favor near-term rewards ($\gamma < 1$) or give all rewards equal weight ($\gamma = 1$).

A policy π has a state-value or action-value function, represented as $V_\pi(S)$ and $Q_\pi(S, A)$, respectively. The state-value function $V_\pi(S)$ tells us the maximum returns from being in state S and using π to make decisions starting from that state. The action-value function $Q_\pi(S, A)$ tells us the maximum returns from being in state S, taking a specific action A, and from there using the policy π to make decisions. One of the fundamental problems in RL is to learn a policy that has the highest return amongst all policies. This policy is called an optimal policy, π^*, and it has value functions $V^*_\pi(S)$ and $Q^*_\pi(S, A)$.

3.2 Policy Control

Algorithms to learn policies that act optimally in an environment were developed as early as the late nineteen eighties. Temporal difference (TD) methods presented in [23] and Q-learning [28] are two well known algorithms. Some of these algorithms follow the off-policy paradigm, where the agent learns using trajectory data generated by a different policy. And others follow the on-policy paradigm, where the agent interacts with an environment, either directly or through a simulator, to generate trajectory data. What they all have in common is the use rewards observed for actions chosen by a policy to learn. For an overview of policy control algorithms, we refer the reader to [24].

3.3 Delayed, Aggregate, Anonymous Feedback

In this paper, we study policy control when rewards are observed with some delay. Additionally, they are observed on aggregate, in that the rewards observed at time t correspond to the sum of rewards for the last P steps. This aggregation makes the contribution of each state-action in the feedback window anonymous. In our experiments, we make P a constant, but our solutions easily extend to P coming from any discrete and bounded distribution $p \sim \tau(t)$. Mathematically, with a constant P, if we denote by R^o_t the reward signal observed by the agent at time t then we have:

$$R^o_t = \begin{cases} \sum_{i=t-P+1}^{t+P} R_i & \text{if (t mod P = 0)} \\ \emptyset & \text{otherwise.} \end{cases} \tag{1}$$

Here $\emptyset$ denotes the empty reward, meaning that the agent receives no reward signal. Whenever the feedback is observed the value of the equation is equivalent to:

$$R^o_t = \sum_{i=t-P+1}^{t+P} R(s, a)_i. \tag{2}$$

One known method for policy control is SARSA [24], and it uses the update rule:

$$Q(S, A) \leftarrow Q(S, A) + \alpha * (R + \gamma * Q(S', A') - Q(S, A)) \tag{3}$$

Q-learning has a similar update rule to SARSA, with the distinction that the value used for the next state-action pair is $argmax_a Q(S', A')$. Without a reward, the update function of the algorithms cannot be executed. In the case of DAAF, only a fraction of transition steps have an observed reward value - albeit, an aggregate one. This creates two issues: (1) sample inefficiency and (2) biased value updates. The first issue, sample inefficiency, is due to the fact that many transitions can be ignored, and the fraction of transitions ignored is proportional to the reward period. For example, if the reward period is fixed as $P = 5$, then we would observe an aggregate reward on every fifth step, and update the value function estimate while ignoring the previous four steps. This would effectively mean we would reject 80% of the data in this case. More generally, $\frac{P-1}{P}$ transitions are ignored. The second issue, though, can have more dire consequences. As we increase the reward period P, we reduce data efficiency but also bias the value of the true state-action rewards, since the observed reward is compounded from multiple state transitions. We study these issues, through extensive empirical experiments where we simulate DAAF. To overcome the aforementioned issues created with DAAF, we propose four approaches. The first employs least squares for true reward estimation; two and three adapt Q-learning and SARSA, to deal with our unique setting; and the fourth leverages a policy with options framework. We describe each approach in the coming sections.

4 Algorithms Development

Linear Estimation of State-Action Rewards. The first method we propose is to attempt to recover the underlying reward function with data. Doing so gives us a reward model, $\hat{R}(s, a)$, which we can use with standard policy control methods such as SARSA, with the learned reward:

$$Q(S, A) \leftarrow Q(S, A) + \alpha * (\hat{R} + \gamma * Q(S', A') - Q(S, A)) \tag{4}$$

In principle, this approach enables us to use any of the pre-existing policy control algorithms, and the results hinge on the bias of our reward model. For problems with deterministic dynamics, we present *linear estimation of state-action rewards (LEAST)* - an algorithm for recovering the reward function. With the learned rewards, we can replace both the aggregate anonymous and missing rewards in a trajectory with their estimate, $\hat{R}_t$.

To understand how LEAST works, we first take note of the structure of the problem. Say the aggregated anonymous rewards are observed at fixed time step intervals, P. We denote by $R(s, a)$ the average reward from the state-action pair (s, a). Then our goal is to use the data observed from the aggregate rewards to estimate $R(s, a)$ for all pairs (s, a). This is naturally formulated as a least-squares problem. Mathematically, we can denote by the vector of $R(s, a)$ for all pairs (s, a), i.e., $x = [R(s, a)]_{(s,a)}$ with $(s, a) \in S \times A$.

We can define a matrix $B \in \mathbb{R}^{N \times (|S| * |A|)}$ and vector $c \in \mathbb{R}^N$ such that our estimated reward is the solution to the least squares problem [14]:

$$\min_x \mid\mid Bx - c \mid\mid_2^2 \tag{5}$$

Each row in the matrix B is constructed from a single reward window P, and each column corresponds to the number of observations of state-action pairs within that same window. Our factors to be learned, x, are the average rewards for each state-action pair that when multiplied by a row i in B yield c_i, the DAAF observed in the window. To illustrate this structure, assume we have an MDP with two states and one action, and a reward of 10 for every action in any state. Assuming our reward period $P = 2$, and that over two transitions we observe both states and the same action, an entry row for our regression estimation would be $B_i = [1, 1]$ with each 1 indicating a single observation for both $(s = 0, a = 0)$ and $(s = 1, a = 0)$; and $c_i = 20$ the undiscounted DAAF observed.

Our formulation is similar to that described by the authors in [20] for the task of learning a reward function for a policy that mimics some trajectory data through supervised learning. When the approximation of $R(s, a)$ is unbiased, LEAST can provide a more reliable observation of returns to estimate $V_\pi(s)$ or $Q_\pi(s, a)$. Note that this formulation also works on problems where the reward period is sampled from any discrete and bounded distribution $p \sim \tau(t)$ within an episode, i.e. the delay changes in each window.

Impute Missing Rewards. The second method, as the name suggests, is an approach whereby we simply make the assumption that an absent reward corresponds to a reward value of zero. The intuition is that because the observed feedback is aggregate, it is still worth updating the steps in-between feedback steps with values that do not alter the episodic return (zero being a base assumption) to allow for more frequent TD updates. Note that the rewards used for estimating $Q_\pi(s, a)$ can still differ from the true rewards. However, this method requires no computation, and the returns observed from a given *starting* state are closer to the true returns when the discount factor $\gamma = 1$, matching exactly when the episode ends with the observation of aggregate feedback. Otherwise, the learned $Q_\pi(s, a)$ can be biased.

N-Step TD with Selective Updates. As a third method, we introduce a modification to the standard n-step SARSA algorithm. The change we make is to only update steps that lie $t - n$ steps before the current time step t when t is a step on which we observe aggregate feedback, i.e. $(t \mod P) = 0$. This variant is called *nTD-SU*. Following the update rule for n-step SARSA, these updates are expected to be unbiased when $\gamma = 1$; however, skipping updating other states should make the variant less sample efficient. We do this to simulate how one would realistically use aggregate feedback with n-step methods.

MDP with Options. Options, as described in [25], are action choices that last longer than one step. Each option can be comprised of one or more actions, called primitive actions, which can be shared amongst options, e.g. going up or down, picking up an object or dropping it. More formally, an option is comprised of a triplet $< I, \pi, \beta >$, where I is a set initiation states where the option can be chosen; π is a policy that chooses primitive actions; and β is a stochastic termination function, and when it returns 1, we terminate the current option and choose another. For our formulation, we assume every state is a valid initiation state, and termination happens P steps after the option starts. Since an option can take several primitive actions until it terminates, the reward in options is the sum of discounted rewards for the duration of the option:

$$r_s^o = E\{r_{t+1} + \gamma r_{t+2} + ... + \gamma^{k-1} r_{t+k} \mid \varepsilon(o, s, t)\} \tag{6}$$

where $\varepsilon(o, s, t)$ denotes the event that option o is initiated in state s at time t. And the transition function similarly follows:

$$p_{ss'}^o = \sum_{k=1}^{\infty} p(s^{'}, k)\gamma^k \tag{7}$$

where $p(s^{'}, k)$ is the probability the option ends at state $s^{'}$ after k steps, and γ^k factors in the transition delay.

Upon close inspection, Eq. 6 has some striking similarity to the aggregate reward definition in Eq. 2. In fact, if we consider a sequence of actions, we get an MDP with options where the options are composite actions. The distinction lies in that options assume rewards are observed per primitive action step, and can thus be discounted accordingly. Since with DAAF the only reward we observe is aggregate with respect to the a sequence of actions, this reward corresponds to the reward for an option specified in Eq. 6, when $\gamma = 1$. It follows that we can learn a policy that uses sequences of actions as options in undiscounted settings.

The authors in [25] suggest that for episodic tasks, if the goal of the underlying MDP coincides with the terminal state of an option, there are advantages to using options, but otherwise, it can be sub-optimal. We extend this analysis, with an observation about the dynamics. Let us consider an environment with DAAF, where delays are fixed as a constant P, with $P > 1$. To observe feedback for options, we have to use options that last for exactly P steps. Assuming we have deterministic environments, there is no guarantee that episodic termination will coincide with the observation of a aggregate reward. For example, if the options are of length P, but the episode ends one step after an option starts. Such a condition can prevent a policy from learning how to act in the final stages of the environment. Turning to environments with stochastic dynamics, can we fair any better? On the one hand, non-deterministic dynamics may allow options to coincide with episodic termination in some cases. However, they also make it difficult to learn the best option in a state because the state where the option terminates and its aggregate reward would vary in proportion to the number of actions and reward period. Consider a maze without obstacles where the agent has to navigate to an exit by moving up, down, left or right. If every action has a ϵ probability moving the agent in its opposite direction, then the larger the reward period P gets, the more possible ending states and rewards each option has. While it may be possible to learn an optimal policy that is as good as a primitive actions policy, under certain conditions, the process would be less efficient with respect to data samples. Thus, our expectation is that addressing problems with DAAF using options policies can lead to sub-optimal solutions, but adequate results under the specific conditions described beforehand.

5 Experiments

For our experiments, we evaluate impute missing rewards (IMR), LEAST, and options policies of composite actions on two single-step TD algorithms: SARSA and Q-learning. For n-step algorithms, we evaluate IMR, LEAST and n-step-SU on n-step SARSA. The reason for this is that the parameter n in n-step SARSA defines how many steps of look ahead are used to update the value of a state-action, and only options policies with option of length n would have updates, which in turn would reduce the problem to standard SARSA on an options trajectory that we already evaluate.

For each method and algorithm combination, we vary the discount ($\gamma \in \{1, 0.99\}$), the reward period ($P \in \{2, 4, 6, 8\}$), and use exploration rate of $\epsilon = 20\%$, and a learning rate of $\alpha = 0.1$. We test each configuration by running policy learning for 2500 episodes, 20 times with different seeds. We note that for LEAST, we wait until we observe every state-action combination to get a full ranked matrix B and estimate the rewards. Thus, before we have the reward estimates, we use IMR and switch to the predicted rewards once available. The samples used to learn the reward for LEAST are factored into our evaluation, by being part of the same 2500 episodes every method gets. Lastly, due to their memory requirements, we could only run options policies experiments with $P = 6$ as a maximum, but provide results of larger values of P for the other methods.[1] The code for our experiments, including notebooks with analyses of the results, are available in github[2].

Baselines. For the one step TD methods SARSA and Q-learning, our baselines are (1) to drop missing rewards (DMR), since we cannot perform TD updates and (2) to use options policies; and for multi step TD method n-step SARSA, we use nTD-SU as the baseline. Under scenarios with undiscounted reward, i.e. $\gamma = 1$, both the options policy and n-TD-SU are expected to yield unbiased updates. Additionally, in order to understand the gap in performance for each our proposed methods, we carry out on-policy learning with rewards observed at each step for each environment and policy learning algorithm. This serves as a benchmark and oracle.

[1] An environment where $P = 8$, $|S| = 30$ and $|A| = 6$ yields a Q-table with 50M values for an options policy.

[2] https://github.com/dsv-data-science/rl-daaf/tree/ec21d0c9cd581d2e641ba20e138f7a621d1ffc06.

Environments. We use a set of environments to represent a range of tasks with different challenges, and with commonalities with real world applications including from navigation, discovery, sequence ordering, and item sorting.

Grid World - an environment where the goal is to navigate from a starting position to an exit, avoiding falling into cliffs; from Gymnasium[3]; *Ice World* - an environment similar to Grid World, with the distinction that there are holes in the ice, and falling into one ends the game. An agent fails to reach their goal in the episode in such case. There are two configurations: a 4x4 and 8x8 grid; *ABC Sequence* - an environment where the goal is to select a set of actions in order, each corresponding to the correct next step in the sequence; this environment has configurable complexity, since the number of actions grows with the number of states. A parameter n defines the number of states and actions; *RedGreen* - an environment where the goal is to choose one of three options (red, green, wait) correctly at each step of a given sequence. We use a sequence of length $n = 9$ (red, green, wait, green, red, red, green, wait); *Tower of Hanoi* - given n disks stacked in one of three pegs, from smallest at the top to the largest at the bottom, the goal is to move all disks from the leftmost peg to the rightmost peg, whilst never placing a large disk on top of a smaller one. More details are provided in the Appendix.

5.1 Returns

Our first results are the return curves for select environments, due to space constraints. In Fig. 1 we have plots for Q-learning and in Fig. 2 for n-step SARSA on *Grid World*, *Ice World (8x8)*, and *Hanoi Tower*. The results for SARSA are similar to Q-learning. Starting with Fig. 1, the plots show how the performance of options policies degrade as the reward period P increases across the three environments. This pattern occurs in every environment tested. With more and lengthier options (actions) due to longer reward delay, signal propagation becomes more inefficient. Generally, DMR achieves better results than the options policy, while IMR and LEAST have better results than both. Note that in *Ice World*, every method we test degrades in performance as the reward period increases, but they maintain their performance rank (IMR/LEAST > DMR > Options).

Turning to n-step SARSA in Fig. 2, we can observe the delay in learning for nTD-SU compared to the both IMR and LEAST. In the Ice World results, there is higher variance in the returns for every method, including the full rewards policies, due to the nature of the problem, whereby several terminal states with different returns exist. And in some of the GridWorld configurations, LEAST had worst returns in some of the runs. This occurred in cases where the estimated reward was incorrect for certain state-actions - in particular for state-actions near the goal, leading to delayed termination.

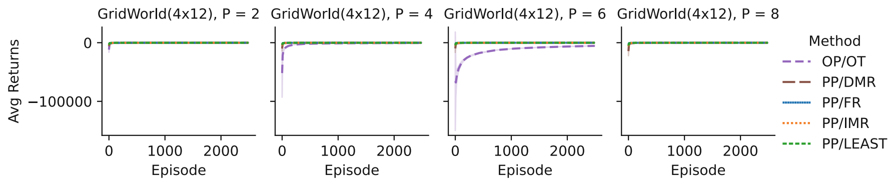
[3] https://gymnasium.farama.org/.

5.2 Sample Efficacy

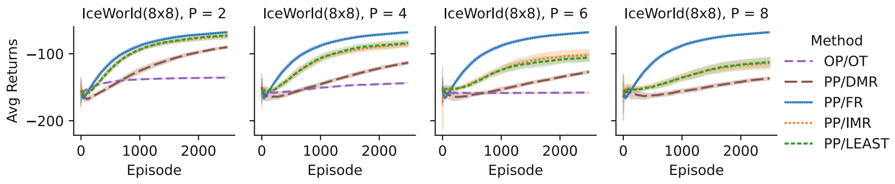
To measure sample efficacy, we compute a returns ratio using a policy learned with full rewards (FR) as the oracle. First, we take the average return over all episodes at the final episode K, $\bar{G}_K = \frac{1}{K}\sum_{k=1}^{K} G_k$, where G_k is the return for an episode $k \in [1...K]$. We then average the value over the 20 runs for each configuration, yielding $\mu_{\bar{G}_K}$ Finally, we compute a returns ratio with the formula:

$$\frac{\mu_{\bar{G}^m_K} - \mu_{\bar{G}^{FR}_K}}{|\mu_{\bar{G}^{FR}_K}|} \tag{8}$$

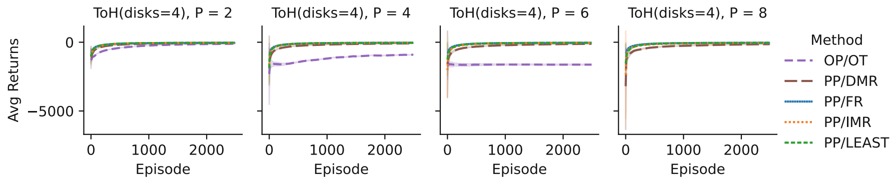
Returns ratio measures the percentage difference in average return between the policy trained with method m compared to the full rewards policy, as a ratio. A positive number indicates the method performs better than a full rewards policy, and a negative value indicates the opposite. The results are given in Fig. 3a and 3b, for one-step and n-step algorithms respectively. For one step



(a) Grid World, with $\gamma = 0.99$

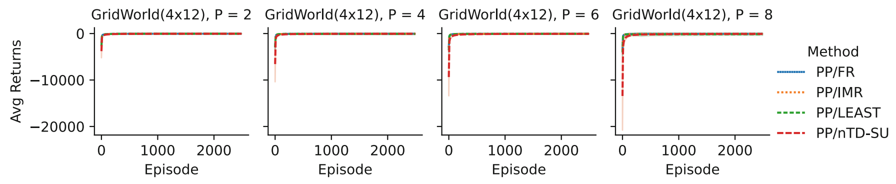


(b) Ice World (8x8), with $\gamma = 1$

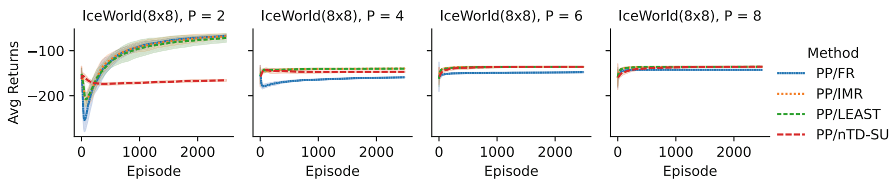


(c) Tower of Hanoi (disks=4), with $\gamma = 0.99$

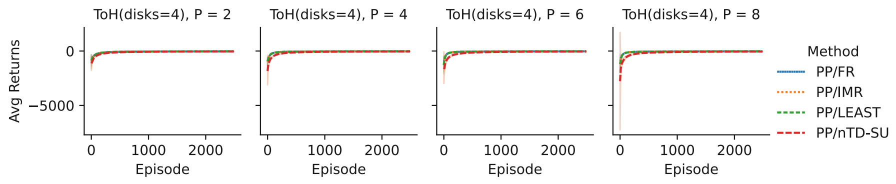
Fig. 1. Return plots for on-policy control using Q-learning. The solid blue line represents returns for policies learned with full rewards, while the others are the methods under comparison. Each line is the average and standard deviation of 20 runs. Note: results for options policies are limited to $P = 6$ due to its memory requirements. (Color figure online)



(a) Grid World, with $\gamma = 0.99$



(b) Ice World (8x8), with $\gamma = 1$



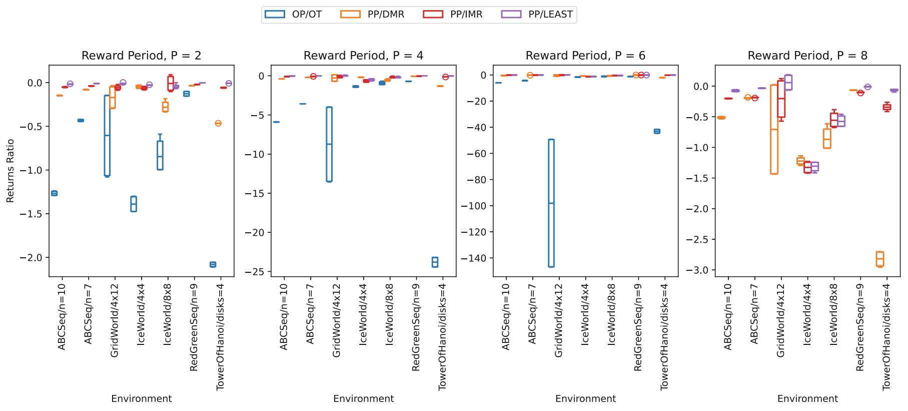
(c) Tower of Hanoi (disks=4), with $\gamma = 0.99$

Fig. 2. Return plots for on-policy control using n-step SARSA. The solid blue line represents returns for policies learned with full rewards, while the others are the methods under comparison. Each line is the average and standard deviation of 20 runs.

methods, the options policy has worse returns ratio than IMR and LEAST for every environment, with returns worsening as the delay increases. In the majority of cases, LEAST either has the best returns ratio or it performs on par with IMR. The exception is the environment *Ice World (4x4)*, where DMR has the best returns ratio when $P > 2$. The pattern with n-step SARSA is similar for most environments, with nTD-SU having the worst returns ratio. Here, the exceptions are *Grid World* and *Ice World (4x4) & (8x8)* where IMR has a better returns ratio than LEAST.

5.3 Stochastic Similarity to Full Rewards Policies

To study the difference in returns between the methods under comparison and policies with full rewards information, we average the returns from the 20 runs for each configuration and run a one-sided non-parametric Mann-Whitney U test [17] comparing them to returns from the full rewards policies. The null hypotheses is that the results come from the same distribution (i.e. stochastically similar), and the alternative is that the results are lower. The *p-value* for the test



(a) Results for options policy (OP/OT), DMR (PP/DMR), IMR (PP/IMR) and LEAST (PP/LEAST) on primitive actions policies. Averaged across SARSA and Q-learning, and discount factors $\gamma \in \{0.99, 1\}$.

