

Hocine Cherifi · Murat Donduran ·
Luis M. Rocha · Chantal Cherifi ·
Onur Varol *Editors*

Complex Networks & Their Applications XIII

Proceedings of The Thirteenth
International Conference on Complex
Networks and Their Applications:
COMPLEX NETWORKS 2024 - Volume 3



Series Editor

Janusz Kacprzyk, *Polish Academy of Sciences, Warsaw, Poland*

Editorial Board Members

Marco Dorigo , *Université Libre de Bruxelles, Bruxelles, Belgium*

Andries Engelbrecht, *University of Stellenbosch, Stellenbosch, South Africa*

Vladik Kreinovich, *University of Texas at El Paso, El Paso, TX, USA*

Francesco Carlo Morabito, *Mediterranea University of Reggio Calabria, Reggio Calabria, Italy*

Roman Slowinski, *Poznan University of Technology, Poznan, Poland*

Yingxu Wang, *Schulich School of Engineering, Calgary, AB, Canada*

Yaochu Jin, *Westlake University, Hangzhou, China*

The series “Studies in Computational Intelligence” (SCI) publishes new developments and advances in the various areas of computational intelligence—quickly and with a high quality. The intent is to cover the theory, applications, and design methods of computational intelligence, as embedded in the fields of engineering, computer science, physics and life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, soft computing, fuzzy systems, and hybrid intelligent systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

Indexed by SCOPUS, DBLP, WTI AG (Switzerland), zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

Hocine Cherifi · Murat Donduran ·
Luis M. Rocha · Chantal Cherifi · Onur Varol
Editors

Complex Networks & Their Applications XIII

Proceedings of The Thirteenth International
Conference on Complex Networks and Their
Applications: COMPLEX NETWORKS 2024 -
Volume 3

Editors

Hocine Cherifi
ICB
University of Burgundy
Dijon, France

Luis M. Rocha
Thomas J. Watson College of Engineering
and Applied Science
Binghamton University
Binghamton, USA

Onur Varol
Faculty of Engineering and Natural Sciences
Office
Sabancı University
Istanbul, Türkiye

Murat Donduran
Department of Economics
Yildiz Technical University
Istanbul, Türkiye

Chantal Cherif
IUT Lumière
Université Lyon 2
Bron, France

ISSN 1860-949X ISSN 1860-9503 (electronic)

Studies in Computational Intelligence

ISBN 978-3-031-82434-0 ISBN 978-3-031-82435-7 (eBook)

<https://doi.org/10.1007/978-3-031-82435-7>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2025

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

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

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

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

If disposing of this product, please recycle the paper.

Preface

Dear Colleagues, Participants, and Readers,

We are thrilled to present the proceedings of the 13th International Conference on Complex Networks and Their Applications, held in the vibrant city of Istanbul, Türkiye, from December 10 to 12, 2024. This year's conference marks yet another milestone in the growth of our interdisciplinary community, reflecting the ever-expanding significance of complex network research. Understanding the structures and dynamics that govern complex systems has never been more critical as our world becomes more interconnected.

The Complex Networks Conference has evolved over the past thirteen years into a crucial platform where scholars, researchers, and professionals from various disciplines converge. Through these proceedings, we showcase groundbreaking work that explores the diverse applications of network science, from biological systems to social networks, technological infrastructures, and economic systems. Each edition brings new ideas and breakthroughs, and this year is no exception, with significant contributions from a global cohort of experts.

We are particularly honored to feature keynote lectures from internationally renowned scholars, whose talks highlight some of the most pressing issues and exciting developments in the field:

- Federico Battiston (Central European University, Austria) presents “Higher-order network science”.
- Tina Eliassi-Rad (Northeastern University, USA) explored the “Stability of Graph Machine Learning Models”.
- Frank Emmert-Streib (Tampere University, Finland) discussed “From Digital Twins to Complexity Data Science”.
- Filippo Menczer (Indiana University, USA) talked about “AI and Social Media Manipulation: The Good, the Bad, and the Ugly”.
- Luciano Pietronero (University of Rome “Sapienza,” Italy) provided deep insights into “Economic Fitness: Concepts, Methods, and Applications”.

These keynote talks reflect the richness and diversity of this year's conference, touching on cutting-edge issues in both theory and application. In addition, we are excited to have offered two pre-conference tutorials on December 9, 2024:

- Alessandro Galeazzi (University of Padova, Italy) led a tutorial on “Unveiling Echo Chambers and Users' Opinions on Online Social Media”.
- Clara Stegehuis (University of Twente, The Netherlands) presented a session on “Mathematical Analysis and Applications of Network Motifs”.

The proceedings of COMPLEX NETWORKS 2024, published by Springer, consist of four volumes that capture a wide range of topics, including machine learning networks, network embedding, link analysis ranking, social networks, biological networks, higher-order interactions, and community structure, among others. Each paper included here

has undergone a rigorous peer-review process, ensuring the high quality and scientific relevance that our community expects.

We are also pleased to announce that extended versions of selected papers will be invited for publication in prestigious journals, including PLOS Complex Systems, Applied Network Science, Social Network Analysis and Mining, and Advances in Complex Systems, furthering the reach and impact of the research presented at this conference.

We would like to extend our sincere gratitude to the authors, presenters, and reviewers for their contributions, which are at the core of this conference's success. Our most profound appreciation goes to the organizing committees and volunteers, whose dedication has made this year's edition successful.

We invite you to explore the rich content within these proceedings and hope that they will inspire future research, foster innovation, and stimulate collaborations within the complex networks' community. As you explore these contributions, we are confident you will share the enthusiasm and excitement we felt in organizing COMPLEX NETWORKS 2024.

Murat Donduran
Hocine Cherifi
Luis M. Rocha
Chantal Cherifi
Onur Varol

Organization and Committees

General Chairs

Hocine Cherifi

University of Burgundy, France

Luis M. Rocha

Binghamton University, USA

Advisory Board

Jon Crowcroft

University of Cambridge, UK

Raissa D'Souza

Univ. of California, Davis, USA

Eugene Stanley

Boston University, USA

Ben Y. Zhao

University of Chicago, USA

Program Chairs

Chantal Cherifi

University of Lyon, France

Murat Donduran

Yildiz Technical University, Turkey

Lightning Chairs

Konstantin Avrachenkov

Inria Université Côte d'Azur, France

Mathieu Desroches

Inria Université Côte d'Azur, France

Huijuan Wang

TU Delft, Netherlands

Poster Chairs

Christophe Crespelle

Université Côte d'Azur, France

Manuel Marques Pita

Universidade Lusófona, Portugal

Laura Ricci

University of Pisa, Italy

Special Issues Chair

Sabrina Gaito University of Milan, Italy

Publicity Chairs

Fabian Braesemann University of Oxford, UK
Zachary Neal Michigan State University, USA
Xiangjie Kong Dalian University of Technology, China

Tutorial Chairs

Luca Maria Aiello Nokia-Bell Labs, UK
Leto Peel Maastricht University, Netherlands

Social Media Chair

Brennan Klein Northeastern University, USA

Sponsor Chairs

Sustainability Chair

Madeleine Aurelle City School International De Ferney-Voltaire,
France

Local Committee Chair

Charlie Joyez Université Côte d'Azur, France

Publication Chair

Matteo Zignani

University of Milan, Italy

Submission Chair

Cheick Ba

Queen Mary University of London, UK

Web Chairs

Stephany Rajeh

Sorbonne University, France

Alessia Galdeman

University of Milan, Italy

Program Committee

Jacobo Aguirre

Centro de Astrobiología (CAB), Spain

Luca Maria Aiello

ITU Copenhagen, Denmark

Esra Akbas

Georgia State University, USA

Sinan G. Aksoy

Pacific Northwest National Laboratory, USA

Mehmet Aktas

Georgia State University, USA

Tatsuya Akutsu

Kyoto University, Japan

Reka Albert

Pennsylvania State University, USA

Alberto Aleta

University of Zaragoza, Spain

Claudio Altafini

Linkoping University, Sweden

Viviana Amati

University of Milano-Bicocca, Unknown

Frederic Amblard

Université Toulouse 1 Capitole, IRIT, France

Enrico Amico

EPFL, Switzerland

Yuri Antonacci

University of Palermo, Italy

Alberto Antonioni

Carlos III University of Madrid, Spain

Nino Antulov-Fantulin

ETH Zurich, Switzerland

Mehrnaz Anvari

Fraunhofer SCAI, Germany

David Aparicio

Zendesk, Portugal

Nuno Araujo

Univ. de Lisboa, Portugal

Panos Argyrakis

Aristotle University of Thessaloniki, Greece

Oriol Artíme

University of Barcelona, Spain

Malbor Asllani

Florida State University, USA

Tomaso Aste

University College London, UK

Martin Atzmueller

Osnabrück University & DFKI, Germany

Konstantin Avrachenkov

Inria Sophia-Antipolis, France

Giacomo Baggio	University of Padova, Italy
Franco Bagnoli	Università di Firenze, Italy
James Bagrow	University of Vermont, USA
Yiguang Bai	Xidian University, China
Sven Banisch	Karlsruhe Institute of Technology, Germany
Annalisa Barla	Università degli Studi di Genova, Italy
Nikita Basov	The University of Manchester, UK
Anais Baudot	CNRS, AMU, France
Gareth J. Baxter	University of Aveiro, Portugal
Loredana Bellantuono	University of Bari Aldo Moro, Italy
Andras Benczur	SZTAKI, Hungary
Rosa M. Benito	Universidad Politécnica de Madrid, Spain
Ginestra Bianconi	Queen Mary University of London, UK
Ofer Biham	The Hebrew University, Israel
Romain Billot	IMT Atlantique, France
Livio Bioglio	University of Turin, Italy
Hanjo D. Boekhout	Leiden University, Netherlands
Anthony Bonato	Toronto Metropolitan University, Canada
Anton Borg	Blekinge Institute of Technology, Sweden
Cecile Bothorel	IMT Atlantique, France
Federico Botta	University of Exeter, UK
Romain Bourqui	University of Bordeaux, France
Alexandre Bovet	University of Zurich, Switzerland
Dan Braha	New England Complex Systems Institute, USA
Ulrik Brandes	ETH Zürich, Switzerland
Rion Brattig Correia	Instituto Gulbenkian de Ciência, Portugal
Chico Camargo	University of Exeter, UK
Gian Maria Campedelli	Fondazione Bruno Kessler, Italy
M. Abdullah Canbaz	University at Albany SUNY, USA
Vincenza Carchiolo	DIEEI, Italy
Dino Carpentras	ETH Zürich, Switzerland
Giona Casiraghi	ETH Zürich, Switzerland
Douglas Castilho	Federal Inst. of South of Minas Gerais, Brazil
Costanza Catalano	University of Florence, Italy
Lucia Cavallaro	Free University of Bozen/Bolzano, Italy
Remy Cazabet	University of Lyon, France
Jianrui Chen	Shaanxi Normal University, China
Po-An Chen	National Yang Ming Chiao Tung Univ., Taiwan
Xihui Chen	University of Luxembourg, Luxembourg
Sang Chin	Boston University, USA
Daniela Cialfi	Institute for Complex Systems, Italy
Giulio Cimini	University of Rome Tor Vergata, Italy

Matteo Cinelli	Sapienza University of Rome, Italy
Salvatore Citraro	University of Pisa, Italy
Jonathan Clarke	Imperial College London, UK
Richard Clegg	QMUL, UK
Reuven Cohen	Bar-Ilan University, Israel
Jean-Paul Comet	Université Côte d'Azur, France
Marco Coraggio	Scuola Superiore Meridionale, Italy
Michele Coscia	ITU Copenhagen, Denmark
Christophe Crespellé	Université Côte d'Azur, France
Regino H. Criado Herrero	Universidad Rey Juan Carlos, Spain
Marcelo V. Cunha	Instituto Federal da Bahia, Brazil
David Soriano-Paños	Instituto Gulbenkian de Ciência, Portugal
Joern Davidsen	University of Calgary, Canada
Toby Davies	University of Leeds, UK
Caterina De Bacco	Max Planck Inst. for Intelligent Systems, Germany
Pietro De Lellis	University of Naples Federico II, Italy
Pasquale De Meo	University of Messina, Italy
Domenico De Stefano	University of Trieste, Italy
Fabrizio De Vico Fallani	Inria-ICM, France
Charo I. del Genio	Coventry University, UK
Robin Delabays	HES-SO, Switzerland
Yong Deng	Univ. of Electronic Science and Tech., China
Mathieu Desroches	Inria Centre at Université Côte d'Azur, France
Carl P. Dettmann	University of Bristol, UK
Zengru DI	Beijing Normal University, China
Riccardo Di Clemente	Northeastern University London, UK
Branco Di Fátima	University of Beira Interior (UBI), Portugal
Alessandro Di Stefano	Teesside University, UK
Ming Dong	Central China Normal University, China
Constantine Dovrolis	Georgia Tech, USA
Maximilien Dreveton	EPFL, Switzerland
Ahlem Drif	University of Setif, Algeria
Johan L. Dubbeldam	Delft University of Technology, Netherlands
Jordi Duch	Universitat Rovira i Virgili, Spain
Cesar Ducruet	CNRS, France
Mohammed El Hassouni	Mohammed V University in Rabat, Morocco
Frank Emmert-Streib	Tampere University, Finland
Gunes Ercal	Southern Illinois University Edwardsville, USA
Alejandro Espinosa-Rada	ETH Zürich, Switzerland
Alexandre Evsukoff	Universidade Federal do Rio de Janeiro, Brazil
Mauro Faccin	University of Bologna, Italy

Max Falkenberg	City University, UK
Guilherme Ferraz de Arruda	CENTAI Institute, Italy
Andrea Flori	Politecnico di Milano, Italy
Manuel Foerster	Bielefeld University, Germany
Emma Fraxanet Morales	Pompeu Fabra University, Spain
Angelo Furno	LICIT-ECO7, France
Sergio Gómez	Universitat Rovira i Virgili, Spain
Sabrina Gaito	Università degli Studi di Milano, Italy
José Manuel Galán	Universidad de Burgos, Spain
Alessandro Galeazzi	Ca' Foscari university of Venice, Italy
Lazaros K. Gallos	Rutgers University, USA
Joao Gama	INESC TEC - LIAAD, Portugal
Jianxi Gao	Rensselaer Polytechnic Institute, USA
David Garcia	University of Konstanz, Germany
Floriana Gargiulo	CNRS, France
Michael T. Gastner	Singapore Institute of Technology, Singapore
Alexander Gates	University of Virginia, USA
Alexandra M. Gerbasi	Exeter Business School, UK
Fakhteh Ghanbarnejad	Potsdam Inst. for Climate Impact Res., Germany
Cheol-Min Ghim	Ulsan National Inst. of Science and Tech., South Korea
Tommaso Gili	IMT School for Advanced Studies Lucca, Italy
Silvia Giordano	Univ. of Applied Sciences of Southern Switzerland, Switzerland
Rosalba Giugno	University of Verona, Italy
Kimberly Glass	Brigham and Women's Hospital, USA
David Gleich	Purdue University, USA
Antonia Godoy Lorite	UCL, UK
Kwang-Il Goh	Korea University, South Korea
Carlos Gracia	University of Zaragoza, Spain
Oscar M. Granados	Universidad Jorge Tadeo Lozano, Colombia
Michel Grossetti	CNRS, France
Guillaume Guerard	ESILV, France
Jean-Loup Guillaume	Université de la Rochelle-France, France
Furkan Gursoy	Bogazici University, Turkey
Philipp Hövel	Saarland University, Germany
Meesoon Ha	Chosun University, South Korea
Bianca H. Habermann	AMU, CNRS, IBDM UMR 7288, France
Chris Hankin	Imperial College London, UK
Yukio Hayashi	JAIST, Japan
Marina Hennig	Johannes Gutenberg University of Mainz, Germany

Takayuki Hiraoka	Aalto University, Finland
Marion Hoffman	Institute for Advanced Study in Toulouse, France
Bernie Hogan	University of Oxford, UK
Seok-Hee Hong	University of Sydney, Australia
Yujie Hu	University of Florida, USA
Flavio Iannelli	UZH, Switzerland
Yuichi Ikeda	Kyoto University, Japan
Roberto Interdonato	CIRAD, France
Antonio Iovanella	Univ. degli Studi Internazionali di Roma, Italy
Arkadiusz Jędrzejewski	CY Cergy Paris Université, France
Tao Jia	Southwest University, China
Jiaojiao Jiang	UNSW Sydney, Australia
Di Jin	University of Michigan, USA
Ivan Jokifá	Technology University of Delft, Netherlands
Charlie Joyez	GREDEG, Université Côte d'Azur, France
Bogumil Kamiński	SGH Warsaw School of Economics, Poland
Marton Karsai	Central European University, Austria
Eytan Katzav	Hebrew University of Jerusalem, Israel
Mehmet Kaya	Firat University, Turkey
Domokos Kelen	SZTAKI, Hungary
Mohammad Khansari	Sharif University of Technology, Iran
Jinseok Kim	University of Michigan, USA
Pan-Jun Kim	Hong Kong Baptist University, Hong Kong
Maksim Kitsak	TU Delft, Netherlands
Mikko Kivelä	Aalto University, Finland
Brennan Klein	Northeastern University, UK
Konstantin Klemm	IFISC (CSIC-UIB), Spain
Xiangjie Kong	Zhejiang University of Technology, China
Onerva Korhonen	University of Eastern Finland, Finland
Miklós Krész	InnoRenew CoE, Slovenia
Prosenjit Kundu	DA-IICT, Gandhinagar, Gujarat, India
Haewoon Kwak	Indiana University Bloomington, USA
Richard La	University of Maryland, USA
José Lages	Université de Franche-Comté, France
Renaud Lambiotte	University of Oxford, UK
Aniello Lampo	UC3M, Spain
Jennifer Larson	Vanderbilt University, USA
Paul J. Laurienti	Wake Forest, USA
Anna T. Lawniczak	University of Guelph, Canada, Canada
Deok-Sun Lee	KIAS, South Korea
Harlin Lee	Univ. of North Carolina at Chapel Hill, USA
Juergen Lerner	University of Konstanz, Germany

Lasse Leskelä	Aalto University, Finland
Petri Leskinen	Aalto University/SeCo, Finland
Inmaculada Leyva	Universidad Rey Juan Carlos, Spain
Cong Li	Fudan University, China
Longjie Li	Lanzhou University, China
Ruiqi LI	Beijing Univ. of Chemical Technology, China
Xiangtao Li	Jilin University, China
Hao Liao	Shenzhen University, China
Fabrizio Lillo	Università di Bologna, Italy
Giacomo Livan	University of Pavia, Italy
Giosue' Lo Bosco	Università di Palermo, Italy
hao long	Jiangxi Normal University, China
Juan Carlos Losada	Universidad Politécnica de Madrid, Spain
Laura Lotero	Universidad Nacional de Colombia, Colombia
Yang Lou	National Yang Ming Chiao Tung Univ., Taiwan
Meilian Lu	Beijing Univ. of Posts and Telecom., China
Maxime Lucas	CENTAI, Italy
Lorenzo Lucchini	Bocconi University, Italy
Hanbaek Lyu	UW-Madison, USA
Vince Lyzinski	University of Maryland, College Park, USA
Morten Mørup	Technical University of Denmark, Denmark
Leonardo Maccari	Ca'Foscari University of Venice, Italy
Matteo Magnani	Uppsala University, Sweden
Maria Malek	CY Cergy Paris University, France
Giuseppe Mangioni	University of Catania, Italy
Andrea Mannocci	CNR - ISTI, Italy
Rosario N. Mantegna	University of Palermo, Italy
Manuel Sebastian Mariani	University of Zurich, Switzerland
Radek Marik	CTU in Prague, Czech Republic
Daniele Marinazzo	Ghent University, Belgium
Andrea Marino	University of Florence, Italy
Malvina Marku	INSERM, CRCT, France
Antonio G. Marques	King Juan Carlos University, Spain
Christoph Martin	Hamburg University of Applied Sciences, Germany
Samuel Martin-Gutierrez	Complexity Science Hub Vienna, Austria
Cristina Masoller	Universitat Politècnica de Catalunya, Spain
Rossana Mastrandrea	IMT School for Advanced Studies, Italy
John D. Matta	Southern Illinois Univ. Edwardsville, USA
Carolina Mattsson	CENTAI Institute, Italy
Fintan McGee	Luxembourg IST, Luxembourg
Matus Medo	University of Bern, Switzerland

Ronaldo Menezes	University of Exeter, UK
Humphrey Mensah	Epsilon Data Management, LLC, USA
Anke Meyer-Baese	Florida state university, USA
Salvatore Micciche	UNIPA DiFC, Italy
Letizia Milli	University of Pisa, Italy
Marija Mitrovic	Institute of Physics Belgrade, Serbia
Andrzej Mizera	University of Warsaw, Poland
Chiara Mocenni	University of Siena, Italy
Roland Molontay	Budapest UTE, Hungary
Sifat Afroz Moon	University of Virginia, USA
Alfredo Morales	MIT, USA
Andres Moreira	UTFSM, Chile
Greg Morrison	University of Houston, USA
Igor Mozetic	Jozef Stefan Institute, Slovenia
Sarah Muldoon	State University of New York, Buffalo, USA
Tsuyoshi Murata	Tokyo Institute of Technology, Japan
Jose Nacher	Toho University, Japan
Nishit Narang	NIT Delhi, India
Filipi Nascimento Silva	Indiana University, USA
Muaz A. Niazi	National Univ. of Science & Technology, Pakistan
Peter Niemeyer	Leuphana University Lüneburg, Germany
Jordi Nin	ESADE, Universitat Ramon Llull, Spain
Rogier Noldus	Ericsson, Netherlands
Masaki Ogura	Osaka University, Japan
Andrea Omicini	Università di Bologna, Italy
Gergely Palla	Eötvös University, Hungary
Fragkiskos Papadopoulos	Cyprus University of Technology, Cyprus
Symeon Papadopoulos	Centre for Research & Technology, Greece
Alice Patania	University of Vermont, USA
Leto Peel	Maastricht University, Netherlands
Hernane B. B. Pereira	Senai Cimatec, Brazil
Josep Perelló	Universitat de Barcelona, Spain
Anthony Perez	Université d'Orléans, France
Juergen Pfeffer	Technical University of Munich, Germany
Carlo Piccardi	Politecnico di Milano, Italy
Pietro Hiram Guzzi	Univ. Magna Gracia of Catanzaro, Italy
Yoann Pigné	Université Le Havre Normandie, France
Bruno Pinaud	University of Bordeaux, France
Flavio L. Pinheiro	Universidade Nova de Lisboa, Portugal
Manuel Pita	Universidade Lusófona, Portugal
Clara Pizzuti	CNR-ICAR, Italy