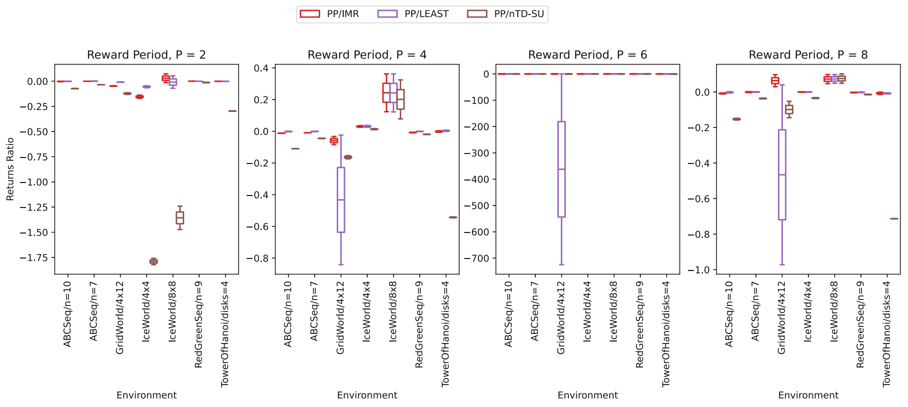


Fig. 3. The returns ratio for each method compared to control with full rewards. Values closer to zero indicate parity with full rewards, positive values indicate better performance than full rewards, and negative values indicate worse returns than full rewards.

is 0.05. We then count the configurations where the null hypothesis is rejected (Less than) and failed to be rejected (Similar) for each method in the case of one-step and n-step methods, separately. In the case of one step methods, DMR has similar results to the full rewards policy in 3.8% of the configurations (out of 160), the options policies in 4.2% (out of 94), IMR in 7.5% (out of 160) and LEAST in 10% (out of 160). For n-step, the outcomes are substantially different. On the one hand, the returns ratio for LEAST has higher worse case values than the other methods in *Grid World*. Despite that, the proportion of configurations with results statistically similar to the policies with full rewards are 81.3% (out of 64) for LEAST, 52% (out of 64) for IMR, and 14.1% for the baseline nTD-SU (out of 64).

6 Discussion

The results obtained in the experiments corroborate our analysis from Sect. 4. When carrying out control with DAAF, the performance of options policies is inversely related to delay, as their action space grows exponentially with it. Out of our proposed methods, IMR and LEAST are more sample efficient. This particularly striking considering that LEAST first collects data to estimate rewards, and still manages to achieve a better returns ratio than the other methods in all but few cases: with *Ice World (4x4)* on one-step TD methods, where DMR is better; and for *Grid World* and *Ice World (4x4) & (8x8)* when used with n-step SARSA, where IMR is better.

Both IMR and LEAST have results that can be, based on Mann-Whitney U statistical testing, indistinguishable from the results of a policy with full rewards in a reasonable number of cases when used with n-step SARSA (37.9% and 61.3% respectively) indicating we can do well despite the limited rewards information. With the exclusion of two environments, LEAST's better returns ratio would align with our hypothesis that recovering the rewards can lead to less biased policy control. Both *Grid World* and *Ice World*, where LEAST did not perform best, are the more challenging environments, though the reasons for the results with LEAST are different in each one. For the former, *Grid World*, we encountered cases where reward estimation was incorrect for certain actions just before the terminal state. This is because the observed data in those instances had many possible solutions, since the rewards in terminal states are zero. By analysing reward estimation in other environments, we found that this problem isn't exclusive to *Grid World*. The results with LEAST for *Ice World* have a different reason: there are several terminal states (ice holes) that prevent the agent from reaching the preferred exit. In fact, in *Ice World 8x8*, even the policy with full rewards did not reach the goal in any of our runs - thus, its returns are based on ending the episode early by falling onto the nearest ice hole. In these cases, the matrix required to estimate the true rewards with LEAST was incomplete, and thus LEAST was equivalent to IMR. Such idiosyncrasies serve as guide for challenges to be addressed in future work, which we discuss in Sect. 7. The stochastic similarity of results for IMR and LEAST to the full rewards policy when coupled with n-step SARSA is likely an artifact of the efficiency of temporal feedback propagation inherit in n-step methods, which our methods can leverage well; n-step methods, after all, combine the best of both worlds between monte-carlo and TD approaches.

One advantage of IMR is that it requires no additional computational resources. Its superior performance against the baselines makes it a viable initial candidate for settings with DAAF. For LEAST, the main advantage is that its structure only depends on observing every state-action pair, or the most relevant ones, irrespective of the delay. Approximating the reward function with LEAST requires resources of the order $O(|A| * |S| * m)$ for storage and a one-time $O(|A|^2 * (m + |A|))$ compute cost, where m is the number of samples collected. This can make it unsuitable for environments with constrained resources. Despite this, the expectation of lower relative regret with the bounded compute costs make it, in our view, an attractive solution for real-world applications along with IMR when compute resources are available. Furthermore, should the data be too large to fit in memory, one can resort to gradient based methods for least squares, which allow smaller samples to be processed at a time, making computation viable for large problems.

It is worth stating that many applications where DAAF would be observed are managed settings, where it is possible to choose P. Consider, for instance, networking systems where we wish to reduce the frequency at which nodes communicate feedback to a central control system. The savings in networking, which are typically the highest cost in such settings, are proportional to the delay. With our proposed solutions, it is now possible to achieve those savings without sacrificing performance as much.

One final aspect we deliberate on is the bias in IMR. We hypothesized that using aggregate feedback as the reward for a given state-action pair and zero otherwise would yield a biased Q_π. What we have observed is that, despite these biased updates, the agent can still learn a reasonably good policy. Our hypothesis for this is that this bias does not prevent the agent from learning which action is the best in a given state, at least in expectation, which is why IMR performs adequately well.

7 Conclusion

In this paper, we study policy control in environments with DAAF. We formalise the problem and use the framework of MDPs with options to study it. We propose four methods for policy control with DAAF, and run extensive experiments to validate their performance. From our analyses, we conclude that MDPs with options are ill suited for problems with fixed long delays while our other proposed methods IMR and LEAST, on the other hand, can allow policy control with higher returns than the baselines, in exchange for extra compute and storage in the case of LEAST.

There are several avenues to extend the work presented here. For starters, we have focused our study on environments with finite state spaces. Many real-world applications though require very large or continuous state spaces. Thus, studying reward imputation and formulating reward recovery in such environments is of interest. Second, one of our methods assumes a linear relationship between state-actions and aggregate rewards. Whilst this assumption makes it practical to use reword recovery in our setting, its effectiveness in environments with non-linear reward aggregation is unknown. Addressing that would allow wider application of the solution. Third, with LEAST we solve an unconstrained optimisation problem, which leads to incorrect estimates in cases where the samples collected yield a matrix for which the solution for certain state-action pairs can be many. Given prior knowledge about the environment, e.g. the terminal states which are generally known, one can solve a constrained optimisation problem, and reduce errors in the estimation. Finally, several applications of RL involve optimising a composite reward function that is made up of different objectives. Disentangling the effect of a sequence of actions on each objective is an interesting problem area to explore as well, to expand the range of applications of RL.

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

References

1. Agogino, A.K., Tumer, K.: Unifying temporal and structural credit assignment problems. In: AAMAS '04, USA, pp. 980–987. IEEE Computer Society (2004)
2. Arjona-Medina, J.A., Gillhofer, M., Widrich, M., Unterthiner, T., Brandstetter, J., Hochreiter, S.: RUDDER: return decomposition for delayed rewards. In: Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019). https://proceedings.neurips.cc/paper/2019/hash/16105fb9cc614fc29e1bda00dab60d41-Abstract.html
3. Cesa-Bianchi, N., Gentile, C., Mansour, Y.: Nonstochastic Bandits with Composite Anonymous Feedback, pp. 750–773. PMLR (2018). iSSN: 2640-3498
4. Chen, H., et al.: Large-scale interactive recommendation with tree-structured policy gradient **33**(1), 3312–3320 (2019). number: 01
5. Chen, M., Beutel, A., Covington, P., Jain, S., Belletti, F., Chi, E.H.: Top-k off-policy correction for a REINFORCE recommender system. In: WSDM '19, pp. 456–464. Association for Computing Machinery (2019)
6. Choi, J., Kim, K.E.: Inverse reinforcement learning in partially observable environments. J. Mach. Learn. Res. **12**, 691–730 (2011)
7. Christensen, M.H., Ernewein, C., Pinson, P.: Demand response through price-setting multi-agent reinforcement learning. In: RLEM'20, pp. 1–5. Association for Computing Machinery
8. El Asri, L., Piot, B., Geist, M., Laroche, R., Pietquin, O.: Score-based inverse reinforcement learning. In: AAMAS '16, International Foundation for Autonomous Agents and Multiagent Systems, pp. 457–465 (2016)
9. Gupta, Y., Bhargava, L.: Reinforcement learning based routing for cognitive network on chip. ICTCS '16, pp. 1–3. Association for Computing Machinery (2016)
10. Han, B., Ren, Z., Wu, Z., Zhou, Y., Peng, J.: Off-policy reinforcement learning with delayed rewards. In: Proceedings of the 39th International Conference on Machine Learning, pp. 8280–8303. PMLR (2022). https://proceedings.mlr.press/v162/han22e.html, iSSN: 2640-3498
11. Jin, C., Jin, T., Luo, H., Sra, S., Yu, T.: Learning adversarial Markov decision processes with bandit feedback and unknown transition. In: Proceedings of the 37th International Conference on Machine Learning, pp. 4860–4869. PMLR (2020). iSSN: 2640-3498
12. Jindal, I., Qin, Z.T., Chen, X., Nokleby, M., Ye, J.: Optimizing taxi carpool policies via reinforcement learning and spatio-temporal mining, pp. 1417–1426 (2018)
13. Kathirgamanathan, A., Twardowski, K., Mangina, E., Finn, D.P.: A centralised soft actor critic deep reinforcement learning approach to district demand side management through CityLearn. In: RLEM'20, pp. 11–14. Association for Computing Machinery
14. Lawson, C.L., Hanson, R.J.: Least-squares approximation, pp. 963–964. John Wiley and Sons Ltd., GBR (2003)

15. Li, M., et al.: Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In: WWW '19, pp. 983–994. Association for Computing Machinery (2019)
16. Luo, H., Wei, C.Y., Lee, C.W.: Policy optimization in adversarial MDPs: improved exploration via dilated bonuses. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems. vol. 34, pp. 22931–22942. Curran Associates, Inc. (2021)
17. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. **18**(1), 50–60 (1947)
18. Naug, A., Quiñones-Grueiro, M., Biswas, G.: Continual adaptation in deep reinforcement learning-based control applied to non-stationary building environments. In: RLEM'20, pp. 24–28. Association for Computing Machinery (2020)
19. Pike-Burke, C., Agrawal, S., Szepesvari, C., Grunewalder, S.: Bandits with delayed, aggregated anonymous feedback. In: Proceedings of the 35th International Conference on Machine Learning, pp. 4105–4113. PMLR (2018). iSSN: 2640-3498
20. Ratliff, N.D., Bagnell, J.A., Zinkevich, M.A.: Maximum margin planning. In: ICML '06, New York, NY, USA, pp. 729–736. Association for Computing Machinery (2006)
21. Shahryari, S., Doshi, P.: Inverse reinforcement learning under noisy observations. In: AAMAS '17, pp. 1733–1735. International Foundation for Autonomous Agents and Multiagent Systems (2017)
22. Shiarlis, K., Messias, J., Whiteson, S.: Inverse reinforcement learning from failure. In: AAMAS '16, International Foundation for Autonomous Agents and Multiagent Systems, pp. 1060–1068. (2016)
23. Sutton, R.S.: Learning to predict by the methods of temporal differences. Mach. Lang. **3**(1), 9–44 (1988)
24. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, Adaptive Computation and Machine Learning Series, 2nd edn. The MIT Press, Cambridge (2018)
25. Sutton, R.S., Precup, D., Singh, S.: Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. Artif. Intell. **112**(1), 181–211 (1999)
26. Tucker, A.D., Biddulph, C., Wang, C., Joachims, T.: Bandits with costly reward observations. In: Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence, pp. 2147–2156. PMLR (2023). iSSN: 2640-3498
27. Wang, Z., Qin, Z., Tang, X., Ye, J., Zhu, H.: Deep reinforcement learning with knowledge transfer for online rides order dispatching, pp. 617–626. ISSN: 2374-8486
28. Watkins, C., Dayan, P.: Q-learning. Machine Learning (2004)
29. Xu, Z., et al.: Large-scale order dispatch in on-demand ride-hailing platforms: a learning and planning approach. In: KDD '18, pp. 905–913. Association for Computing Machinery (2018)
30. Zhang, C., Kuppannagari, S.R., Kannan, R., Prasanna, V.K.: Building HVAC scheduling using reinforcement learning via neural network based model approximation. In: BuildSys '19, pp. 287–296. Association for Computing Machinery (2019)
31. Zhao, Y., Zhou, Y.H., Ou, M., Xu, H., Li, N.: Maximizing cumulative user engagement in sequential recommendation: an online optimization perspective. In: KDD '20, New York, NY, USA, pp. 2784–2792. Association for Computing Machinery (2020)

32. Zheng, H., Louri, A.: An energy-efficient network-on-chip design using reinforcement learning. In: DAC '19, pp. 1–6. Association for Computing Machinery (2019)
33. Zou, L., Xia, L., Ding, Z., Song, J., Liu, W., Yin, D.: Reinforcement learning to optimize long-term user engagement in recommender systems. In: KDD '19, New York, NY, USA, pp. 2810–2818. Association for Computing Machinery (2019)

DiffVersify: a Scalable Approach to Differentiable Pattern Mining with Coverage Regularization

Thibaut Chataing[1,2], Julien Perez[3], Marc Plantevit[3](✉), and Céline Robardet[2]

[1] PALO IT, Lyon, France
[2] INSA Lyon, CNRS, LIRIS UMR 5205, 69621 Villeurbanne, France
[3] EPITA Research Laboratory (LRE), 94276 Le Kremlin-Bicêtre, France
marc.plantevit@epita.fr

Abstract. Pattern mining addresses the challenge of automatically identifying interpretable and discriminative patterns within data. Recent approaches, leveraging differentiable approach through neural autoencoder with class recovery, have achieved encouraging results but tend to fall short as the magnitude of the noise and the number of underlying features increase in the data. Empirically, one can observe that the number of discovered patterns tend to be limited in these challenging contexts. In this article, we present a differentiable binary model that integrates a new regularization technique to enhance pattern coverage. Besides, we introduce an innovative pattern decoding strategy taking advantage of non-negative matrix factorization (NMF), extending beyond conventional thresholding methods prevalent in existing approaches. Experiments on four real-world datasets exhibit superior performances of DiffVersify in terms of the ROC-AUC metric. On synthetic data, we observe an increase in the similarity between the discovered patterns and the ground truth. Finally, using several metrics to finely evaluate the quality of the patterns in regard to the data, we show the global effectiveness of the approach.

1 Introduction

Pattern mining is a crucial field for extracting meaningful and easily interpretable insights from data. Traditional frequent pattern mining techniques [24], while widely used, often fail to capture all the underlying regularities in the data and tend to produce results that are overly general and redundant. To address this limitation, various techniques [6,10] have emerged, aimed at identifying a smaller yet more informative set of patterns. However, these approaches are computationally intensive due to the use of enumeration-based strategies on large search space. Thus, they often struggle to scale effectively, particularly in scenarios with large and complex datasets. To mitigate these challenges, many methods use heuristic approaches [3,8] and consider data of limited size, particularly on the number of features. This restricted scope excludes many potential application areas, such as biological and large-scale complex problems.

A. Bifet et al. (Eds.): ECML PKDD 2024, LNAI 14946, pp. 407–422, 2024.
https://doi.org/10.1007/978-3-031-70365-2_24

The differentiable pattern mining framework, as introduced in recent works [9,21], represents a significant advancement in leveraging neural network architectures to extract interpretable relations from data. Grounded in neuro-symbolic learning principles, it harnesses the computational power of neural networks to learn fully interpretable patterns within a constrained neural architecture. In the seminal paper by Fischer et al. [9], a novel binarized autoencoder is proposed to uncover human-interpretable sets of conjunctive patterns using gradient-based optimization. This model projects input data into an interpretable latent space, striving to faithfully reconstruct the data from patterns encoded in this space. Interpretability is achieved through the use of binary weights and activations during the forward pass, while scalability is ensured through efficient continuous optimization during backpropagation. Building upon this foundation, Walter et al. [21] extend the framework to learn patterns that can effectively differentiate between classes. Their approach combines a binary autoencoder with a classifier attached to the hidden layer, allowing for joint optimization of reconstruction and classification tasks. This integrated model enhances the interpretability of learned patterns while enabling effective classification of data instances based on these patterns.

While the neural autoencoder with class recovery shows promise, particularly on high-dimensional data, its performance tends to degrade as noise magnitude and the number of underlying classes and features in the data increase. In such scenarios, patterns may become redundant, leaving a substantial portion of the data uncovered. To address this limitation, we propose integrating a novel regularization technique into the differentiable binary model, aimed at promoting the extraction of patterns that provide better coverage of the data while enforcing pattern diversity. Our experiments demonstrate that the orthogonality regularization term in the loss function yields significant improvements in pattern extraction. Additionally, we introduce an innovative pattern decoding strategy that utilizes non-negative matrix factorization (NMF), extending beyond conventional thresholding methods prevalent in existing approaches. This robust and original decoding strategy adapts well to diverse datasets and enhances the overall performance of the model.

The experiments show the scalability of the proposed DiffVersify method concerning the number of features, classes and noise levels, which are pivotal factors in real-world pattern mining scenarios. The evaluation of the effectiveness of the approach on synthetic datasets shows its ability to improve the detection of ground truth patterns with the increase, compared to the baselines, of their similarity with the extracted patterns. Additionally, DiffVersify demonstrates superior performance in terms of the ROC-AUC metric across four real-world datasets. Recognizing that the assessment of pattern collections associated with classes requires more than just supervised classification measures, we introduce novel evaluation metrics to better characterize the appropriateness of discovered patterns relative to the data, with a particular focus on pattern coverage. Our results reveal the effectiveness of DiffVersify on this aspect.

Section 2 reviews related literature. Section 3 introduces notations and concepts while Sect. 4 outlines the approach; Sect. 5 presents the experiments, and Sect. 6 resumes the contributions and discusses their limitations.

2 Related Work

The problem of pattern set and association rule mining was introduced as a method for identifying local structures within data [1]. Rule-based classification, as studied in [5,11,15,19], aims to derive interpretable classification conjunctive rules. Despite featuring interpretability, these methods predominantly prioritize prediction over description, leading to a loss of important contextual details. Furthermore, their reliance on combinatorial optimization techniques hampers their scalability, particularly when applied to high-dimensional datasets. Neuro-symbolic classification [7,14,22] offers a solution to these computational limitations. These methods devise neural architectures that, following training, allow for the extraction of symbolic classification rules. Despite their focus on optimization, these approaches share similarities with traditional rule-based classifiers in their emphasis on classification accuracy rather than descriptive rule discovery. Association discovery research domain experienced a period of robust activity characterized by a plethora of studies, yielding significant insights. Following this first collection of work, the field experienced a resurgence with the introduction of a pioneering neural approach [9], revitalizing research efforts and bringing renewed attention to the domain. In [9], Fischer and Vreeken propose a novel approach, BiNaps, for discovering high-quality and noise-robust pattern sets. Unlike existing methods limited by combinatorial search, BiNaps employs a gradient-based optimization strategy, bridging the discrete search space and continuous optimization. This approach involves a neural autoencoder with binary activations and binarized weights, termed BiNaps, which directly represent conjunctive patterns. By optimizing a data-sparsity aware reconstruction loss, the authors achieve effective pattern discovery, demonstrating scalability to real-world datasets such as supermarket transactions and biological datasets. The patterns are effectively decoded using a thresholded binarisation of the weight matrix of the model after convergence.

In [21], the authors build on BiNaps to propose DiffNaps, a novel binary neural network architecture that builds class-specific patterns. Similarly to BiNaps, DiffNaps also uses a binary autoencoder but combined to a separate classification head. The model is learnt by jointly optimizing reconstruction and classification. Succinct class-specific patterns are promoted thanks to elastic-net regularizers.

Table 1 details the composition of the respective losses and pattern decoding strategies of the main state-of-the-art approaches of the recent literature. While numerous attempts have proposed to take into account various elements of constraints for pattern discovery, our approach DiffVersify proposes two novel contributions. First, we explicitly promote diversity over the resulting pattern set

Table 1. Comparison of the losses and pattern decoding of baselines and DiffVersify.

		RL-NET [7]	RRL [22]	R2N [14]	BiNaps [9]	DiffNaps [21]	DiffVersify
Losses	Reconstruction	✓	✗	✗	✓	✓	✓
	Classification	✗	✓	✓	✗	✓	✓
	$\mathcal{L}_1$ regularization	✗	✗	✓	✗	✓	✓
	$\mathcal{L}_2$ regularization	✓	✓	✗	✗	✓	✓
	Coverage regularization	✗	✗	✗	✗	✗	✓
Decoding							
	Threshold	✗	✗	✗	✓	✓	✓
	NMF	✗	✗	✗	✗	✗	✓

with a dedicated differentiable loss. Second, to enable the discovery of more specific patterns, we propose a novel pattern decoding strategy using latent variable model inference using Non-Negative Matrix factorization for pattern extraction.