Jan Platos	VSB - Technical University of Ostrava, Czech Republic
Pawel Pralat	Toronto Metropolitan University, Canada
Rafael Prieto-Curiel	Complexity Science Hub, Austria
Daniele Proverbio	University of Trento, Italy
Giulia Pullano	Georgetown University, USA
Rami Puzis	Ben-Gurion University of the Negev, Israel
Christian Quadri	Università degli Studi di Milano, Italy
Hamid R. Rabiee	Sharif University of Technology, Iran
Filippo Radicchi	Indiana University, USA
Giancarlo Ragozini	University of Naples Federico II, Italy
Juste Raimbault	IGN-ENSG, France
Sarah Rajtmajer	Penn State, USA
Gesine D. Reinert	University of Oxford, UK
Elisabeth Remy	Institut de Mathématiques de Marseille, France
Xiao-Long Ren	Univ. of Electronic Science and Tech., China
Laura Ricci	University of Pisa, Italy
Albano Rikani	INSERM, France
Luis M. Rocha	Binghamton University, USA
Luis E C Rocha	Ghent University, Belgium
Fernando E. Rosas	Imperial College London, UK
Giulio Rossetti	CNR-ISTI, Italy
Camille Roth	CNRS/CMB/EHESS, France
Celine Rozenblat	UNIL, Switzerland
Giancarlo Ruffo	Univ. degli Studi del Piemonte Orientale, Italy
Arnaud Sallaberry	University of Montpellier, France
Hillel Sanhedrai	Northeastern University, USA
Iraj Saniee	Bell Labs, Nokia, USA
Antonio Scala	CNR Institute for Complex Systems, Italy
Michael T. Schaub	RWTH Aachen University, Germany
Irene Sendiña-Nadal	Universidad Rey Juan Carlos, Spain
Mattia Sensi	Politecnico di Torino, Italy
Ke-ke Shang	Nanjing University, China
Julian Sienkiewicz	Warsaw University of Technology, Poland
Per Sebastian Skardal	Trinity College, Ireland
Fiona Skerman	Uppsala University, Sweden
Oskar Skibski	University of Warsaw, Poland
Keith M. Smith	University of Strathclyde, UK
Igor Smolyarenko	Brunel University, UK
Zbigniew Smoreda	Orange Innovation, France
Annalisa Socievole	ICAR-CNR, Italy
Igor M. Sokolov	Humboldt University Berlin, Germany

Albert Solé-Ribalta	Universitat Oberta de Catalunya, Spain
Sara Sottile	University of Trento, Italy
Sucheta Soundarajan	Syracuse University, USA
Jaya Sreevalsan-Nair	IIT Bangalore, India
Christoph Stadtfeld	ETH Zürich, Switzerland
Clara Stegehuis	University of Twente, Netherlands
Lovro Šubelj	University of Ljubljana, Slovenia
Xiaoqian Sun	Beihang University, China
Michael Szell	IT University of Copenhagen, Denmark
Boleslaw Szymanski	Rensselaer Polytechnic Institute, USA
Andrea Tagarelli	University of Calabria, Italy
Kazuhiro Takemoto	Kyushu Institute of Technology, Japan
Frank W. Takes	Leiden University, Netherlands
Fabien Tarissan	CNRS & ENS Paris Saclay, France
Laura Temime	Cnam, France
François Théberge	TIMC, France
Guy Theraulaz	Université Paul Sabatier and CNRS, France
I-Hsien Ting	National University of Kaohsiung, Taiwan
Michele Tizzani	ISI Foundation, Italy
Michele Tizzoni	University of Trento, Italy
Olivier Togni	University of Burgundy, France
Leo Torres	Northeastern University, USA
Sho Tsugawa	University of Tsukuba, Japan
Francesco Tudisco	The University of Edinburgh, UK
Melvyn S. Tyloo	Los Alamos National Lab, USA
Stephen M. Uzzo	National Museum of Mathematics, USA
Lucas D. Valdez	IFIMAR-UNMdP, Argentina
Pim Van der Hoorn	Eindhoven University of Technology, Netherlands
Piet Van Mieghem	Delft University of Technology, Netherlands
Fabio Vanni	University of Insubria, Italy
Christian L. Vestergaard	Institut Pasteur, France
Tiphaine Viard	Télécom Paris, France
Julian Vicens	Eurecat, Spain
Blai Vidiella	CSIC, Spain
Pablo Villegas	Enrico Fermi Research Center (CREF), Italy
Maria Prosperina Vitale	University of Salerno, Italy
Pierpaolo Vivo	King's College London, UK
Johannes Wachs	Corvinus University of Budapest, Hungary
Huijuan Wang	Delft University of Technology, Netherlands
Lei Wang	Beihang University, China
Guanghui Wen	Southeast University, Nanjing, China
Mateusz Wilinski	Los Alamos National Laboratory, USA

Dirk Witthaut	Forschungszentrum Jülich, Germany
Bin Wu	Beijing Univ. of Posts and Telecom., China
Mincheng Wu	Zhejiang University of Technology, China
Tao Wu	Chongqing Univ. of Posts and Telecom., China
Haoxiang Xia	Dalian University of Technology, China
Gaoxi Xiao	Nanyang Technological University, Singapore
Nenggang Xie	Anhui University of Technology, China
Takahiro Yabe	MIT, USA
Kaicheng Yang	Northeastern University, USA
Yian Yin	Cornell University, USA
Jean-Gabriel Young	University of Vermont, USA
Irfan Yousuf	Univ. of Engineering and Technology, Pakistan
Yongguang Yu	Beijing Jiaotong University, China
Paolo Zeppini	University Cote d'Azur, France
Shi Zhou	University College London (UCL), UK
Wei-Xing Zhou	East China Univ. of Science and Techno., China
Eugenio Zimeo	University of Sannio, Italy
Lorenzo Zino	Politecnico di Torino, Italy
Michał R. Zochowski	University of Michigan, USA
Claudia Zucca	Tilburg University, Netherlands

Contents

Anomaly Detection

DCOR: Anomaly Detection in Attributed Networks via Dual Contrastive Learning Reconstruction	3
<i>Hossein Rafiee Zade, Hadi Zare, Mohsen Ghassemi Parsa, Hadi Davardoust, and Meshkat Shariat Bagheri</i>	
Enhancing Recommender Systems with Anomaly Detection: A Graph Neural Network Approach	16
<i>Bahareh Rahmatikargar, Pooya Moradian Zadeh, and Ziad Kobti</i>	
FlowSeries: Anomaly Detection in Financial Transaction Flows	29
<i>Arthur Capozzi, Salvatore Vilella, Dario Moncalvo, Marco Fornasiero, Valeria Ricci, Silvia Ronchiadin, and Giancarlo Ruffo</i>	

Community Structure

Group Fairness Metrics for Community Detection Methods in Social Networks	43
<i>Elze de Vink and Akrati Saxena</i>	
FastEnsemble: A New Scalable Ensemble Clustering Method	57
<i>Yasamin Tabatabaee, Eleanor Wedell, Minhyuk Park, and Tandy Warnow</i>	
Examining Different Research Communities: Authorship Network	71
<i>Shrabani Ghosh</i>	
Constructing Telegram Channels Digital Profiles	83
<i>V. A. Popov and A. A. Chepovskiy</i>	
Fair-mod: Fair Modular Community Detection	91
<i>Georgios Panayiotou and Matteo Magnani</i>	
Improved Community Detection Using Stochastic Block Models	103
<i>Minhyuk Park, Daniel Wang Feng, Siya Digra, The-Anh Vu-Le, George Chacko, and Tandy Warnow</i>	
A Knowledge Graph for Monitoring and Controlling Public Policies	115
<i>Álvaro Carvalho and Raimir Holanda Filho</i>	

High-Performance Implementation of Louvain Algorithm with Representational Optimizations	127
<i>Subhajit Sahu, Kishore Kothapalli, and Dip Sankar Banerjee</i>	
Branching Out: How Academic Genealogy Networks Shape Academic Groups	140
<i>Gustavo Moraes, Angelo Brayner, and Ronaldo Menezes</i>	
Entropy-Based Attack on Community Detection in Directed Networks	152
<i>Saif Aldeen Madi and Giuseppe Pirrò</i>	
Synthetic Networks That Preserve Edge Connectivity	166
<i>Lahari Anne, The-Anh Vu-Le, Minhyuk Park, Tandy Warnow, and George Chacko</i>	
Human Behavior	
Criminal Network Construction and Analysis from Italian Civil Cases: A Case Study	181
<i>Claudia Licari, Gabriele Rinaldi, and Annamaria Ficara</i>	
Improving Accuracy of Image Geolocalization Based on Region Segmentation Using Proximity Graph Partitioning	194
<i>Rinto Koike, Takumu Toyama, and Takayasu Fushimi</i>	
Harmony in Diagnosis: Exploring Consensus and Variability in Clinical Judgement	207
<i>Sanad Satel, George Kour, Eyal Zinger, Yoav Ganzach, and Sarel Cohen</i>	
Information Spreading in Social Media	
Characterize Fake News Diffusion Patterns Using Temporal Logic Rules	221
<i>Valeria Fionda</i>	
Tracking Narrative Dynamics Using Churn Management on Social Networks	233
<i>Ahmed Al-Taweel, Nitin Agarwal, and Abiodun Quadri</i>	
Weakly Supervised Contrastive Representation Learning to Encode Narrative Viewpoint of COVID-19 Tweets	246
<i>Kin Wai Ng, Nathan Wendt, Jasmine Eshun, and Emily Saldanha</i>	

Enhancing Multilingual Fake News Detection Through LLM-Based Data Augmentation	258
<i>Razieh Chalehchaleh, Reza Farahbakhsh, and Noel Crespi</i>	
Anomalous Channel Detection for YouTube Through Label Propagation	271
<i>Ridwan Amure and Nitin Agarwal</i>	
Decoding Political Polarization in Social Media Interactions	282
<i>Giulio Pecile, Niccolò Di Marco, Matteo Cinelli, and Walter Quattrociocchi</i>	
Modelling the Influence of Deinfluencers	294
<i>Jiarui Hu and Tzu-Yi Chen</i>	
Multilayer/Multiplex	
The Structure of Illegal Online Gambling Network	309
<i>Ardian Maulana, Andhika Bernad, Fausto Keiluhu, and Hokky Situngkir</i>	
Early Prediction of Power Outage Duration Through Hierarchical Spatiotemporal Multiplex Networks	320
<i>Rafaa Aljurbua, Jumanah Alshehri, Shelly Gupta, Abdulrahman Alharbi, and Zoran Obradovic</i>	
Robustness of Short-Term Memory Models on Networks in Spatial Prisoner’s Dilemma	335
<i>Akihiro Takahara and Tomoko Sakiyama</i>	
Hierarchical Structure of Inter-Regional Input-Output Network: A Multilayer Approach	342
<i>Ardian Maulana, Hokky Situngkir, Pradono, and Adiwan Fahlan Aritenang</i>	
Resilience	
Structural and Flow-Based Approaches to Vulnerability Analysis of Complex Network Systems	353
<i>Olexandr Polishchuk</i>	
Sublinear Cuts are the Exception in BDF-GIRGs	366
<i>Marc Kaufmann, Raghu Raman Ravi, and Ulysse Schaller</i>	

Structural Network Measures

Elucidation of the Relationship Between the Box and Cluster Dimensions Using a Small-World Fractal Tree Model	381
<i>Nobutoshi Ikeda</i>	
Cell Signaling Characterization for Spatial Transcriptomics (ST) Data Using Network Analysis	394
<i>Azka Javaid and H. Robert Frost</i>	
Exploiting Vertex-Cut Partitioning in Distributed Graph Generation	406
<i>Furkan Atas and Mehmet Burak Akgun</i>	
Flow-Model Capacity for Large Scale Quantum Complex Networks	418
<i>Ricardo Quintas and Bruno Coutinho</i>	
Topological Features vs Structural Features in Drug Interactions	430
<i>Dimitrios Vogiatzis</i>	
Author Index	441

About The Editors

Hocine Cherifi is a professor of Computer Science at the University of Burgundy, Dijon, France, since 1999. He completed his Ph.D. degree at the National Polytechnic Institute, Grenoble, in 1984. Before moving to Dijon, he held faculty positions at Rouen University and Jean Monnet University, in France. He has held visiting positions at Yonsei, Korea, University of Western Australia, Australia, National Pingtung University, Taiwan, and Galatasaray University, Turkey. His recent research interests are in computer vision and complex networks. He has published more than 200 scientific papers in international refereed journals and conference proceedings. He held leading positions in more than 15 international conference organizations (general chair, program chair), and he served on more than 100 program committees. He is the founder of the International Conference on Complex Networks & their Applications. Currently, he is also a member of the Editorial Board of Computational Social Networks, PLOS One, IEEE Access, Journal of Imaging, Complex Systems, Quality and Quantity, and Scientific Reports. He is Founding Editor-in-Chief of the Applied Network Science journal.

Luis M. Rocha is the George J. Klir Professor of Systems Science at the Thomas J. Watson College of Engineering and Applied Science, Binghamton University, where he leads the Complex Adaptive Systems and Computational Intelligence (CASCI) lab. He is also a senior fellow at the Instituto Gulbenkian da Ciencia, co-director of the Consortium for Social and Biomedical Complexity between Binghamton University and Indiana University, and a founding partner of the International Center for Excellence in Mental Health Sciences. He has been the director of the NSF-NRT Interdisciplinary Training Program in Complex Networks and Systems and professor of Informatics at the Luddy School of Informatics, Computing, and Engineering at Indiana University. His research is on complex networks and systems, computational and systems biology, and computational intelligence. He received his PhD in Systems Science in 1997 from the State University of New York at Binghamton. From 1998 to 2004, he was a permanent staff scientist at the Los Alamos National Laboratory, where he founded and led a Complex Systems Modeling Team during 1998–2002, and was part of the Santa Fe Institute research community. He has organized major conferences in the field such as Alife X, ECAL 2007, and Complex Networks 2019–2023. He has published many articles in scientific and technology journals and has been the recipient of several scholarships and awards.

Chantal Cherifi received her PhD in Computer science from Corsica University, France, in 2011. Since 2014, she is working as Associate Professor at the DISP laboratory, at the University of Lyon 2, France. Her main research interests are focused on information systems agility and big data management with applications on enterprise information systems and smart cities, using complex networks, ontologies, and product lifecycle management (PLM) systems tools. She is involved in several international

conference organizations such as Complex Networks, CompleNet, PLM, and DICTAP. She serves as a member of international conferences program committees of Complex Networks, CompleNet, Complexis, ISCRAM-med, CSCESM, SITIS, and ICIEIS; and journal referee for IJCIM, EPL, Scientific Reports, and Nature. She is a member of EU Erasmus-Mundus programs (SmartLink and cLink program). Her local responsibilities include Committee Lab member, since 2016, and Lab seminar co-organizer, since 2014.

Murat Donduran is Professor of Economics at the Department of Economics, YILDIZ Technical University İstanbul Türkiye, where he is Director of the Graduate School of Social Sciences. He is also Board Member of in Turkish Economic Foundation. His research is on microeconomics, computational economics, firm dynamics, and game theory. He received his Bachelor's and Master's Degrees in Economics from Marmara University, İstanbul, Türkiye, and a Ph.D. in Economics in 2000 from the YILDIZ Technical University. Since 2000, he has been a faculty member in the Faculty of Economic and Administrative Sciences at the YILDIZ Technical University. He has organized several conferences in the field such as the YILDIZ International Conference on Social Sciences, the Annual International Conference on Social Sciences (2016–2020), and the International Conference on Economics (IceTea) 2019–2023. He has published many articles in scientific journals such as Games and Economic Behavior, Review of Industrial Organization, Physica A, and North American Journal of Economics and Finance.

Dr. Onur Varol is Assistant Professor at the Sabancı University Faculty of Engineering and Natural Sciences and Principal Investigator at the VIRAL Lab. His research focuses on developing techniques to analyze online behaviors to improve individual well-being and address societal problems using online data. He is awarded by The Turkish Science Academy to Young Scientist Awards (2022) and Research Incentive Award by METU Parlar Foundation (2022). He also received TUBITAK 2247-D National Leader Researcher Grant in 2022. Prior to joining Sabancı University, he was a postdoctoral researcher at Northeastern University at the Center for Complex Network Research. He completed his PhD in Informatics at Indiana University, Bloomington (USA). His thesis focuses on the analysis of manipulation and threats on social media and he was awarded the 2018 University Distinguished Ph.D. Dissertation Award. He has developed a system called Botometer to detect social bots on Twitter and his team ranked top 3 worldwide at the 2015 DARPA Bot Detection Challenge. His research published in prestigious venues such as International Conference of Web and Social Media (ICWSM), Nature Communications, Nature Human Behavior, World Wide Web (WWW) conference, and Communications of the ACM.

Anomaly Detection



DCOR: Anomaly Detection in Attributed Networks via Dual Contrastive Learning Reconstruction

Hossein Rafiee Zade, Hadi Zare^(✉), Mohsen Ghassemi Parsa,
Hadi Davardoust, and Meshkat Shariat Bagheri

Department of Data Science and Technology, School of Intelligent Systems
Engineering, University of Tehran, Tehran, Iran

{hossein.rafiee,h.zare,mgparsa,davardoust,m.shariatbagheri}@ut.ac.ir

Abstract. Anomaly detection using a network-based approach is one of the most efficient ways to identify abnormal events such as fraud, security breaches, and system faults in a variety of applied domains. While most of the earlier works address the complex nature of graph-structured data and predefined anomalies, the impact of data attributes and emerging anomalies are often neglected. This paper introduces DCOR, a novel approach on attributed networks that integrates reconstruction-based anomaly detection with Contrastive Learning. Utilizing a Graph Neural Network (GNN) framework, DCOR contrasts the reconstructed adjacency and feature matrices from both the original and augmented graphs to detect subtle anomalies. We employed comprehensive experimental studies on benchmark datasets through standard evaluation measures. The results show that DCOR significantly outperforms state-of-the-art methods. Obtained results demonstrate the efficacy of proposed approach in attributed networks with the potential of uncovering new patterns of anomalies.

Keywords: Anomaly detection · Attributed networks · Graph Neural Networks · Graph Autoencoders · Contrastive learning

1 Introduction

Anomaly detection in attributed networks plays a critical role in modern networked systems due to their increasing complexity and interconnectivity [1]. Anomalies often occur in applied domains such as financial systems, computer networks, and the internet known as fraud, security breaches, or system faults with billions of costs for companies [2].

Traditional anomaly detection methods often have problems when facing with attributed graph-structured data. Techniques such as Local Outlier Factor (LOF) [3] and Structural Clustering Algorithm for Networks (SCAN) [4] primarily rely on local density and structural deviations. However, these approaches

suffered to deal with node attributes and connectivity structure of the network simultaneously.

The rise of deep learning brought new technologies in the identification of anomalies, such as autoencoders and Graph Neural Networks (GNNs). AnomalyDAE [5] uses dual autoencoders to reconstruct both the adjacency matrix and node features for detecting anomalies that hurt either the structure or network attributes. Recent works still build upon those foundations, like Dominant [6], which demonstrated good results on attributed networks based on graph convolutional neural networks.

Additionally, contrastive learning is a robustness-enhancing method, which learns to distinguish between augmented versions of data. It has shown promise within graph representation learning, for example, GraphCL [7], which uses data augmentation in order to create contrasting views of graphs. That is, contrastive learning maximizes the agreement among different augmented views of the same data (positive pairs) while minimizing the agreement between the views of different data points (negative pairs). The aforementioned pre-training strategies for GNNs were experimented upon by Hu et al. [8], who showed the capability of the model to capture global and local structural information.

Based on these insights, we propose a new way of integrating dual reconstruction-based anomaly detection with contrastive learning. Our method, called Dual Reconstruction Contrastive Learning (DCOR), introduces a GNN-based framework that contrasts the reconstructed adjacency and feature from both original and augmented graphs. This contrast allows DCOR to better adapt in capturing both structural and attribute anomalies, due to the learned robust reconstructions through contrastive learning, ensuring that even subtle anomalies missed by previous methods are detected. Furthermore, this contrast enhances the quality of our reconstruction, which in turn improves the overall performance of the model. By increasing the quality of the main network reconstruction, we indirectly enhance the representations and consequently strengthen the model’s ability to detect anomalies.

In this paper, we present the design and implementation of DCOR, along with a detailed performance evaluation across multiple datasets. After summarizing recent related work, we define our proposed approach, which includes the graph augmentation process, the anomaly detection model, and the contrastive learning framework.

We evaluate DCOR’s performance against several baseline methods using five datasets from diverse domains, including Flickr, Amazon, Enron, and Facebook. The results demonstrate that DCOR achieves notable improvements compared to state-of-the-art methods.

2 Problem Definition

Formally, let $G = (A, X)$ represent an attributed network, where $A \in \mathbb{R}^{n \times n}$ is the adjacency matrix and $X \in \mathbb{R}^{n \times d}$ is the attribute matrix. Each node v_i in the graph is represented by a row in both A and X . The adjacency matrix A

captures the structural relationships between nodes, with $A_{ij} = 1$ if there is an edge between nodes v_i and v_j , and $A_{ij} = 0$ otherwise. The attribute matrix X contains attribute vectors for each node, where d is the dimensionality of the attribute space.

The goal of anomaly detection is to recognize a subset of nodes $V_{\text{anomaly}} \subseteq V$ where each node $v \in V_{\text{anomaly}}$ shows anomalous behavior. This anomalous behavior could happen in three ways: structural anomalies, attribute anomalies, and jointly together.

Specifically, we define our anomaly detection problem as follows:

- **Input:** An attributed network $G = (A, X)$ with adjacency matrix A and node attribute matrix X .
- **Output:** A ranked ordering of nodes via their anomaly scores, with higher scores for a more likely of being anomalous.

Our proposed method aims to produce an anomaly score $S(v)$ for each node $v \in V$, where nodes with the highest scores are identified as anomalies.

3 Related Work

3.1 Autoencoder and GNN-Based Anomaly Detection

Some works combine autoencoders with GNNs for better anomaly detection because they capture complex dependencies in graph data. For instance, Zhou and Liu proposed AnomalyDAE [5], which is a dual autoencoder architecture to simultaneously reconstruct the adjacency matrix and node features for anomaly detection. Some analogical approaches, such as Dominant [6], based on GNNs, try to learn node embeddings from a GNN to acquire good performance in node-wise anomaly detection. A recent and likely the most relevant one is by Liu et al. [9], presenting contrastive learning on attributed graphs to improve anomaly detection. The focus of such an approach lies in generating effective negative samples to distinguish normal patterns from anomalous ones, thereby achieving enhanced robustness of the model. Such approaches underline the strength of autoencoders and GNNs in capturing structural nuances and dependencies in graph-based tasks of anomaly detection.

3.2 Graph Contrastive Learning and Self-supervised Learning

Some of the latest techniques in this line, which have considerably advanced the robustness and generalization of GNNs, are graph contrastive learning and self-supervised learning with data augmentation.

Graph Contrastive Learning: GraphCL [7] introduces a way to improve the GCNs by providing different views of graphs with respect to data augmentations, like node dropping and edge perturbations. It works in a way in which the augmented views between the same graph are brought to maximum agreement, but minimized with those of different graphs.

Self-supervised Learning with Data Augmentation: In [10], robust embeddings of anomaly detection are generated by adding node and graph augmentations. Curriculum negative sampling is introduced to adjust the difficulty of sampling, which has resulted in strengthening the model’s ability to improve the identification of anomalous patterns [11].

Some other recent works: adversarial training of GCL to enhance robustness in [12]; a self-supervised framework for integrating graph augmentation and multi-view learning in [13]; an SSL-over-graph federated learning approach in [14]. Some strategies for pre-training GNNs are also discussed in reference [8] for node and graph-level tasks.

3.3 Recent Advances in Graph Anomaly Detection

A dynamic graph anomaly detection using GNNs was proposed in [15], followed by the introduction of the first contrastive learning method for detecting anomalies in temporal networks. A domain-specific contrastive learning technique that is agnostic to the graph was proposed in [16]. They proposed a hybrid model that combined graph neural networks with reinforcement learning for anomaly detection within dynamic graphs.

4 Proposed Method

Our method starts by generating both the original network and an augmented version. We then focus on comparing the reconstructed adjacency and feature matrices from the original network with those from the augmented one.

What sets our approach apart from traditional contrastive learning frameworks [17], which usually compare embeddings from original and augmented networks, is our focus on reconstructing the actual graph data. By shifting attention to reconstruction, we can uncover more intricate anomalies and subtle differences in the graph’s structure and features-details that might be missed if we were only looking at the embeddings. Our main goal is to refine the quality of these reconstructions within the autoencoder, allowing the model to more effectively distinguish between normal and anomalous data by emphasizing the key differences.