3 Preliminaries

We assume a supervised input dataset $\mathcal{D} : (\mathbf{X}, \mathbf{Y})$ with $\mathbf{X} \in \{0, 1\}^{n \times m}$ composed with n samples and m features, and $\mathbf{Y} \in [0, 1]^{n \times k}$, the probability for each sample to be assigned to one of the class labels $K = \{1, \ldots, k\}$. Our purpose consists in finding a set of patterns P, where each pattern $p \in P$ is a set of feature indices $p \subset \{1, 2, \ldots, m\}$ representing feature co-occurrences. To find such sets of patterns, it has been recently proposed to learn a binarized autoencoder type of neural network, where $\mathbf{W} \in \mathbb{R}^{m,h}$ is its weight matrix with h hidden dimensions. We denote by $\mathbf{W}_i$, the i-th row of $\mathbf{W}$. $\mathbf{W}^d$ indicate the binarized version of $\mathbf{W}$. We also consider b for a bias, and b^d for its discretized value. For a given binary database, our aim is to find a diverse set of patterns P that describes the data. One interpretation of this claim consists in defining a set of patterns as correct if it can marginally reconstruct the database.

4 Differentiable Pattern Mining with Coverage Regularization

Pattern mining has been recently tackled using autoencoders, minimizing reconstruction loss with additional class prediction, facilitating robust pattern discovery. As a first contribution, we introduce a diversity objective to minimize collapsing among neurons of the encoder layer during training. Secondly, we propose a novel decoding process after training, promoting the creation of longer patterns with respect to solely thresholding using Non-Negative Matrix Factorization (NMF).

4.1 Neural Model for Pattern Mining

For various data, autoencoders have proven to be a successful approach for capturing the main regularities in the data by minimizing reconstruction loss. An autoencoder is a neural network consisting of task-specific encoding layers that end in an embedding layer, and a symmetric decoder to reconstruct the input from the embedding layer. The embedding layer is usually small compared to the input layer, imposing an information bottleneck and forcing the network to learn relevant and shared structure between inputs.

To support interpretability, a novel type of neural autoencoder as been recently proposed, where weights and activations are discretized in $\{0, 1\}$ during the forward pass. To learn in small noisy steps during backpropagation, for training continuous versions of the weights are used, optimizing reconstruction loss with respect to these continuous weights. The autoencoder consists of one linear hidden layer - a so-called pattern layer - and one linear output layer. For each neuron in the hidden layer, incoming binary weights indicate whether an input item is part of the encoded pattern. For example, a binarized weight $\mathbf{W}^d_{i,j}$ means that input item j is part of the pattern given by hidden neuron i. Thus, each neuron in the hidden layer corresponds to a pattern p, while all neurons together correspond to the pattern set P.

Concretely, one binarized version of the weights is construct for compute the forward pass, and used for reconstruction. To ensure that the hidden neurons correspond to interpretable patterns, the auto-encoder architecture is symmetrical as the weight of the decoding layer is the transpose of the weight of the encoding layer.

4.2 Learning Algorithm

The architecture of differentiable pattern recognition usually consists of a binary autoencoder. The encoding and decoding layers of the autoencoder share a set of continuous weights $\mathbf{W}$. The forward pass uses a binarized version of this weight matrix $\mathbf{W}^d$ following [9]. Each hidden neuron j represents a pattern, and a feature i is part of the pattern corresponding to neuron j if $\mathbf{W}^d_{i,j} = 1$. The decoding layer performs the transposed linear transformation of the encoding layer which enforces the patterns formed during optimization to describe the data. In recent work, a classifier has been added to the pattern layer with continuous weights $\mathbf{W}^c$ to act as an additional regularizer. This classifier is linear and hence interpretable.

The overall objective function consists of a series of terms for the autoencoder reconstruction, the classification error, and various regularization terms.

Reconstruction Loss. First, the autoencoder reconstruction loss from the input points is defined with a weighted XOR function as proposed in [9]. As binary data tends to be sparse and dominated toward zeros, a sparsity-aware reconstruction loss weighs the importance of reconstructing a 1 proportional to the sparsity of the data.

$$\mathcal{L}_e(\mathbf{X}_i, \widehat{\mathbf{X}_i}) = \sum_{j=1}^{m} ((1 - \mathbf{X}_{i,j})\alpha + \mathbf{X}_{i,j}(1 - \alpha))|\widehat{\mathbf{X}_{i,j}} - \mathbf{X}_{i,j}|, \tag{1}$$

with $\alpha = \frac{\#1s}{\#1s+\#0s}$ the sparsity of $\mathbf{X}$ and $\widehat{\mathbf{X}}$ the reconstruction of $\mathbf{X}$ by the autoencoder.

Classification Loss. Second, to optimize the classifier, the cross-entropy loss is naturally optimized between the predicted logits $\widehat{\mathbf{Y}}$ and the true label $\mathbf{Y}$:

$$\mathcal{L}_c(\mathbf{Y}_i, \widehat{\mathbf{Y}_i}) = -\sum_{\ell=1}^{k} \mathbf{Y}_{i,\ell} \log(\widehat{\mathbf{Y}}_{i,\ell}) \tag{2}$$

L_2 **-regularizer.** Next, to promote parsimonious patterns, the L_2-regularizer is leveraged to penalize long patterns, i.e., rows with many 1 s. The function $r_s(\mathbf{W})$ is defined as:

$$r_s(\mathbf{W}) = \sum_{i=1}^{m} \left(\sum_{j=1}^{h} \mathbf{W}_{i,j} \right)^2 \tag{3}$$

This function computes the squared sum of each row of the weight matrix $\mathbf{W}$. This loss penalizes a pattern as a whole as it defines a quadratic cost on the length of the pattern. Hence, the regularizer is promoting shorter patterns discovery.

W-Shaped Regularizer. To further force the weights towards a binary solution, a W-shaped regularizer is defined. The function $r_b(\mathbf{W})$ is defined as:

$$r_b(\mathbf{W}) = \min_i \{ r(\mathbf{W}_i), r(\mathbf{W}_i - 1) \}, \tag{4}$$

where $r(\mathbf{W}_i)$ is defined as $r(\mathbf{W}_i) = \kappa \|\mathbf{W}_i\|_1 + \lambda \|\mathbf{W}_i\|_2^2$. Here, κ and λ are hyperparameters specifying the trade-off between the L_1 and L_2 regularization penalties. This regularizer takes the classic form of an elastic-net, where the κ and λ hyperparameters respectively specify the trade-off between the ridge and lasso penalty.

Coverage Regularization. Finally, to enforce diversity and data-coverage among the patterns in the model's representations, we introduce a orthogonality component into the loss function. By including the orthogonality constraint in the loss function, the model is encouraged to learn diverse and independent features, which can lead to improved generalization performance. The orthogonality constraint is defined through a cosine similarity between each pair of the neurons, which correspond to a line of $\mathbf{W}$. By encouraging orthogonality, the loss function helps prevent the model from collapsing to specific features and encourages it to learn more informative representations. Formally, the loss is defined as follows:

$$\mathcal{L}_{\text{cov}}(\mathbf{W}) = \frac{1}{m(m-1)} \sum_{i \neq j} \left(\frac{\mathbf{W}_i \cdot \mathbf{W}_j}{\|\mathbf{W}_i\| \|\mathbf{W}_j\|} \right). \tag{5}$$

This orthogonality component is combined with other regularization terms to form the complete loss function. As a result, given the parameters of the network $\{\mathbf{W}, \mathbf{W}^c\}$, the loss function is given by:

$$\begin{aligned}\mathcal{L}(\mathcal{D}, \mathbf{W}, \mathbf{W}^c\}) = \sum_{i=1}^{n} \Big[\mathcal{L}_e(\mathbf{X}_i, \widehat{\mathbf{X}}_i) + \lambda_c \mathcal{L}_c(\mathbf{Y}_i, \widehat{\mathbf{Y}}_i)\Big] + r_s(\mathbf{W}) \\ + r_b(\mathbf{W}) + r_b(\mathbf{W}^c) + \mathcal{L}_{\text{cov}}(\mathbf{W}),\end{aligned} \tag{6}$$

where λ_c is a parameter that weighs the classification loss.

4.3 Pattern Decoding from Latent Representation

To extract differential patterns at convergence, $\mathbf{W}$ and $\mathbf{W}^c$ are classically thresholded with τ_e and τ_c, respectively. As described above, a pattern p_j is given by the index set of all i's such that $\mathbf{W}^d_{i,j} = 1$. However, one limitation can be mentioned: the decoding process does not consider the creation of long patterns, resulting from the coverage loss. These longer patterns capture intricate dependencies and interactions between features, possibly offering a deeper understanding of the underlying data structure. Unfortunately, the decoding mechanism may overlook these longer patterns, potentially leading to a loss of information during the reconstruction phase. As a result, the reconstructed data may lack the finer details captured by these longer patterns, hindering the fidelity of the reconstructed dataset. This limitation underscores the need for a decoding strategy that can effectively incorporates the information encoded in longer patterns. So, we propose to improve the pattern decoding process using Non-Negative Matrix Factorization (NMF) over $\mathbf{M}$ define as

$$\mathbf{M}_{i,j} = \frac{\sum_{\ell=0}^{m} \mathbf{X}_{i,\ell}.\mathbf{W}^d_{\ell,j}}{\sum_{\ell=0}^{m} \mathbf{W}^d_{\ell,j}} \tag{7}$$

This way, one can improve the quality and accuracy of the reconstructed patterns.

Non-negative Matrix Factorization is a popular technique of dimensionality reduction that has been explored in numerous applications, like topic modelling and recommendation systems [2]. Given a non-negative matrix $\mathbf{M}$ with dimensions $m \times h$, NMF seeks to factorize this matrix into two non-negative matrices $\mathbf{U}$ and $\mathbf{V}$, such that $\mathbf{M} \approx \mathbf{UV}$. Here, $\mathbf{U}$ represents a basis matrix with dimensions $m \times g$, where g is typically chosen to be smaller than m and h, and $\mathbf{V}$ denotes a coefficient matrix with dimensions $g \times h$. The factorization is constrained to be non-negative, meaning that all elements of $\mathbf{U}$ and $\mathbf{V}$ are non-negative. The resulting factorization aims to represent $\mathbf{M}$ as a linear combination of a reduced set of basis vectors from $\mathbf{U}$, weighted by the coefficients in $\mathbf{V}$. The objective loss function of NMF is defined as the Frobenius norm of the difference between the original matrix and the reconstructed matrix:

$$\mathcal{L}_{\text{NMF}}(\mathbf{M}, \widehat{\mathbf{M}}) = \|\mathbf{M} - \widehat{\mathbf{M}}\|_F^2 = \|\mathbf{M} - \mathbf{UV}\|_F^2. \tag{8}$$

As for topic modeling in natural language processing [20,23], we build new and longer patterns by aggregating the top-p patterns defined as

$$\text{top-p} = \text{argmax}_{j=1\ldots p}\mathbf{V}_{d,j}$$

that corresponds to the most co-occurred patterns for each latent dimension $d \in [1, g]$ of the matrix V. This approach aims at regrouping the patterns that frequently co-occurs in the data. This composition is complementary to the proposed coverage loss which tend to create shorter and orthogonal patterns.

5 Experiments

In the following experiments, we address the following questions. First, we assess how our proposed model performs with respect to the current state-of-the-art methods in terms of scalability, robustness, and overall effectiveness, across both real-world and synthetic datasets. Second, we assess how the diversity regularizer allows to discover a larger variety of discriminative patterns. Third, we question how the proposed NMF decoding build more specific patterns beyond generalist ones in real and synthetic data[1].

5.1 Metrics

We use a set of metrics to assess the pertinence of the discovered patterns. Indeed, the sheer volume and complexity of the patterns generated makes it challenging to identify the most relevant and informative ones. So, several metrics can help to assess the quality, novelty, and usefulness of patterns, and to identify those that are most likely to be of interest to domain experts or end-users. Let P be the set of patterns found and $\mathbf{z}_p$ is the binary vector denoting the assignment of the dataset point to the support of pattern p. We use the following measures to describe the collection of patterns:

- Cover: It computes the proportion of the dataset samples covered by at least one pattern: $\frac{||\bigvee_{p\in P}\mathbf{z}_p||}{n}$.
- Purity: It measures the purity of a pattern with respect to $\mathbf{y}$: $\frac{1}{|P|}\sum_{p\in P}\frac{\max_\ell ||\mathbf{y}_\ell \bigwedge \mathbf{z}_p||}{||\mathbf{z}_p||}$

Then, we use a set of measures to evaluate the collection of patterns as a prediction model:

- Weighted-F1: For each sample, we take the set of patterns that support it. Among those patterns, we select one with the highest purity and associate this class as the predicted label for the sample. In the case where no pattern is supporting a sample, the majority class is associated to it. Weighted-F1 score calculates the F1 score for each class and then computes a weighted average based on the number of samples in each class.

[1] Code and data are available: https://chataingt.github.io/DiffVersify/.

- ROC-AUC based on PURITY and COVER: Since ground truth labels are not available for real-world data, we evaluate the collection of patterns as presented in [21], by using the area under the curve of the percentage of data covered by patterns (COVER measure) once patterns are sorted according to their PURITY that is proportional to the probability of predicting the target class). This evaluation can be interpreted as a trade-off between sensitivity, i.e. the proportion of the dataset covered, and specificity, i.e. the pattern's relevance to a particular class. To eliminate spurious patterns, we only consider those with a predictive probability of $\frac{1}{k} + 0.1$ or higher, indicating a slightly greater likelihood than chance.
- SOFT-F1: When the ground truth patterns are accessible, as usually in synthetic datasets, we utilizes the Jaccard distance instead of strict equality for calculating recall and precision as it prevents an excessive penalty for methods that only partially recover individual patterns [12]. The SOFT-F1 score is defined as the harmonic mean of soft precision and soft recall defined by: soft precision$(P_d, P_g) = \frac{1}{|P_d|} \sum_{p_d \in P_d} \max_{p_g \in P_g} \frac{|p_d \cap p_g|}{|p_d \cup p_g|}$ soft recall$(P_d, P_g) = \frac{1}{|P_g|} \sum_{p_g \in P_g} \max_{p_d \in P_d} \frac{|p_d \cap p_g|}{|p_d \cup p_g|}$ where soft recall and soft precision are computed using the Jaccard distance between the recovered and ground truth patterns. The soft F1 score allows to take into accounts partial matches between recovered and ground truth patterns.

We also consider other description measures of pattern collection:

- # PATTERNS: The number of patterns in P.
- AVG. SUPP.: The average support of the patterns in P: $\frac{\sum_{p \in P} ||\mathbf{z}_p||}{\#\ \text{PATTERNS}}$.

5.2 Baselines

We evaluate our model against the seminal proposal of BINAPS and DIFFNAPS, its improvement in class-specific pattern set mining. By transitivity, we challenge the current state-of-the-art methodologies including decision trees, significant pattern mining [18], MDL-based label-descriptive approaches [12], classification rule learning [19], neuro-symbolic classification rule learning [22], top-k subgroup discovery [16], difference description [4], falling rule lists [17], optimal sparse decision trees [11], and class-specific BMF [13] reported by DIFFNAPS [21]. Indeed, DIFFNAPS has superior performance than these baselines. Throughout all experiments, we utilize the replication package of DIFFNAPS to establish parameters for consistency across the subsequent experiments.

5.3 Experiments on Real-World Benchmarks

First, we evaluate DIFFVERSIFY on four biology-related benchmarks with the variant DIFFVERSIFY-ABL to do an ablation study over the use of the non-negative factorization. The impact of the diversity regularizer is evaluated through the comparison with DIFFNAPS.

Datasets. We consider a phenotypical CARDIO dataset[2], a DISEASE diagnosis dataset[3] and two high-dimensional binarized gene expression datasets for breast cancer, BRCA-N and BRCA-S, both derived from The Cancer Genome Atlas (TCGA)[4]. The number of descriptive features are respectively 45, 131, 1976 and 1976. The number of individuals are respectively, 68k, 5k, 222 and 187. The number of classes are respectively 2, 41, 2 and 4. We use the hyper-parameters reported in DIFFNAPS and BINAPS, which were optimized on these dataset, and we use cross-validation to define the ones for DIFFVERSIFY. In particular, the rank g of NMF decoding determined for each real-world dataset is as follows: CARDIO: 10, DISEASE: 15, BRCA-N: 100 and BRCA-S: 500.

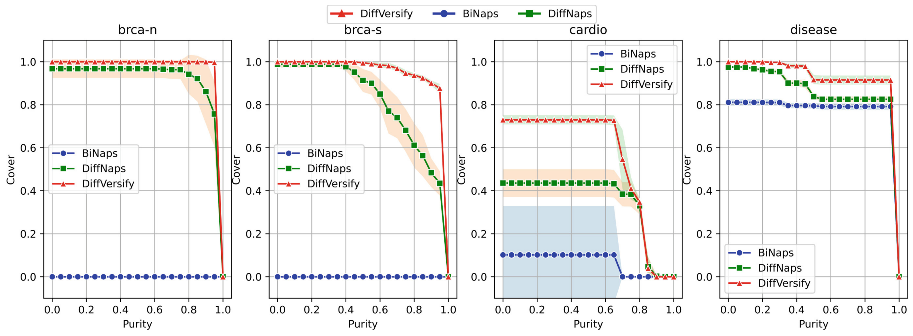


Fig. 1. ROC curve on four biology-related benchmarks, BRCA-N, BRCA-S, CARDIO and DISEASE.

Results. Table 2 reports the measure values obtained by the different methods on the 4 datasets. First, notice that BINAPS returns 0 patterns on both BRCA datasets and therefore the measures can not be evaluated. For all dataset, we can observe that DIFFVERSIFY's patterns exhibit an almost perfect COVER, indicating a complete representation of the data. Notice that there is a large number of patterns on BRCA datasets due to their high number of features compared to their number of data points.

The ROC-AUC shows that DIFFVERSIFY's patterns consistently provide superior COVER with better PURITY, suggesting that they are more effective in describing the classes. Figure 1 shows the ROC curves. For BRCA-S and BRCA-N datasets, we can observe the critical role of regularization for this metric (see the increase compared to DIFFNAPS). This is particularly pertinent with datasets that exhibit a disparity between the number of rows and columns. When applied

[2] https://www.kaggle.com/datasets/sulianova/cardiovasculardisease-dataset.
[3] https://www.kaggle.com/datasets/itachi9604/diseasesymptom-description-dataset.
[4] The BRCA datasets were derived from data made available by the TCGA Research Network.

to datasets such as CARDIO and DISEASE, both DiffVersify and DiffNaps demonstrate comparable performance levels. However, as the complexity of the dataset increases, DiffVersify seems to generate better patterns. Indeed, these patterns offer better coverage while maintaining good Purity, thereby underscoring the potential efficacy of DiffVersify.

Table 2. Comparison of performance metrics across four real-world datasets. Average values and standard-deviations are reported over 5 runs of the methods.

	Measures	Methods	Datasets			
			BRCA-N	BRCA-S	CARDIO	DISEASE
Model evaluation	ROC AUC	BiNaps	nan ± nan	nan ± nan	0.06 ± 0.15	0.76 ± 0.01
		DiffNaps	0.90 ± 0.05	0.79 ± 0.04	0.34 ± 0.05	0.84 ± 0.0
		DiffVersify-Abl	0.92 ± 0.00	0.89 ± 0.03	0.54 ± 0.02	0.86 ± 0.01
		DiffVersify	**0.95 ± 0.00**	**0.93 ± 0.03**	**0.55 ± 0.18**	**0.90 ± 0.01**
	Weighted-F1	BiNaps	nan ± nan	nan ± nan	0.34 ± 0.0	0.76 ± 0.03
		DiffNaps	0.55 ± 0.21	0.18 ± 0.19	**0.71 ± 0.01**	0.88 ± 0.04
		DiffVersify-Abl	0.63 ± 0.28	0.20 ± 0.07	0.68 ± 0.02	0.98 ± 0.00
		DiffVersify	**0.79 ± 0.25**	**0.38 ± 0.14**	0.69 ± 0.02	**1.00 ± 0.01**
Model description	Cover	BiNaps	nan ± nan	nan ± nan	0.87 ± 0.07	0.79 ± 0.03
		DiffNaps	**1.00 ± 0.00**	**1.00 ± 0.00**	0.66 ± 0.17	0.99 ± 0.01
		DiffVersify-Abl	**1.00 ± 0.00**	**1.00 ± 0.00**	**0.98 ± 0.02**	**1.00 ± 0.00**
		DiffVersify	**1.00 ± 0.00**	**1.00 ± 0.00**	**0.98 ± 0.02**	**1.00 ± 0.00**
	Purity	BiNaps	nan ± nan	nan ± nan	0.54 ± 0.05	**0.98 ± 0.01**
		DiffNaps	**0.84 ± 0.03**	**0.37 ± 0.05**	**0.77 ± 0.04**	0.13 ± 0.00
		DiffVersify-Abl	0.59 ± 0.01	0.35 ± 0.02	0.73 ± 0.03	0.10 ± 0.00
		DiffVersify	0.61 ± 0.02	**0.37 ± 0.02**	**0.77 ± 0.02**	0.22 ± 0.00
Other measures	# Patterns	BiNaps	0 ± 0	0 ± 0	5.80 ± 1.92	124.60 ± 2.07
		DiffNaps	182.60 ± 40.83	939.20 ± 336.21	10.06 ± 1.14	3693.69 ± 206.63
		DiffVersify-Abl	2674.80 ± 1088.92	9630.00 ± 1398.29	8.20 ± 1.64	2626.40 ± 59.58
		DiffVersify	2874.80 ± 1088.93	11630.00 ± 1398.29	22.2 ± 1.64	3241.40 ± 59.58
	Avg. Supp.	BiNaps	nan ± nan	nan ± nan	49849.89 ± 8703.88	95.22 ± 3.61
		DiffNaps	33.02 ± 0.58	26.08 ± 1.79	9731.81 ± 4711.73	275.12 ± 7.21
		DiffVersify-Abl	31.05 ± 0.58	26.26 ± 1.79	18231.42 ± 2514.64	273.41 ± 5.07
		DiffVersify	29.83 ± 0.60	24.29 ± 1.61	14938.36 ± 4172.42	263.72 ± 4.86

The Weighted-F1 score, in conjunction with the perfect coverage, underscores the ability of DiffVersify's pattern set to discriminate between classes, even when the pattern set covers all the samples of the dataset. This indicates that DiffVersify not only provides thorough coverage but also maintains a high discriminating capability. The ablation study, where the NMF step is excluded, demonstrates that although DiffVersify-Abl may outperform DiffVersify on ROC-AUC, as on BRCA-N, DiffVersify systematically benefits from this post-processing in all other performance measures.