Initially, we use an autoencoder to help us in the reconstruction of the adjacency and feature matrices of the graph. These reconstructions are the basis for carrying out anomaly detection. Now begins the process of contrastive learning. At this stage, the model is going to compare the reconstructed adjacency and feature matrices of the original and augmented graphs. The aim is to augment the accuracy of anomaly detection through this contrast. Such an approach enables the model to detect slight differences in the structure and features of normal versus anomalous graphs, thus making anomaly detection much more accurate.

During reconstruction, anomalies found in the augmented graph lead the model to try deviating from their reconstruction of the original network. At the same time, the model will also try keeping its reconstruction for the nodes that

are anomalously far away from theirs. For instance, if an augmented network node is infected, it tries to predict the reconstruction of that node back toward its corresponding node in the original network. Such contrasts are made both in terms of node features and the adjacency matrix. Such contrast will therefore let the model learn how it should accordingly reconstruct the graph. It implies that the model would be more skilled at detecting anomalies because differences and similarities at both the feature and structural level between normal and abnormal nodes could be distinguished. Ultimately, we are interested in improving the power of the model to capture subtle yet significant anomalies present within graph data by ameliorating the quality of the reconstructions supplied by the autoencoder.

4.1 Graph Data Augmentation

Anomaly detection graph augmentation refers to the incorporation of artificially generated anomalies into graph data to enhance the effectiveness of detection algorithms [9]. In order to enable effective testing and verification, we incorporate unconventional nodes or edges into the network, hence producing concrete abnormalities. Both structural and feature anomalies are included, with augmentation rates optimally adjusted to the unique characteristics of each dataset. Upon encountering an anomaly, a node is allocated the label of anomalous (anomaly label), which facilitates straightforward discrimination during testing. By incorporating tailored anomaly creation, the model's robustness and capacity to generalize to real-world scenarios are improved.

Graph augmentation can be expressed as:

$$G' = G + A \quad (1)$$

where G' is the augmented graph, G is the original graph, and A represents the anomalies introduced to the graph.

Node-Level Augmentation. We perturb node features by adding noise to simulate anomalies and increase data diversity. Perturbation improves the model's ability to generalize. Techniques include:

- **Feature Copying:** Replicating the feature vector of a node with maximum deviation to a target node creates attribute anomalies [18], useful in detecting subtle deviations.
- **Feature Scaling:** Adjusting feature values by a scaling factor creates anomalies in feature distribution, enhancing detection.

Structural Augmentation. Structural augmentation alters the graph by adding or removing edges to simulate unusual connectivity patterns. This helps the model generalize and detect structural anomalies. Techniques include:

- **Adding Subgraph:** Introducing dense subgraphs around nodes creates structural anomalies, simulating unusual connectivity patterns.
- **Drop all connections:** Disconnecting a node isolates it, simulating scenarios like faults or network issues, and test the model's ability to detect isolated nodes.

Our approach customizes augmentation rates based on each dataset's characteristics, rather than using uniform strategies. This improves anomaly detection performance by optimizing model robustness and generalization.

4.2 Anomaly Detection Model and Contrastive Learning

Traditional contrastive learning frameworks [17] focus on comparing embeddings from different augmented views of the same data to maximize agreement between similar pairs and minimize agreement between dissimilar pairs. Our approach builds upon this concept by first reconstructing the graph data using a dual autoencoder model. Since typical representations may lose some sensitive information about the nodes, which can hinder accurate contrasts, we compare the reconstructed adjacency matrices and feature matrices directly. By feeding both the main graph and the augmented graph into the dual autoencoder, we enhance the detection of anomalies and ensure more precise results.

Dual Autoencoder Model. The dual autoencoder model reconstructs the graph's adjacency and feature matrices using both structure and attribute autoencoders. The main and augmented graphs are fed into the model to obtain reconstructed matrices.

Structure Autoencoder. The structure autoencoder encodes the graph structure into a latent representation and reconstructs the adjacency matrix.

- **Encoder:** Compresses the input graph (A, X) into a latent representation Z_V :

$$\tilde{Z}_V = \sigma(XW_V^{(1)} + b_V^{(1)}) \quad (2)$$

where X is the feature matrix, $W_V^{(1)}$ and $b_V^{(1)}$ are weight and bias parameters, and σ is the activation function.

The attention scores (e_{ij}) between nodes i and j are computed as follows:

$$e_{ij} = \text{attn}(\tilde{Z}_i^V, \tilde{Z}_j^V) = \sigma \left(a^T \cdot [W_V^{(2)}\tilde{Z}_i^V || W_V^{(2)}\tilde{Z}_j^V] \right) \quad (3)$$

where \tilde{Z}_i^V and \tilde{Z}_j^V are the latent representations of nodes i and j , $W_V^{(2)}$ is the weight matrix, a is the attention weight vector, and $[||]$ represents concatenation of the vectors.

A graph attention layer then aggregates the representation from neighboring nodes:

$$\gamma_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in N_i} \exp(e_{ik})} \quad (4)$$

where γ_{ij} is the attention coefficient for nodes i and j , computed based on the attention scores e_{ij} .

The final embedding for node i , denoted as Z_V^i , is computed as:

$$Z_V^i = \sum_{k \in N_i} \gamma_{ik} \tilde{Z}_V^k \quad (5)$$

where N_i represents the neighbors of node i .

- **Decoder:** Reconstructs the adjacency matrix \hat{A} from Z_V :

$$\hat{A} = \text{Sigmoid}(Z_V Z_V^T) \quad (6)$$

where Z_V is the latent node representation and \hat{A} is the reconstructed adjacency matrix.

Attribute Autoencoder. The attribute autoencoder encodes node attributes into latent space and reconstructs the attribute matrix.

- **Encoder:** Transforms node attributes X into a latent representation Z_A :

$$\tilde{Z}_A = \sigma(X^T W_A^{(1)} + b_A^{(1)}) \quad (7)$$

where $W_A^{(1)}$ and $b_A^{(1)}$ are weight and bias parameters for attribute encoding.

$$Z_A = \tilde{Z}_A W_A^{(2)} + b_A^{(2)} \quad (8)$$

where Z_A is the final latent representation of the node attributes.

- **Decoder:** Reconstructs the attribute matrix \hat{X} from Z_V and Z_A :

$$\hat{X} = Z_V (Z_A)^T \quad (9)$$

where \hat{X} is the reconstructed attribute matrix.

Loss Function. The objective is to minimize the reconstruction errors of both structure and attributes:

$$\mathcal{L}_{\text{rec}} = \alpha \|A - \hat{A}\|_F^2 + (1 - \alpha) \|X - \hat{X}\|_F^2 \quad (10)$$

where α controls the trade-off between structure A and attribute X reconstruction, and $\|\cdot\|_F^2$ represents the Frobenius norm.

Nodes with higher reconstruction errors, indicated by the above loss function, are more likely to be anomalies.

Contrastive Loss Function. After obtaining the reconstructed adjacency and feature matrices from both the main and augmented graphs, the contrastive loss, which consists of both structural contrastive loss and feature contrastive loss, is applied to these reconstructed matrices. The margin parameter m used in Equations (12) and (13) is set to 0.5. To further enhance the model's ability to distinguish between the original and augmented graphs, contrastive learning

is incorporated. This allows the network to effectively differentiate between the reconstructions of the main graph and those of the augmented graph containing labeled anomalies.

The contrastive loss is a combination of structural contrastive loss and feature contrastive loss:

$$\mathcal{L}^{sc} = \mathcal{L}_{\text{struct}} + \mathcal{L}_{\text{feat}} \quad (11)$$

where:

$$\mathcal{L}_{\text{struct}} = \frac{1}{n} \sum_{i=1}^n \left(I_{y_i=0} \cdot d(A_i, \hat{A}_i) + I_{y_i=1} \cdot \max\{0, m - d(A_i, \hat{A}_i)\} \right) \quad (12)$$

$$\mathcal{L}_{\text{feat}} = \frac{1}{n} \sum_{i=1}^n \left(I_{y_i=0} \cdot d(X_i, \hat{X}_i) + I_{y_i=1} \cdot \max\{0, m - d(X_i, \hat{X}_i)\} \right) \quad (13)$$

where, $d(\cdot, \cdot)$ represents the distance between matrices, $I_{y_i=0}$ and $I_{y_i=1}$ are indicator functions for normal and anomalous instances, and m is a margin parameter.

The total contrastive loss \mathcal{L}^{sc} includes:

- **Positive Pair Loss:** Minimizes the distance between the reconstructed matrices of the main and augmented graphs.
- **Negative Pair Loss:** Ensures the distance between reconstructed matrices of different graphs exceeds the margin m .

As part of the contrastive learning process, the model evaluates the reconstruction of anomalous nodes in the augmented network during the reconstruction of the main network. The goal is to push the reconstruction of anomalous nodes further from their counterparts in the main network, while pulling the reconstruction of normal nodes closer to their corresponding nodes in the main network. Through these contrasts, the model improves its ability to reconstruct the graph and enhances its anomaly detection capabilities. Additionally, by leveraging the augmented network reconstructions, the model generates a more accurate reconstruction of the main network.

Total Loss. The total loss is a weighted sum of the reconstruction loss and the contrastive loss. This combined loss function ensures that the model not only accurately reconstructs the graph but also effectively distinguishes between normal and anomalous graphs.

$$\mathcal{L}_{\text{total}} = \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{sc}} \mathcal{L}^{sc} \quad (14)$$

where λ_{rec} and λ_{sc} determine the balance between reconstruction and contrastive losses.

The methodology can be summarized as follows:

1. **Graph Augmentation:** Create augmented versions of the graph using feature and structural augmentations.
2. **Reconstruction:** Apply the dual autoencoder to both the main and augmented graphs to reconstruct adjacency and feature matrices.
3. **Contrastive Learning:** Use contrastive loss to compare the reconstructed matrices, improving anomaly detection by highlighting differences between the main and augmented graphs.

5 Experiments

5.1 Datasets

We evaluate DCOR using four real-world datasets-Flickr [19], Amazon [20], Enron [21], and Facebook [22]-from diverse domains such as social networks, e-commerce, and email communications.

Table 1 provides key statistics for these datasets.

Table 1. Dataset Statistics

Dataset	Nodes	Edges	Features	Ground Truth Anomalies
Flickr	7,575	23,938	12,047	600
Amazon	1,418	3,695	21	28
Enron	13,533	176,987	18	5
Facebook	4,039	88,234	576	400

5.2 Experimental Settings

We evaluate the method on four real-world datasets with consistent hyperparameters across all:

- Adam optimizer - Embedding and hidden dimensions set to 128 - Learning rate of 1e-2 (except for Facebook) - One hidden layer in the encoder

Dataset-specific hyperparameters are detailed in Table 2.

We implemented DCOR using PyTorch and ran the experiments on Google Colab with an NVIDIA T4 GPU and 15 GB of memory.

Table 2. Hyperparameters for each dataset

Hyperparameter	Enron	Amazon	Facebook	Flickr
Epochs	200	200	200	200
Learning Rate	0.01	0.01	0.2	0.01
Alpha	0.1	0.5	0.1	0.9
Structure Anomaly Rate	0.9	0.1	0.5	0.2
Feature Anomaly Rate	0.2	0.1	0.1	0.9
Reconstruction Loss Weight (λ_{rec})	0.8	0.1	0.7	0.5
Contrastive Loss Weight (λ_{sc})	0.9	0.4	0.4	0.7

Table 3. AUC scores of all methods on four datasets.

Method	Enron	Amazon	Facebook	Flickr
LOF	0.581	0.510	0.522	0.661
Dominant	0.716	0.592	0.554	0.749
AEGIS	0.602	0.556	0.659	0.765
AnomalyDAE	0.552	0.611	0.741	0.694
CONAD	0.731	0.635	0.863	0.782
DCOR	0.861	0.762	0.926	0.822

5.3 Experimental Results

We compare DCOR with several existing methods including LOF [3], Dominant [6], AEGIS [23], AnomalyDAE [5], and Conad [17]. The AUC scores for anomaly detection are shown in Table 3.

To better understand how the various augmentation strategies influence the performance of DCOR, we conducted an ablation study on the Amazon dataset under three different scenarios: feature augmentation only, adjacency augmentation only, and without contrastive learning. We show the results in Table 4, giving the corresponding AUC scores for each augmentation type.

Table 4. Ablation study results for DCOR with different augmentation strategies on the Amazon dataset.

Augmentation Type	AUC Score
Feature Augmentation Only	0.712
Adjacency Augmentation Only	0.673
Without Contrastive Learning	0.592

In this paper, we show that incorporating contrastive learning with reconstruction-based methods provides a more robust anomaly detection app-

roach for attributed networks. Table 3 shows that DCOR consistently outperforms state-of-the-art methods with significantly higher AUC scores on different datasets. The improvement is highly significant in the datasets of Facebook and Enron, where DCOR attained AUC scores of 0.926 and 0.861, respectively. These results indeed promise DCOR as a great potential to enhance anomaly detection, considering its impressively good performance across a variety of settings.

One challenge with traditional contrastive learning methods is that such embeddings may not be able to capture all intricacies within the structure and attributes of the graph, especially when anomalies involve complex patterns. During the process of embedding, subtle details might be overlooked, which in turn reduces anomaly detection accuracy.

Unlike in traditional methods, DCOR does not focus on a comparison of node embeddings but shifts its attention to the reconstruction of adjacency and feature matrices for the graph. This allows the model to learn structural and attribute anomalies that are complex and hard to find otherwise.

Furthermore, DCOR detects effective contrasting views by using customized augmentation techniques; hence, it equips the model for better discrimination between normal and anomaly nodes. This two-dimensional analysis of structure and attributes strengthens the power of anomaly detection, as evidenced by the excellent performance on challenging datasets such as Facebook and Enron.

Finally, the DCOR framework integrates dual reconstruction with contrastive learning to provide an end-to-end anomaly detection approach that significantly enhances its robustness and adaptability to complex network data. This allows the model to accurately reconstruct the graph while effectively distinguishing between normal and abnormal patterns, resulting in superior performance on anomaly detection.

6 Conclusion

In this paper, we proposed DCOR a dual contrastive learning based approach for anomaly detection in attributed networks. By contrasting the reconstructed adjacency and feature matrices in both original and augmented graphs, DCOR successfully identified the low-level irregularities likely to be neglected by prior methods. This technique improved the quality of reconstructions in the autoencoder, thereby enabling the model to capture important anomalies. Extensive experiments conducted on a number of public datasets including Flickr, Amazon, Enron, and Facebook revealed that DCOR outperformed the current state-of-the-art algorithms with higher AUC scores. These results demonstrated the strength of the integrated proposed approach using contrastive learning with reconstruction-based methods for detecting anomalies in attributed networks.

References

1. Ding, K., Li, J., Liu, H.: Interactive anomaly detection on attributed networks. In: Proceedings of the Twelfth ACM International Conference on Web Search

- and Data Mining (WSDM), pp. 357–365 (2019). <https://doi.org/10.1145/3289600.3290964>
2. Irofti, P., Pătrașcu, A., Baltoiu, A.: Fraud detection in networks. In: Springer, vol. 517–536. Springer (2020). https://doi.org/10.1007/978-3-030-52067-0_23
 3. Breunig, M. M., Kriegel, H. P., Ng, R. T., Sander, J.: LOF: Identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 93–104 (2000). <https://doi.org/10.1145/342009.335388>
 4. Xu, X., Yuruk, N., Feng, Z., Schweiger, T. A.: SCAN: a structural clustering algorithm for networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 824–833 (2007). <https://doi.org/10.1145/1281192.1281280>
 5. Zhou, F., Liu, Q.: AnomalyDAE: dual autoencoder for anomaly detection on attributed networks. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1295–1303 (2020). <https://doi.org/10.1145/3394486.3403272>
 6. Ding, K., Li, J., Bhanushali, R., Liu, H.: Deep anomaly detection on attributed networks. In: Society for Industrial and Applied Mathematics eBooks, pp. 594–602 (2019). <https://doi.org/10.1137/1.9781611975673.67>
 7. You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. In: Advances in Neural Information Processing Systems, pp. 5812–5823 (2020)
 8. Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., Leskovec, J.: Strategies for Pre-training Graph Neural Networks. International Conference on Learning Representations (2019)
 9. Liu, Y., Zheng, V.W., Dou, D., Malik, J.: Anomaly detection in attributed networks via contrastive self-supervised learning. arXiv preprint <arXiv:2102.06162> (2021)
 10. Duan, H., Xie, C., Li, B., Tang, P.: Self-supervised contrastive graph representation with node and graph augmentation. Neural Netw. **167**, 223–232 (2023). <https://doi.org/10.1016/j.neunet.2023.08.039>
 11. Yan, R., Bao, P.: ConCur: self-supervised graph representation based on contrastive learning with curriculum negative sampling. Neurocomputing **551**, 126525 (2023). <https://doi.org/10.1016/j.neucom.2023.126525>
 12. Zhu, X., et al.: Adversarial contrastive learning for robust graph neural networks (2023)
 13. Wang, H., et al.: Multi-view learning for self-supervised anomaly detection in graphs (2023)
 14. Li, J., et al.: Federated self-supervised learning for anomaly detection in graphs (2023)
 15. Wang, X., Wang, P., Zhang, Y., Zhang, Y.: Dynamic graph anomaly detection via contrastive self-supervised learning. In: Proceedings of the 2022 IEEE International Conference on Data Mining (ICDM), pp. 1323–1328 (2022). <https://doi.org/10.1109/ICDM55829.2022.00178>
 16. Zhang, T., Wang, H., Zhang, Y.: A Hybrid Model Combining GNNs with Reinforcement Learning for Dynamic Graph Anomaly Detection. In: Proceedings of the 2024 AAAI Conference on Artificial Intelligence (2024)
 17. Xu, Z., Huang, X., Zhao, Y., Dong, Y., Li, J.: Contrastive attributed network anomaly detection with data augmentation. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), pp. 3–15. Springer (2022). https://doi.org/10.1007/978-3-031-05936-0_1

18. Sun, Y., Wang, W., Wu, N., Yu, W., Chen, X.: Anomaly Subgraph Detection with Feature Transfer. Proceedings of the 29th ACM International Conference on Information & Knowledge Management (2020). https://doi.org/10.1007/978-3-031-05936-0_1
19. Huang, X., Li, J., Hu, X.: Label informed attributed network embedding. In: WSDM 2017
20. Sánchez, P.I., Müller, E., Laforet, F., Keller, F., Böhm, K.: Statistical Selection of congruent subspaces for mining attributed graphs. In: ICDM (2013)
21. Müller, E., Sánchez, P.I., Mülle, Y., Böhm, K.: Ranking outlier nodes in subspaces of attributed graphs. In: ICDE Workshop (2013)
22. McAuley, J.J., Leskovec, J.: Learning to discover social circles in ego networks. In: NeurIPS (2012)
23. Ding, K., Li, J., Agarwal, N., Liu, H.: Inductive anomaly detection on attributed networks. In: IJCAI (2020)



Enhancing Recommender Systems with Anomaly Detection: A Graph Neural Network Approach

Bahareh Rahmatikargar^(✉), Pooya Moradian Zadeh, and Ziad Kobti

School of Computer Science, University of Windsor, Ontario, Canada
`{rahmatib,moradiap,kobti}@uwindsor.ca`

Abstract. In this study, we investigate the incorporation of anomaly detection into session-based recommender systems using Graph Neural Networks (GNNs) to enhance recommendation accuracy. We explore various integration stages, including item embedding, fusion layers, and session representation. Our methodology involves initially zeroing out anomalous embeddings and subsequently replacing them with what we consider to be the best embeddings based on anomaly scores derived from the Isolation Forest algorithm. Through extensive experimentation on a real-world dataset, we observed that anomaly detection at the session representation layer yields significant improvements in hit rate (HR) and mean reciprocal rank (MRR). Our findings demonstrate that strategically integrating anomaly detection can effectively improve recommendation quality and system performance.

Keywords: recommender systems · accuracy · graph neural networks (GNNs) · anomaly detection

1 Introduction

In recent years, recommender systems have gained significant attention in the field of artificial intelligence. They are often developed to assist users in finding and selecting items, content, or products that match their preferences and interests [10]. Indeed, the majority of these systems aim to forecast users' potential interest in content or items they haven't yet encountered. Consequently, they can provide personalized recommendations based on these predictions, enhancing the user's experience by introducing them to relevant choices.

Recently, various approaches have emerged for recommendation systems ranging from traditional content-based filtering that suggests items based on past preferences to more modern techniques such as collaborative filtering, hybrid methods, knowledge-based systems, context-aware systems, graph-based models, sequential, session-based, social network-based, and cross-domain recommender systems. Among them, Session-Based methods stand out for their ability to provide real-time recommendations based on a user's immediate interactions within

an ongoing session. These algorithms prove especially valuable when dealing with unregistered or new users, situations where long-term data is often insufficient [6]. Leveraging the capabilities of deep learning techniques, these methods aim to comprehend a user’s short-term intent while taking into account contextual factors that influence their current session.

Session-based recommender systems have evolved from using recurrent neural networks (RNNs) to adopting GNNs for modeling user browsing behavior. Using graphs, in particular, offers the advantage of representing relationships between items and users in a highly structured manner [3,8], enabling better recommendations. This transition to GNNs brings notable advantages, such as considering item attributes and extracting complex item relationships [15]. However, an important aspect that has been overlooked in current research is the potential negative impact of anomalous embeddings on recommendation accuracy.

In this research work, we focus on exploring various strategies for incorporating anomaly detection into the recommendation process. We also present a new method to enhance the quality of the session-based recommendations by utilizing the GNN approach and incorporating anomaly detection. Anomaly detection in this context involves identifying embeddings or data points that significantly deviate from the expected or normal patterns within the system. It aims to pinpoint unusual or rare instances that may represent anomalies or outliers in the data, potentially indicating noise, errors, or novel user behaviors that need special attention or investigation.

The rest of the paper is structured as follows. In the next section, we review various strategies within session-based recommender systems. After that, our methodology is presented. We will then discuss the experiments and evaluation results. Finally, we summarize our findings and outline our future work.

2 Related Work

In session-based recommender systems, various approaches have been proposed. In [7], the authors introduced a novel approach that combines a Transformer with a target-attentive GNN, leading to richer representations of user behavior and preferences. The authors in [18], proposed a method to address the problem with the Dual-channel Graph Transition Network, which captures item transitions not only within the target session but also among similar neighbor sessions. The STAR framework, introduced in [17], enhances the recommendation by leveraging session temporal information and focusing on the time intervals between session events. In [16], SR-GNN, utilizes GNNs to capture complex item transitions within session sequences. It combines global preferences and current interests in sessions using an attention network. In [12], the authors introduced the Interval-enhanced Graph Transformer, which enhances recommendations by incorporating time intervals, item interactions, and user preference prediction.

In [13], RN-GNN has been introduced to handle repeat consumption, emphasizing intra-session learning and addressing item popularity bias through innovative techniques like mini-batch sessions and dropout layers. The authors in

[11] have proposed a method to tackle the recommendation challenges with the Enhanced GNN, combining global and local item graphs and introducing novel fusion algorithms. The authors in [2] proposed a method to deal with the challenge of recommending new items in session-based systems. They incorporated a dual-intent network and zero-shot learning to predict user intent for new items.

There are also a few studies utilizing anomaly detection on graphs for recommendation purposes. In [14], the authors proposed a graph anomaly detection method, TA-Detector, which uses a trust classifier to differentiate trustworthy and untrustworthy connections. By integrating a novel GNN (DTNN) and residual networks, their approach significantly improves GAD performance, as shown by experiments on multi-relation datasets. Moreover, in [9], the authors introduced GAD-NR, a graph anomaly detection method based on a graph autoencoder that reconstructs neighborhood information from node representations. By capturing self-features, degree reconstruction, and neighboring node distributions, GAD-NR effectively detects contextual, structural, and joint anomalies, outperforming state-of-the-art baselines across multiple datasets.