However, it is worth noting that DiffVersify identified a substantially higher number of patterns (# Patterns) in three of the datasets. This can be attributed to the diversity constraint, which facilitates the generation of more general patterns, and the subsequent NMF decoding that refines these patterns into more precise ones.

This limitation can be addressed through a straightforward yet efficient procedure: sorting the patterns based on their Cover value and selecting patterns until achieving a coverage of 1. Figure 2 illustrates the performance metrics achieved with an increasing set of selected patterns. The results indicate that this post-processing leads to a more compact and effective pattern set. In BRCA-N

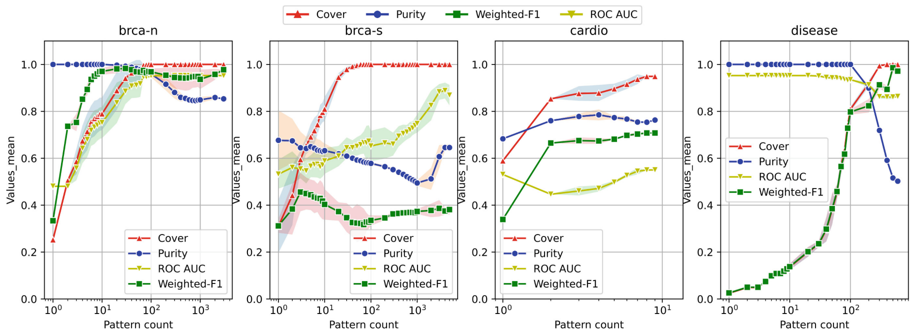


Fig. 2. Evolution of the performance metrics of DiffVersify over subsets of patterns defined by increasing Cover values across brca-n, brca-s, cardio and disease.

and disease, merely one hundred patterns yield satisfactory performance, aligning the # Patterns value with the minimum observed in other methods. On brca-s, a trade-off between Weighted-F1 and ROC-AUC can be achieved applying an elbow method on Cover measure. However, in cardio, where the pattern set is already small, this post-processing step is deemed unnecessary.

Finally Table 3 reports qualitative results. Considering in detail top patterns with respect to Cover found per class on cardio by all three methods (BiNaps, DiffNaps and DiffVersify), Table 3 reveals that DiffVersify consistently identifies at least the same set of patterns as the baseline DiffNaps. Remarkably, DiffVersify outperforms DiffNaps by uncovering additional patterns, characterized by high coverage and purity scores, that the latter fails to detect.

5.4 Experiments on Synthetic Data

To enhance the understanding and comparison of the different methods, we use synthetic data to readily access ground truth patterns, providing a controlled environment for evaluating the properties of the considered approaches.

Dataset Generation. For the data generation process, we use the publicly available DiffNaps replication package. Within each class, ten patterns are randomly sampled across features, with lengths drawn from a uniform distribution ($U(5, 15)$). Also, 20 common patterns are sampled, with lengths drawn from $U(0.01 \times m, 0.025 \times m)$ to maintain data density. Each class comprises an equal number of samples, each containing two common and three class-specific patterns randomly embedded. We introduce additive and destructive noise by flipping ten 0 s to 1 s and flipping 1 s affected by a pattern to 0 s with a 2.5% probability, respectively. Class labels are assigned to satisfy $\frac{(\mathbf{z}_p^t \mathbf{Y})_k}{n} = 0.9$. Means and std of measures across four independently generated datasets are reported. We set the rank g for NMF decoding equal to the number of ground truth patterns.

Table 3. Analysis of the top 6 patterns in terms of Cover by class on cardio and thus for the three methods: BiNaps, DiffNaps and DiffVersify. A pattern is mentioned as Unique if only one of the methods discovered it.

c	Method	Unique	Cover	Cover[c]	Purity	Features
Heart attack	DiffVersify	✓	0.28	0.44	0.77	ap_lo_High_2
	DiffNaps		0.26	0.45	0.84	ap_hi_High_2
	DiffVersify		0.26	0.45	0.84	ap_hi_High_2
	DiffNaps		0.20	0.35	0.84	ap_hi_High_2, ap_lo_High_2
	DiffVersify		0.20	0.35	0.84	ap_hi_High_2, ap_lo_High_2
	DiffVersify	✓	0.16	0.27	0.83	ap_hi_High_2, cholesterol_normal
	DiffNaps		0.14	0.20	0.70	age_(60.0, 64.0)
	DiffVersify		0.14	0.20	0.70	age_(60.0, 64.0)
	DiffNaps		0.11	0.18	0.76	cholesterol_way_above
	DiffVersify		0.11	0.18	0.76	cholesterol_way_above
Healthy	DiffVersify	✓	0.71	0.86	0.62	ap_lo_Normal_Elevated
	DiffNaps		0.59	0.80	0.68	ap_hi_Normal
	DiffVersify		0.59	0.80	0.68	ap_hi_Normal
	DiffVersify	✓	0.57	0.74	0.66	ap_lo_Normal_Elevated, cholesterol_normal
	DiffNaps		0.57	0.77	0.69	ap_hi_Normal. ap_lo_Normal_Elevated
	DiffVersify		0.57	0.77	0.69	ap_hi_Normal, ap_lo_Normal_Elevated
	BiNaps	✓	0.23	0.33	0.73	gender_women, ap_hi_Normal
	DiffNaps		0.18	0.24	0.69	age_(29.0, 45.0)
	DiffVersify		0.18	0.24	0.69	age_(29.0, 45.0)

Scalability in m. One significant challenge in existing pattern-set mining approaches is handling high-dimensional data. We thus vary the number of features m within $\{10^2, 5 \times 10^2, 10^3, 5 \times 10^3, 10^4, 1.5 \times 10^4, 2 \times 10^4, 2.5 \times 10^4, 5 \times 10^4, 10^5\}$. We set the number of classes to $k = 2$ and the number of rows to $n = 10^4$. To mitigate pattern overlap in low-dimensional data ($m < 10^3$), we sample 5 patterns per class without sharing.

Multi-classes. We assess the methods' capability to classify data as the number of distinct classes increases. The number of classes k ranges from 2 to 50, with 4×10^3 samples generated per class and $m = 5 \times 10^3$ features.

Robustness to Additive Noise. We evaluate the robustness of our model to additive noise, by simulating scenarios in which data may be corrupted or perturbed. Setting $k = 2$, $m = 5 \times 10^3$, and $n = 10^3$, we introduce additive noise by varying the number of randomly added 1 s per row from 0 to 100.

Robustness to Destructive Noise. The robustness of the model against destructive noise, a significant challenge in extracting meaningful patterns, is evaluated by varying the probability of flipping 1 s to 0 s from 0% to 60%.

Results. The results are shown in Fig. 3 and Table 4. In the feature and noise experiments in Table 4, we expect to identify 20 ground truth patterns. Remarkably, DiffVersify is the only method that consistently achieves near-perfect coverage, irrespective of the magnitude of the noise or the value of dimensionality. Both DiffVersify and DiffNaps discover class-specific patterns with an average purity surpassing 0.8 across all experiments. Notably, DiffVersify shows

Table 4. Performance comparison on synthetic datasets.

	Measures	Method	# Features	# Classes	Add. noise	Dest. noise
Model evaluation	Soft-F1	BiNaps	0.24 ± 0.3	0.52 ± 0.21	0.09 ± 0.08	0.07 ± 0.03
		DiffNaps	**0.89 ± 0.12**	0.59 ± 0.09	0.65 ± 0.05	0.5 ± 0.17
		DiffVersify	**0.81 ± 0.23**	**0.66 ± 0.09**	**0.73 ± 0.07**	**0.68 ± 0.13**
	Weighted-F1	BiNaps	0.63 ± 0.15	0.11 ± 0.16	0.61 ± 0.04	0.57 ± 0.03
		DiffNaps	**0.88 ± 0.03**	0.65 ± 0.12	0.76 ± 0.01	0.56 ± 0.18
		DiffVersify	**0.89 ± 0.02**	**0.75 ± 0.11**	**0.84 ± 0.06**	**0.62 ± 0.2**
Model description	Cover	BiNaps	**1.0 ± 0.01**	**1.0 ± 0.0**	**1.0 ± 0.0**	**0.99 ± 0.01**
		DiffNaps	**0.95 ± 0.06**	0.79 ± 0.2	0.73 ± 0.03	0.38 ± 0.28
		DiffVersify	**0.99 ± 0.02**	**0.85 ± 0.17**	0.84 ± 0.9	0.48 ± 0.35
	Purity	BiNaps	0.71 ± 0.08	0.53 ± 0.05	0.73 ± 0.05	0.7 ± 0.02
		DiffNaps	**0.9 ± 0.01**	**0.83 ± 0.08**	**0.9 ± 0.02**	**0.88 ± 0.06**
		DiffVersify	**0.87 ± 0.04**	**0.84 ± 0.08**	**0.9 ± 0.02**	**0.9 ± 0.06**
Other measures	# Patterns	BiNaps	369.51 ± 259.53	–	323.35 ± 93.86	203.02 ± 22.48
		DiffNaps	17.8 ± 4.87	–	14.13 ± 2.05	9.73 ± 4.95
		DiffVersify	41.36 ± 29.23	–	16.85 ± 2	13.47 ± 6.7
	Avg. Supp.	BiNaps	1896.77 ± 1136.14	1206.58 ± 679.08	119.8 ± 32.5	118.38 ± 8.06
		DiffNaps	2667.88 ± 1106.17	210.69 ± 88.06	170 ± 28.1	106.17 ± 64.27
		DiffVersify	2664.29 ± 1294.1	188.88 ± 38.18	188.51 ± 24.31	102.27 ± 75.84

higher similarity to the ground truth patterns relative to its baselines, a phenomenon attributable to the NMF decoding process. It is worth mentioning that the addition of obtained patterns to the existing ones invariably introduces similarity among patterns. Furthermore, both DiffVersify and DiffNaps manage to identify the majority of the ground-truth, with an average soft-F1 score exceeding 0.70. The exception to this observation is in the experiment involving the number of classes, where the complexity of the classes diminished performance to a level akin to the baseline, BiNaps. In terms of weighted-F1 results, DiffVersify and DiffNaps show comparable performance, although DiffVersify exhibits superior average results. As the number of class is varying, the number of groundtruth patterns is varying accordingly. As a consequence, we do not compute the number of patterns for these specific experimental settings in the table. In Fig. 3, DiffVersify exhibits better robustness to both additive and destructive noises. Furthermore, DiffVersify demonstrates scalability with respect to the number of features, particularly in high-dimensional settings where it outperforms DiffNaps in terms of robustness.

The results suggest that our proposed method is able to discover more diverse and discriminative patterns compared to the baseline methods, while maintaining high coverage and purity. The use of diversity regularization and NMF decoding in DiffVersify allows for the discovery of longer and more specific patterns, which lead to improved generalization performance. Further work could be done to improve the proposed approach in several ways. One potential direction is to explore other decoding strategies beyond NMF, such as using more advanced matrix factorization techniques or incorporating domain-specific knowledge into the decoding process. Another direction is to investigate the use of other regularization techniques, such as group-sparsity regularizers, to further encourage diversity and interpretability in the learned patterns. One limitation of our proposed approach is that it relies on the assumption that the data can be well-represented by a set of binary patterns. However, in some cases, the data may

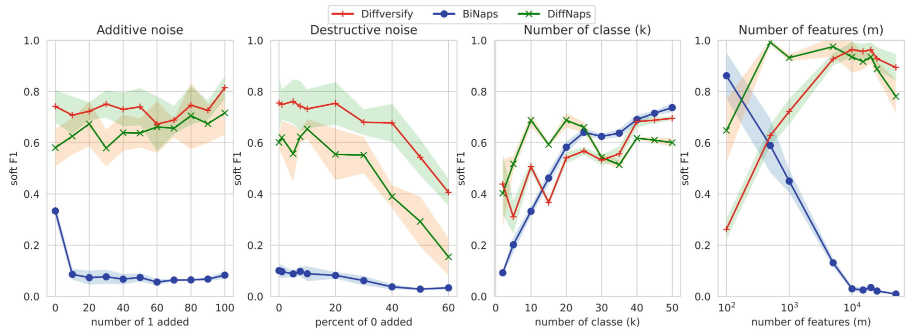


Fig. 3. F1 scores obtained on the synthetic datasets by varying (from left to right) the additive noise level, the destructive noise level, k and m.

contain more complex relationships that cannot be captured by binary patterns alone. In such cases, it may be necessary to extend the approach to allow for more complex pattern representations, such as real-valued or continuous patterns.

6 Conclusion

We introduced a novel differentiable binary model for pattern mining that incorporates a regularization loss emphasizing pattern coverage and a pattern decoding strategy using non-negative matrix factorization (NMF). Our approach demonstrates superior performance in terms of ROC-AUC on four real-world biology-related datasets and improves pattern detection by increasing similarity measure to ground truth patterns on synthetic data. Through extensive evaluations, we show the appropriateness of discovered patterns relative to the data, focusing on pattern coverage, indicating the efficacy of our approach in handling challenging scenarios with high noise levels and multiple classes. One possible future direction is to search for alternative techniques for pattern decoding from differentiable model. This could help to better capture complex patterns and improve the overall accuracy of our approach. Another direction is to incorporate additional regularizers such as those proposed in neuro-symbolic approaches.

Acknowledgement. This work benefited from state aid managed by the National Research Agency under France 2030 with the reference “ANR-22-PEAE-0008”.

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: SIGMOD, pp. 207–216. ACM Press (1993)
2. Berman, A., Plemmons, R.J.: Nonnegative matrices in the mathematical sciences. In: Classics in Applied Mathematics (1979)
3. Bosc, G., Boulicaut, J., Raïssi, C., Kaytoue, M.: Anytime discovery of a diverse set of patterns with Monte Carlo tree search. DAMI **32**(3), 604–650 (2018)

4. Budhathoki, K., Vreeken, J.: The difference and the norm: characterising similarities and differences between databases. In: Mach (2015)
5. Dash, S., Günlük, O., Wei, D.: Boolean decision rules via column generation. In: NeurIPS, pp. 4660–4670 (2018)
6. De Bie, T.: Maximum entropy models and subjective interestingness: an application to tiles in binary databases. Data Min. Knowl. Discov. **23**(3), 407–446 (2011)
7. Dierckx, L., Veroneze, R., Nijssen, S.: RL-net: interpretable rule learning with neural networks. In: PAKDD, pp. 95–107 (2023)
8. Dzyuba, V., van Leeuwen, M., Raedt, L.D.: Flexible constrained sampling with guarantees for pattern mining. Data Min. Knowl. Discov. **31**(5), 1266–1293 (2017)
9. Fischer, J., Vreeken, J.: Differentiable pattern set mining. In: SIGKDD, pp. 383–392. ACM (2021)
10. Gionis, A., Mannila, H., Mielikäinen, T., Tsaparas, P.: Assessing data mining results via swap randomization. ACM Trans. Knowl. Discov. Data **1**(3), 14 (2007)
11. Hayden, M., et al.: Fast sparse decision tree optimization via reference ensembles. In: AAAI, vol. 36 (2022)
12. Hedderich, M., Fischer, J., Klakow, D., Vreeken, J.: Label-descriptive patterns and their application to characterize classification errors. In: ICML (2022)
13. Hess, S., Morik, K.: C-SALT: mining class-specific alterations in boolean matrix factorization. In: ECML PKDD, vol. 10534, pp. 547–563 (2017)
14. Kusters, R., Kim, Y., Collery, M., Marie, C.d.S., Gupta, S.: Differentiable rule induction with learned relational features. arXiv preprint arXiv:2201.06515 (2022)
15. Lakkaraju, H., Bach, S.H., Leskovec, J.: Interpretable decision sets: a joint framework for description and prediction. In: SIGKDD, pp. 1675–1684 (2016)
16. Lemmerich, F., Becker, M.: pysubgroup: easy-to-use subgroup discovery in python. In: Brefeld, U., et al. (eds.) ECML PKDD 2018. LNCS (LNAI), vol. 11053, pp. 658–662. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-10997-4_46
17. Lin, J.J., Zhong, C., Hu, D., Rudin, C., Seltzer, M.I.: Generalized and scalable optimal sparse decision trees. In: ICML (2020)
18. Pellegrina, L., Riondato, M., Vandin, F.: Spumante: significant pattern mining with unconditional testing. In: SIGKDD, pp. 1528–1538 (2019)
19. Proença, H.M., van Leeuwen, M.: Interpretable multiclass classification by mdl-based rule lists. Inf. Sci. **512**, 1372–1393 (2020)
20. Shi, T., Kang, K., Choo, J., Reddy, C.K.: Short-text topic modeling via NMF enriched with local word-context correlations. In: WWW (2018)
21. Walter, N.P., Fischer, J., Vreeken, J.: Finding interpretable class-specific patterns through efficient neural search. In: AAAI (2024)
22. Wang, Z., Zhang, W., Liu, N., Wang, J.: Scalable rule-based representation learning for interpretable classification. In: NeurIPS, vol. 34, pp. 30479–30491 (2021)
23. Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: ACM SIGIR (2003)
24. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of association rules. In: SIGKDD, pp. 283–286 (1997)

Hierarchical Graph Contrastive Learning for Review-Enhanced Recommendation

Changsheng Shui, Xiang Li, Jianpeng Qi, Guiyuan Jiang, and Yanwei Yu(✉)

Ocean University of China, Qingdao, China
{scs,lixiang1202}@stu.ouc.edu.cn,
{qijianpeng,jiangguiyuan,yuyanwei}@ouc.edu.cn

Abstract. In comparison to numerical ratings and implicit feedback, textual reviews offer a deeper understanding of user preferences and item attributes. Recent research underscores the potential of reviews in augmenting recommendation capabilities, thereby advancing the deployment of review-enhanced recommendation systems. However, existing methodologies often neglect the significance of rating magnitudes and are susceptible to challenges such as data sparsity and long-tail distribution in real-world contexts. To address these challenges, we propose Hierarchical Graph Contrastive Learning (HGCL) for advancing review-enhanced recommendation systems. HGCL dynamically learns hypergraph structures to capture higher-order correlations among nodes and simultaneously integrates local and global collaborative relations through global-local contrastive learning. Additionally, we propose hierarchical graph contrastive learning methods to better model the intrinsic correlation between ratings and reviews, encompassing aspects such as local-global, cross-rating, and edge-level contrastive learning. Extensive experimentation on five public datasets demonstrates that the proposed method notably outperforms state-of-the-art approaches.

Keywords: Graph Representation Learning · Hypergraph Learning · Contrastive Learning · Recommender Systems

1 Introduction

In recent years, the explosion of information has propelled recommender systems [2,9,31,35], into indispensable tools for e-commerce and social media platforms. Among the core challenges in recommender systems, predicting ratings stands prominently [16,21,34,43]. However, a notable disparity between ratings and implicit feedback (such as purchases, clicks, or favorites) is the concurrent provision of textual reviews by many users. While ratings offer a direct reflection of user preferences for items [10], they often fall short in capturing nuanced sentiments. In contrast, textual reviews encapsulate users' personalized needs and attention to specific aspects, providing essential insights into genuine user

C. Shui and X. Li—Contribute equally to this work.

preferences. The rapid advancement of Natural Language Processing (NLP), has significantly augmented semantic understanding capabilities, such as BERT [11] and its variants [24], and BERT4Rec [25]. These NLP models excel in transforming textual reviews into high-fidelity vector representations, effectively capturing users' semantic sentiments. Consequently, review-based recommendation systems, empowered by these advancements, have become ubiquitous in recommendation technology landscapes.

Traditional review-based recommendation approaches usually rely on the Bag-of-Words model, neglecting contextual and sequential information within the text, which results in the loss of critical semantic nuances. With the advent of deep learning, various methodologies have integrated review information into graph neural network (GNN) architectures [5,33,41]. Notably, GNN-based methods have emerged as potent models for comprehensively understanding user-item interactions. Integrating reviews into GNN-based recommendation models has thus become pivotal in enhancing recommendation effectiveness. Wu et al. [32] propose a novel method named RMG which stands as a pioneering effort, integrating graph signals and review information for recommender systems. Subsequently, Shuai et al. [23] build upon this foundation by leveraging semantic information embedded within review features and then propose their RGCL model. It fine-tunes the influence of neighboring nodes and employs contrastive learning techniques to achieve superior interaction modeling. These GNN-based methods for review-enhance recommendation adeptly model user-item bipartite graphs using graph convolution, seamlessly incorporating review vectors obtained through NLP models as semantic cues for interaction edges. They not only surmount limitations in traditional approaches but also enrich recommender systems with precise semantic understanding, amalgamating interaction details with contextual insights gleaned from reviews.

Despite their efficacy, existing methods grapple with several challenges. *First*, they often rely on stacking multiple layers to capture high-order correlations and constraints, leading to noise accumulation and over-smoothing issues. *Second*, inherent susceptibility to data sparsity and distribution imbalances in user-item interaction data poses a significant hurdle, exacerbated by long-tail distribution patterns and rating imbalances. *Last*, the intrinsic correlation between reviews and ratings remains inadequately addressed, with many recommender systems neglecting rating magnitude and order, contrary to implicit feedback mechanisms. Efforts to mitigate these challenges and further enhance the effectiveness of GNN-based recommender systems are imperative for advancing recommendation technology in evolving information landscapes.

Desired by addressing the aforementioned challenges, we propose a novel **H**ierarchical **G**raph **C**ontrastive **L**earning for review-enhanced recommendation named **HGCL**. Specifically, to mitigate the over-smoothing issue, we design a hypergraph learner to capture high-order dependencies between nodes and employ hierarchical contrastive learning to achieve mutual enhancement between two views. Additionally, we propose multi-rating contrastive learning and global-local contrastive learning to strengthen dependencies between adjacent ratings, and further enhance the intrinsic correlation between ratings and reviews through edge-level contrastive learning.