The proposed model in [4] is one of the leading works on the Yoochoose dataset in terms of accuracy metrics and serves as the basis for our research in this paper. In that work, the authors proposed a dual GNN framework to capture implicit and explicit item relationships. The framework has four main components: the adaptive GNN (A-GNN) module for implicit information aggregation, the single gate (SG-GNN) module for explicit information aggregation, the session representation layer, and the prediction layer. An inter-session graph is constructed by collecting item sets based on neighbor sessions from one batch and converting them into a unified low-dimensional embedding space. The A-GNN module employs multi-head correlation to capture implicit correlations between any two items dynamically, while the SG-GNN module uses a gate mechanism to leverage the explicit dependencies among items reflected by sequential information in sessions. The fusion layer combines the updated item representation from these two modules, while the session representation layer generates the next item's prediction probability. They proposed two kinds of session representation: local and global representations to capture users' short-term and long-term preferences [4]. The final session representation is obtained by concatenating the local and global representations and applying a linear transformation. The item scores are then computed, followed by generating probabilities for the next items.

3 Methodology

Let $U = \{u_1, u_2, \dots, u_n\}$ represents a finite set of n users where each user $u_i, 1 \leq i \leq n$, has the capability to interact with items and is also a potential recipient of recommendations from the recommender system. The interactions between users and items can indeed vary based on the domain and context. Some examples of such interactions may include actions such as purchasing, viewing, watching, clicking on, or rating items, depending on the specific application and use case of the recommender system. Let $I = \{i_1, i_2, \dots, i_m\}$, denote a set of m items

that users can interact with or be recommended to. These items can vary widely depending on the application, too. For example, they can be physical products, digital content, movies, books, events, advertisements, music tracks, or even user actions based on the particular goals and domain of the recommendation system.

Accordingly, a session can be represented as a finite non-empty list, $s = [u, i_1, i_2, \dots, i_k]$, where $i_k \in I$ are the sequence of items interacted with or viewed by user $u \in U$ during the session. Therefore $S = \{s_1, \dots, s_{|S|}\}$ represents a set of these sessions. Consequently, the main goal of the recommender function and algorithm, $R : S \mapsto 2^I$, is to take user sessions as input and, based on them, recommend sets of items that the user might be interested in, considering all possible combinations of items from the set I .

Many GNN session-based recommender systems follow a specific set of steps and phases to process the input data. They can be categorized as Graph Construction, Item embedding, Fusion layer, and Session representation. The first step is to create a graph representation of user-session-item interactions, where nodes may represent users, sessions, or items, and edges capture relationships or interactions between them. The second step is to use a method to represent items as embeddings in a high-dimensional space. These embeddings capture the inherent characteristics and properties of items in a way that facilitates recommendation algorithms to make personalized and accurate item suggestions to users. The embeddings are used as input for GNN models. These GNN models can include multiple GNNs to thoroughly capture the complex connections between items (In our work, the A-GNN and SG-GNN modules are employed for implicit and explicit information aggregation, respectively).

After that, the fusion layer serves as an integration point where information from various sources or representations, such as user and item embeddings, is combined. The goal is to create a unified representation, allowing the system to capture complex patterns and relationships. Session representation is the next step, which is a representation specific to the user's current session, capturing their recent or overall interactions and behaviors. This final representation is then used by the algorithm to make personalized recommendations.

In session-based recommender systems, some sessions or interactions may not carry meaningful information and, more importantly, they may sometimes act as noise in the system. We hypothesize that identifying and removing them from the process may enhance the system's accuracy. Consequently, we explore various scenarios for integrating anomaly detection into the system. Anomaly detection targets both anomalous sessions and individual data points, focusing on those that deviate notably from the system's normal behavior. By identifying these outliers, whether at the session level or within specific embeddings, the system is equipped to address potential noise, errors, or unusual and misleading user behaviors, ensuring more accurate and reliable recommendations. The proposed framework is shown in Fig. 1.

Graph Construction Phase: in this scenario, anomaly detection can be incorporated during the graph construction phase. It involves examining the graph

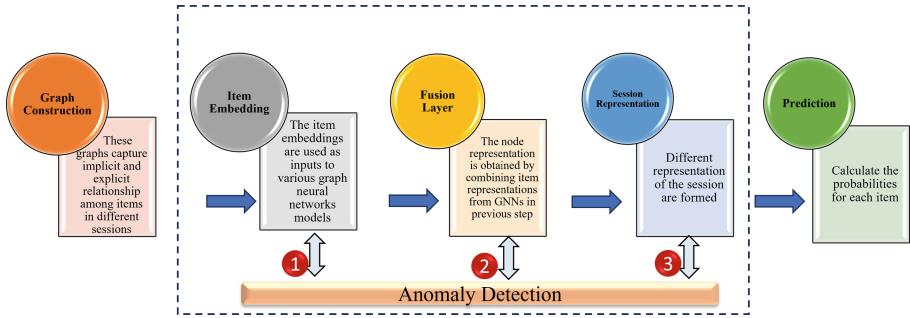


Fig. 1. The proposed framework

itself as input for anomaly detection algorithms, with the aim of identifying unusual items within its structure.

Let $G = (V, E)$ be the graph constructed from the set of items I where V is the set of nodes representing items, and E is the set of edges representing connections or interactions between items in sessions. Edges are generated based on the sequential occurrence of items within user sessions. The across-session edges are also formed based on implicit correlations between non-adjacent items across sessions. The graph includes all sessions, so interactions from multiple users shape its structure.

The problem can be defined as finding a subset $A \subseteq V$, where A represents the anomalous nodes in the graph G . In this approach, anomalies are detected and removed at a very early stage. Consequently, because the graph's structure may not be fully revealed at this point, there is a potential risk of inaccurate anomaly identification and removal. Furthermore, eliminating the identified anomalies during this phase could lead to the loss of valuable information. They might have played a role in certain patterns or relationships within the data that could be significant to the overall recommendation process. Consequently, in this paper, we do not consider applying anomaly detection in this phase.

Item Embedding Phase: integrating anomaly detection involves using item embeddings as input for the GNN model. Let $E(I)$ be the set of item embeddings obtained from the graph construction phase. Typically, these embeddings are based on the graph structure or derived from node features of the items. The problem can be defined as finding a subset $B \subseteq E(I)$, where B represents the anomalous item embeddings. Algorithm 1 represents the pseudocode for this case. However, removing anomalies during item embedding may lead to the loss of valuable data patterns that could be relevant for generating recommendations. This action could introduce inconsistencies into the training data, potentially impeding the model's effective learning process. Moreover, inadequate handling of anomalies within item embeddings may negatively impact the model's capacity to generalize and make accurate predictions for new, unseen data. In addition, sometimes GNN inputs may consist of random embeddings,

lacking inherent meaning. These initial representations undergo refinement during network training, rendering anomaly detection ineffective in this phase.

Algorithm 1. Recommender System with Anomaly Detection on Item Embeddings

Initialization: $U = \{u_1, u_2, \dots, u_n\}$ Set of users, $I = \{i_1, i_2, \dots, i_m\}$ Set of items, $S = \{s_1, s_2, \dots, s_{|S|}\}$ Set of user sessions

Input: I, S

$G = \text{create graph}(S)$ *Construct graph(s) from sessions*

$E(I) = \text{embed items}(G)$ *Generate item embeddings*

$\text{IsolationForest.fit}(E(I))$ *Fit the Isolation Forest on the entire set of item embeddings*

$a_Scores = \text{IsolationForest.decision_function}(E(I))$ *Compute anomaly scores for all item embeddings*

$ZscoreEmbeddings = \text{stats.zscore}(a_Scores)$ *Compute z-scores for the anomaly scores*

for $\text{itemEmb} \in E(I)$ **do**

if $ZscoreEmbeddings[\text{itemEmb}]$ exceeds threshold **then**

 Remove item from recommendations

end if

end for

$IEmb1 = \text{GNN1}(E(I))$ *Apply GNN1 on item embeddings (Here, GNN1 = A-GNN)*

$IEmb2 = \text{GNN2}(E(I))$ *Apply GNN2 on item embeddings (Here, GNN2 = SG-GNN)*

... *(We can have more GNNs if needed)*

$F(E(I)) = \text{fusion layer}(IEmb1, IEmb2, \dots)$ *Combine embeddings of multiple GNNs*

$\text{SessionRep} = \text{session representation}(F(E(I)))$ *Capture session-specific features//Final representations = Local representations for short-term preferences + Global representations for long-term interests, concatenated and transformed as mentioned in Related work.*

$\text{ItemScores} = \text{compute}(\text{item scores}(\text{normal Sessions}))$

Recommend top k (ItemScores , $k = 20$)

Fusion Layer: when incorporating anomaly detection into this phase, the input is the item embeddings and the objective is to identify and exclude anomalous embeddings. Let $E(I)$ be the set of item embeddings obtained from the graph construction phase. Let $F(E(I))$ be the output of the fusion layer, which combines insights from multiple GNNs. The task can be seen as finding a subset $C \subseteq F(E(I))$, where C represents the anomalous embeddings after fusion. The pseudocode is shown in Algorithm 2.

During this phase, anomaly detection might fail to recognize the unique contributions of GNNs. For instance, when employing both the A-GNN and the SG-GNN, it could result in missed opportunities to finely adjust and optimize each component individually to address specific types of anomalies. Additionally, the fusion layer often operates on higher-dimensional representations, which can be computationally more intensive for anomaly detection. Meanwhile, since it

combines information from multiple sources, anomalies in one component might disproportionately influence the overall anomaly detection.

Session Representation: when integrating anomaly detection at this stage, two perspectives are possible: identifying anomalies within session items (i.e., on items in each session) or at the session level. In the first approach, we detect and remove anomalies in the session item representation. In the second, we conduct session-level anomaly detection and remove anomalous sessions.

The session representation layer typically features lower dimensions when compared to the fusion layer. This dimensionality reduction simplifies the anomaly detection process, enhancing computational efficiency. It also can yield more interpretable results and potentially detect different types of anomalies associated with short-term and long-term user preferences, providing more insights into distinct patterns of abnormal behavior. We believe this scenario may provide a better and more meaningful opportunity to apply anomaly detection. The pseudocode is shown in Algorithm 3.

We utilize the isolation forest algorithm [5] for anomaly identification. It has proven to be quite promising in efficiently detecting anomalies within complex data spaces. It also has low complexity and needs minimal parameter tuning. By computing and employing the Z-score to establish a threshold for anomaly detection, we proceed to eliminate anomalous entities in different scenarios. We use the z-score for setting this threshold due to its effectiveness in quantifying the degree of deviation of individual data points from the dataset’s mean. Other than removing the anomalous values also, we replaced the anomalous values with what we deemed as the best embedding results. we did this in two ways: first, by replacing anomalous values with the most non-anomalous values using the Isolation Forest decision function, and second, by replacing anomalous values with those whose absolute z-scores were closest to zero.

At last, we calculate item scores and provide users with recommendations for suitable items.

4 Experiment and Results

We have employed the work presented in [4] as our baseline model. We then evaluated the effectiveness of the scenarios along with our proposed algorithm. We implemented our work using the Python and the PyTorch library. Our study was conducted on the real-world dataset Yoochoose 1/64 [1]. This dataset consists of a stream of six-month user clicks on an e-commerce website. Given the massive size of the training set in Yoochoose, we opted to use two subsamples as our training sets. Based on the work presented in [4], we conducted filtering by excluding sessions with a length of 1 and items that occurred fewer than 5 times. Additionally, we consistently utilized the Adam optimizer with a learning rate of 0.0005, which yielded the best results across all scenarios. Our learning rate will decay by 0.5 after every five training epochs, and we used L_2 regularization (10^{-5}) and early stopping to prevent overfitting.

Algorithm 2 Recommender System with Anomaly Detection on Fusion Layer Embeddings

Initialization: $U = \{u_1, u_2, \dots, u_n\}$ Set of users, $I = \{i_1, i_2, \dots, i_m\}$ Set of items, $S = \{s_1, s_2, \dots, s_{|S|}\}$ Set of user sessions

Input: I, S

$G = \text{create graph}(S)$ *Construct graph(s) from sessions*

$E(I) = \text{embed items}(G)$ *Generate item embeddings*

$\text{IEmb1} = \text{GNN1}(E(I))$ *Apply GNN1 on item embeddings (Here, GNN1 = A-GNN)*

$\text{IEmb2} = \text{GNN2}(E(I))$ *Apply GNN2 on item embeddings (Here, GNN2 = SG-GNN)*

$F(E(I)) = \text{fusion layer}(\text{IEmb1}, \text{IEmb2}, \dots)$ *Combine embeddings of multiple GNNs*

$\text{IsolationForest.fit}(F(E(I)))$ *Fit Isolation Forest on the fusion embeddings*

$a_Scores = \text{IsolationForest.decision_function}(F(E(I)))$ *Compute anomaly scores*

$ZscoreEmbeddings = \text{stats.zscore}(a_Scores)$ *Compute z-scores*

for itemEmb in $F(E(I))$ **do**

Scenario 1: Zeroing Anomalous Embeddings

if $ZscoreEmbeddings[\text{itemEmb}]$ exceeds threshold **then**

Remove item from recommendations *(Zeroing out anomalous embeddings)*

end if

end for

for itemEmb in $F(E(I))$ **do**

Scenario 2: Replacing Anomalous Embeddings with Best Embeddings based on Anomaly Scores

$\text{bestEmbedding} = F(E(I))[a_Scores.\text{argmin}()]$

Find the best fitted embeddings

if $ZscoreEmbeddings[\text{itemEmb}]$ exceeds threshold **then**

Replace itemEmb with bestEmbedding

end if

end for

for itemEmb in $F(E(I))$ **do**

Scenario 3: Replacing Anomalous Embeddings with Best Embeddings - Zero Proximity

if $ZscoreEmbeddings[\text{itemEmb}]$ exceeds threshold **then**

$\text{bestEmbeddingNearZero} = \text{find embeddings where } ZscoreEmbeddings \text{ is near zero}$

Replace itemEmb with $\text{bestEmbeddingNearZero}$ *(Using near-zero embeddings)*

end if

end for

$\text{SessionRep} = \text{session representation}(F(E(I)))$ *Capture session-specific features // Final representations = Local representations for short-term preferences + Global representations for long-term interests, concatenated and transformed.*

$\text{ItemScores} = \text{compute}(\text{item scores}(\text{normal Sessions}))$

Recommend top k (ItemScores , $k = 20$)

We initialized parameters with Gaussian distribution ($\text{mean} = 0$, $\text{std} = 0.1$), used fixed embedding dimension and batch size (100), and varied GC-GNN lay-

ers (1–4), A-GNN blocks (4–6), and dropout ratios (0.1, 0.5, 0.9) following the approach used in [4].

To evaluate the effectiveness of our scenarios and the performance of our proposed model, we considered HR (hit rate) and MRR (mean reciprocal rank) as our evaluation metrics. HR measures the proportion of users who receive at least one relevant recommendation. MRR measures the quality of the recommendations by calculating the average reciprocal rank of the first relevant recommendation for each user.

Algorithm 3 Recommender System with Anomaly Detection on Session Representation

Initialization: $U = \{u_1, u_2, \dots, u_n\}$ Set of users, $I = \{i_1, i_2, \dots, i_m\}$ Set of items, $S = \{s_1, s_2, \dots, s_{|S|}\}$ Set of user sessions

Input: I, S

$G = \text{create graph}(S)$ *Construct graph(s) from sessions*

$E(I) = \text{embed items}(G)$ *Generate item embeddings*

$\text{IEmb1} = \text{GNN1}(E(I))$ *Apply GNN1 on item embeddings (Here, GNN1 = A-GNN)*

$\text{IEmb2} = \text{GNN2}(E(I))$ *Apply GNN2 on item embeddings (Here, GNN2 = SG-GNN)*

... *(We can have more GNNs if needed)*

$F(E(I)) = \text{fusion layer}(\text{IEmb1}, \text{IEmb2}, \dots)$ *Combine embeddings of multiple GNNs*

$\text{SessionRep} = \text{session representation}(F(E(I)))$ *Capture session-specific features// Final representations = Local representations for short-term preferences + Global representations for long-term interests, concatenated and transformed.*

$\text{IsolationForest.fit}(\text{SessionReps})$ *Fit the Isolation Forest on all session representations*

$a_Scores = \text{IsolationForest.decision_function}(\text{SessionReps})$ *Compute anomaly scores for all sessions*

$ZscoreEmbeddings = \text{stats.zscore}(a_Scores)$ *Compute z-scores for anomaly scores*

for (sessionRep, Zscore) in (SessionReps, ZscoreEmbeddings) **do**

if Zscore exceeds threshold **then**

 Remove session from recommendations *(Mark as anomalous)*

end if

end for

$\text{ItemScores} = \text{compute}(\text{item scores(normal_Sessions)})$

Recommend top k (ItemScores, k = 20)

Hit Rate (HR):

$$HR = \frac{1}{|U|} \sum_{u \in U} \frac{|R_u \cap G_u|}{|G_u|} \quad (1)$$

where $|U|$ is the total number of users, R_u is the set of recommended items for user u , and G_u is the set of relevant (ground-truth) items for user u .

Mean Reciprocal Rank (MRR):

$$MRR = \frac{1}{|U|} \sum_{u \in U} \frac{1}{\text{rank}_u} \quad (2)$$

where rank_u is the rank of the first relevant item for user u .

Additionally, to evaluate each scenario, we examined several z-score threshold intervals, including $(-1.5, 1.5)$, $(-2, 2)$, $(-2.5, 2.5)$, and $(-3, 3)$. This approach allowed us to conduct a detailed assessment of anomalies at varying levels of strictness in the Isolation Forest.

To evaluate the first scenario, we performed anomaly detection on item embeddings. However, we observed that the initial embeddings generated using PyTorch's class introduce randomness. As this randomly initialized data serves as inputs for anomaly detection in this phase, the process becomes ineffective due to the inherent randomness, resulting in invalid outcomes.

Then we performed anomaly detection on the fusion layer. At first, we zero out the anomalous embeddings. The results are shown in Table 1.

Table 1. DGNN with anomaly detection (Zeroing Anomalies) on fusion layer

Z-Score Range	HR	MRR
$(-1.5, 1.5)$	65.4031	30.8831
$(-2, 2)$	72.6878	38.4424
$(-2.5, 2.5)$	78.4572	45.6115
$(-3, 3)$	79.0136	47.535

Subsequently, we replaced the anomalous embeddings with the best embeddings. In our exploration, we observed that anomalous embeddings, particularly in the case of the Sklearn Isolation Forest library, often have decision function values within the range of $(-0.5, 0.5)$. Lower values within this range typically indicate the most anomalous embeddings. Consequently, we opted to utilize the embeddings that minimize the anomaly scores, as the best embedded items, derived from the output of the decision function in the Isolation Forest. The results are presented in Table 2 in the Lowest Anomaly Score section. Despite our initial expectations, the results indicate that this approach did not have a significant impact on improving the quality of the recommendations.

Therefore, in subsequent iterations, we considered alternative strategies. One involved identifying items with z-scores of anomalous scores near zero as the best embeddings, replacing anomalous items with these values. The results are shown in Table 2 in the Zero Proximity section. As we gained the best result when zeroing out the anomalous embeddings, we used this strategy in the last scenario on session representation.

In our last scenario, we aimed to remove anomalous sessions by leveraging Isolation Forest. We conducted experiments on session representation using both global and final session representations, as discussed in the previous section.

Table 2. DGNN with anomaly detection and replacement approaches on fusion layer

Z-Score Range	Best Embedding (Lowest Anomaly Score)			Zero Proximity	
	HR	MRR	Z-Score Range	HR	MRR
(-1.5, 1.5)	43.3933	54.2381	(-1.5, 1.5)	38.8708	12.2481
(-2, 2)	43.7207	14.4710	(-2, 2)	34.2213	9.8915
(-2.5, 2.5)	26.2478	7.7710	(-2.5, 2.5)	44.8567	14.8033
(-3, 3)	54.1254	19.8473	(-3, 3)	36.6364	11.0104

Our baseline DGNN model [4] achieved hit rate and Mean reciprocal rank values of 79.629 and 48.739, respectively. However, as Table 3 shows, our best results were obtained by running anomaly detection within the global session representation, particularly when the Z-scores were within (-2, 2).

Table 3. DGNN with Anomaly Detection on various Session Representations

Z-Score Range	HR (Global)	MRR (Global)	HR (Final)	MRR (Final)
(-1.5, 1.5)	79.2175	47.3588	79.5027	48.0485
(-2, 2)	80.6308	49.0166	79.5019	48.1486
(-2.5, 2.5)	80.0225	47.7766	80.0825	48.1271
(-3, 3)	80.2998	48.8955	80.0834	48.1276

5 Discussion

Based on our experiments, the best result in anomaly detection across various stages of GNNs was observed in session representation, consistent with our expectations. In the initial approach of replacing anomalies with zero embeddings, the system achieved relatively high HR and MRR scores across different Z-score ranges, indicating decent recommendation accuracy. However, upon replacing anomalies with what we were considered as the best embeddings based on Isolation Forest anomaly scores, the performance saw a notable decrease in HR and MRR scores, particularly in Z-score ranges of (-2, 2) and (-2.5, 2.5). This unexpected drop suggests that the replacement embeddings may not have effectively captured the underlying patterns in the data, leading to less accurate recommendations.

Further exploration involved identifying anomalies with near-zero Z-scores and replacing them with alternative embeddings. This strategy yielded mixed results. While some Z-score ranges showed improvements in HR and MRR compared to the initial replacement method, others saw a decline, indicating a trade-off between anomaly removal and recommendation accuracy.

Overall, our experiments demonstrate the complexity of anomaly handling in recommendation systems and highlight the importance of carefully considering the trade-offs between anomaly removal and recommendation quality.

6 Conclusion

In this study, we investigated the effects of applying anomaly detection at various stages of session-based recommender systems. Additionally, we introduced an approach to enhance the accuracy of these systems by integrating the Isolation Forest anomaly detection algorithm into the session representation layer of GNN-based frameworks. Our experimental analysis on a real-world dataset demonstrates the effectiveness of our proposed approach, as evidenced by improved HR and MRR evaluation metrics.

In the future, we plan to gain insights into the patterns and characteristics of anomalous embeddings and use these insights to improve the model. Additionally, we will explore advanced GNN-based anomaly detection methods with additional datasets to further enhance the system's accuracy.