We highlight the key contributions of this work as follows:

- HGCL explicitly proposes the idea of hierarchical contrastive learning, which not only acquires node features from both local and global views between user-item node pairs but also learns complex interaction information from multi-rating and edge-level perspectives.
- HGCL leverages hypergraph learning to partition the user-item interaction graph into distinct subgraphs, which can dynamically learn hypergraph structures to capture higher-order correlations during training.
- Comprehensive experiments on five real-world datasets are conducted to evaluate model performance, and results show HGCL achieves up to 4.10% and 4.35% performance improvement in terms of MSE and MAE, respectively.

2 Related Work

2.1 GNNs for Review-Enhanced Recommendation

To explicitly leverage collaborative signals within user-item interaction graphs, an increasing number of studies in collaborative filtering (CF) have recently integrated graph convolutional networks (GCNs) to enhance model performance. Traditional GNN-based methods [12,27,38] extract information from neighbors in the user-item graph through multi-hop convolutions to exploit high-order correlations defined by initial pairwise links. Although these methods have shown promising results, they overlook the fact that neighbors should be target-relevant. Despite some existing methods utilizing attention mechanisms to assign different weights to neighbors, they encounter two main challenges. Firstly, their performance is hindered by the lack of fine-grained details. Secondly, attention learned solely from interaction data may not accurately capture users' preferences.

To tackle these challenges, review-based recommender systems have been proposed [14,20,23,29]. For instance, RMG [32] is a pioneering effort, integrating graph signals and review information for recommendations. Subsequently, RGCL [23] builds upon this foundation by leveraging semantic information embedded within review features. It fine-tunes the influence of neighboring nodes and employs contrastive learning to achieve superior interaction modeling.

2.2 Contrastive Learning

Contrastive learning has emerged as a powerful self-supervised framework, with its core idea being to promote the closeness of representations among different views of the same object, while simultaneously increasing the dispersion of representations among different objects. The essence of contrastive learning lies in bringing an 'anchor' and a 'positive' sample closer in the embedding space, while simultaneously pushing the anchor away from multiple 'negative' samples [15]. Drawing inspiration from the successes of contrastive learning across various domains [3,7,13,17,39], many researchers have integrated contrastive learning into graph data to address the challenges posed by insufficient supervision signals in recommender systems recently [19,26,28,37,44]. RGCL [23] introduces review-based contrastive learning, where corresponding review representations are selected as positive samples. This innovative approach effectively leverages the semantic richness encapsulated within reviews, contributing to more informed and review-enhanced recommendations. Contrastive learning has emerged as a promising avenue for addressing data sparsity and distribution issues within recommender systems. The adoption of these techniques underscores their potential to enhance recommendation models and enrich the user experience.

3 Problem Definition

We let $\mathbf{U} = \{u_1, u_2, \ldots, u_M\}$ denote the set of users and let $\mathbf{V} = \{v_1, v_2, \ldots, v_N\}$ denote the set of items, where M and N denote the number of users and items, respectively. Each interaction can be defined as a tuple $(u_i, v_j, y_{u_i v_j}, c_{u_i v_j})$, where $y_{u_i v_j}$ indicates the interaction between u_i and v_j and $c_{u_i v_j}$ is the textual review. Additionally, we utilize a pre-trained BERT-Whitening model to process user u_i's reviews on item v_j, obtaining the vector $\tilde{c}_{u_i v_j} \in \mathbb{R}^d$ as the review embedding. The collection of all review embeddings can be represented as $\widetilde{\mathrm{E}} \in \mathbb{R}^{M \times N \times d}$. We typically use a rating matrix $\mathbf{R}$ to store historical ratings from users to items and $\mathcal{R}$ is the set of all ratings (*e.g.*, $\mathcal{R} = \{1, 2, 3, 4, 5\}$), where $r_{u_i v_j}$ indicates a historical rating from u_i to v_j while $r_{u_i v_j} = 0$ means that v_j is unexposed to u_i.

Here we represent the interaction network as a user-item bipartite graph G, and there are only interactions between users and items. The task of review-enhanced recommendation is to predict missing ratings in the bipartite graph.

4 Methodology

In this section, we provide a comprehensive exposition of the proposed HGCL, as illustrated in Fig. 1. HGCL encompasses five pivotal processes: **(1) Subgraph Partition and Review-aware Graph Convolution**, **(2) Multi-Rating Contrastive Learning**, **(3) Hypergraph Structure Learning**, **(4) Global-Local Contrastive Learning**, and **(5) Edge-Level Contrastive Learning**. Generally, through these five integral components, HGCL stands as a holistic solution designed to enhance the accuracy and efficiency of recommendation, by effectively leveraging collaborative signals, topological features, and high-order correlations across varying ratings within user-item interaction data.

4.1 Subgraph Partition and Review-Aware Graph Convolution

Inspired by previous models [22,40,42], we divide the bipartite graph into different rating subgraphs, the initial feature for each node consists of three different indices (identical index, role-aware index, and one-hot index) in each subgraph.

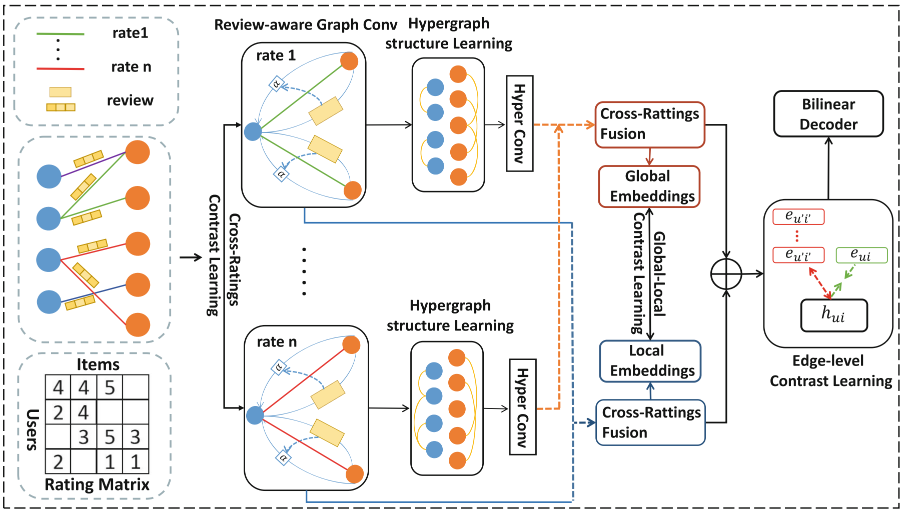


Fig. 1. The overall architecture of the proposed HGCL.

With the feature of each node, we can utilize it to derive the initial embedding of the node from the trainable parameter matrix $\mathrm{E} \in \mathbb{R}^{|\mathcal{R}|\times(M+N+3)\times d}$, where $|\mathcal{R}|$ represents the number of rating categories.

As previously mentioned, in contrast to simple rating information, review text provides a more nuanced description of the current interaction edge. It can reflect the user's preference for the item and the popularity of the item itself in greater detail. Therefore, we compute the weight coefficient of different neighbors based on the reviews corresponding to each edge as follows:

$$x_r^{l+1}[u_i] = \sum_{j\in\mathcal{N}_r(u_i)} \frac{\sigma(w_r^l \tilde{c}_{u_i v_j})}{\sqrt{|\mathcal{N}_r(u_i)|\cdot|\mathcal{N}_r(v_j)|}} x_r^l[v_j], \tag{1}$$

where $x_r^{l+1}[u_i]$ denotes the embedding of the u_i at layer $l+1$ in the r-th rating subgraph, $\mathcal{N}_r(u_i)$ is the set of neighbors for u_i in the r-th rating subgraph, w_r^l represents the trainable parameter matrix of the rating subgraph r at the l layer, and $\sigma(\cdot)$ indicates the sigmoid activation function. These embeddings can represent the collaborative signal and the topological information of each user and item on different subgraphs.

4.2 Multi-rating Contrastive Learning

Compared to traditional heterogeneous graphs with multiple edge types, the importance of ratings suggests that users giving nearby ratings share similar preferences, thus leading to similar representations. However, real datasets with imbalanced rating distributions add complexity to this scenario. Partitioning the heterogeneous graph into rating subgraphs unintentionally results in the loss of sequential rating information.

To tackle this challenge, we propose an innovative multi-rating contrastive learning module that captures the correlation between different ratings by leveraging multi-view contrastive learning. Specifically, we adopt the well-established InfoNCE loss [8] based on multi-rating representations of users and items. We treat a user's embedding learned from the current rating subgraph and the user embedding from the next adjacent rating subgraph as positive pairs while respecting the order of ratings.

$$\mathcal{L}^{\mathbf{U}}_{MR(r)} = \sum_{u \in \mathcal{U}} -\log \frac{exp((x_u^{(r)} \cdot x_u^{(r+1)}/\tau))}{\sum_{v \in \mathbf{U}} exp((x_v^{(r)} \cdot x_v^{(r+1)}/\tau))}, \tag{2}$$

where $x_u^{(r)}$ is the normalized output embedding of rating subgraph r and τ is the temperature hyper-parameter of softmax.

As an illustrative example, consider a 5-point Likert scale. In this case, $L^{\mathbf{U}}_{MR(1)}$ denotes the multi-rating contrastive loss between the subgraphs with a rating of '1' and '2'. Similarly, $L^{\mathbf{U}}_{MR(2)}$ signifies the multi-rating contrastive loss between the subgraphs with a rating of '2' and '3', while $L^{\mathbf{U}}_{MR(3)}$ captures the multi-rating contrastive loss between the subgraphs with a rating of '3' and '4'. The overall multi-rating contrastive loss for the entire user can be expressed as follows:

$$\mathcal{L}^{\mathbf{U}}_{MR} = \mathcal{L}^{\mathbf{U}}_{MR(1)} + \mathcal{L}^{\mathbf{U}}_{MR(2)} + ... + \mathcal{L}^{\mathbf{U}}_{MR(|\mathcal{R}|-1)}. \tag{3}$$

where $|\mathcal{R}|$ denotes the number of rating categories. Similarly, we can obtain the loss function of the item side $\mathcal{L}^{\mathbf{V}}_{MR}$. The complete multi-rating contrastive objective function $\mathcal{L}_{MR}$ is the sum of the above two losses:

$$\mathcal{L}_{MR} = \mathcal{L}^{\mathbf{U}}_{MR} + \mathcal{L}^{\mathbf{V}}_{MR}. \tag{4}$$

4.3 Hypergraph Structure Learning

As previously discussed, the embedding derived from Eq. (2) effectively encapsulates collaborative signals and topological information pertaining to users and items within distinct rating subgraphs. It is worth noting that the inherent topological similarities among nodes often mirror implicit similarities in user preferences or item features. In light of this insight, we endeavor to transcend the conventional limitation inherent in bipartite graphs, where edges typically link nodes of the same type. Drawing inspiration from prior methodologies [31,35], we employ a multi-layer perceptron (MLP) in the computation of hyperedge assignments for users and items within each rating subgraph. This process yields hypergraphs $\mathbf{H}_u^r$ for users and $\mathbf{H}_v^r$ for items, each reflecting the intricate relations and dependencies that exist within the respective rating contexts:

$$\begin{aligned} \mathbf{H}_u^r &= norm(LeakyReLU(x_u^r \mathbf{W}_u^r)), \\ \mathbf{H}_v^r &= norm(LeakyReLU(x_v^r \mathbf{W}_v^r)), \end{aligned} \tag{5}$$

where $\mathbf{W}_u^r \in \mathbb{R}^{K \times d}$ and $\mathbf{W}_v^r \in \mathbb{R}^{K \times d}$ are trainable weight matrices, in which K is the number of hyperedges. Upon deriving the user hypergraph and item hypergraph associated with various ratings, we possess the means to incorporate implicit high-order correlations among nodes into the updated representations of both users and items. The hypergraph convolution, which operates independently on each rating subgraph, is formally defined as follows:

$$h_u^r = \mathbf{H}_u^r \mathbf{L}_u^{-1} \mathbf{H}_u^{r\,\mathsf{T}} x_u^r, \quad h_v^r = \mathbf{H}_v^r \mathbf{H}_v^{-1} \mathbf{H}_v^{r\,\mathsf{T}} x_v^r. \tag{6}$$

Here, $\mathbf{L}_u$ and $\mathbf{L}_v$ represent the vertex degree matrices associated with the learned user hypergraph $\mathbf{H}_u^r$ and item hypergraph $\mathbf{H}_v^r$, respectively. These matrices play a crucial role in the re-scaling of the resulting embeddings.

4.4 Global-Local Contrastive Learning

As elucidated in Eq. (2) and Eq. (10), each node is endowed with a local embedding capturing the nuances of its local topology, as well as a global embedding encapsulating higher-order dependencies within each rating subgraph. However, in downstream tasks, there often arises a necessity for more concise node embeddings. Conventional approaches typically employ simplistic pooling operations (such as average or sum pooling) on embeddings from various perspectives to derive condensed representations, inadvertently overlooking the importance of node representations pertaining to diverse relations.

To overcome this drawback, we propose a global-local contrastive learning aimed at reinforcing relationships between adjacent ratings. Formally, let $\mathcal{N}_r$ denote the two ratings adjacent to rating r, and let x^r represent the local embeddings of nodes within the r-th rating subgraph. The process of message passing between adjacent ratings unfolds as follows:

$$e^r = x^r + \sum_{r' \in \mathcal{N}_r} \zeta_{r'} \cdot x^{r'}, \tag{7}$$

where $\zeta_{r'}$ is the normalized relevance of rating r' to r, which is calculated by:

$$\zeta_{r'} = \frac{\exp\left(LeakyReLU\left(q^r x^{r'}\right)\right)}{\sum_{r' \in \mathcal{N}(r)} \exp\left(LeakyReLU\left(q^r x^{r'}\right)\right)}. \tag{8}$$

q^r constitutes a trainable attention mechanism specifically designed for rating r. It plays a crucial role in governing the flow of information between rating r and its neighboring ratings within $\mathcal{N}(r)$. Similarly, we utilize the global-local contrastive learning in parallel to derive global embeddings γ^r.

Finally, we further process this intermediate output through a linear operator:

$$e[i] = tanh(\mathbf{W}_1 \sum_{r \in \mathcal{R}} z^r[i]), \quad \gamma[i] = tanh(\mathbf{W}_2 \sum_{r \in \mathcal{R}} gamma^r[i]), \tag{9}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ denote the weight parameters of the nonlinear function $tanh(\cdot)$, $e[\cdot]$ and $\gamma[\cdot]$ represent local and global embeddings after multi-view aggregation, and the final representation of users and items is the combination of local embeddings and global embeddings.

Specifically, we differentiate between the local and global embeddings as two distinct perspectives. In this context, we consider different views of the same user or item as positive pairs, while treating divergent views of users/items as negative pairs. Formally, we introduce our global-local contrastive loss function for user representations, leveraging the InfoNCE framework, as follows:

$$\mathcal{L}_{GL}^{\mathbf{U}} = \sum_{u \in \mathbf{U}} -\log \frac{exp((\mathrm{e}[u] \cdot \gamma[u]/\tau))}{\sum_{u' \in \mathbf{U}} exp((e[u'] \cdot \gamma[u']/\tau))}. \tag{10}$$

The final global-local contrastive loss is the sum of user and item objectives:

$$\mathcal{L}_{GL} = \mathcal{L}_{GL}^{\mathbf{U}} + \mathcal{L}_{GL}^{\mathbf{V}}. \tag{11}$$

4.5 Edge-Level Contrastive Learning

The preceding multi-rating and local-global methods represent variants of multi-view contrastive learning. These methodologies aim to foster the congruence of identical nodes across diverse views while emphasizing the divergence of distinct nodes, thereby facilitating synergistic augmentation and information complementarity across varied perspectives. Nevertheless, while embeddings predominantly capture the characteristics of associated interaction edges, the aforementioned contrastive learning techniques fail to account for the self-supervised cues inherent within interaction edges and their corresponding reviews. Drawing inspiration from prior methodologies, we capitalize on the richness of review data by employing edge-level contrastive learning to fully harness the potential of review information.

To capture the interaction edge between users and items, we employs a Multi-layer Perceptron (MLP) to construct the interaction embedding $h_{u_i v_j}$. This embedding is derived by concatenating the final representations of users and items.

$$h_{u_i v_j} = \mathbf{MLP}(e[u_i], e[v_j]), \tag{12}$$

where $e[u_i]$ and $e[v_j]$ represent the final embeddings of user u_i and item v_j respectively, and MLP comprises two hidden layers with a ReLU activation function. Subsequently, all interaction embeddings serve as anchor examples, while the corresponding review embeddings are designated as positive samples. However, in the case of an interaction embedding with an actual rating of '1', for instance, the ideal negative sample should be chosen from the review embeddings with a rating of '5'. Nevertheless, RGCL (presumably a previous method) imposes no constraints on the random sampling range of negative samples, and each anchor of RGCL corresponds to only one negative sample, thereby diminishing the effectiveness of contrastive learning. Hence, HGCL takes into account the sequentiality of ratings and the distribution of the number of interactions corresponding to different ratings and introduces a method to extract negative samples based on ratings. In essence, each rating selectively chooses negative samples from reviews associated with ratings that are farther away. The edge-level contrastive learning mechanism can be formulated as follows:

$$\mathcal{L}_{EL} = \sum_{u_i, v_j \in \varepsilon} - \log \frac{exp((h_{u_i v_j} \cdot \tilde{c}_{u_i v_j}/\tau))}{\sum_{c'_{u_i v_j} \in \mathcal{N}(h_{u_i v_j})} exp((h_{u_i v_j} \cdot c'_{u_i v_j}/\tau))}. \tag{13}$$

4.6 Model Optimization

In line with prior work [1,22], we adopt a bilinear decoder for the purpose of reconstructing edges within the user-item graph. The decoder functions by bilinearizing the potential rating levels, generating a probability distribution, and subsequently applying a softmax function:

$$p(\check{\mathbf{R}}_{u_i v_j} = r) = \frac{e^{(e[u_i])^T Q_r e[v_j]}}{\sum_{r'=1}^{\mathcal{R}} e^{(e[u_i])^T Q_{r'} e[v_j]}}, \tag{14}$$

where Q_r is the learnable parameter matrix of rating r. The predicted rating matrix is calculated as:

$$\check{\mathbf{R}}_{u_i v_j} = \sum_{r \in \mathcal{R}} rp\left(\check{\mathbf{R}}_{u_i v_j} = r\right). \tag{15}$$

We minimize the cross-entropy loss (denoted by CE) between the predictions and the ground truth ratings. While assigning a value of '1' to each ground truth label, we also set two adjacent labels to a smaller value, denoted as l_{close}, mitigating the problem that CE loss ignores differences between incorrect labels. Typically, we set l_{close} to 0.1 to strike an appropriate balance.

$$\mathcal{L}_{main} = \frac{1}{|(u_i, v_j)|\Omega_{u_i, v_j} = 1|} \sum_{(u_i, v_j)|\Omega_{u_i, v_j} = 1} \mathbf{CE}(r[u_i, v_j], \hat{r}[u_i, v_j]), \tag{16}$$

where we represent the expanded true rating label as $r[u_i, v_j]$, while $\hat{r}[u_i, v_j]$ signifies the predicted rating for the pair (i, j). In parallel, we introduce hyper-parameters α and β to weigh and optimize the contrastive learning tasks. This leads us to the final loss function, which is defined as follows:

$$\mathcal{L} = \mathcal{L}_{main} + \alpha \mathcal{L}_{MR} + \beta \mathcal{L}_{GL} + \theta \mathcal{L}_{EL} + \varphi \mathcal{L}_{NRR}. \tag{17}$$

Here, we introduce $\varphi \mathcal{L}_{NRR}$ as a regularization mechanism inspired by prior models [1,22]. This regularization method is designed to account for both magnitude and order information, which aims to foster similarity in the representation of each node in the rating subgraph that is adjacent to each other.

5 Experiment

5.1 Experimental Settings

Datasets. We use two publicly benchmark data source, Amazon and Yelp, where Amazon includes "Digital_Music", "Toys_and_Games", "Clothing" and "CDs_and_Vinly" four domains. Following the preprocessing steps outlined in [23], we convert all datasets to "5-core" format, which means there are at least five reviews for each user or item. Table 1 presents the summary of dataset statistics.

Table 1. Statistical summaries of datasets, U: number of users; I: number of items; R: rating counts; We present percentage distributions of user ratings.

Dataset	U	I	R	density(%)	Ratings%				
					1	2	3	4	5
Yelp	8,423	3,742	88,647	0.281	5.88	9.41	15.29	37.13	32.09
Digital_Music	5,541	3,568	64,706	0.330	4.31	4.65	10.49	25.55	54.98
Clothing	39,387	23,033	278,677	4.01	5.55	10.91	27.14	20.94	58.57
Toys_and_Games	19,412	11,924	167,597	0.072	2.80	3.75	9.75	22.34	61.33
CDs_and_Vinly	75,258	64,443	1,097,592	0.023	4.06	4.06	8.86	21.47	57.06

Evaluation Protocols and Metrics. We split our training, validation, and test datasets in the ratio of 8:1:1 following the same settings in the most recent studies [22,40]. In order to mitigate the sampling bias and to fairly evaluate the prediction results of the ratings, we set up MAE and MSE, two widely used metrics in the field of matrix complementation.

Baselines. The baselines can be broadly classified into three distinct groups: (1) **Traditional methods**: SVD [18] is a matrix factorization model that uses the inner product of users' and items' latent factors to estimate ratings. NNMF [4] is a classical matrix factorization model that uses the user-item ratings only as the target value of their objective function. (2) **Review-based methods**: DeepCoNN [1] is one of the pioneer works that applies a neural network to a review recommendation system, using two parallel CNNs to capture the latent factors of users and items. NARRE [22] is a model based on the review level attention mechanism, which estimates the importance of different reviews. DAML [36] uses both local and interactive attention layers to learn the embedding of users and items when doing CNN convolution. TransNets [42] converts the learned information into an approximation of the corresponding review through a transformation layer, and uses it for rating prediction. (3) **Graph and review fusion**

methods: RMG [32] fuses review information in graph structure to select important words, sentences, and comments through a three-level attention network. SSG [6] augments the existing single-view approach by merging two additional views and constructs a three-way encoder to capture the long-term, short-term, and collaborative features of users and items for recommendation. RGCL [23] is a graph-based contrastive learning framework that creates a review-aware user-item graph with edge features enriched by review content and introduces two additional contrastive learning tasks to provide self-supervised signals.