References

1. Ben-Shimon, D., Tsikinovsky, A., Friedmann, M., Shapira, B., Rokach, L., Hoerle, J.: RecSys challenge 2015 and the YOOCHOOSE dataset. In: Proceedings of the 9th ACM Conference on Recommender Systems, pp. 357–358 (2015)
2. Jin, D., et al.: Dual intent enhanced graph neural network for session-based new item recommendation. In: Proceedings of the ACM Web Conference, pp. 684–693 (2023)
3. Jouyandeh, F., Sadeghi, S., Rahmatikargar, B., Zadeh, P.M.: Fake news and COVID-19 vaccination: a comparative study. In: Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 525–531 (2021)
4. Li, Z., Wang, X., Yang, C., Yao, L., McAuley, J., Xu, G.: Exploiting explicit and implicit item relationships for session-based recommendation. In: Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, pp. 553–561 (2023)
5. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 413–422. IEEE (2008)
6. Ludewig, M., Mauro, N., Latifi, S., Jannach, D.: Empirical analysis of session-based recommendation algorithms: a comparison of neural and non-neural approaches. User Model. User-Adap. Inter. **31**, 149–181 (2021)
7. Mitheran, S., Java, A., Sahu, S.K., Shaikh, A.: Introducing self-attention to target attentive graph neural networks. arXiv preprint [arXiv:2107.01516](https://arxiv.org/abs/2107.01516) (2022)
8. Rahmatikargar, B., Zadeh, P.M., Kobti, Z.: Social isolation detection in palliative care using social network analysis. In: 2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid), pp. 905–912. IEEE (2022)
9. Roy, A., et al.: GAD-NR: graph anomaly detection via neighborhood reconstruction. In: Proceedings of the 17th ACM International Conference on Web Search and Data Mining, pp. 576–585 (2024)

10. Shao, B., Li, X., Bian, G.: A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph. *Expert Syst. Appl.* **165**, 113764 (2021)
11. Sheng, Z., Zhang, T., Zhang, Y., Gao, S.: Enhanced graph neural network for session-based recommendation. *Expert Syst. Appl.* **213**, 118887 (2023)
12. Wang, H., Zeng, Y., Chen, J., Han, N., Chen, H.: Interval-enhanced graph transformer solution for session-based recommendation. *Expert Syst. Appl.* **213**, 118970 (2023)
13. Wang, J., Xie, H., Wang, F.L., Lee, L.K., Wei, M.: Jointly modeling intra-and inter-session dependencies with graph neural networks for session-based recommendations. *Inf. Process. Manage.* **60**(2), 103209 (2023)
14. Wen, J., et al.: Ta-Detector: a GNN-based anomaly detector via trust relationship. *ACM Trans. Multimedia Comput. Commun. Appl.* (2024)
15. Wu, S., Sun, F., Zhang, W., Xie, X., Cui, B.: Graph neural networks in recommender systems: a survey. *ACM Comput. Surv.* **55**(5), 1–37 (2022)
16. Shu, W., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. *Proc. AAAI Conf. Artif. Intell.* **33**, 346–353 (2019)
17. Yeganegi, R., Haratizadeh, S.: STAR: a session-based time-aware recommender system. arXiv preprint [arXiv:2211.06394](https://arxiv.org/abs/2211.06394) (2022)
18. Zheng, Y., Liu, S., Li, Z., Wu, S.: DGTN: dual-channel graph transition network for session-based recommendation. In: 2020 International Conference on Data Mining Workshops (ICDMW), pp. 236–242. IEEE (2020)



FlowSeries: Anomaly Detection in Financial Transaction Flows

Arthur Capozzi¹, Salvatore Vilella^{2(✉)}, Dario Moncalvo³, Marco Fornasiero³, Valeria Ricci³, Silvia Ronchiadin³, and Giancarlo Ruffo²

¹ Computational Social Science, ETH Zürich, Zürich, Switzerland
arthur.capozzi@gess.ethz.ch

² DISIT, Università degli Studi del Piemonte Orientale “A. Avogadro”, Alessandria, Italy

{salvatore.vilella,Giancarlo.Ruffo}@uniupo.it

³ Anti Financial Crime Digital Hub, Turin, Italy
{dario.moncalvo,marco.fornasiero,valeria.ricci,silvia.ronchiadin}@intesasanpaolo.com

Abstract. In recent years, the digitization and automation of anti-financial crime (AFC) investigative processes have faced significant challenges, particularly the need for interpretability of AI model results and the lack of labeled data for training. Network analysis has emerged as a valuable approach in this context.

In this paper we present *FlowSeries*, a top-down search pipeline for detecting potentially fraudulent transactions and non-compliant agents. In a transaction network, fraud attempts are often based on complex transaction patterns that change over time to avoid detection. The *FlowSeries* pipeline requires neither an *a priori* set of patterns nor a training set. In addition, by providing elements to explain the anomalies found, it facilitates and supports the work of an AFC analyst.

We evaluate *FlowSeries* on a dataset from Intesa Sanpaolo (ISP) bank, comprising 80 million cross-country transactions over 15 months, benchmarking our implementation of the algorithm. The results, corroborated by ISP AFC experts, highlight its effectiveness in identifying suspicious transactions and actors, particularly in the context of the economic sanctions imposed in EU after February 2022. This demonstrates *FlowSeries'* capability to handle large datasets, detect complex transaction patterns, and provide the necessary interpretability for formal AFC investigations.

Keywords: Network analysis · Anti Financial Crime · Anti Money Laundering · Temporal Networks · Graph Search Algorithm

1 Introduction and Related Work

Financial markets are complex, adaptive systems involving diverse actors such as hedge funds, individual investors, and banks. Complex networks effectively model these systems, focusing on stabilization and destabilization dynamics [11],

stock price correlations [22], shareholder networks [3], and the resilience of financial networks [10]. International trade networks based on linguistic, cultural, or religious ties have also been examined [18], along with applications to financial crises, including the 2008 global financial crisis [1]. Beyond stock markets, network theory models competitive dynamics on the World Wide Web [14] and the cryptocurrency market, analyzing dominant cryptocurrencies [16], transaction structures [19], and Ethereum network attributes [12].

Network analysis is vital for anti-financial crime (AFC) and anti-money laundering (AML), detecting anomalies in transaction networks and automating fraud detection to enhance efficiency. Tools like VISFAN [5] use network metrics for identifying suspicious transactions. García et al. [8] applied network analysis in the Spanish Revenue Agency’s Tax Control Study, using algorithms for rapid fraud detection and community detection techniques for economic landscape representation. Colladon et al. [6] highlighted social network metrics for identifying money laundering through relational graphs of economic sectors, regions, transaction volumes, and ownership links. The CoDetect framework [9] integrates network and feature data for fraud detection.

AMLSim [24] generates synthetic bank transaction data with identifiable money laundering patterns. Preliminary results indicate that graph learning for AML is feasible even in large, sparse networks, with graph compression techniques like Ligra+ achieving significant compression [20]. Garcia-Bedoya et al. [7] advocate for AI and network analysis in AML, identifying three interaction types that conceal money laundering: path (money sent through intermediaries), cycle (money returns to its origin), and smurf (dividing a transaction into smaller ones). Liu et al. [13] suggest transaction cycles can indicate fraud.

Machine learning and deep learning, particularly graph neural networks, are explored for fraud detection [4]. Unsupervised anomaly detection is often used due to a lack of annotated training data, with innovative approaches like zero-shot learning [4]. Pan [15] proposes a deep-set algorithm combining meta-learning and zero-shot learning for money laundering detection. Reviews highlight effective anomaly detection strategies in fraud detection [2, 17], using social network analysis to uncover organized criminal behavior [25].

1.1 Our Contribution: FlowSeries

In this paper, we introduce *FlowSeries*, a top-down search pipeline based on network analysis, designed to aid Anti-Financial Crime (AFC) analysts in detecting illicit transactions and non-compliant agents. We evaluate *FlowSeries* on a dataset of 80 million cross-country bank transactions over 15 months, provided anonymously by Intesa Sanpaolo bank, compliant with legal privacy and security standards. This evaluation focuses on identifying attempts to bypass economic sanctions against Russia following the Ukraine invasion on February 24, 2022.

Financial fraud often involves complex, evolving transaction patterns to evade detection. *FlowSeries* identifies these by defining a *transaction flow* as a set of payment lines from x to y through intermediaries. Unlike other methods, *FlowSeries* does not rely on predefined patterns, addressing the challenge

of unlabeled data in AFC tools. Moreover, it provides interpretability, crucial for formal investigations by domain experts, unlike many black-box models [4].

The application of *FlowSeries* involves two steps:

1. **Determine Structural Granularity:** create a transaction network using financial transactions data. Each node can be any sort of entity, such as an *ISO code*, *BIC*, or *IBAN*. ISO 3166 codes define country names, BIC (Bank Identifier Code) identifies banks in international payments, and IBAN (International Bank Account Number) identifies bank accounts across borders.
2. **Compute paths on temporal networks:** Generate multiple weighted directed temporal networks by aggregating wire transfers over time (e.g., weekly, monthly). An algorithm computes all possible paths from a selected node for each temporal aggregation. Analysts explore these paths to identify trends, patterns, or anomalies and estimate the total money sent between nodes through intermediaries.

Section 3 presents the application of *FlowSeries*, demonstrating:

1. Its effectiveness in identifying anomalous transaction flows;
2. Its scalability to millions of transactions;
3. Its interpretability, facilitating formal investigations.

We formalize transaction networks and transaction flows in the next section, describe the algorithm for identifying transaction flows, and propose a technique for weighting flows between nodes. Section 3 covers the application experiments of *FlowSeries* at different granularities, while the last section discusses the limitations and the possible extensions of the work.

2 Methodology

2.1 Weighing a Transaction Flow

Transactions networks, usually represented as weighted directed temporal networks, are very useful to model financial transactions over time. On such a network $G_T = (V, L_T)$, we can define a path as an ordered finite collection of n distinct edges connecting vertices i and j during interval T :

$$\begin{aligned} P_{i,j}^T &= \{e(i, v_1), e(v_1, v_2), e(v_2, v_3), \dots, e(v_{n-1}, j)\}, \\ &\text{with } v_x \in V \text{ and } e(v_x, v_{x+1}) \in L_T \\ &\text{for all } x \in \{1, \dots, n-1\} \end{aligned} \tag{1}$$

The weight of the path $P_{i,j}^T$ is then given by

$$\sum_{i=1}^{n-1} w(e(v_i, v_{i+1})) \tag{2}$$

and $W(P_i) = \{w(e(v_1, v_2)), \dots, w(e(v_{n-1}, v_n))\}$ is the set containing the weights of the edges of the path P_i . Finally, $\text{Paths}_{i,j}^n(G_T)$ is the set of all possible paths of length n from i to j in graph G at interval T .

In real-world scenarios, to avoid detection of fraudulent activity, an agent may use intermediaries to transfer money. The number of intermediary groups or the length of each path connecting any two nodes x to y often cannot be known a priori.

A transaction through n intermediaries in a transaction network forms a path of length $n + 1$. The *FlowSeries* pipeline aims to identify the transaction flow from a given input node x and verify the maximum possible amount of money sent from x to other nodes within a maximum distance n .

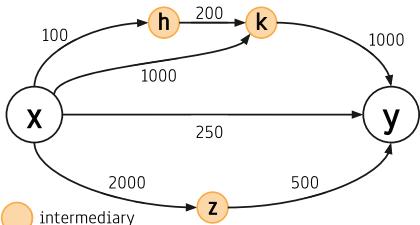
Equation 3 defines $\text{Flow}^n(x, y)$ as the set of all paths from x to y with a maximum length n :

$$\text{Flow}^n(x, y) = \{P_1, \dots, P_m\}, \text{ with } P_i \in \text{Paths}_{i,j}^n(G_T) \quad (3)$$

In a transaction network, the maximum amount node x could send to node y through intermediaries is the minimum weight of the edges in the path between x and y . For multiple intermediary groups, the transaction flow weight between x and y is the sum of the minimum weights of each path. The flow weight of maximum length n from x to y is defined as:

$$w(\text{Flow}^n(x, y)) = \sum_{i=1}^m \min(W(P_i)), \text{ with } P_i \in \text{Paths}_{i,j}^n(G_T) \quad (4)$$

Figure 1a illustrates a transaction network. To study the hypothetical amount of money sent from x to y , considering only the weight of edge $e_{x,y}$ is insufficient. Intermediaries like nodes h , k , and z must be considered. Table 1b lists the paths and minimum weights for each path. The flow weight from x to y is the sum of the minimum weights for each path, totaling 1850.



(a) Figure

P_i	$\min(W(P_i))$
$\{e(x, h), e(h, k), e(k, y)\}$	100
$\{e(x, k), e(k, y)\}$	1000
$\{e(x, z), e(z, y)\}$	500
$\{e(x, y)\}$	250

(b) Table

Fig. 1. In the transaction network in Fig. 1a, node x has an edge of weight 250 towards node y . The paths between nodes x and y through nodes h , k and z could be an attempt to hide a direct edge of higher weight. Table 1b lists all paths of maximum distance 3 from x to y and their respective minimum weights.

Note that the temporal aggregation of a network for the interval T is an approximation. In fact, each transaction has a timestamp t and therefore all edges involved in a valid path should satisfy the constraint $t_{e_i} < t_{e_{i+1}}$.

2.2 Pseudocode

The pseudocode of the algorithm underlying *FlowSeries* is shown in 1. It is a variant of the DFS graph search algorithm [21], and it consists of a recursive search in a network G of all paths of maximum length n starting from a given input node u . The search explores the nodes in depth, starting from node u , and each search ends when the length of the path has reached the maximum length n . The Algorithm 1 returns as output a list of lists, each representing a path and containing the ordered list of nodes traversed. It can be proven that the time complexity of the algorithm is less than or equal to $\max(d_{out})^n(n + 1)$, where d_{out} is the out-degree of one of the nodes in the network.

Algorithm 1. FlowSeries

```

1: procedure FINDPATHS( $G, u, n, n\_recursive$ )
2:    $paths \leftarrow []$ 
3:   if  $n\_recursive == 0$  then
4:     return  $[[u]]$ 
5:   end if
6:   if  $n\_recursive < n$  then
7:      $paths.append([u])$ 
8:   end if
9:   for neighbor in  $G.neighbors(u)$  do       $\triangleright$  loop through all neighbours of  $u$ 
10:    for path in FindPaths( $G, neighbor, n, n\_recursive - 1$ ) do
11:      if  $u$  not in path then
12:        path.insert(0,  $u$ )            $\triangleright$  append node  $u$  on the head of path list
13:        paths.append(path)
14:      end if
15:    end for
16:   end for
17:   return paths
18: end procedure

```

2.3 Time Series of Flows

Anomaly detection typically focuses on statistically significant changes in the weight of existing edges rather than identifying new edges. In many real world applications, detecting anomalies between two nodes at the path level involves discovering significant changes in weight, not new paths. In large financial transaction networks, connectivity is influenced by periodic payments forming repetitive patterns, such as salary payments or transfers to service providers. By analyzing a node's history, one can model transaction patterns and identify anomalies.

Applying *FlowSeries* in an AFC investigation involves examining the time series of $w(\text{Flow})$. A flow comprises multiple paths, so a single edge-level anomaly may not significantly impact the overall flow weight.

Figure 3 shows the time series of a transaction flow's weight between two nodes, with the flow weight in orange, the weighted moving average (WMA) in blue, and the exponentially weighted moving average (EWMA) in green. Anomaly detection can involve checking if the percentage difference between the expected value (WMA or EWMA) and the actual flow weight exceeds a threshold. In Fig. 3, this difference is indicated by vertical dotted lines, green if positive and red if negative.

2.4 *FlowSeries* Pipeline

We formalised the concepts described above in the *FlowSeries* pipeline, i.e., a sequence of steps that allow us to analyse all the possible flows of money between entities in a financial graph, that can be built at different levels of spatial and temporal aggregation. While this methodology can easily be generalised to any suitable graph, we will refer to financial graphs as this pipeline was originally designed to support an AFC investigation and we will illustrate its application specifically to financial graphs in this Sect. 3.

The *FlowSeries* pipeline can be described in the following steps:

1. Choose a structural and temporal aggregation T of the data in order to build the graph;
2. For each temporal aggregation, create a weighted temporal transaction network G_T ;
3. Select a node x from which to start the investigation, and, for each network G_T , execute the Algorithm 1.
4. Store the results of the previous step in a table. Each row of the table represents a path in time period T . For each path at period T , we have the list of the weights of the edges involved in that path. If available, additional information of the nodes involved in the path can be included;
5. The AFC analyst can now examine the table by filtering and sorting the paths. It is possible to filter the table by selecting all paths that end at a node y and obtain the transaction flow from x to y . If other information about the nodes is available, it can be used to further filter the results. For each time period T and transaction flow, we can compute several time series metrics, such as the moving average or the exponentially weighted moving average. These metrics can be used for sorting the transaction flows, as described in Sect. 2.3.

In summary, by executing Algorithm 1 on a properly constructed network, the AFC analyst will discover all the money paths of length n between any two nodes and will be able to analyze statistically as time series, in order to identify potentially abnormal connections.

3 Results

We test the potential of the *FlowSeries* tool by applying it to a dataset of cross-border transactions in a period of fifteen months, from September 2021 to November 2022. The dataset¹ is provided by Intesa Sanpaolo (ISP), the largest Italian bank operating across Europe, and it includes 80 million transactions involving ISP customers either as sender or receiver, 8008 unique BICs, and 218 countries. The data can be aggregated either by IBANs, BIC codes or Countries - modeling transactions respectively between IBANs, BICs or nations at a progressively more coarse-grained resolution. More details of this dataset are reported in [23].

3.1 Use Case: Money Flows Between Prominent European Countries During the Russia-Ukraine War

In this section, we show an example of how the *FlowSeries* tool can be used in a financial crime investigation. The example involve networks with different types of node aggregation and show how *FlowSeries* can assist in the identification of complex patterns that can conceal financial fraud. For privacy and security reasons, measures are taken in accordance with the legal provisions and the requirements of AFC Digital Hub consortium. In particular, transaction amounts are normalized to the maximum amount of the time series and BIC codes, that were already provided to the researchers in complete anonymisation, are further anonymised. Furthermore, all Country names will be anonymized as $C_1, C_2 \dots$ for the sake of data protection as requested by the data provider.

This use case investigates transaction flows between two prominent European countries at a macro scale, specifically focusing on the period between the outbreak of Russia-Ukraine war and the subsequent imposition of economic sanctions by the EU. Figure 2 illustrates the weekly aggregated direct amounts transferred from C_2 BICs to C_1 BICs. The grey dashed line marks the start of the war (24 February 2022). The trend remains stable throughout the period, indicating that economic sanctions did not significantly alter the transaction amounts from C_2 to C_1 . Figure 3 displays the time series of $w(Flow^3(C_2, C_1))$, representing the maximum potential amount sent from C_2 to C_1 BICs through one intermediary. Both figures, especially before the war outbreak, show peaks but no discernible patterns. To further investigate the issue, an AFC analyst can examine the intermediaries in the $Flow^3(C_2, C_1)$ transaction flow. There are 86 paths of maximum length 3 between C_2 and C_1 . Table 1 highlights the intermediaries that exhibit the largest percentage increase in actual value over the expected moving average after February 24. For example, countries C_4 and

¹ The data supporting the findings of this study is available from ISP upon request to AFC Digital Hub. Please note that restrictions for data availability apply. Researchers interested in having access to data for academic purposes will be asked to sign a non-disclosure agreement. Contact adh@pec.afcdigitalhub.com for further information.

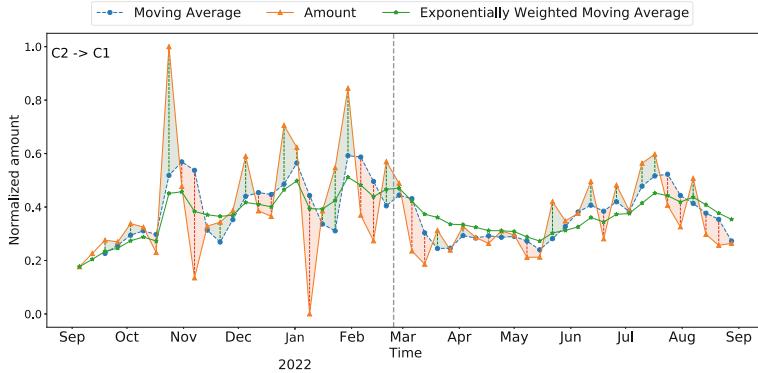


Fig. 2. Direct transactions from C2 BICs to C1 BICs. The vertical grey dotted line represents the begin of the Russian invasion of Ukraine (24 February 2022).

C5 have a percentage increase of 42% and 49%, respectively: we can investigate this evidence by analyzing all the transactions that pass through C4 and C5.

In Figs. 4 and 5, the left side shows the flow weights computed on weekly aggregated networks. On the right side, individual edges involved in the flows are depicted: C2 to C4 and C4 to C1 in Fig. 4 and C2 to C5 and C5 to C1 in Fig. 5. From May 2022, the weight of $w(\text{Flow}^3(C2, C1))$ shows rapid and abnormal growth. Analyzing these individual edges, discovered through *FlowSeries*, helps the analyst understand this growth and the actors involved and the actors

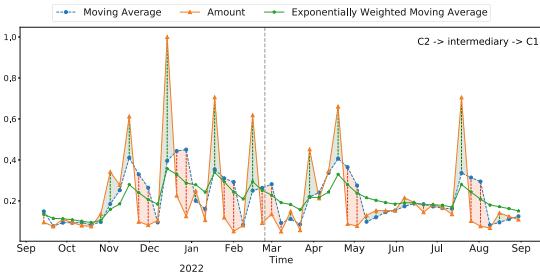


Fig. 3. Weight of the transaction flow $\text{Flow}^3(C2, C1)$, i.e., the hypothetical maximum amount of money sent from C2 to C1 through several payment lines, each with a maximum of one intermediary. The vertical grey dotted line represents the begin of the war in Ukraine.

Table 1. Intermediaries in $\text{Flow}^3(C2, C1)$ that show the largest percentage increase between the expected value given by the moving average and the actual value.

Country	Difference
C4	0.427
C6	0.436
C5	0.493
C7	0.535
C8	0.539
C9	0.549
C10	0.604
C11	0.636
C12	0.662
C13	0.665

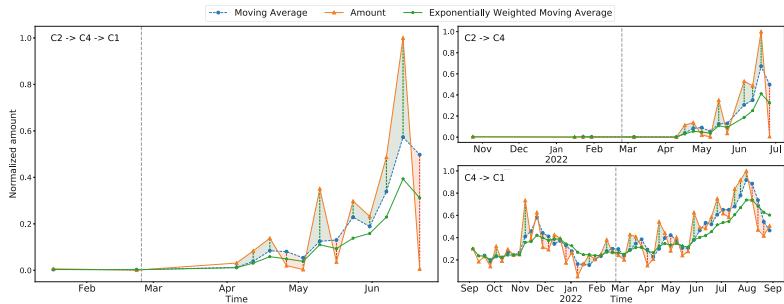


Fig. 4. Transaction flow from C2 to C1 through C4.

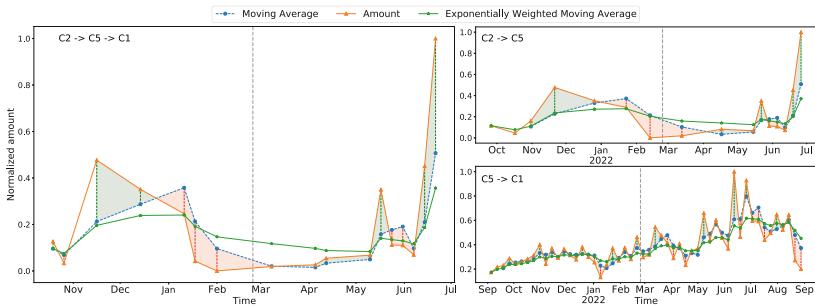


Fig. 5. Transaction flow from C2 to C1 through C5.

involved in these suspicious transfers. Incidentally, very similar patterns of intermediaries used to bypass international sanctions have been well documented².

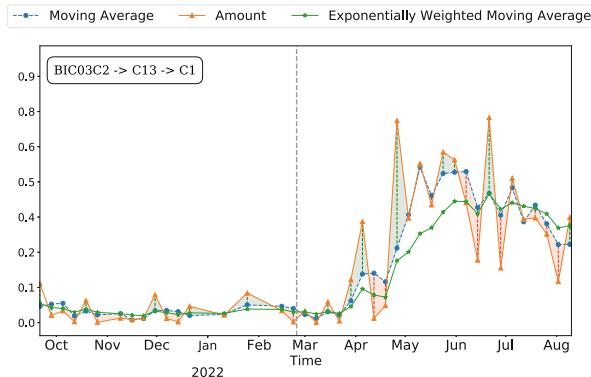


Fig. 6. Transactions towards country C1, passing through intermediaries in country C13, originating from a specific BIC (BIC03C2) involved in malicious operations.

² <https://www.ft.com/content/0fc846f7-aac8-4a34-a7dd-3b0615bce983>.