Parameter Setting. We employ the Adam optimizer for the entirety of model training, initializing all trainable parameters using the Xavier method. We systematically explore several key hyperparameters: The number of layers is varied within the range $\{1, 2, ..., 5\}$. The hyperparameters α, β and θ in a range of $\{0.001, 0.01, 0.02, 0.05, 0.1, 0.2\}$, λ is searched in $\{1e^{-7}$, $1e^{-6}$, $1e^{-5}$, $1e^{-4}\}$. In the embedding layer, we consider different vector sizes, selecting from 30, 60, ..., 1200, and the dimension of the review embedding generated by BERT-Whitening is set to 64. The number of hyperedges is searched in $\{4, 8, 16, 32, 128\}$. Additionally, we implement early stopping to mitigate the risk of overfitting. The complete model implementation is carried out using the Deep Graph Library [30] and PyTorch, harnessing the computational power of an Intel Xeon Gold 5320T CPU (2.3GHz) with two NVIDIA 3090 GPUs. The source code of our model is available at: https://github.com/lx970414/PKDD24/tree/master.

5.2 Comparison and Result Analysis

Performance comparison of baselines and our proposed HGCL has been shown in the Table 2. Compared with traditional recommendation models based matrix factorization, SVD and NNMF, other GNN-based baselines that fusing graph structure information have achieved better performance, which proves the advantages of graph learning for modeling the interaction between nodes.

Specifically, for review-based methods, Deep- CoNN and DAML both use CNNs to capture the latent factors of users and items, which performs well but ignores important graph-level information. NARRE is a review-based model with a review-level attention mechanism, which estimates the importance of different reviews, but it neglects high-order information of nodes. TransNets converts the learned information into an approximation of the corresponding review through a transformation layer. However, it brings a lot of extra time overhead and the experimental performance is inferior to our proposed HGCL.

For the graph and review fusion method, RMG, uses review information in graph structure to select important context through a three-level attention network, but it only focuses on local information of nodes and reviews which leads to the lack of high-order information. For SSG and RGCL, which also incorporate graph structure information, perform worse than the other two fusion methods. This can be attributed to the fact that SSG has difficulty in to adequately capturing collaborative information shared between users and items. Furthermore, HGCL achieves the best performance across all datasets and shows a significant improvement compared to the latest graph-review fusion method RGCL. This can be attributed to hierarchical graph contrastive learnings of HGCL, which alleviative long-tail distribution by rendering the learned representation distribution more uniform. Additionally, HGCL captures high-order correlations between nodes through the hypergraph learner and uses cross-view contrastive learning to achieve mutual enhancement of information between different views. Finally, HGCL also captures the intrinsic correlation between ratings and reviews through edge-level contrastive learning.

Table 2. Performance comparison of the proposed model and different baselines.

Method	Digital_Music		Toys_and_Games		Clothing		CDs_and_Vinly		Yelp	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
SVD	0.8523	0.6879	0.8301	0.6812	1.1167	0.8082	0.8662	0.8246	1.1741	0.8705
NNMF	0.8316	0.6960	0.8296	0.6822	1.1156	0.8095	0.8663	0.8257	1.1695	0.8637
DeepCoNN	0.8378	0.7032	0.8207	0.6739	1.1184	0.8113	0.8621	0.8231	1.1671	0.8611
NARRE	0.8172	0.6616	0.8175	0.6730	1.1064	0.8077	0.8495	0.8065	1.1673	0.8551
DAML	0.8237	0.6597	0.8143	0.6714	1.1065	0.8082	0.8483	0.8072	1.1588	0.8471
TransNets	0.8273	0.6788	0.8163	0.6782	1.1141	0.8127	0.8440	0.7986	1.1661	0.8447
RMG	0.8074	0.6635	0.8092	0.6586	1.1064	0.8067	0.8425	0.7991	1.1507	0.8433
SSG	0.8218	0.6780	0.8194	0.6770	1.1228	0.8135	0.8458	0.8013	1.1613	0.8529
RGCL	0.7735	0.6524	0.7984	0.6502	1.0858	0.7783	0.8180	0.7729	1.1397	0.8394
HGCL	**0.7585**	**0.6240**	**0.7657**	**0.6355**	**1.0753**	**0.7731**	**0.8059**	**0.7624**	**1.1172**	**0.8097**

5.3 Ablation Study

To evaluate the effectiveness of each component, we further conduct the ablation study on different HGCL variations. We report the results of the ablation study on five datasets in Table 3. Specifically, we generate the following variants:

- **w/o R** - which removes review-aware graph convolution.
- **w/o CLH** - which removes all the hierarchical contrastive learnings and hypergraph convolution.
- **w/o M-R** - which removes the multi-ratings contrastive learning.
- **w/o G** - which removes the hypergraph structure learning module and hypergraph convolution on learned hypergraph, so that the final representation of the node contains only local embeddings.
- **w/o ECL** - which removes edge-level contrastive learning.

Based on the results, we have the following four observations: (1) The results of **w/o ECL** and **w/o R** demonstrate the necessity of considering the intrinsic correlation between ratings and reviews in HGCL. Furthermore, **w/o CLH** performs worse than all variants on all datasets, which is equivalent to degenerating into a normal GAE-based model. (2) Both **w/o M-R** and **w/o G** have decreased compared to HGCL, which indicates the effectiveness of these two

Table 3. Performance comparison of different variants (MSE)

Dataset	Digital_Music	Toys_and_Games	Clothing	CDs_and_Vinly	Yelp
w/o R	0.7638	0.7731	1.0829	0.8133	1.1229
w/o CLH	0.7922	0.7989	1.1124	0.8122	1.1381
w/o M-R	0.7809	0.7863	1.0848	0.8225	1.1204
w/o G	0.7694	0.7742	1.0913	0.8159	1.1259
w/o ECL	0.7642	0.7701	1.0811	0.8127	1.1241
HGCL	0.7585	0.7657	1.0753	0.8059	1.1172

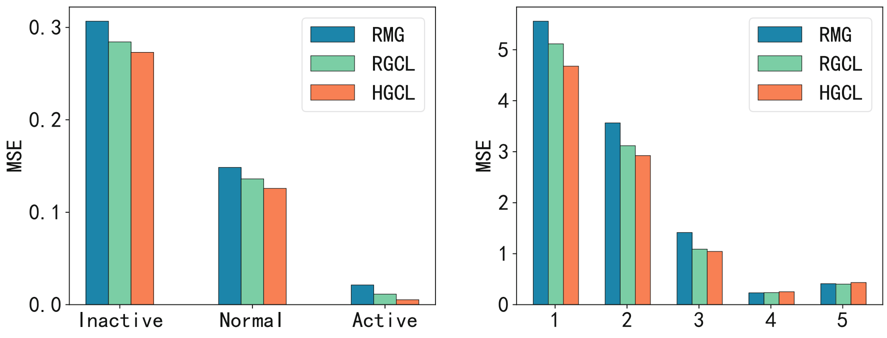
modules. Specifically, we can see that the two modules contribute differently to different datasets, which may be affected by the situation of specific datasets. For example, multi-ratings contrastive learning has significantly improved performance in Toys and CDs, which may be due to the most prominent issue of rating imbalanced distribution. The better performance of hypergraph learning for Yelp and Clothing may be due to the higher density of these two datasets and more complex dependency relations between nodes, thus requiring hypergraphs to represent the global collaborative information.

5.4 Impact of Imbalanced Data Distribution

To validate HGCL's efficacy in addressing popularity bias, we conduct experiments assessing the impact of imbalanced node interaction distribution and rating distribution. In the interest of brevity, we present the results specifically for Digital_Music datasets. However, it is worth noting that similar observations are made across other datasets, reinforcing the robustness of our findings.

Impact of Imbalanced Node Interactions. In this section, we assess the performance of HGCL across users exhibiting varying degrees of sparsity in their interactions. To achieve this, we categorize all users into three distinct subsets, based on their interaction frequencies. Specifically, we designate 80% of users with the fewest interactions as 'Inactive', classify the top 5% of users with the most interactions as 'Active', and assign the remaining users as 'Normal'. The resulting finding is visually depicted in Fig. 2(a). Notably, this figure reveals a consistent trend wherein HGCL consistently outperforms the two baseline models. Intriguingly, the observed improvements are most pronounced among users with lower interaction frequencies. This phenomenon can be attributed to the effectiveness of InfoNCE losses, which address the popularity bias issue by promoting uniformity in the learned representations.

Impact of Imbalanced Rating Interactions. In this section, we present an analysis of HGCL's performance across different rating categories. The outcomes of this evaluation are visually depicted in Fig. 2(b). Our findings underscore the significant improvements achieved, particularly in the context of long-tail ratings



(a) Each group on Digital_Music (b) Each rating on Digital_Music

Fig. 2. Impact of imbalanced data distribution.

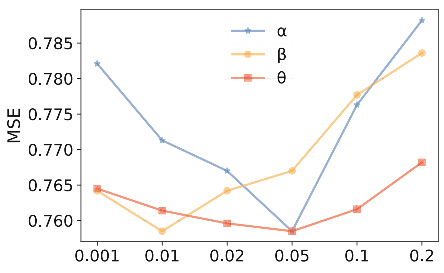
('1' and '2'). Importantly, these enhancements do not come at the expense of performance in more common data categories (head class). This compelling result reaffirms the role of InfoNCE in reshaping the overall distribution of embeddings, aligning them more closely with the characteristics of long-tail rating distributions.

5.5 Parameter Sensitivity

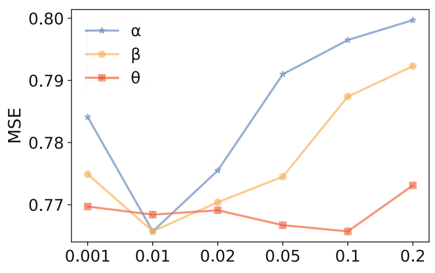
In our evaluation, we systematically investigate the sensitivity of HGCL with regard to four primary hyperparameters: α, β, θ, and the number of hyperedges. The corresponding MSE results obtained under different hyperparameter settings are presented Fig. 3. For the sake of conciseness, we present results exclusively for the ML-100K and Amazon datasets; however, it's important to emphasize that our observations hold true across other datasets as well.

Effect of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$. To assess the impact of three contrastive learning losses, we conduct a series of parameter sensitivity experiments. The range for α, β and θ is varied from 0.001 to 0.2. The resulting insights of three hyperparameters are presented in Fig. 3(a) and 3(b), respectively. We observe similar trends in the variation of three parameters across the two datasets. Initially, as the values increase, the performance of HGCL improves on both datasets. This indicates that contrastive learning effectively facilitates coordinated node representation learning within HGCL. However, it is noteworthy that when the parameters become too large, they might affect the importance of the loss function of the specific task in the model learning, thus damaging the model performance.

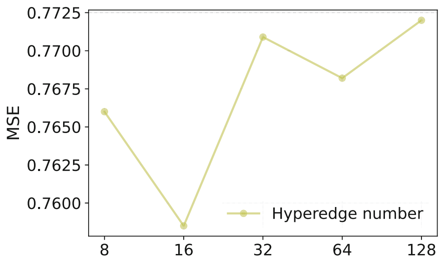
Effect of Hyperedge Number. This part delves into the exploration of the optimal number of hyperedges. We conduct experiments by varying the hyperedge numbers, ranging from 4 to 128 on two datasets. The resulting insights are presented in Fig. 3(c) and 3(d). For Digital Music, the optimal number of



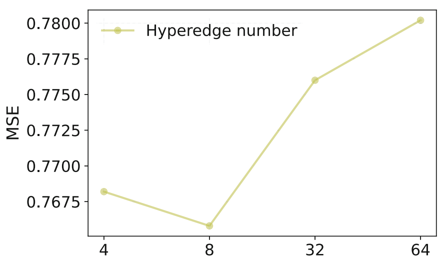
(a) Digital_Music



(b) Toys_and_Games



(c) Digital_Music



(d) Toys_and_Games

Fig. 3. The parameter sensitivity results.

hyperedges appears to be 16, while on Toys and Games, the model's performance reaches its peak at 8. Interestingly, further increasing the number of hyperedges beyond these optimal values tends to degrade model performance to varying degrees, which can be attributed to the fact that an excessive number of hyperedges can introduce unnecessary noise into the representations.

6 Conclusion

In this paper, we introduce a novel Hierarchical Graph Contrastive Learning (HGCL) for review-enhanced recommendation. First, we apply review-aware graph convolution on each rating subgraph, dynamically learning the hypergraph structure of different subgraphs. Second, we employ a cross-rating contrastive learning to enhance mutual reinforcement by encouraging alignment between adjacent views. Third, we use global-local contrastive learning to collaboratively learn node representations. Additionally, we design an edge-level contrastive learning to strengthen the intrinsic correlation between reviews and ratings. Experiment results on five benchmark datasets show the superiority and effectiveness of the proposed HGCL.

Acknowledgments. This work is partially supported by the National Natural Science Foundation of China under grant Nos. 62176243 and 62372421, the Project funded by China Postdoctoral Science Foundation (No. 2023M743328), the Postdoctoral Fellowship Program of CPSF (No. GZC20232500), and the Postdoctoral Project funded by Qingdao (No. QDBSH20240102093).

References

1. Berg, R.V.D., Kipf, T.N., Welling, M.: Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263 (2017)
2. Cai, X., Huang, C., Xia, L., Ren, X.: Lightgcl: simple yet effective graph contrastive learning for recommendation. In: ICLR (2022)
3. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: ICML, pp. 1597–1607. PMLR (2020)
4. Dziugaite, G.K., Roy, D.M.: Neural network matrix factorization. arXiv preprint arXiv:1511.06443 (2015)
5. Fu, C., Zheng, G., Huang, C., Yu, Y., Dong, J.: Multiplex heterogeneous graph neural network with behavior pattern modeling. In: SIGKDD, pp. 482–494 (2023)
6. Gao, J., Lin, Y., Wang, Y., Wang, X., Yang, Z., He, Y., Chu, X.: Set-sequence-graph: a multi-view approach towards exploiting reviews for recommendation. In: CIKM, pp. 395–404 (2020)
7. Giorgi, J., Nitski, O., Wang, B., Bader, G.: Declutr: deep contrastive learning for unsupervised textual representations. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 879–895 (2021)
8. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: a new estimation principle for unnormalized statistical models. In: ICAIS, pp. 297–304. JMLR Workshop and Conference Proceedings (2010)
9. Ji, S., Feng, Y., Ji, R., Zhao, X., Tang, W., Gao, Y.: Dual channel hypergraph collaborative filtering. In: SIGKDD, pp. 2020–2029 (2020)
10. Kalofolias, V., Bresson, X., Bronstein, M., Vandergheynst, P.: Matrix completion on graphs. In: Neural Information Processing Systems Workshop (2014)
11. Kenton, J.D.M.W.C., Toutanova, L.K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT, pp. 4171–4186 (2019)
12. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR (2016)
13. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: a lite BERT for self-supervised learning of language representations. In: ICLR (2019)
14. Liu, M., Xu, X., Li, J., Li, G.: A review-based feature-level information aggregation model for graph collaborative filtering. Neurocomputing 126697 (2023)
15. Ma, Y., He, Y., Zhang, A., Wang, X., Chua, T.S.: Crosscbr: cross-view contrastive learning for bundle recommendation. In: SIGKDD, pp. 1233–1241 (2022)
16. Mahadevan, A., Arock, M.: A class imbalance-aware review rating prediction using hybrid sampling and ensemble learning. Multimedia Tools Appl. **80**, 6911–6938 (2021)
17. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)

18. Rao, N., Yu, H.F., Ravikumar, P.K., Dhillon, I.S.: Collaborative filtering with graph information: consistency and scalable methods. In: Advances in Neural Information Processing Systems, vol. 28 (2015)
19. Ren, Y., Liu, B.: Heterogeneous deep graph infomax. In: Workshop of Deep Learning on Graphs: Methodologies and Applications co-located with the Thirty-Fourth AAAI Conference on Artificial Intelligence (2020)
20. Ren, Y., et al.: Disentangled graph contrastive learning for review-based recommendation. arXiv preprint arXiv:2209.01524 (2022)
21. Seo, S., Huang, J., Yang, H., Liu, Y.: Interpretable convolutional neural networks with dual local and global attention for review rating prediction. In: RecSys, pp. 297–305 (2017)
22. Shen, W., Zhang, C., Tian, Y., Zeng, L., He, X., Dou, W., Xu, X.: Inductive matrix completion using graph autoencoder. In: CIKM, pp. 1609–1618 (2021)
23. Shuai, J., et al.: A review-aware graph contrastive learning framework for recommendation. In: SIGIR, pp. 1283–1293 (2022)
24. Su, J., Cao, J., Liu, W., Ou, Y.: Whitening sentence representations for better semantics and faster retrieval. arXiv preprint arXiv:2103.15316 (2021)
25. Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., Jiang, P.: Bert4rec: sequential recommendation with bidirectional encoder representations from transformer. In: CIKM, pp. 1441–1450 (2019)
26. Thakoor, S., Tallec, C., Azar, M.G., Munos, R., Veličković, P., Valko, M.: Bootstrapped representation learning on graphs. In: ICLR Workshop (2021)
27. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018)
28. Veličković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D.: Deep graph infomax. In: ICLR (2018)
29. Wang, K., Zhu, Y., Zang, T., Wang, C., Liu, K., Ma, P.: Multi-aspect graph contrastive learning for review-enhanced recommendation. ACM TOIS **42**(2), 1–29 (2023)
30. Wang, M.Y.: Deep graph library: towards efficient and scalable deep learning on graphs. In: ICLR Workshop (2019)
31. Wei, C., Liang, J., Bai, B., Liu, D.: Dynamic hypergraph learning for collaborative filtering. In: CIKM, pp. 2108–2117 (2022)
32. Wu, C., Wu, F., Qi, T., Ge, S., Huang, Y., Xie, X.: Reviews meet graphs: enhancing user and item representations for recommendation with hierarchical attentive graph neural network. In: EMNLP-IJCNLP, pp. 4884–4893 (2019)
33. Wu, J., et al.: Self-supervised graph learning for recommendation. In: SIGIR, pp. 726–735 (2021)
34. Wu, Y., Huang, X.: A gumbel-based rating prediction framework for imbalanced recommendation. In: CIKM, pp. 2199–2209 (2022)
35. Xia, L., Huang, C., Xu, Y., Zhao, J., Yin, D., Huang, J.: Hypergraph contrastive collaborative filtering. In: SIGIR, pp. 70–79 (2022)
36. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W.L., Leskovec, J.: Graph convolutional neural networks for web-scale recommender systems. In: SIGKDD, pp. 974–983 (2018)
37. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. In: NeurIPS, pp. 5812–5823 (2020)
38. Yu, P., Fu, C., Yu, Y., Huang, C., Zhao, Z., Dong, J.: Multiplex heterogeneous graph convolutional network. In: SIGKDD, pp. 2377–2387 (2022)

39. Yuan, X., et al.: Multimodal contrastive training for visual representation learning. In: CVPR, pp. 6995–7004 (2021)
40. Zhang, C., Chen, H., Zhang, S., Xu, G., Gao, J.: Geometric inductive matrix completion: a hyperbolic approach with unified message passing. In: WSDM, pp. 1337–1346 (2022)
41. Zhang, J., Shi, X., Zhao, S., King, I.: Star-GCN: stacked and reconstructed graph convolutional networks for recommender systems. In: IJCAI, p. 4264 (2019)
42. Zhang, M., Chen, Y.: Inductive matrix completion based on graph neural networks. In: ICLR (2020)
43. Zhao, X., Zhu, Z., Zhang, Y., Caverlee, J.: Improving the estimation of tail ratings in recommender system with multi-latent representations. In: WSDM, pp. 762–770 (2020)
44. Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Deep graph contrastive representation learning. arXiv preprint arXiv:2006.04131 (2020)

Linear Contextual Bandits with Hybrid Payoff: Revisited

Nirjhar Das(✉) and Gaurav Sinha

Microsoft Research India, Bangalore, India
nirjhar.das@alumni.iitd.ac.in, gauravsinha@microsoft.com

Abstract. We study the Linear Contextual Bandit (`LinearCB`) problem in the hybrid reward setting. In this setting, every arm's reward model contains arm specific parameters in addition to parameters shared across the reward models of all the arms. We can easily reduce this setting to two closely related settings; (a) *Shared* - no arm specific parameters, and (b) *Disjoint* - only arm specific parameters, enabling the application of two popular state of the art algorithms - `LinUCB` and `DisLinUCB` (proposed as Algorithm 1 in Li et al. 2010). When the arm features are stochastic and satisfy a popular diversity condition, we provide new regret analyses for both `LinUCB` and `DisLinUCB` that significantly improves upon the known regret guarantees of these algorithms. Our novel analysis critically exploits the structure of the hybrid rewards and diversity of the arm features. Along with proving these new guarantees, we introduce a new algorithm `HyLinUCB` that crucially modifies `LinUCB` (using a new exploration coefficient) to account for sparsity in the hybrid setting. Under the same diversity assumptions, we prove that at the end of T rounds, `HyLinUCB` also incurs only $\tilde{O}(\sqrt{T})$ regret. We perform extensive experiments on synthetic and real-world datasets demonstrating strong empirical performance of `HyLinUCB`. When the number of arm specific parameters is much larger than the number of shared parameters, we observe that `DisLinUCB` incurs the lowest regret. In this case, regret of `HyLinUCB` is the second best and it is extremely competitive to `DisLinUCB`. In all other situations, including our real-world dataset, `HyLinUCB` has significantly lower regret than `LinUCB`, `DisLinUCB` and other state of the art baselines we considered. We also empirically observe that the regret of `HyLinUCB` grows much slower with the number of arms K, compared to baselines, making it suitable even for very large action spaces.

Keywords: Linear Contextual Bandits · Hybrid Payoff · Stochastic Contexts

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-70365-2_26.

1 Introduction

The LinearCB problem is a popular choice to model sequential decision making scenarios such as news recommendations [16], web-page optimization [11] etc. In LinearCB, at each round, a learner receives (from the environment) a set of arms (actions) specified as feature vectors, selects one of them, and receives a reward sampled from a linear model (over the features of the selected arm). The goal of the learner is to design a policy of selecting arms that minimizes its cumulative regret. Here, regret of a round is calculated as the difference between the best possible reward in that round and the actual reward received.

The LinearCB problem is most commonly studied under the *shared setting*. Here, the linear reward model is shared across all the arms and algorithms such as LinUCB [15,16] achieve state of the art regret. However, this setting can be quite restrictive in applications where the reward model is different for different arms. For example, in news recommendation, where arms are different news items, the reward model (say click through rate) for news items about sports could be very different from that of politics, movies etc. As a result, the problem has also been studied under the *disjoint* and *hybrid* settings [16]. While in the disjoint setting, the reward model of each arm only contains parameters specific to the arm (called arm-specific parameters), in the hybrid setting they also contain parameters shared across all the arms (called shared parameters). Algorithms were developed for the disjoint and hybrid settings in Algorithm 1 and Algorithm 2 (respectively) of [16]. While Algorithm 1 (called DisLinUCB from here on-wards) can be analyzed using ideas similar to the analysis of LinUCB for the shared setting [15], Algorithm 2 requires tuning of a hyper-parameter. Analyzing the regret of this algorithm is quite non-trivial and to the best of our knowledge tight regret guarantees for the hybrid setting are not known. Designing algorithms for the hybrid setting that overcome this challenge and also have strong regret guarantees is the main focus of our work. For each arm's reward model, let d_1, d_2 be the number of shared and arm specific parameters respectively, and let K be the total number of arms. Further, define $d \coloneqq d_1 + d_2K$. We make the following contributions.

1.1 Our Contributions

1. First, we reduce the hybrid setting to the shared setting with d parameters and provide a new analysis for LinUCB [15]. We prove that if features of arms pulled by LinUCB satisfy Assumption 1, then, at the end of T rounds, LinUCB incurs a regret of $\tilde{O}(\sqrt{dKT})$, when $T = \tilde{\Omega}(K^3)$. Note that the standard guarantee of LinUCB for this reduced problem is $\tilde{O}(d\sqrt{T})$.
2. Next, by reducing to the disjoint setting, we provide a new analysis for DisLinUCB. We prove that, under the same assumption as above, at the end of T rounds, DisLinUCB incurs a regret of $\tilde{O}\left(\sqrt{(d_1+d_2)KT}\right)$, when $T = \tilde{\Omega}(1)$. The standard analysis (following analysis of LinUCB in [15]) implies $\tilde{O}((d_1+d_2)\sqrt{KT})$ regret, which is much worse.

3. Finally, we modify `LinUCB` using a tighter exploration parameter and develop a new algorithm `HyLinUCB`. Under the same diversity assumption, we prove a $\tilde{O}(\sqrt{K^3T} + \sqrt{dKT})$ regret guarantee for this algorithm. While our guarantee has a weaker dependence on K compared to the above algorithms, we empirically observe it to be much stronger compared to them. By performing extensive experiments on synthetic (capturing a wide range of problem settings) and real-world datasets (Yahoo! Front Page Dataset [25]), we demonstrate that in almost all cases `HyLinUCB` has much lower regret than `LinUCB`, `DisLinUCB` and other baselines. When the number of shared parameters is larger than the arm-specific parameters, regret of `HyLinUCB` is significantly lower than all state of the art baselines we compare it with. In the other case, i.e., when the number of arm-specific parameters are larger, `HyLinUCB` is extremely competitive with the best algorithm i.e. `DisLinUCB`. We also empirically study the variation in regret with respect to the number of arms K, and observe that regret of `HyLinUCB` grows much slower with K, compared to baselines, making it suitable even for very large action spaces.

1.2 Additional Remarks on Contributions

We make some additional remarks to provide more clarity on our contributions.

Reduction to the Shared Setting: The hybrid setting is easily reduced to the shared setting by combining the parameters of linear reward models of all the actions into a common reward model. We provide complete details of this reduction in Sect. 2. This reduction helps us apply most of the known algorithms for `LinearCB` (discussed in detail in Sect. 1.3). Note that the reduction significantly increases the dimensionality of the arm features and the known regret guarantees of these `LinearCB` algorithms might not be optimal anymore. The additional structure in the problem needs to be exploited in the analysis to improve these guarantees. Our first contribution does this for `LinUCB`. Similar improved analysis might exist for other state of the art `LinearCB` algorithms we discuss in Sect. 1.3. We leave this problem for future work.

Regret Guarantee of `HyLinUCB`: Even though our regret guarantee for `HyLinUCB` has a slightly worse dependence on K, we believe this is an artifact of our proof and that it can be improved further. Our synthetic and real-data experiments provide strong evidence of this as `HyLinUCB` performs much better than `LinUCB` in all the different problem settings we consider. Regret guarantee that improves this dependence is an interesting direction for future work.

Diversity Assumption: Our regret guarantees are derived under the assumption that the arms pulled by the algorithms satisfy a diversity condition (Assumption 1) that has been studied in [5,9,10]. The assumption states that the minimum eigenvalue of the expected outer product of the pulled arm's feature vector with itself, is bounded away from zero. This assumption allows the algorithms to perform parameter estimation in conjunction to regret minimization, the former being crucial to derive better dependence on the number of arms in the regret upper bound. The necessity of such diversity assumptions and their implications on better regret rates has been studied in [19]. Even though the assumption is algorithm dependent, there are problem instances for which all algorithms satisfy it (Sect. 3, [19]). We empirically show that Assumption 1 is indeed satisfied by the algorithms we study in this work.

1.3 Related Work

As highlighted in Sect. 1.2, the `LinearCB` problem in the hybrid setting can be reduced to the shared setting, enabling the application of all `LinearCB` algorithms designed for the shared setting. Over the last couple of decades, there has been a substantial progress on `LinearCB` for the shared setting. Many state of the art algorithms follow the optimism in the face of uncertainty approach [14], and provide regret analysis with near optimal guarantees [1,4,6,8,20]. For general stochastic arm features, the best known regret upper bound (independent of number of arms K) provided in these works is $\tilde{O}(d\sqrt{T})$ and the best known lower bound [6] is $\tilde{\Omega}(\sqrt{dT})$, where d is the dimensionality of the arm features and T is the number of rounds. The `LinUCB` algorithm [16] attains this upper bound ($\tilde{O}(d\sqrt{T})$) [15] and is also a popular choice for applications such as news recommendation [16] due to it's strong empirical performance. Moreover, it was also adapted to the hybrid setting in [16]. As a result we choose it as one of our main candidate algorithms to study in this paper. The popular `SupLinUCB` [6] algorithm improves the dependence on d and guarantees $\tilde{O}(\sqrt{dT}\log^{3/2} K)$ regret where K is the number of actions. A recent variant of the `SupLinUCB` algorithm [18] improves this further to $\tilde{O}(\sqrt{dT}\log K)$. While the dependence on d improves, it is traded off with a logarithmic dependence on K. It has also been noted [13] that despite strong regret guarantees, `SupLinUCB` and its variants [17,18] explore excessively and perform worse in practice due to computational inefficiency. Recently, [13] developed a new algorithm `HyRan` which does not depend on K in the leading terms and attains a regret guarantee of $\tilde{O}(\sqrt{dT})$ (improving significantly over `SupLinUCB` both theoretically and empirically). While they prove this under an additional assumption on stochasticity of the arm features, they also provide a matching lower bound showing tightness of their guarantee. Their experiments validate their strong upper bound by demonstrating superior performance compared to many state of the art `LinearCB` baselines for a large variety of problem instances. Due to their strong guarantee and empirical performance, we compare our algorithms with `HyRan` in Sect. 4. Another family of algorithms (based on randomized exploration) referred to as Thompson Sampling [22] based methods are an active area of research. Linear contextual version of this method was recently developed [2,3] and best known regret guarantees are either $\tilde{O}(d\sqrt{T\log K})$ or $\tilde{O}(d^{3/2}\sqrt{T})$. Both guarantees are worse than that of `LinUCB`. Even though Thompson Sampling does well practically, the `HyRan` algorithm from [13] was shown to be superior to them for all problem instances considered in [13]. Since we already compare with `HyRan` in our experiments in Sect. 4, we do not compare to Thompson Sampling based methods.

2 Problem Formulation

Notations: $[N]$ denotes the set $\{1, \ldots, N\}$. $K, d_1, d_2 \in \mathbb{N}$ denote the number of arms, shared parameters and arm-specific parameters respectively. We denote by $\|\cdot\|_2$ the ℓ_2 norm of a vector and by $\|\cdot\|$ the operator norm of a matrix. We denote all matrices with boldface and vectors in small case. For two symmetric matrices $\mathbf{A}$ and $\mathbf{B}$, we write $\mathbf{A} \succcurlyeq \mathbf{B}$ to indicate that $\mathbf{A} - \mathbf{B}$ is positive semi-definite. Further, we will denote by $\mathbf{I}_p$ the $p \times p$ identity matrix and $\mathbf{0}$ as the zero matrix/vector (whose dimension will be clear from context when not specified).

`LinearCB` **with Hybrid Rewards:** We assume there exist unknown parameter vectors $\theta^* \in \mathbb{R}^{d_1}$ and $\beta_i^* \in \mathbb{R}^{d_2}$ for $i \in [K]$. At each round $t \in \mathbb{N}$, the learner receives a set of K feature vector tuples $\mathcal{X}_t = \{(x_{i,t}, z_{i,t}) : i \in [K]\}$. For each arm $i \in [K]$, the features $x_{i,t} \in \mathbb{R}^{d_1}$ correspond to the parameter vector θ^* and the features $z_{i,t} \in \mathbb{R}^{d_2}$ correspond to the arm-specific parameters β_i^*, i.e. if the learner selects arm $i_t \in [K]$ at round t, she receives a reward

$$r_t = \langle x_{i_t,t}, \theta^* \rangle + \langle z_{i_t,t}, \beta_{i_t}^* \rangle + \eta_t,$$

where η_t is a conditionally 1-subgaussian random noise. Specifically, let $\mathcal{F}_t$ be the filtration $(\mathcal{X}_1, i_1, r_1, \ldots, r_{t-1})$, then, $\mathbb{E}[\eta_t \mid \mathcal{F}_t] = 0$, and, $\mathbb{E}[e^{\alpha \eta_t} \mid \mathcal{F}_t] \leq e^{\alpha^2/2}$, for all $\alpha \in \mathbb{R}$. Note that we do not assume the noise variable η_t to depend on the arms selected by the learner. This later allows us to reduce the problem to the shared setting albeit in a larger dimension. We define the (cumulative) regret of the learner `Alg` at the end of round T as,

$$\texttt{Reg}(T, \texttt{Alg}) = \sum_{t=1}^{T} \max_{j \in [K]} \left(\langle x_{j,t}, \theta^* \rangle + \langle z_{j,t}, \beta_j^* \rangle \right) - \sum_{t=1}^{T} \left(\langle x_{i_t,t}, \theta^* \rangle + \langle z_{i_t,t}, \beta_{i_t}^* \rangle \right).$$

Assumptions: The key assumption in this work is the diversity assumption, similar to the one made in [5,9,10] for the shared setting. We extend this assumption (stated below) to the hybrid setting by adding one extra assumption that arises from the hybrid nature of the problem.

Assumption 1 (Diversity). *There exists a constant $\rho > 0$, such that features of the arm selected by the algorithm, i.e., $x_{i_t,t}$, $z_{i_t,t}$, for all rounds $t \in \mathbb{N}$, satisfy,*

1. $\mathbb{E}[x_{i_t,t} \mid \mathcal{F}_{t-1}] = \mathbf{0}, \mathbb{E}[z_{i_t,t} \mid \mathcal{F}_{t-1}] = \mathbf{0}$, *and* $\mathbb{E}\left[x_{i_t,t} z_{i_t,t}^{\mathsf{T}} \mid \mathcal{F}_{t-1}\right] = \mathbf{0}$,
2. $\mathbb{E}\left[x_{i_t,t} x_{i_t,t}^{\mathsf{T}} \mid \mathcal{F}_{t-1}\right] \succcurlyeq \rho \mathbf{I}_{d_1}, \mathbb{E}\left[z_{i_t,t} z_{i_t,t}^{\mathsf{T}} \mid \mathcal{F}_{t-1}\right] \succcurlyeq \rho \mathbf{I}_{d_2}$

We note that this assumption is algorithm dependent, however, there are problem instances in the shared setting for which any algorithm satisfies it (Section 3, [19]). We empirically validate this assumption in the hybrid setting for the algorithms studied in this work, thereby demonstrating that the assumption and its crucial implications indeed hold in practice. The details of this empirical study is presented in Appendix F. We also make the following standard assumption on the reward parameters.

Assumption 2. *There is a fixed constant $S \in \mathbb{R}$ (known to us) such that the reward parameters $\|\theta^*\|_2 \leq S$ and $\|\beta_i^*\|_2 \leq S$ for all $i \in [K]$. Further, $\|x_{i,t}\|_2 \leq 1$ and $\|z_{i,t}\|_2 \leq 1$, for all arms $i \in [K]$ and rounds $t \in \mathbb{N}$.*

Reduction to Shared Setting: We will now formalize our discussion from Sect. 1.2 about reducing the hybrid setting to the shared setting. We first embed the arm features into a sparse feature vector in $\mathbb{R}^{d_1+d_2K}$ via transformation $\mathcal{P} : [K] \times \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \to \mathbb{R}^{d_1+d_2K}$ which takes as input arm $i \in [K]$ and its features $x \in \mathbb{R}^{d_1}, z \in \mathbb{R}^{d_2}$ and maps it as follows:

$$\mathcal{P}(i, x, z) = (x^\intercal, 0, \ldots, 0, z^\intercal, 0, \ldots, 0)^\intercal \tag{1}$$

that is, the coordinate entries of x are copied to the first d_1 coordinates of $\mathcal{P}(i, x, z)$ while the coordinate entries of z are copied to the location starting from $d_1+(i-1)d_2+1$ and ending at d_1+id_2. We also combine the true shared and arm-specific parameters into a global parameter vector $\phi^* = (\theta^{*\intercal}, \beta_1^{*\intercal}, \ldots, \beta_K^{*\intercal})^\intercal$. It's easy to see that this transformation converts the hybrid setting to the shared setting while preserving the mean reward of every arm, i.e.,

$$\langle \mathcal{P}(i, x, z), \phi^* \rangle = \langle x, \theta^* \rangle + \langle z, \beta_i^* \rangle \quad \forall\, i \in [K].$$

Moreover, since the noise variable η_t at round t is assumed to be independent of the arm selected, the noisy rewards $r_t = \langle \mathcal{P}(i, x, z), \phi^* \rangle + \eta_t$ in this shared setting exactly captures the reward received by arm i in the hybrid setting.

More Notations. For a matrix $\mathbf{M} \in \mathbb{R}^{p \times p}$ and vector $a \in \mathbb{R}^p$, for any $p > 0$, we define $\|a\|_{\mathbf{M}} = \sqrt{a^\intercal \mathbf{M} a}$. For a square matrix $\mathbf{A} \in \mathbb{R}^{p \times p}$, we will denote by $\lambda_{max}(\mathbf{A})$ and $\lambda_{min}(\mathbf{A})$ its maximum and minimum eigenvalues respectively. Finally, for $p, q \in \mathbb{R}$, we will write $p \vee q = \max\{p, q\}$ and $p \wedge q = \min\{p, q\}$.

3 Algorithms and Analysis

In this section, we present our main algorithms and their analyses. First, in Sect. 3.1 we present the `LinUCB` and `DisLinUCB` algorithms for the hybrid setting in Algorithms 1 and 2 respectively. These algorithms are well known, however, for completeness we present them with the minor modifications we need to make to apply them to the hybrid setting. Following this, in Theorem 3 and Corollary 1, we present the improved regret guarantees for `LinUCB` and `DisLinUCB` (under Assumptions 1 and 2) respectively. Next, in Sect. 3.2, we introduce the `HyLinUCB` algorithm which is exactly the same as `LinUCB` presented in Algorithm 1 but with a new confidence parameter γ that we describe. We provide some intuition regarding our choice of this parameter and then in Theorem 4 present a regret guarantee for this algorithm under Assumptions 1 and 2.

3.1 LinUCB and DisLinUCB

The LinUCB algorithm (Algorithm 1 with appropriate choices described below) takes as input the total number of rounds T, a regularization parameter λ, an exploration coefficient γ (whose exact values we provide later) that controls the weightage to be put on the UCB (Upper Confidence Bound) bonus term, and a desired failure probability $\delta \in (0, 1)$. In *Step 1*, it performs initialization of the reward parameter vector and the design matrix $\mathbf{M}$. Note that all initialization is done assuming a feature space of dimension $d_1 + d_2K$ which is the dimensionality after reducing the hybrid setting to the shared setting. The algorithm runs for rounds $t = 1$ to $t = T$ (*Steps 2–9*). In *Steps 3,4*, it receives the set $\mathcal{X}_t$ of arm features and for each arm i, converts its features to appropriate shared setting features $\tilde{x}_i \in \mathbb{R}^{d_1+d_2K}$, using the mapping $\mathcal{P}$ described in Sect. 2. Then, in *Step 5*, it selects the arm i_t with the best UCB estimate, and receives a reward r_t corresponding to it. This is then used to update the design matrix and the estimated reward parameters in *Steps 8,9*. For our LinUCB implementation, we set the regularizer $\lambda = 1$ and carefully choose the exploration coefficient $\gamma = S\sqrt{K} + \sqrt{2(d_1 + d_2K)\log(T/\delta)}$[1] (in accordance with Theorem 2 in [1]). This choice of λ and γ is standard for LinUCB in the $d_1 + d_2K$ dimensional shared setting. For these values of λ and γ, the regret analysis of LinUCB [1,15,16], under Assumption 2, leads to a regret guarantee of $\tilde{O}((d_1 + d_2K)\sqrt{T})$. In Theorem 3, we present a much stronger guarantee when Assumption 1 is also satisfied.

Algorithm 1. LinUCB (λ, γ)

```
Require: Regularizer $\lambda$, exploration coefficient $\gamma$, failure probability $\delta \in (0,1)$
 1: Initialize $\widehat{\phi} = \mathbf{0} \in \mathbb{R}^{d_1+d_2K}$, $\mathbf{M} = \lambda\mathbf{I}_{d_1+d_2K}$, $u = \mathbf{0} \in \mathbb{R}^{d_1+d_2K}$
 2: for $t = 1, \ldots, T$ do
 3:     Receive context $\mathcal{X}_t = \{(x_i, z_i)\}_{i\in[K]}$
 4:     $\tilde{x}_i := \mathcal{P}(i, x_i, z_i)$ for all $i \in [K]$
 5:     $i_t = \text{argmax}_{i\in[K]} \langle \tilde{x}_i, \widehat{\phi}\rangle + \gamma \|\tilde{x}_i\|_{\mathbf{M}^{-1}}$
 6:     Play arm $i_t$ and observe reward $r_t$
 7:     $u \leftarrow u + r_t\tilde{x}_t$
 8:     $\mathbf{M} \leftarrow \mathbf{M} + \tilde{x}_t\tilde{x}_t^\intercal$
 9:     Update: $\widehat{\phi} = \mathbf{M}^{-1}u$
10: end for
```

Theorem 3 (Regret of LinUCB). *At the end of T rounds, the regret of LinUCB (Algorithm 1 with $\lambda = 1$, $\gamma = S\sqrt{K} + \sqrt{2(d_1 + d_2K)\log(T/\delta)}$) under Assumptions 1 and 2 is upper bounded by $C\sqrt{(d_1 + d_2K)T\log(T/\delta)}$ with probability at least $1 - 4\delta$. Here, $C > 0$ is a universal constant and T is assumed to be $\tilde{\Omega}(K^4)$.*

The reduction from the hybrid setting to the shared setting leads to a sparse design matrix. Our proof of Theorem 3 critically exploits the structure of this

[1] S is a known upper bound on the magnitude of the reward parameter vector.

sparse matrix and its inverse, along with some key technical ideas involving geometry of random vectors and eigenvalue bounds on random matrices applied to the stochastic arm features and the design matrix. For brevity, we present the complete proof in Appendix A.

Our second algorithm `DisLinUCB` presented in Algorithm 2 (same as Algorithm 1 in [16]) takes the same set of inputs as Algorithm 1 above, i.e., the total number of rounds T, a regularization parameter λ, an exploration coefficient γ and a desired failure probability $\delta \in (0, 1)$. In *Step 1*, for each arm $i \in [K]$, it performs initialization of the reward parameter vectors ϕ_i and $(d_1 + d_2) \times (d_1 + d_2)$ size design matrices $\mathbf{V}_i$. This algorithm treats the hybrid setting as a disjoint setting and therefore the dimensionality of the problem remains the same as the hybrid setting i.e. $d_1 + d_2$. The algorithm runs for rounds $t = 1$ to $t = T$ (*Steps 2–9*). In *Steps 3,4*, it receives the set $\mathcal{X}_t$ of arm features. For each arm $i \in [K]$, it concatenates the shared and disjoint arm features to obtain the $d_1 + d_2$ dimensional feature vector $\overline{x}_i$ and treats it as the vector of arm-specific parameters in the algorithm. Then, in *Step 5*, it selects the arm i_t with the best UCB estimate, and receives a reward r_t corresponding to it. This is then used to update the corresponding design matrix (i.e. $\mathbf{V}_{i_t}$) and the estimated reward parameter vector (i.e. ϕ_{i_t}) in *Steps 8,9*. A key difference from the previous algorithm is that in `DisLinUCB` the UCB bonus for arm $i \in [K]$ is computed using the arm specific design matrix i.e. $\mathbf{V}_i$, whereas in `LinUCB` it is computed using the overall design matrix $\mathbf{M}$. In our `DisLinUCB` instantiation, we set the regularizer $\lambda = 1$ and choose the exploration coefficient $\gamma = 2\sqrt{S} + \sqrt{2(d_1 + d_2)\log(KT/\delta)}$. This choice of λ and γ is standard for `LinUCB` in the $d_1 + d_2$ dimensional shared setting. For these values of λ and γ, the regret analysis of `LinUCB` (under Assumption 2) can be appropriately modified to provide a regret guarantee of $\tilde{O}((d_1 + d_2)\sqrt{KT})$ for `DisLinUCB`. In Corollary 1, we present a much stronger guarantee when Assumption 1 is also satisfied.