Having exogenous information helps: by deploying the pipeline of anomaly detection on financial graphs described in [23], we can identify suspicious BICs in C2, use them as seeds for *FlowSeries*, and proceed to a more fine-grained analysis. An example of this procedure is shown in Fig. 6, which shows the *FlowSeries* from a BIC of the country C2, named BIC03C2, to C1. In this case, only paths through the intermediary C13, which stands out in Table 1 as a possible intermediary between C2 and C1 with a 66% increase, are filtered out. The weight of $w(\text{Flow}^3(BIC03C2, C1))$ has a strong increasing trend after the outbreak of the war in Ukraine, peaking in the months of May and June.

4 Conclusions, Limitations and Future Work

The *FlowSeries* pipeline supports AFC analysts in identifying suspicious actors and anomalous transactions but has some limitations. First, there is no certainty that transaction flows between nodes indicate attempts to conceal financial crimes. The flow's weight represents the maximum amount transferred through intermediaries and multiple payment lines, and anomalies in these weights require an AFC analyst's expertise and additional data to verify financial crime attempts. Second, an AFC analyst needs a root node to start the investigation. However, investigations can reveal new suspicious nodes at various granularity levels, as shown in the examples in Sect. 3. The pipeline can then be applied to these new nodes, starting a new investigation.

Interpretability and the lack of labeled data are significant challenges in using automated systems for AFC. Interpretability is crucial for reporting anomalies to authorities, and the lack of labeled data stems from the confidential nature of financial data and the complexity of labeling vast amounts of transactions. Studies such as [8] highlight the diverse and complex nature of fraudulent transaction patterns. Although *FlowSeries* requires an initial search point, as demonstrated in Sect. 3, an exploratory search can start from a broad level, such as a country, and progressively zoom in to identify anomalous BICs or specific suspicious transactions.

Future improvements could include enhanced path filtering tools based on additional information available to AFC analysts, refined anomaly detection on transaction flow weights using ARIMA or SARIMA models as well as other machine learning based models, and improved node filtering by removing irrelevant nodes, according to some given metric. Such enhancements could reduce computational costs and streamline the analysis process, making *FlowSeries* more efficient and effective in real-world applications.

Acknowledgements. The research that led to this paper has been conducted started in 2022, when AC, SV and GR where at the University of Turin. **Authors' Contribution:** AC: Conceptualization, Methodology, Software, Validation, Visualization, Writing - Original Draft. SV: Methodology, Software, Validation, Writing - Review & Editing. DM, MF, VR, and SR: Resources, Data Curation, Validation. GR: Methodology, Supervision, Writing - Review & Editing. **Funding:** This research has been funded by AFC Digital Hub (Anti Financial Crime Digital Hub) consortium, whose members

are Intesa Sanpaolo Innovation Center, University of Turin, Polytechnic University of Turin, and CENTAI. GR and SV also acknowledge funding by the European Union - Next Generation EU, Mission 4 Component 2 - CUP C53D23005810006.

References

1. Crises in economic complex networks: black swans or dragon kings? *Economic Anal. Policy* **62**, 105–115 (2019). <https://doi.org/10.1016/j.eap.2019.01.009>, <https://www.sciencedirect.com/science/article/pii/S0313592617302783>
2. Bolton, R.J., Hand, D.J.: Statistical fraud detection: a review. *Stat. Sci.* **17**(3), 235–249 (2002). <http://www.jstor.org/stable/3182781>
3. Caldarelli, G., Battiston, S., Garlaschelli, D., Catanzaro, M.: Emergence of Complexity in Financial Networks, vol. 650, pp. 399–423 (2004)
4. Chen, Z., Le, D.V.K., Teoh, E., Nazir, A., Karuppiah, E., Lam, K.: Machine learning techniques for anti-money laundering (AML) solutions in suspicious transaction detection: a review. *Knowl. Inf. Syst.* **57** (2018). <https://doi.org/10.1007/s10115-017-1144-z>
5. Didimo, W., Liotta, G., Montecchiani, F., Palladino, P.: An advanced network visualization system for financial crime detection, pp. 203–210 (2011). <https://doi.org/10.1109/PACIFICVIS.2011.5742391>
6. Fronzetti Colladon, A., Remondi, E.: Using social network analysis to prevent money laundering. *Expert Syst. Appl.* **67**, 49–58 (2017). <https://doi.org/10.1016/j.eswa.2016.09.029>, <https://www.sciencedirect.com/science/article/pii/S0957417416305139>
7. Garcia-Bedoya, O., Granados, O., Burgos, J.: Ai against money laundering networks: the Colombian case. *J. Money Laundering Control* ahead-of-print (2020). <https://doi.org/10.1108/JMLC-04-2020-0033>
8. García, I.G., Mateos, A.: Use of social network analysis for tax control in Spain. *Hacienda Pública Española / Rev. Public Econ.* **239**(4), 159–197 (2021). <https://ideas.repec.org/a/hpe/journl/y2021v239i4p159-197.html>
9. Huang, D., Mu, D., Yang, L., Cai, X.: Codetect: Financial fraud detection with anomaly feature detection. *IEEE Access* **6**, 19161–19174 (2018). <https://doi.org/10.1109/ACCESS.2018.2816564>
10. Kauê Dal'Maso Peron, T., da Fontoura Costa, L., Rodrigues, F.A.: The structure and resilience of financial market networks. *Chaos Interdiscip. J. Nonlinear Sci.* **22**(1), 013117 (2012). <https://doi.org/10.1063/1.3683467>
11. Lillo, F., Moro, E., Vaglica, G., Mantegna, R.N.: Specialization and herding behavior of trading firms in a financial market. *New J. Phys.* **10**(4), 043019 (2008). <https://doi.org/10.1088/1367-2630/10/4/043019>
12. Lin, D., Wu, J., Yuan, Q., Zheng, Z.: Modeling and understanding Ethereum transaction records via a complex network approach. *IEEE Trans. Circ. Syst. II Express Briefs* **67**(11), 2737–2741 (2020). <https://doi.org/10.1109/TCSII.2020.2968376>
13. Liu, Z., Zhou, D., Zhu, Y., Gu, J., He, J.: Towards fine-grained temporal network representation via time-reinforced random walk. *Proc. AAAI Conf. Artif. Intell.* **34**(04), 4973–4980 (2020). <https://doi.org/10.1609/aaai.v34i04.5936>, <https://ojs.aaai.org/index.php/AAAI/article/view/5936>
14. López, L., A. Almendral, J., Sanjuán, M.A.: Complex networks and the www market. *Phys. Stat. Mech. Appl.* **324**(3), 754–758 (2003). [https://doi.org/10.1016/S0378-4371\(02\)01867-8](https://doi.org/10.1016/S0378-4371(02)01867-8), <https://www.sciencedirect.com/science/article/pii/S0378437102018678>

15. Pan, J.: Deep set classifier for financial forensics: an application to detect money laundering (2022). <https://doi.org/10.48550/ARXIV.2207.07863>, <https://arxiv.org/abs/2207.07863>
16. Papadimitriou, T., Gogas, P., Gkatzoglou, F.: The evolution of the cryptocurrencies market: a complex networks approach. *J. Comput. Appl. Math.* **376**, 112831 (2020). <https://doi.org/10.1016/j.cam.2020.112831>, <https://www.sciencedirect.com/science/article/pii/S0377042720301229>
17. Phua, C., Lee, V.C.S., Smith-Miles, K., Gayler, R.W.: A comprehensive survey of data mining-based fraud detection research. *CoRR* **abs/1009.6119** (2010). <http://arxiv.org/abs/1009.6119>
18. Rauch, J.E.: Business and social networks in international trade. *J. Econ. Lit.* **39**(4), 1177–1203 (2001). <https://doi.org/10.1257/jel.39.4.1177>, <https://www.aeaweb.org/articles?id=10.1257/jel.39.4.1177>
19. Serena, L., Ferretti, S., D'Angelo, G.: Cryptocurrencies activity as a complex network: analysis of transactions graphs. *Peer-to-Peer Netw. Appl.* **15** (2022). <https://doi.org/10.1007/s12083-021-01220-4>
20. Shun, J., Dhulipala, L., Blelloch, G.E.: Smaller and faster: parallel processing of compressed graphs with ligra+. In: 2015 Data Compression Conference, pp. 403–412 (2015). <https://doi.org/10.1109/DCC.2015.8>
21. Tarjan, R.: Depth-first search and linear graph algorithms. In: 12th Annual Symposium on Switching and Automata Theory (swat 1971), pp. 114–121 (1971). <https://doi.org/10.1109/SWAT.1971.10>
22. Vandewalle, N., Brisbois, F., Tordoir, X.: Non-random topology of stock markets. *Quant. Finan.* **1**(3), 372–374 (2001). <https://doi.org/10.1088/1469-7688/1/3/308>, <https://ideas.repec.org/a/taf/quantf/v1y2001i3p372-374.html>
23. Vilella, S., Lupi, A.T.E.C., Fornasiero, M., Moncalvo, D., Ricci, V., Ronchiadini, S., Ruffo, G.: Anomaly detection in cross-country money transfer temporal networks (2023)
24. Weber, M., Chen, J., Suzumura, T., Pareja, A., Ma, T., Kanezashi, H., Kaler, T., Leiserson, C.E., Schardl, T.B.: Scalable graph learning for anti-money laundering: a first look. *CoRR* **abs/1812.00076** (2018). <http://arxiv.org/abs/1812.00076>
25. Šubelj, L., Štefan Furlan, Bajec, M.: An expert system for detecting automobile insurance fraud using social network analysis. *Expert Syst. Appl.* **38**(1), 1039–1052 (2011). <https://doi.org/10.1016/j.eswa.2010.07.143>, <https://www.sciencedirect.com/science/article/pii/S0957417410007712>

Community Structure



Group Fairness Metrics for Community Detection Methods in Social Networks

Elze de Vink and Akrati Saxena^(✉)

Leiden Institute of Advanced Computer Science, Leiden University, Leiden,
The Netherlands
a.saxena@liacs.leidenuniv.nl

Abstract. Understanding community structure has played an essential role in explaining network evolution, as nodes join communities which connect further to form large-scale complex networks. In real-world networks, nodes are often organized into communities based on ethnicity, gender, race, or wealth, leading to structural biases and inequalities. Community detection (CD) methods use network structure and nodes' attributes to identify communities, and can produce biased outcomes if they fail to account for structural inequalities, especially affecting minority groups. In this work, we propose group fairness metrics (Φ_p^{F*}) to evaluate CD methods from a fairness perspective. We also conduct a comparative analysis of existing CD methods, focusing on the performance-fairness trade-off, to determine whether certain methods favor specific types of communities based on their size, density, or conductance. Our findings reveal that the trade-off varies significantly across methods, with no specific type of method consistently outperforming others. The proposed metrics and insights will help develop and evaluate fair and high performing CD methods.

Keywords: Community Detection · Algorithmic Fairness · Group Fairness Metrics

1 Introduction

In social networks, nodes are organized into communities. As Barabási defines, “In network science, we call a community a group of nodes that have a higher likelihood of connecting to each other than to nodes from other communities” [3]. Social networks have structural inequalities as the phenomenon of people joining communities is strongly driven by their ethnicity, gender, race, or wealth. These networks contain communities that vary in size, density, and their connectivity within the network based on human behavior. If such inequalities are not considered while designing Social Network Analysis (SNA) algorithms, they might lead to a biased outcome, especially for minorities [34]. Community detection (CD) methods use network structure and nodes' attributes to identify communities, and if they overlook structural inequalities, they cannot efficiently identify

small-size groups [12]. Misclassifying communities can further affect the fairness of other SNA methods. For example, fairness-aware methods proposed for influence maximization [10], influence minimization [31], link prediction [32, 33], and centrality ranking [38] use community membership to get the final fair outcome. It is crucial to consider structural inequalities while designing SNA methods to ensure fairness and mitigate bias for or against all users and groups, irrespective of their size or type.

Ghasemian et al. [12] compared 16 CD methods and found that the number of identified communities varies a lot across different methods. The authors did not examine the impact of these methods on different types of nodes or communities with varying sizes and densities. While many metrics exist to assess the quality of detected communities [6], there is no metric to evaluate the fairness of a method, particularly in relation to bias against minority groups. Fairness is not yet well defined and studied for CD methods, given that it has a vast literature [11].

In this work, we first propose fairness metrics to compute fairness for each community and use them further to propose group fairness metrics (Φ) for CD methods. Next, we perform a comparative analysis of existing community detection methods using the performance-fairness trade-off to study if high-performing methods are biased towards communities of different sizes, densities, or conductance. The size of a community is the total number of nodes, density is the ratio of the number of internal edges and total possible internal edges, and conductance is the fraction of the total edge volume that connects outside the community. We classify CD methods into six classes: (i) Optimization, (ii) Spectral, (iii) Propagation, (iv) Dynamics, (v) Representation Learning, and (vi) Miscellaneous. Experiments are performed on LFR and real-world networks. Results show that it is not the case that a specific class of methods always performs better. The performance-fairness trade-off is highly dependent on particular methods, and some of the best-performing methods identifying high-quality and fair communities include Infomap, RSC-V, RSC-K, Significance, and Walktrap.

To the best of our knowledge, this is the first work of its kind. Next, we discuss the proposed metrics, followed by experimental setup, results, and conclusion.

2 The Proposed Group Fairness Metric (Φ)

To compute the proposed fairness metrics, we begin by mapping the ground-truth and identified communities. Next, we compute the community-wise scores using the four proposed metrics, and finally, we compute the fairness of a CD method using these scores. The complete methodology is explained below.

2.1 Community Mapping

Let's assume that a given network $G(V, E)$ has m ground-truth communities, defined as $C = \{c_1, c_2, \dots, c_m\}$. We apply a CD method on G which returns a set of predicted communities P of size k defined as $P = \{p_1, p_2, \dots, p_k\}$. To evaluate the bias in a community detection method, it is essential to assess the fairness of

the detection process for each ground-truth community, i.e., the extent to which each community has been accurately identified. This is achieved by mapping the ground-truth communities to the corresponding predicted communities.

For mapping, perform the following steps until there is at least one ground-truth and one predicted community that is not-mapped.

1. Compute Jaccard-similarity for each pair of ground-truth and predicted communities, as follows:

$$J(c_i, p_j) = \frac{|c_i \cap p_j|}{|c_i \cup p_j|}, \forall i \& j$$

2. A pair having the highest similarity score is chosen and these ground-truth and predicted communities are mapped. In case of a tie, a ground-truth community is randomly chosen and mapped.

After this process, if there is an unmapped ground-truth community, it is marked as completely misclassified and will be mapped with an empty set.

2.2 Community-Wise Fairness Metrics

We propose the following metrics to evaluate how effectively a ground-truth community is represented by its corresponding predicted community, taking into account both the community's nodes and edges.

1. **Fraction of Correctly Classified Nodes (FCCN):** A simple metric to assess how well a predicted community (p_j) represents the ground-truth community (c_i) is the fraction of ground-truth nodes present in the predicted community. The FCCN is computed as follows:

$$FCCN(c_i, p_j) = \frac{|c_i \cap p_j|}{|c_i|} \quad (1)$$

2. **F1 Score:** The FCCN focuses mainly on common nodes in the ground-truth and predicted communities, but it does not account for extra nodes present in the predicted community. To address this, we propose the F1 score (inspired by the F1 [7] score used in machine learning), i.e., calculated as:

$$F1(c_i, p_j) = \frac{2|c_i \cap p_j|}{|c_i| + |p_j|} \quad (2)$$

3. **Fraction of Correctly Classified Edges (FCCE):** Community structure is primarily driven by edges, and a thorough evaluation of a CD method's performance should include an analysis of community edges. The FCCE metric quantifies the number of edges from the ground-truth community that are present in the mapped predicted community. It is calculated as:

$$FCCE(c_i, p_j) = \frac{|E(c_i) \cap E(p_j)|}{|E(c_i)|} \quad (3)$$

where $E(c_i)$ is the set of intra-community edges of a community c_i and it is computed as: $E(c_i) = \{(u, v) \in E \mid u \in c_i \text{ and } v \in c_i\}$.

2.3 Group Fairness Metric (Φ)

Our aim is to investigate whether a given CD method favors communities of specific properties based on their size, density, or conductance, using the aforementioned community-wise fairness metrics. A straightforward approach is to plot these metrics for each community against their corresponding attribute values. For example, Fig. 1 shows FCCN vs. normalized community size for a dummy network, showing that larger communities are more accurately identified than smaller ones. To ensure comparability across different networks, we normalize community attribute using min-max scaling so that they range from 0 to 1.

To compute the group fairness metric (Φ), we fit a linear line using least squares approximation [15] on the proposed community-wise fairness metrics (represented by F^*) versus normalized community property; e.g., in Fig. 1, the dashed line is the best fit linear line on the community-wise fairness metric (FCCN) versus normalized community size. The fairness metric is measured using the slope of the line, which ranges from $(-1, 1)$. The regression line provides Δx and Δy , and to compute the angle θ we use the arctangent: $\arctan\left(\frac{\Delta y}{\Delta x}\right)$. This is further multiplied by $\frac{2}{\pi}$ to get the fairness metric value in $(-1, 1)$ range as the arctangent angle is in radians within $(-\frac{\pi}{2}, \frac{\pi}{2})$. Finally, the fairness of a CD method, with respect to a per-community fairness metric (F^*) and community property (p) is computed as:

$$\Phi_p^{F^*} = \frac{2}{\pi} \arctan\left(\frac{\Delta y}{\Delta x}\right) = \frac{2}{\pi} \arctan(\Delta y) \quad (4)$$

The x-axis values are normalized using min-max scaling, so $\Delta x = 1$ and can be removed. Using the example from Fig. 1, given the $\Delta y = 0.70$, we calculate its fairness score as follows: $\Phi_{size}^{FCCN} = \frac{2}{\pi} \cdot \arctan(0.70) \approx 0.39$.

The proposed metric ($\Phi_p^{F^*}$) ranges $(-1, 1)$, where 0 (a straight best fit line) indicates fair result, meaning all communities are identified either equally well or equally poorly. Negative and positive values indicate that the method favors communities with lower and higher property values (p), respectively.

2.4 Metric Behavior

To study the behavior of the proposed metric, we create a homophilic network (homophily factor 0.9) using HICH-BA model [31] having two communities: majority (70 nodes) and minority (40 nodes), and ~ 900 edges. Initially, predicted communities are the same as the ground truth. We then swap nodes between

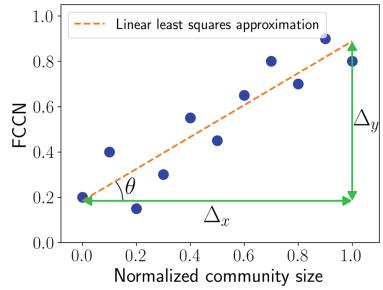


Fig. 1. FCCN vs. normalized community size for each community on a dummy network. The best fit line shows the trend of a CD method.

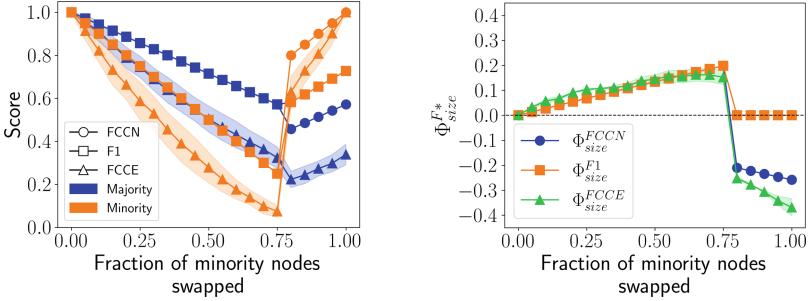


Fig. 2. Analyzing behavior of community-wise fairness metric and group fairness on a network having minority and majority community.

minority and majority from 0 to 40, and map the resulted predicted and ground-truth communities. Figure 2 shows community-wise fairness (left) and group-fairness (right) scores with respect to swapped nodes. FCCE score can vary based on which nodes are moved, and therefore, we show the average and deviation over 20 iterations. FCCE is lower than FCCN and F1 due to homophilic characteristic of the network.

As we switch an equal number of nodes between minority and majority, the community-wise fairness is always lower for minority than majority till a certain point (~ 0.75), but after that, the mapping switches between majority and minority and the fairness for minority increases as compared to the majority. Similar insights are conveyed by Φ_{size}^{F*} that initially favors the majority, but once the mapping is switched, the method either favors the minority or remains unbiased. The Φ_{size}^{F1} is fair after that point, as it also considers the nodes from the predicted communities that are not present in the ground-truth. Just to clarify, this example creates extreme conditions that might not happen in real-world.

The proposed fairness metric (Φ) offers an understanding of biases, ensuring that CD methods are evaluated not just on their overall performance but also on their fairness towards communities of different sizes, densities, and conductance.

3 Experimental Setup

3.1 Community Detection Methods

We compare 24 CD methods (refer Table 1) falling under the following six classes.

1. **Optimization** methods aim to optimize a quality function that describes the quality of the given partition. All methods considered in this category optimize modularity [21] except the Significance [36].
2. **Dynamics** methods infer communities by analyzing traversal on the network, typically through random walks as they tend to remain within communities due to their dense connectivity compared to rest of the network.
3. **Spectral** methods identify communities using the spectral properties of matrices representing the network, such as adjacency or Laplacian matrix [11].

Table 1. Overview of CD methods used in experimentation from 6 classes.

Optimization	Spectral Properties	Representational
<ul style="list-style-type: none"> • Clauset-Newman-Moore Algorithm (CNM) [8] • Combo [35] • Leiden [37] • Louvain [4] • Paris [5] • Reichardt-Bornholdt - configuration null model (RB-C) [29] • Reichardt-Bornholdt - Erds-Rnyi null model (RB-ER) [29] • Significance [36] 	<ul style="list-style-type: none"> • Eigenvector [20] • Regularized Spectral Clustering with k-means (RSC-K) [39] • RSC sklearn Spectral Embedding (RCS-SSE) [39] • RSC - Vanilla (RSC-V) [39] • Spectral Clustering [16] 	<ul style="list-style-type: none"> • Deepwalk [26] • Fairwalk [28] • Node2Vec [14]
Dynamics	Propagation	Miscellaneous
<ul style="list-style-type: none"> • Infomap [30] • Spinglass [29] • Walktrap [27] 	<ul style="list-style-type: none"> • Fluid [23] • Label Propagation [9] 	<ul style="list-style-type: none"> • Expectation-Maximization (EM) [22] • Stochastic Block Model (SBM) [24] • SBM - Nested [25]

4. **Propagation** methods utilize the iterative propagation of community labels, where each node's label is updated based on its neighbors' labels, until a stable configuration is reached representing communities in the network.
5. **Representational** Learning methods first generate a network embedding, and then apply clustering algorithm to create partition of the network [2].
6. **Miscellaneous** category contains the methods that do not fit to any of the previously mentioned category due to their underlying working mechanism.

We use CDlib implementation (<https://cdlib.readthedocs.io/>) with default parameters for all CD methods. The code for our experiments and metric computations is available at:

<https://github.com/akratiiet/Group-Fairness-Metrics-for-Community-Detection>.

3.2 Datasets

Synthetic Networks: We use LFR benchmark model [18] to generate networks of varying mixing parameter μ having 10,000 nodes, and the power law exponents of degree and community size to be 2 and 2.5, respectively.

Real-world Networks: These are described in Table 2.

Table 2. Real-world datasets. $|C|$: total number of communities, $|c_{max}|$: size of largest community, $|c_{min}|$: size of smallest community.

Dataset	$ V $	$ E $	Avg deg	Max deg	$ C $	$ c_{min} $	$ c_{max} $
Polbooks [17]	105	441	8.40	25	3	13	49
Football [13]	115	613	10.66	12	12	5	13
Eu-core [19]	986	16,687	33.85	347	42	1	107

4 Results

We study the fairness of CD methods with respect to community size, density and conductance versus the quality of identified communities, i.e., measured using the Normalized Mutual Information (NMI) [1].

4.1 Group Fairness-Performance Trade-Off Versus Community Size

We first study how well different community detection methods identify communities of different sizes. In Fig. 3, we plot NMI versus three group fairness (Φ_{size}^{FCCN} , Φ_{size}^{F1} , and Φ_{size}^{FCCE}) for LFR networks having mixing parameters $\mu = 0.2$, 0.4 , and 0.6 . To perform our experiments, we create 10 LFR networks for each setting and apply CD methods (stochastic CD methods are executed ten times) to compute the average and standard deviation.

For $\mu = 0.2$, all CD methods, except EM, Paris, SBM-nested, and RSC-SSE, favor large-size communities for all fairness metrics F_* . The methods which have high fairness and identify good quality communities include RSC-K, RSC-V, RSC-SSE, Infomap, Fluid, Walktrap, and Significance (NMI ~ 1). SBM-Nested identifies smaller communities (negative Φ_{size}^{F*}) well as this is a hierarchical method, and at lower levels, it perfectly discovers smaller communities. When μ is increased to 0.4 , the SBM-Nested method still favors small communities. However, when communities are highly connected with each other ($\mu = 0.6$), no method identifies small groups well. All methods identify larger communities better, and the fair methods have very low NMI as they identify communities of all sizes equally bad. One important point to note is that across all networks, there is no correlation between the fairness and performance of CD methods.

4.2 Fairness Performance Trade-Off Vs. Community Density

We further study the performance of CD methods for communities of different densities. In Fig. 4, we observe that when the mixing parameter is low, CD methods identify sparse communities better than denser communities. However, this behavior changes as the mixing parameter is increased and the nodes have

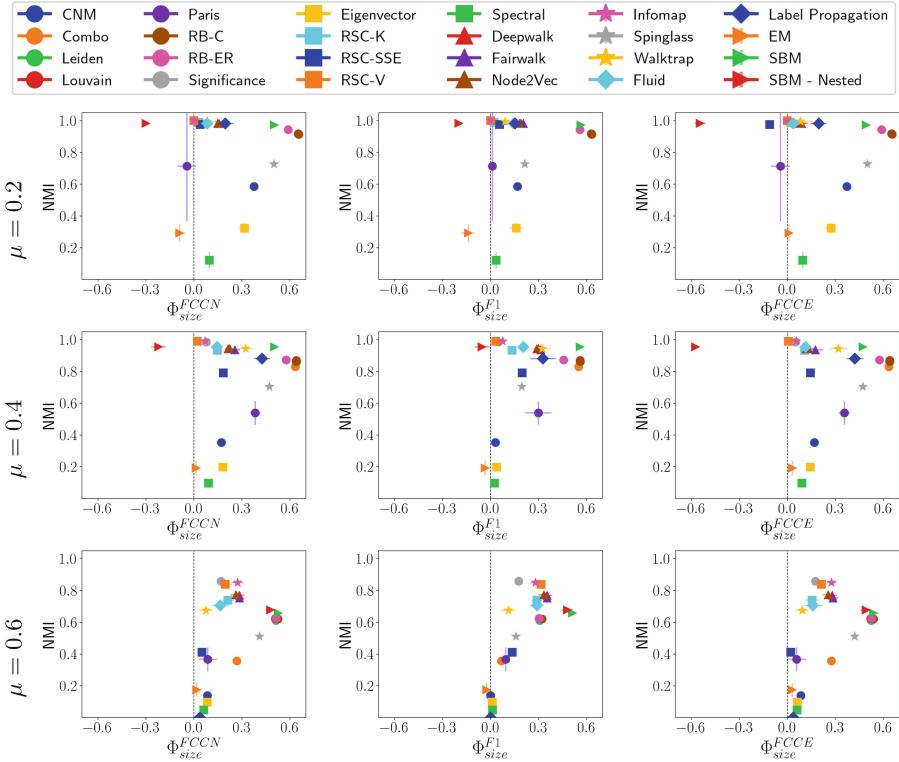


Fig. 3. NMI vs. fairness of community detection methods with respect to community size for LFR networks of 10,000 nodes having different μ values.

more inter-community edges. With $\mu = 0.4$, and 0.6 , Leiden, Louvain, RB-C, RB-ER, and Spinglass CD methods have similarly high NMI and negative Φ_{size}^{F*} . All these methods predict fewer communities than the ground-truth, and predicted communities are mapped with low-density ground-truth communities. At $\mu = 0.6$, methods with high NMI also predict dense communities better. For example, the Significance predicts a lot more communities than the ground-truth, and it identifies denser communities but splits up sparser communities in the prediction.

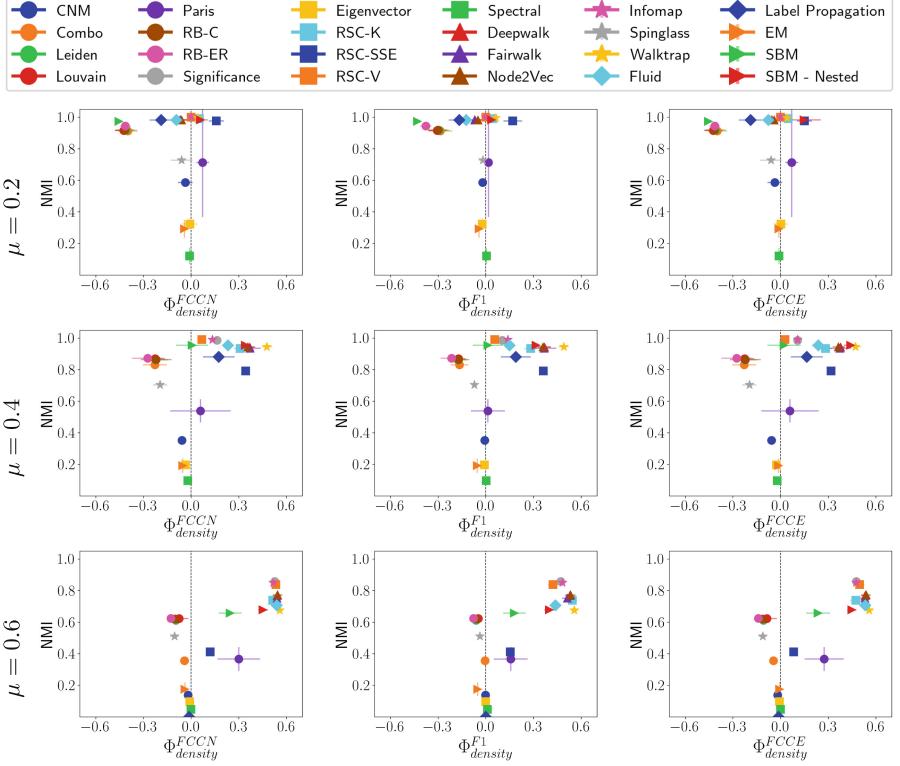


Fig. 4. NMI vs. fairness with respect to community densities on LFR networks.

4.3 Fairness Performance Trade-Off Vs. Community Conductance

Fig. 5 presents NMI vs. fairness with respect to community conductance. Most CD methods favor communities with lower conductance, meaning communities that have more separation. Across all fairness metrics, the bias increases with μ , leading to lower $\Phi_{conductance}^{F*}$ values. With medium mixing ($\mu = 0.4$), RSC-V, SBM-Nested, Infomap, and Significance are both fair and identify high-quality communities. At $\mu = 0.6$, $\Phi_{conductance}^{F*}$ and NMI have linear relations, where fair methods have poor predictions and better predictions are highly biased.

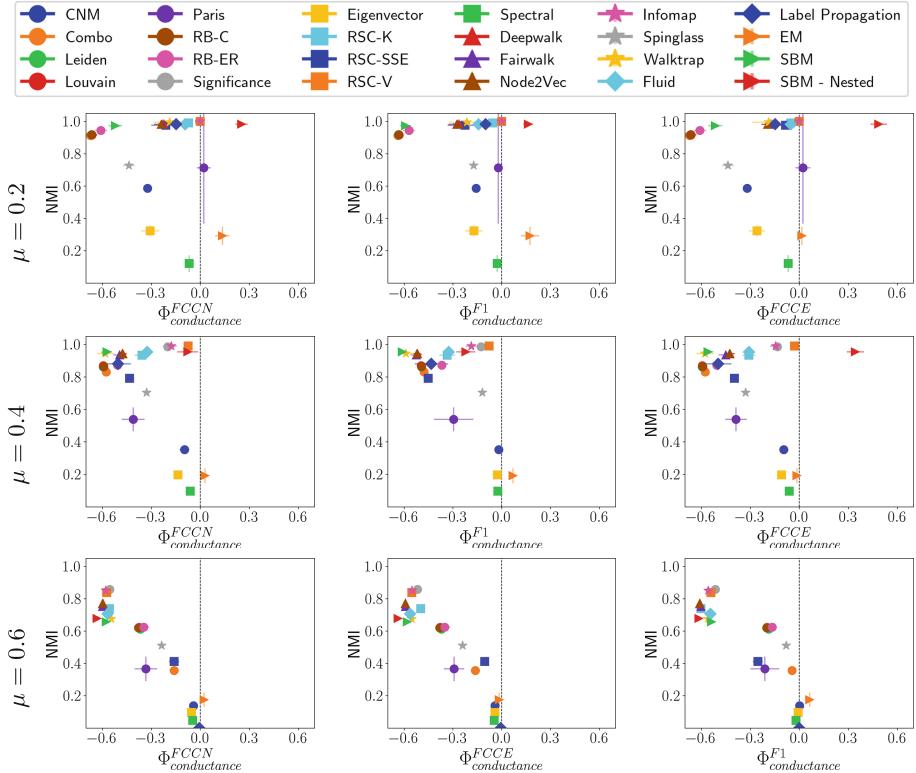


Fig. 5. NMI vs. fairness with respect to conductance on LFR networks.

4.4 Fairness Metric Versus Other Quality Evaluation Metrics

We also compare fairness with three additional quality evaluation metrics: (i) Adjusted Rand Index (ARI), (ii) Reduced Mutual Information (RMI), and (iii) Normalized F1 score (NF1) [6]. Figure 6 shows the results. The aim of this analysis was to identify CD methods which are both fair and perform well across various metrics, so that they can be recommended for broader use. Additionally, understanding the working dynamics of these methods will help in designing CD methods with high fairness-performance trade-off. From this analysis, the stand-out CD methods include: RSC-V, RSC-K, Walktrap, Infomap, and Significance.

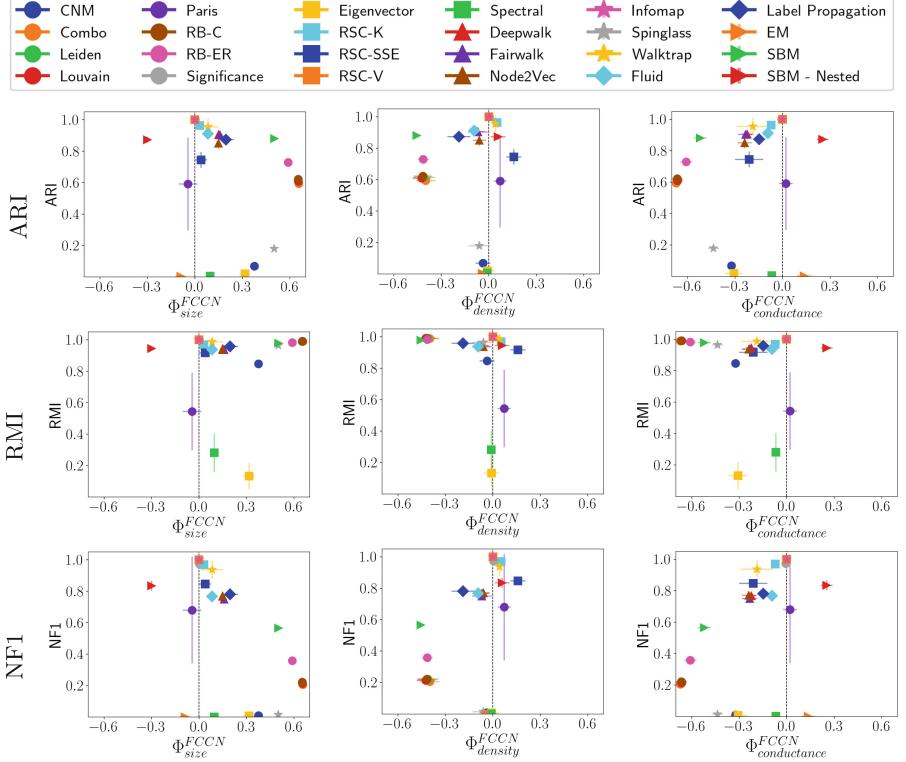


Fig. 6. ARI, RMI, and NF1 vs. Φ on LFR networks having $\mu = 0.2$.

4.5 Fairness Analysis on Real-World Networks

Figure 7 presents NMI vs. Φ_{size}^{F*} for real-world networks. All methods, except EM, tend to favor larger groups, with consistent results across various fairness metrics. It is important to note that no single method is universally fair across all datasets, though Significance consistently performs well.

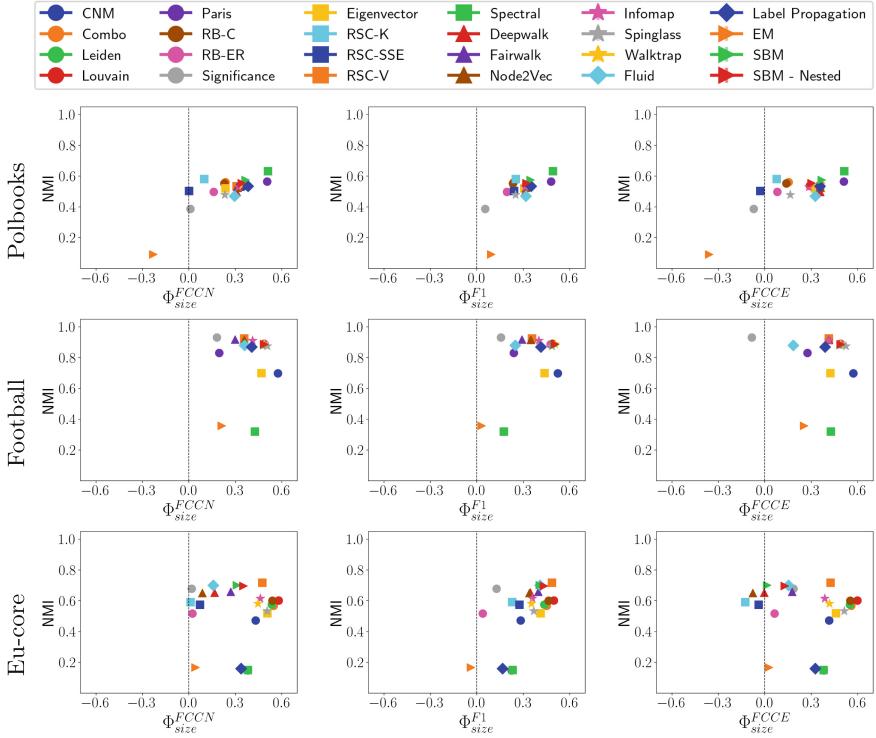


Fig. 7. NMI vs. fairness of CD methods on real-world networks.

5 Conclusion

In real-world networks, communities vary in size, density, and connectivity due to how people organize and communicate. In this work, we first introduce community-wise fairness metrics (FCCN, F1, and FCCE) and use them to propose group fairness metrics (Φ) to evaluate community detection methods with respect to communities' properties. We compare 24 community detection methods from six different classes on LFR benchmark and real-world networks. Our analysis reveals that no single class of methods consistently outperforms others; instead, performance varies for methods. However, some general trends emerge: modularity methods are highly biased towards large and sparse (except Paris and Significance), and low conductance groups. Representational and label propagation methods are biased toward large, low-density (for low μ), and low-conductance groups. Dynamic and spectral methods have a huge variation in their fairness and quality evaluation. For $\mu = 0.2$, the best methods which are fair in all aspects and have high quality scores (NMI, ARI, RMI, and NF1) include Significance, RSC-K, RSC-V, Infomap, and Walktrap. However, Walktrap is biased towards lower conductance. As μ increases, Fluid, SBM-Nested,

SBM, and Label Propagation also perform better in all aspects, though they do not agree on RMI.

This study offers valuable insights that can guide the design of fair community detection methods. In the future, we aim to propose individual fairness metrics for community detection to assess whether similar nodes are treated fairly.

References

1. Ana, L.F., Jain, A.K.: Robust data clustering. In: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2. IEEE (2003)
2. Arya, A., Pandey, P.K., Saxena, A.: Node classification using deep learning in social networks. In: Deep Learning for Social Media Data Analytics, pp. 3–26. Springer (2022). https://doi.org/10.1007/978-3-031-10869-3_1
3. Barabási, A.L.: Network science book. *Netw. Sci.* **625**, 066111 (2014)
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**(10), P10,008 (2008)
5. Bonald, T., Charpentier, B., Galland, A., Hollocou, A.: Hierarchical graph clustering using node pair sampling. In: MLG (2018)
6. Chakraborty, T., Dalmia, A., Mukherjee, A., Ganguly, N.: Metrics for community analysis: a survey. *ACM Comput. Surv. (CSUR)* **50**(4), 1–37 (2017)
7. Chinchor, N.: Muc-4 evaluation metrics. In: Proceedings of the 4th Conference on Message Understanding, pp. 22–29. USA (1992)
8. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. *Phys. Rev. E* **70**(6), 066111 (2004)
9. Cordasco, G., Gargano, L.: Community detection via semi-synchronous label propagation algorithms. In: IEEE International Workshop on BASNA, pp. 1–8 (2010)
10. Farnad, G., Babaki, B., Gendreau, M.: A unifying framework for fairness-aware influence maximization. In: Companion Proceedings of the Web Conference 2020, pp. 714–722 (2020)
11. Fortunato, S., Hric, D.: Community detection in networks: a user guide. *Phys. Rep.* **659**, 1–44 (2016)
12. Ghasemian, A., Hosseini-mardi, H., Clauset, A.: Evaluating overfit and underfit in models of network community structure. *IEEE TKDE* **32**(9), 1722–1735 (2019)
13. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. *Proc. Natl. Acad. Sci.* **99**(12), 7821–7826 (2002)
14. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference, pp. 855–864 (2016)
15. 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>
16. Higham, D.J., Kalna, G., Kibble, M.: Spectral clustering and its use in bioinformatics. *J. Comput. Appl. Math.* **204**(1), 25–37 (2007)
17. Krebs, V.: Unpublished. <http://www.orgnet.com/>
18. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. *Phys. Rev. E* **78**(4), 046110 (2008)
19. Leskovec, J., Krevl, A.: SNAP Datasets: stanford large network dataset collection. <http://snap.stanford.edu/data> (2014)
20. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* **74**(3) (2006)

21. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. *Phys. Rev. E* **69**(2), 026113 (2004)
22. Newman, M.E.J., Leicht, E.A.: Mixture models and exploratory analysis in networks. *Proc. Natl. Acad. Sci.* **104**(23), 9564–9569 (2007)
23. Parés, F., et al.: Fluid communities: a competitive, scalable and diverse community detection algorithm (2017). <https://arxiv.org/abs/1703.09307>
24. Peixoto, T.P.: Efficient monte carlo and greedy heuristic for the inference of stochastic block models. *Phys. Rev. E* **89**(1), 012804 (2014)
25. Peixoto, T.P.: Hierarchical block structures and high-resolution model selection in large networks. *Phys. Rev. X* **4**(1), 011047 (2014)
26. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
27. Pons, P., Latapy, M.: Computing communities in large networks using random walks. *J. Graph Alg. Appl.* **10**(2), 191–218 (2006)
28. Rahman, T., Surma, B., Backes, M., Zhang, Y.: Fairwalk: towards fair graph embedding. In: Proceedings of the 28th IJCAI, pp. 3289–3295 (2019)
29. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. *Phys. Rev. E* **74**(1), 016110 (2006)
30. Rosvall, M., Bergstrom, C.T.: Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci.* **105**(4), 1118–1123 (2008)
31. Saxena, A., Bierbooms, C., Pechenizkiy, M.: Fairness-aware fake news mitigation using counter information propagation. *Appl. Intell.* **53**, 1–22 (2023)
32. Saxena, A., Fletcher, G., Pechenizkiy, M.: HM-EIICT: Fairness-aware link prediction in complex networks using community information. *J. Comb. Optim.*, 1–18 (2021). <https://doi.org/10.1007/s10878-021-00788-0>
33. Saxena, A., Fletcher, G., Pechenizkiy, M.: Nodesim: node similarity based network embedding for diverse link prediction. *EPJ Data Sci.* **11**(1), 24 (2022)
34. Saxena, A., Fletcher, G., Pechenizkiy, M.: FairSNA: algorithmic fairness in social network analysis. *ACM Comput. Surv.* **56**(8), 1–45 (2024)
35. Sobolevsky, S., Campari, R., Belyi, A., Ratti, C.: General optimization technique for high-quality community detection in complex networks. *Phys. Rev. E* **90**(1), 012811 (2014)
36. Traag, V.A., Krings, G., Van Dooren, P.: Significant scales in community structure. *Sci. Rep.* **3**(1) (2013)
37. Traag, V.A., Waltman, L., van Eck, N.J.: From louvain to leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 5233 (2019)
38. Tsoutsouliuklis, S., Pitoura, E., Tsaparas, P., Kleftakis, I., Mamoulis, N.: Fairness-aware link analysis. *CoRR* abs/2005.14431 (2020)
39. Zhang, Y., Rohe, K.: Understanding regularized spectral clustering via graph conductance (2018). <https://arxiv.org/abs/1806.01468>



FastEnsemble: A New Scalable Ensemble Clustering Method

Yasamin Tabatabaei, Eleanor Wedell, Minhyuk Park,
and Tandy Warnow

Siebel School of Computing and Data Science, University of Illinois
Urbana-Champaign, Urbana, IL 61801, USA
warnow@illinois.edu

Abstract. Many community detection algorithms are stochastic in nature, and their output can vary based on different input parameters and random seeds. Consensus clustering methods, such as FastConsensus and ECG, combine clusterings from multiple runs of the same clustering algorithm, in order to improve stability and accuracy. In this study we present a new consensus clustering method, FastEnsemble, and show that it provides advantages over both FastConsensus and ECG. Furthermore, FastEnsemble is designed for use with any clustering method, and we show results using FastEnsemble with Leiden optimizing modularity or the constant Potts model. FastEnsemble is available in Github.

Keywords: Consensus Clustering · Ensemble Clustering · Community Detection · Resolution Limit · Unclusterable networks

1 Introduction

Community detection, also known as clustering, is the problem of dividing the nodes of a given network into disjoint subsets so that each subset displays features of a community, such as increased edge density, separability from the rest of the network, strong internal edge connectivity, etc. Several community detection methods have been developed in the past few decades [5, 26], some of which employ randomness in different ways to produce a desirable community. For example, some methods are based on modularity [16] or the constant Potts model (CPM) [21], which are NP-hard optimization problems. The popular software Louvain [2] and Leiden [25] provide effective heuristics for modularity, and Leiden also provides an effective heuristic for optimizing under the CPM criterion.

One of the difficulties in using clustering methods that optimize modularity or CPM is the variability in the outputs, as searches for NP-hard problems based on different starting points, random seeds, or tie-breaking rules often produce different results [3]. To address these challenges, consensus (or ensemble) clustering approaches have been proposed to combine information from different runs in order to extract a reliable clustering; these consensus approaches can lead

to more robust and stable partitions and improve the accuracy of the output clustering [8, 13, 23].