Algorithm 2. `DisLinUCB` (Algorithm 1 in [16])

```
Require: Regularizer $\lambda$, exploration coefficient $\gamma$, failure probability $\delta \in (0, 1)$
 1: For every arm $i \in [K]$, initialize $\phi_i = \mathbf{0} \in \mathbb{R}^{d_1+d_2}$, $\mathbf{V}_i = \lambda \mathbf{I}_{d_1+d_2}$, $u_i = \mathbf{0} \in \mathbb{R}^{d_1+d_2}$
 2: for $t = 1, \ldots, T$ do
 3:     Receive context $\mathcal{X}_t = \{(x_i, z_i)\}_{i \in [K]}$
 4:     Set $\overline{x}_i = \left[x_i^\intercal \; z_i^\intercal\right]^\intercal$ for all $i \in [K]$
 5:     $i_t = \text{argmax}_{i \in [K]} \langle \overline{x}_i, \phi_i \rangle + \gamma \left\| \overline{x}_i \right\|_{\mathbf{V}_i}$
 6:     Play arm $i_t$ and observe reward $r_t$
 7:     $u_i \leftarrow u_i + r_t \overline{x}_{i_t}$
 8:     $\mathbf{V}_{i_t} \leftarrow \mathbf{V}_{i_t} + \overline{x}_{i_t} \overline{x}_{i_t}^\intercal$
 9:     Update $\phi_i = \mathbf{V}_{i_t}^{-1} u_{i_t}$
10: end for
```

Corollary 1 (Regret of `DisLinUCB`). *At the end of T rounds, the regret of `DisLinUCB` (Algorithm 2 with $\lambda = 1$, $\gamma = 2\sqrt{S} + \sqrt{2(d_1 + d_2)\log(KT/\delta)}$) under*

Assumptions 1 and 2 is upper bounded by $C\sqrt{(d_1+d_2)T\log(KT/\delta)}$ *with probability at least* $1-4\delta$*. Here,* $C>0$ *is a universal constant and* T *is assumed to be* $\tilde{\Omega}(1)$*.*

Proof of Corollary 1 uses techniques similar to the ones used in the proof of Theorem 3 with modifications to adapt it to the disjoint setting. Complete proof is provided in Appendix C.

3.2 `HyLinUCB`

Our new algorithm `HyLinUCB`, is derived from Algorithm 1 by using different values for parameters λ and γ. In our `HyLinUCB` instantiation, we set the regularizer $\lambda = K$ and choose the exploration coefficient $\gamma = 2(S\sqrt{K} + \sqrt{2(d_1+d_2)\log(T/\delta)})$. The rest of the algorithm is exactly the same as Algorithm 1. However, due to this change the known regret analysis of `LinUCB` (under Assumption 2) cannot be directly applied to get optimal regret with respect to T. The optimality of the guarantee relies heavily on choosing γ correctly (since it represents confidence in estimation of model parameters) and therefore, even simple modifications do not seem to work. Our main intuition behind using this new value for γ is that even though the feature vector post reduction to the shared setting i.e. $\tilde{x}_i$ (See Step 4 in Algorithm 1) is $d_1 + d_2K$ dimensional, it has only $d_1 + d_2$ non-zero entries. This leads us to the question whether we can use a tighter exploration parameter that depends on the true intrinsic dimensionality of our feature vectors. In Theorem 4, we prove a strong regret guarantee (optimal with respect to T) when Assumptions 1 and 2 are satisfied. We also provide strong empirical evidence in support of our choice for γ in Sect. 4.

Theorem 4 (Regret of `HyLinUCB`). *At the end of* T *rounds, the regret of* `HyLinUCB` *(Algorithm 1 with* $\lambda = K$, $\gamma = 2(S\sqrt{K} + \sqrt{2(d_1+d_2)\log(T/\delta)})$*) under Assumptions 1 and 2 is upper bounded by*

$$C_1\sqrt{K^3T\log(K(d_1+d_2)/\delta)} + C_2\sqrt{(d_1+d_2K)KT\log(KT(d_1+d_2)/\delta)}$$

with probability at least $1-4\delta$*. Here* $C_1, C_2 > 0$ *are universal constants and* T *is assumed to be* $\tilde{\Omega}(K^4)$*.*

Our proof of Theorem 4 requires many new ideas on top of the techniques used in the proof of Theorem 3 in order to accommodate the new confidence parameter. Complete proof is provided in Appendix B. We finish this section with a few remarks on the algorithm.

Remark 1. The scheme of `HyLinUCB` is similar to Algorithm 2 in [16]. However, in [16], the exploration coefficient γ is treated as a hyperparameter to be tuned. As a result it's extremely difficult to compute a regret guarantee and no such guarantee is known. Moreover, it also brings a practical overhead of experimentally finding the optimal parameter. However, in `HyLinUCB`, we fix an appropriate value of γ and prove optimal (with respect to T) regret guarantees. This is where our algorithm differs from [16].

Remark 2. Line 9 in Algorithm 1 requires an inverse of a square matrix of $d_1 + d_2K$ dimension, which requires $O((d_1 + d_2K)^2)$ computation if one uses the Sherman-Morrison formula (Section 0.7.4 in [12]). This is still quadratic in K. However, it can be made linear in K using block matrix inversion technique (see [16,21]). For simplicity, we present the algorithm in the above format.

4 Experimental Setup

In this section, we provide details of our experimental setup. We perform extensive experiments under different problem settings and on real world datasets to compare `HyLinUCB`, `LinUCB`, `DisLinUCB` and a baseline `HyRan` [13]. We set `HyRan` as our baseline because for the shared setting with similar assumptions, it was recently shown to have much better regret than all popular `LinearCB` baselines described in Sect. 1.3. Our experiments are divided into two types (a) synthetic (b) real-world. We provide further details about the setup below. The code for all the experiments are available at https://github.com/nirjhar-das/HyPay_Bandits.

4.1 Synthetic

For the synthetic experiments, the contextual arm features are generated by sampling randomly from some distribution. Since the hybrid reward setting is characterized by 3 key parameters: d_1, d_2 and K, to be exhaustive in our evaluation, we design 3 parameter settings to study the effect of these parameters on the regret of the various algorithms.

Parameter Settings: In **Setting 1**, we fix $d_1 = 40$ and $d_2 = 5$ to observe the regret when d_2 is small compared to d_1. In **Setting 2**, we reverse the situation and fix $d_1 = 5$ and $d_2 = 40$. For both **Setting 1** and **Setting 2** we fix $K = 25$ and $T = 80,000$. For **Setting 3**, we fix $d_1 = d_2 = 5$ and $T = 30,000$ but vary K over a set of points between 10 and 400. This helps us capture the dependence of regret on the number of arms K for the various algorithms.

Environments: We model the environment as a sequence of T contexts (arm features) and a set of reward parameters. For each of our parameter settings, we create 5 different environments. For each environment, we run 5 parallel trials. In two different trials of the same environment, the sequence of contexts and the reward parameters remain unchanged, but the random noise in reward generation is allowed to vary.

Stochastic Feature Generation: For each environment, the contexts i.e., the shared and arm-specific features of an arm in a particular round are generated by uniformly sampling d_1 and d_2 sized vectors from unit balls in d_1 and d_2 dimensions respectively.

Reward Simulation: Reward parameters for an environment are created by uniformly sampling the shared parameter θ^* from the d_1-dimensional ball of radius 1 (i.e., $S = 1$). Thereafter, K vectors are sampled uniformly from the d_2-dimensional ball of radius 1, which become the disjoint parameters $\{\beta_i^*\}_{i\in[K]}$. The stochastic reward generated for arm $i \in [K]$ with features (x_i, z_i) is

$$r_t = \langle x_i, \theta^* \rangle + \langle z_i, \beta_i^* \rangle + \eta \ ,$$

where noise η is sampled from the Gaussian distribution $\mathcal{N}(0, 0.01)$.

4.2 Real-World

We also perform experiments on the Yahoo! Front Page Dataset [25]. This dataset contains click logs from real time user interactions with the news articles displayed in the *Featured tab* of the *Today module* on Yahoo! Front Page during the first 10 days of May 2009. This dataset was also used in [16] for experiments with the hybrid model.

Dataset Description: At every round, a user arrives and is presented with $K = 20$ news articles. The user is described by 6 features and every article presented also has 6 features. The first feature is always 1 while the remaining 5 features correspond to the 5 membership features constructed via conjoint analysis with a bi-linear model as described in [7]. The dataset contains about 45 million such user-article interaction entries. Further, one of the articles out of these K articles is actually shown to the user, for which click (or no-click) is recorded.

Feature Construction: At a particular round, suppose the user features are denoted by $u \in \mathbb{R}^6$ and the arm (news article) features of arm $i \in [K]$ are denoted by $v_i \in \mathbb{R}^6$. Then, for arm i, the shared features x_i are given by vectorizing $uv_i^\intercal$, while the disjoint features z_i are set equal to v_i. Thus, $d_1 = 36$ and $d_2 = 6$.

Parameter Learning: We first learn a reward model by training a hybrid linear regression model on the first $1M$ data points (contained in May 1, 2009 dataset). For every user and the article that was actually shown to the user, we create the shared and the arm parameters x_i and z_i (respectively) of the arm (article). Note that the index of the arm shown in round n, i.e., i_n, is also an essential part of this dataset (as we need to train the arm parameters too). The target variable y_n is the click feedback, which is a boolean variable. Thus, for $N = 1M$ data points, we have $(x_{i_n,n}, z_{i_n,n}, y_n)_{n=1}^N$. Finally we obtain the parameters as:

$$\theta^*, \{\beta_i^*\}_{i\in[K]} = \underset{\theta, \{\beta_i\}_{i\in[K]}}{\operatorname{argmin}} \sum_{n=1}^{N} \left(\langle x_{i_n,n}, \theta \rangle + \langle z_{i_n,n}, \beta_{i_n} \rangle - y_n\right)^2$$

Reward Simulation: For simulating the bandit experiments, the stochastic reward for an arm i with features x_i, z_i is generated as

$$r_t = \langle x_i, \theta^* \rangle + \langle z_i, \beta_i^* \rangle + \eta \ ,$$

where noise η is sampled from the Gaussian distribution $\mathcal{N}(0, 0.0001)$. The reason for keeping the variance of the noise low is that it was observed that the mean rewards are themselves of the order 0.01, so high noise variance causes the algorithms to take longer to demonstrate sub-linear regret due to smaller signal-to-noise ratio in each arm pull's feedback.

5 Results

In all experiments, we compare `HyLinUCB`, `LinUCB`, `DisLinUCB` and `HyRan` [13].

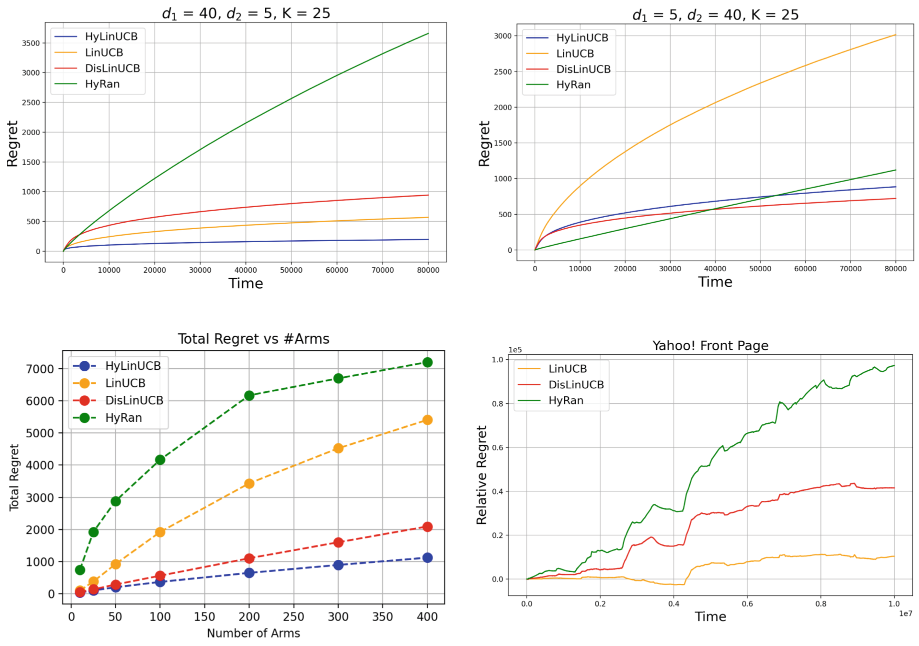


Fig. 1. Results of our experiments. Top-left: Regret vs # of Rounds (T) for **Setting 1**; Top-right: Regret vs # of Rounds (T) for **Setting 2**; Bottom-left: Regret versus # of Arms for **Setting 3**; Bottom-right: Relative regret with respect to `HyLinUCB` for Yahoo! Dataset.

5.1 Synthetic Experiments

Regret vs T: In Fig. 1, in the top row, we present results for **Setting 1** (left) and **Setting 2** (right). We plot the cumulative regret averaged over 5 parallel trials for each of the 5 different environments. In **Setting 1**, with $d_1 \gg d_2$, we observe that the performance of `HyLinUCB` is the best followed by `LinUCB`, thus validating the superiority of these algorithms with higher shared parameters, which resembles more like a fully shared setting. `DisLinUCB` also performs comparably but the regret is higher than `LinUCB`. `HyRan` performs the worst in **Setting 1** although some sub-linear nature can be observed. In **Setting 2**, with $d_2 \gg d_1$, `DisLinUCB` emerges as the best algorithm, while `HyLinUCB` is a very close second. `HyRan` exhibits a linear regret in this regime and eventually crosses beyond both `DisLinUCB` and `HyLinUCB`. Notably, `LinUCB` has the worst performance in this setting, although the regret looks sub-linear. This matches with the intuition that **Setting 2** is closer to the fully disjoint setting, hence `LinUCB` is not well-suited. However, the advantage of `HyLinUCB` is very clear from these experiments as `HyLinUCB` performs very well in both the settings, thus validating its suitability for the hybrid reward problem.

Regret vs K: In Fig. 1, the left plot in the bottom row shows the effect of K on the total regret. This plot corresponds to **Setting 3**, with $d_1 = d_2 = 5$. We again perform 5 parallel trials for (each of the) 5 different environments and then calculate the average of the total regret over T rounds in all these $5 \times 5 = 25$ simulations. The X-axis in the plot represents the number of arms K while the Y-axis denotes the (average) total regret. From the plot, we can observe that `HyLinUCB` has the smallest regret over all values of K while `DisLinUCB` is the second best, followed by `LinUCB` and then `HyRan`. `HyLinUCB` displays a very slow growth with K, leading us to believe that its regret guarantee in Sect. 3 is not tight and can possibly be improved significantly.

5.2 Real-World Experiment

The results of the semi-synthetic experiment for $10M$ rounds (starting from May 2, 2009 dataset and moving to subsequent days' dataset till $10M$ rounds are complete) is shown in Fig. 1 in the bottom right. We plot the regret of other algorithms relative to `HyLinUCB` (subtract the cumulative regret of `HyLinUCB` from the cumulative regret of the algorithm), which has the smallest regret. `LinUCB` comes close to `HyLinUCB` whereas regret of `DisLinUCB` and `HyRan` are much larger. In this experiment, we do not have any control over the context features hence it is possible that the diversity assumption is not satisfied. However, we observe that `HyLinUCB` still performs much better than the other algorithms, demonstrating strong evidence of good performance in hybrid models.

6 Conclusion and Future Work

In this work, we revisited the problem of linear contextual bandits with hybrid rewards which is a useful setting in many applications such as news recommendation. The problem was originally proposed in [16] albeit without any theoretical guarantees. Reducing this problem to the shared and disjoint settings, we derived regret guarantee for the `LinUCB` and `DisLinUCB` (Algorithm 1 in [16]) algorithms, improving over their well known guarantees under a popular diversity assumption. Finally, we propose a new algorithm `HyLinUCB` that employs a tighter exploration coefficient leveraging the sparsity of the problem. We also derive regret guarantee for this algorithm that has optimal dependence on T. We perform extensive empirical evaluation of the three algorithms in various synthetic scenarios (choices of d_1, d_2 and K) as well as on real-world datasets (Yahoo! Front Page Dataset [25]). We empirically compare these algorithms with the state-of-the-art `HyRan` [13] algorithm and demonstrate that `HyLinUCB` outperforms the other algorithms in almost all cases and comes very close to the best in the remaining.

Since `HyLinUCB` performs much better than `LinUCB` empirically, an interesting future direction will be to derive tighter regret bounds that will provably demonstrate the efficacy of `HyLinUCB`. Another future direction is to derive the regret guarantees under a different diversity assumption and understanding the trade-offs therein.

References

1. Abbasi-Yadkori, Y., Pál, D., Szepesvári, C.: Improved algorithms for linear stochastic bandits. In: Advances in Neural Information Processing Systems, vol. 24 (2011)
2. Abeille, M., Lazaric, A.: Linear thompson sampling revisited. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, vol. 54, pp. 176–184. PMLR (2017)
3. Agrawal, S., Goyal, N.: Thompson sampling for contextual bandits with linear payoffs. In: Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28, ICML 2013, pp. III-1220–III-1228. JMLR.org (2013)
4. Auer, P.: Using confidence bounds for exploitation-exploration trade-offs. J. Mach. Learn. Res. **3**, 397-422 (2003). ISSN 1532-4435
5. Chatterji, N., Muthukumar, V., Bartlett, P.: OSOM: a simultaneously optimal algorithm for multi-armed and linear contextual bandits. In: Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, vol. 108, pp. 1844–1854, PMLR (2020). https://proceedings.mlr.press/v108/chatterji20b.html
6. Chu, W., Li, L., Reyzin, L., Schapire, R.: Contextual bandits with linear payoff functions. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 208–214. JMLR Workshop and Conference Proceedings (2011)
7. Chu, W., et al.: A case study of behavior-driven conjoint analysis on yahoo!: front page today module. In: Knowledge Discovery and Data Mining (2009). https://api.semanticscholar.org/CorpusID:18101293
8. Dani, V., Hayes, T.P., Kakade, S.M.: Stochastic linear optimization under bandit feedback. In: Annual Conference Computational Learning Theory (2008). https://api.semanticscholar.org/CorpusID:9134969
9. Gentile, C., Li, S., Kar, P., Karatzoglou, A., Zappella, G., Etrue, E.: On context-dependent clustering of bandits. In: International Conference on Machine Learning, pp. 1253–1262. PMLR (2017)

10. Ghosh, A., Sankararaman, A., Kannan, R.: Problem-complexity adaptive model selection for stochastic linear bandits. In: International Conference on Artificial Intelligence and Statistics, pp. 1396–1404. PMLR (2021)
11. Hill, D.N., Nassif, H., Liu, Y., Iyer, A., Vishwanathan, S.: An efficient bandit algorithm for realtime multivariate optimization. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017, pp. 1813–1821. Association for Computing Machinery, New York (2017). ISBN 9781450348874. https://doi.org/10.1145/3097983.3098184
12. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (2012)
13. Kim, W., Paik, M.C., Oh, M.H.: Squeeze all: novel estimator and self-normalized bound for linear contextual bandits. In: International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, vol. 206, pp. 3098–3124. PMLR (2023). https://proceedings.mlr.press/v206/kim23d.html
14. Lai, T., Robbins, H.: Asymptotically efficient adaptive allocation rules. Adv. Appl. Math. **6**(1), 4–22 (1985). ISSN 0196-8858
15. Lattimore, T., Szepesvári, C.: Bandit Algorithms. Cambridge University Press, Cambridge (2020)
16. Li, L., Chu, W., Langford, J., Schapire, R.E.: A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the 19th International Conference on World Wide Web, pp. 661–670 (2010)
17. Li, L., Lu, Y., Zhou, D.: Provably optimal algorithms for generalized linear contextual bandits. In: Proceedings of the 34th International Conference on Machine Learning - Volume 70, ICML 2017, pp. 2071–2080. JMLR.org (2017)
18. Li, Y., Wang, Y., Zhou, Y.: Nearly minimax-optimal regret for linearly parameterized bandits. In: Conference on Learning Theory, pp. 2173–2174. PMLR (2019)
19. Papini, M., Tirinzoni, A., Restelli, M., Lazaric, A., Pirotta, M.: Leveraging good representations in linear contextual bandits. In: Proceedings of the 38th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol. 139, pp. 8371–8380. PMLR (2021). https://proceedings.mlr.press/v139/papini21a.html
20. Rusmevichientong, P., Tsitsiklis, J.N.: Linearly parameterized bandits. Math. Oper. Res. **35**, 395–411 (2008). https://api.semanticscholar.org/CorpusID:3204347
21. Stanimirović, P.S., Katsikis, V.N., Kolundzija, D.: Inversion and pseudoinversion of block arrowhead matrices. Appl. Math. Comput. **341**, 379–401 (2019). https://api.semanticscholar.org/CorpusID:52988431
22. Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika **25**(3-4), 285–294 (1933). https://doi.org/10.1093/biomet/25.3-4.285. ISSN 0006-3444
23. Tropp, J.A.: User-friendly tail bounds for sums of random matrices. Found. Comput. Math. **12**, 389–434 (2012)
24. Tropp, J.A.: An introduction to matrix concentration inequalities. Found. Trends Mach. Learn. **8**(1-2), 1–230 (2015). https://doi.org/10.1561/2200000048. ISSN 1935-8237
25. Yahoo! Webscope: Yahoo! Front Page Today Module User Click Log Dataset, version 1.0 (2010). http://research.yahoo.com/Academic_Relationsa. Accessed 22 Jan 2024

Author Index