A class of consensus clustering methods, introduced in [13], take a network G as input and run a clustering algorithm (such as Leiden with different random seeds) on it np times to get np different partitions. Next, they summarize the information from these partitions (i.e., nodes that are frequently co-clustered together) into a co-classification matrix and create a new weighted network G' from this matrix. The new network is again given to the clustering algorithm np times to produce np partitions. This procedure is continued until the network G' converges to a stationary network after multiple iterations. Various flavors of this consensus approach have been proposed in the literature [8, 13, 23].

Scalability is an issue for these approaches, as building the co-classification matrix is itself computationally intensive. FastConsensus [23] is a recent method that tries to address the scalability issue by using a sampling technique, where the co-classification matrix is only computed for a subset of node pairs. Another recent and promising consensus method is Ensemble Clustering for Graphs (ECG) [20], but it uses a somewhat simpler technique than FastConsensus, which suggests that it should be even faster.

We report on a new consensus clustering method, FastEnsemble. The algorithmic technique in FastEnsemble is very simple, omitting much of the sophisticated technicalities of FastConsensus so that it can scale to very large networks. FastEnsemble is very similar in design to ECG, but can be used with any given clustering method, whereas ECG is designed for use with Louvain.

In this study we evaluate FastEnsemble for use in modularity-optimization and CPM-optimization on large synthetic networks generated using the LFR [14] software with up to $\sim 3.8M$ nodes, and we compare FastEnsemble to FastConsensus and ECG with respect to accuracy and runtime.

The rest of this paper is organized as follows. We describe FastEnsemble in Sect. 2. In Sect. 3, we describe the performance study, including datasets, methods, and evaluation procedure. Section 4 includes the details of the experiments and their results. We discuss the trends in Sect. 5 and conclude in Sect. 6 with a summary and directions for future work. Due to space limitations, some details and results are provided in the Supplementary Materials [22].

2 Fast Ensemble Clustering

We have developed a basic algorithmic structure that can be used with one or a weighted combination of clustering paradigms, and implemented it for use with Leiden optimizing CPM (“Leiden-CPM”), Leiden optimizing modularity (“Leiden-mod”), and Louvain. In this study we use it in two ways, as we now describe.

In its simplest form, FastEnsemble uses three main parameters: the clustering method, the number of partitions np , and the threshold t . Given an input network N , FastEnsemble uses the specified clustering method to generate np partitions of N , and then builds a new network on the same node and edge

set but with the edges weighted by the fraction of the partitions in which the endpoints are in the same cluster. If a given edge has weight less than t , then the edge is removed from the network; hence the new network can have fewer edges than the original network. The new weighted network is then clustered just once more using the selected clustering method. Increasing the number np of partitions may improve accuracy and stability, but at a computational cost; therefore, for very large networks, np defaults to 10. In Experiment 1 we determine a default setting for the parameter t . However, setting $t = 1$ produces a special case that we refer to as the *Strict Consensus Clustering*.

3 Performance Study

3.1 Networks

We used a selected set of synthetic networks, some available from prior studies, and some generated for this study. Table 1 provides a summary of empirical statistics, including network size and mixing parameters [15] for these networks. Note that networks that have mixing parameters of 0.5 or larger are challenging to cluster while networks with much smaller mixing parameters are generally easy to cluster [9,14].

Training Experiments. For the training experiment, we generated LFR networks using parameters taken from similar networks used in [23], but with the exponent for the cluster size distribution modified to better fit real-world networks (see Supplementary Materials [22] for further discussion). Each of these synthetic networks has 10,000 nodes with calculated mixing parameter values that vary between 0.196–0.978 (note that the model mixing parameters, which are used to generate the networks, are drawn from 0.1, 0.2, . . . , 0.9, but the resultant mixing parameters are different).

Testing Experiments. We use 27 LFR [14] networks from [17]; these were generated using parameters from Leiden-mod and Leiden-CPM clusterings of five real-world networks: cit_hepph, the Curated Exosome Network (CEN), Open Citations (OC), wiki_topcats, and cit_patents. Two of these LFR networks had a substantial percentage of ground truth clusters that were disconnected, and were not included in the experiment in [17], and are also not used in this study, and LFR failed to return a network for one model condition. We also use ring-of-cliques networks [6], Erdős-Rényi graphs [4], and graphs formed by merging Erdős-Rényi graphs with LFR networks.

3.2 Methods

We include FastEnsemble, ECG, FastConsensus, and Leiden, each for modularity optimization. In the final experiment we also examine FastEnsemble and Leiden for CPM-optimization.

Table 1. Empirical statistics of the synthetic networks in the study. We report number of nodes, number of edges, and mixing parameter ranges for each collection of networks. For Erdős-Rényi graphs, the “ground truth” clustering has every node in its own cluster. The last five rows each represent up to six different networks, based on a different clustering of the specified real-world network.

Network	Expt.	nodes	edges	mixing param.	publ.
LFR Training	1,2	10,000	58272-59584	0.196-0.978	this study
Erdős-Rényi	3	1000	470-50,025	1.0	this study
Erdős-Rényi+ LFR	3	2000	4776-53,917	0.486-0.572	this study
Ring-of-Cliques	4	90-10,000	4140-460,000	0.02	this study
LFR cit_hepph	2,5	34,546	$\sim 431K$	0.086-0.781	[17]
LFR wiki_topcats	2,5	1,791,489	$\sim 24M$	0.199-0.793	[17]
LFR cen	2,5	3,000,000	$\sim 21M$	0.180-0.646	[17]
LFR OC	2,5	3,000,000	$\sim 55M$	0.129-0.871	[17]
LFR cit_patents	2,5	3,774,768	$\sim 16M$	0.114-0.807	[17]

The closest method to FastEnsemble is ECG: both have two steps, where they first generate multiple partitions using a clustering algorithm with different random seeds and then combine these into a final clustering by looking at the fraction of times each two nodes are co-clustered in the set of partitions and then applying a clustering algorithm on this weighted network. However, there are two main differences between ECG and FastEnsemble. First, ECG is limited to use with the Louvain algorithm (which optimizes modularity), whereas FastEnsemble is generically designed to combine partitions from different clustering algorithms. Here, when optimizing modularity, we studied FastEnsemble for use with Leiden rather than Louvain, due to the improved connectivity of Leiden clusters [25]. Second, in ECG, if an edge is not in a 2-core in the original graph, then the edge automatically gets the pre-defined minimum edge weight, which is a small positive value. In contrast, in FastEnsemble, a pair of nodes that are co-clustered together in a large fraction of the computed partitions will have a high weight, as the inclusion in a 2-core is not relevant to the weight. As a consequence, ECG will be likely to not co-cluster pairs of nodes in the returned consensus clustering when they are not in 2-cores in the input network.

3.3 Evaluation Criteria

We report accuracy on networks with known ground truth using Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) as implemented by the Scikit-learn [18] library. We also report the F1-score.

We also compute false negative and false positive error rates, defined as follows. By considering true and estimated clusterings each as an equivalence relation, and thus defined by a set of pairs (where (x, y) is in the relation if and only if nodes x and y are in the same cluster), we can define false negatives (pairs

that are in the true clustering but missing in the estimated clustering), false positives (pairs that are in the estimated clustering but not in the true clustering), true positives (pairs that are in both the true and estimated clusterings), and true negatives (pairs that are in neither clustering). Using these, we report the False Negative Rate (FNR) and False Positive Rate (FPR), given by $\frac{fn}{fn+tn}$ and $\frac{fp}{fp+tn}$ respectively, where fn denotes the number of false negatives, fp denotes the number of false positives, tp denotes the number of true positives, and tn denotes the number of true negatives.

3.4 Experiments

We perform five experiments, described below. In each case, synthetic networks were used and accuracy was evaluated in comparison to the ground truth clusterings. Except where indicated, all analyses were given four hours of runtime and 64GB of memory on the University of Illinois Campus Cluster; failures to complete within that time limit were noted.

- Experiment 1: We set the default for the threshold parameter in FastEnsemble
- Experiment 2: We evaluate modularity-based pipelines with respect to both accuracy and scalability on synthetic networks with $\sim 10K$ to $\sim 3.8M$ nodes.
- Experiment 3: We evaluate clusterings on networks that are either entirely or partially Erdős-Rényi graphs.
- Experiment 4: We evaluate robustness to the resolution limit on ring-of-cliques networks with up to $100K$ nodes.
- Experiment 5: We evaluate the accuracy of FastEnsemble on large synthetic LFR networks (up to $\sim 3.8M$ nodes) from [17].

Experiments 1, 2, and 5 use networks with a range of mixing parameters, Experiment 3 uses networks with moderate to high mixing parameters, and Experiment 4 uses networks with extremely low mixing parameters (Table 1).

4 Results

4.1 Experiment 1: Training Experiment

In this first experiment we set the default value for the threshold parameter t , so that edges with support below t are removed from the network. In Fig. 1 (left), we compare results for only three threshold values: $t = 0.2, 0.5$, and 0.8 . We see that overall the best accuracy across all the networks is obtained using $t = 0.8$. In Fig. 1 (right), we show results for just one of the networks with the mixing parameter 0.5 , but allowing $t = 0.1, 0.2, \dots, 1.0$. On this model condition, values for t between 0.7 and 0.9 produce the best accuracy. Based on this experiment, we set $t = 0.8$ as the default.

Note the impact of the mixing parameter: while accuracy is very high for networks with the lowest mixing parameter, it quickly drops as the mixing parameter increases. This reflects the discussion in [9, 14].

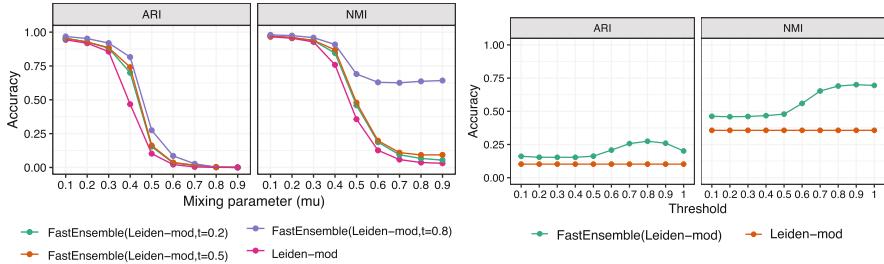


Fig. 1. Experiment 1: Setting the default value for t in FastEnsemble. Left: using LFR networks with 10,000 nodes and varying the model mixing parameter (x-axis), we report ARI and NMI accuracy for both Leiden-Mod and FastEnsemble using three different threshold values. Right: using the LFR network with model mixing parameter 0.5, we report ARI and NMI accuracy for threshold values $t = 0.1, 0.2, \dots, 1.0$. The LFR graphs have 10,000 nodes and parameters $\tau_1 = 3$, $\tau_2 = 1.5$, $d_{avg} = 10$ and $c_{min} = 10$. Based on this experiment, we set $t = 0.8$ as the default for FastEnsemble.

4.2 Experiment 2: Accuracy and Scalability of Clustering Pipelines

We use two collections of synthetic networks in this experiment: the LFR networks based on modularity clusterings of large real-world networks from [17] (which go up to $\sim 3.8M$ nodes) and the training datasets used in Experiment 1.

Results on LFR networks from [17]: For all the methods, the only network they completed on within four hours was cit_hepph, the smallest network with only $\sim 34K$ nodes, and all three methods had nearly perfect ARI and NMI scores (Table 2 and Supplementary Materials). We then allowed all three methods to run for up to 48 h on the four remaining networks. FastEnsemble completed on all the remaining networks, using between 7 and 28 h. FastConsensus completed only on one of these networks (using 14.5 hrs), where it had excellent accuracy that was somewhat better than FastEnsemble. ECG completed on all networks, using from 6 hrs to 36 hrs on each. Thus, FastEnsemble and ECG were both faster than FastConsensus. ECG was less accurate than FastEnsemble on three networks and more accurate on one network, where both ECG and FastEnsemble had very high ARI/NMI scores, indicating that the network was relatively easy to cluster (Table 2 and Supplementary Materials).

Results on the training networks: Fig 2 shows that accuracy decreases for all methods as the model mixing parameter increases. The method with the best accuracy for the two smallest model mixing parameters (0.1 and 0.2) is ECG, but then FastEnsemble has the best accuracy for the larger model mixing parameters. Both ECG and FastEnsemble consistently match or improve on Leiden-mod. FastConsensus improves on Leiden-mod for the large mixing parameter values but is less accurate than Leiden-mod for the smaller mixing parameters.

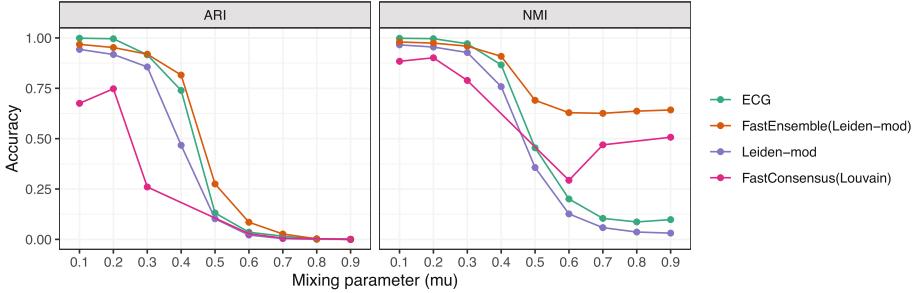


Fig. 2. Experiment 2: Evaluating consensus clustering pipelines. Results are shown on the training data from Experiment 1; mixing parameters on the x-axis are the parameter values for generating the synthetic networks. FastConsensus failed to converge within four hours for mixing parameter $\mu = 0.4, 0.5, 0.8$.

4.3 Experiment 3: Detecting Unclusterable Portions of the Network

While Erdős-Rényi graphs can have regions that appear to be valid communities based on (for example) modularity scores, here we follow the practice of treating Erdős-Rényi as having no valid community structure, as discussed in [12]. Thus, in this study, all valid communities in an Erdős-Rényi graph are singletons, and we use these graphs to evaluate to what extent clustering pipelines are able to reject community structure by returning no or very few non-singleton clusters. We also create networks that are combinations of Erdős-Rényi graphs and LFR networks, and see to what extent the clustering pipelines we examine produce communities that are limited to nodes that are in the LFR subnetwork. We evaluate these questions by examining both the cluster size distribution as well as by examining clustering accuracy.

For Erdős-Rényi graphs with very low values for density p (Fig. 3 (top)), all pipelines tested have good accuracy, returning mostly singletons and clusters of size 2 or 3. However, as the density p increases, both ECG and Leiden-mod return clusters that increase in size, indicating that they are finding community structure in these random networks. At the two largest tested densities, FastConsensus also produces large clusters, while Strict Consensus and FastEnsemble continue to return mainly small clusters.

We see somewhat different trends on networks that are combinations of Erdős-Rényi graphs and LFR networks (Fig. 3 (bottom)). Note that in this setting, FastConsensus and ECG have much better accuracy than on Erdős-Rényi graphs, and only Leiden-mod has really poor accuracy. The two best methods are the two variants of Strict Consensus, with FastEnsemble in third place.

4.4 Experiment 4: The Resolution Limit

The resolution limit for modularity was established in [6], which shows that under some conditions, an optimal modularity clustering will fail to return what

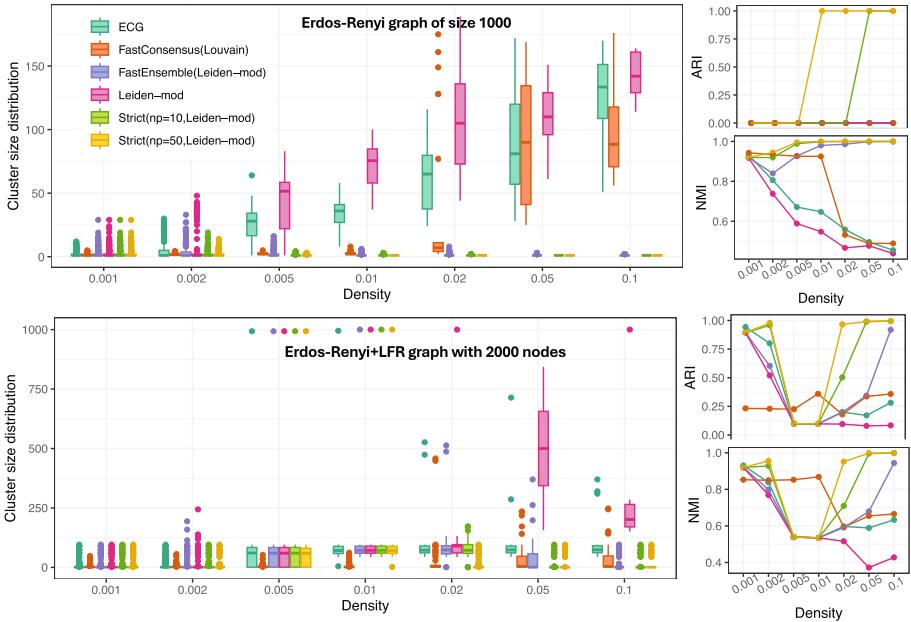


Fig. 3. Experiment 3: Detecting unclusterable portions of networks. Impact of consensus clustering on networks with clusterable and unclusterable components. The unclusterable portion is created using Erdős-Rényi graphs with various densities, and the clusterable portion includes strong community structure created using LFR graphs. Top: Erdős-Rényi graphs with 1000 nodes and various densities. Bottom: Erdős-Rényi graph of size 1000 attached to an LFR graph of size 1000 with 14 communities (2000 nodes in total), with sizes ranging from 45 to 96.

are the “obvious” communities if they are too small. Furthermore, [6] provide as an example the family of ring-of-cliques networks, which are parameterized by the clique size k and the number n of cliques; in a ring-of-cliques network, the cliques are placed in a ring, and each clique is attached to the cliques on each side by a single edge. They establish that when $n \geq k(k - 1) + 2$, then the optimal modularity clustering will put two or more cliques together into a single cluster, instead of the obviously preferred clustering that returns each clique as a separate cluster.

Here we examine whether consensus clustering methods can address this vulnerability of modularity-based clustering from an empirical perspective, using ring-of-clique networks where each clique is of size $k = 10$ but the number n of cliques is allowed to vary. According to the previous paragraph, when $n > 91$ then an optimal modularity clustering will group two or more of the cliques together. Hence, we examine values of n that are both smaller and larger than $n = 91$ in this experiment. We examine Leiden-mod, FastConsensus, ECG, FastEnsemble, and two ways of running the Strict Consensus that vary in terms of the number of partitions (np) on these networks.

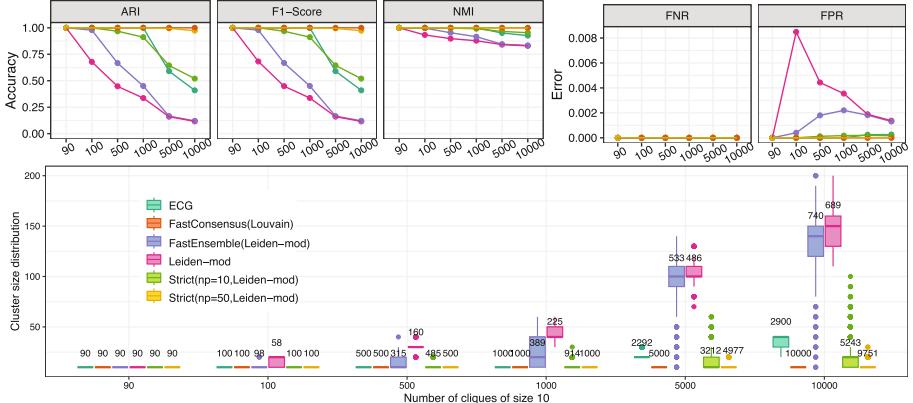


Fig. 4. Experiment 4: Impact of Consensus Clustering on ring-of-cliques networks. We explore accuracy (ARI, NMI, F1-score) and error metrics (FNR and FPR) of modularity-based pipelines (Leiden-mod, ECG, FastConsensus, FastEnsemble, and Strict Consensus), where each network has a variable number of cliques of size 10 arranged in a ring. The numbers shown on the boxplots specify the total number of clusters generated by the method.

As seen in Fig 4, for $n = 90$ clusters, all the methods produce clusterings where each clique is a cluster, as desired. However, as the number of clusters increases, but not their sizes, then Leiden-mod starts merging cliques together, as predicted by the theory from [6]. We also see that the consensus clustering methods (i.e., FastConsensus, ECG, FastEnsemble, and Strict Consensus) reduce the tendency to merge cliques into clusters, but some are more beneficial than others. In particular, the Strict Consensus variants, especially with $np = 50$, have the best accuracy, while FastEnsemble has poor accuracy, especially for the large numbers of clusters, where it is nearly as poor as Leiden-mod.

Figure 4 also shows the FNR and FPR for these methods. Note that all the methods return essentially zero FNR, indicating that no clique in the ring-of-cliques network is ever split apart. On the other hand, the methods differ in terms of FPR, with Leiden-mod having high FPR except for $n = 90$. Again, FastEnsemble is almost as poor as Leiden-mod when there is a large number of cliques, while the other consensus methods have much lower FPR.

The Supplementary Materials shows results for the same data but for clusterings based on CPM-optimization. Note that, in contrast to the theory for modularity, [24] established that for every setting of the resolution parameter r , there will be a value N so that every optimal CPM(r) clustering of a ring-of-cliques network with $n \geq N$ cliques of size k will return the individual cliques as clusters. However, our experimental results show that for large enough numbers of cliques of size 10, Leiden-CPM groups cliques together into clusters. This vulnerability occurs for all of the small resolution values, but disappears when

$r \geq 0.01$. Fortunately, using the Strict Consensus with Leiden-CPM fixes this issue, and returns just the cliques as the clusters.

4.5 Experiment 5: Results on Very Large Networks

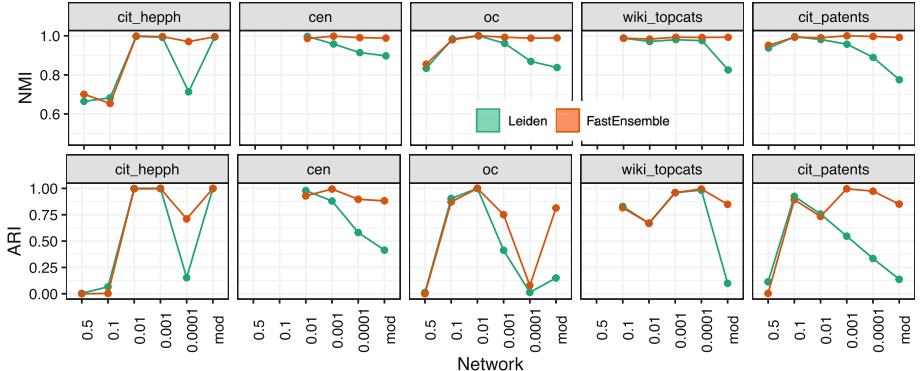


Fig. 5. Experiment 5: Clustering accuracy (ARI and NMI) for CPM and modularity clustering on large synthetic networks. The LFR networks are taken from [17] using Leiden optimizing either modularity or CPM for a specific resolution parameter value, indicated on the x -axis (see Sect. 3.1). Results not shown for three conditions are either for LFR graphs with a large fraction of disconnected ground truth clusters (the two CEN networks) or when the LFR software failed to create a network for the provided parameters (the wiki.topcats network), see [17]. We clustered each network using Leiden and FastEnsemble, using the same optimization criterion as specified on the x -axis.

In this experiment we explore FastEnsemble and Leiden, using both modularity and CPM-optimization, on 27 large LFR synthetic networks based on clustered real-world networks (Materials and Methods) that range up to ~ 3.8 M nodes. FastEnsemble is nearly always at least as accurate as Leiden for all 27 model conditions, and tends to be more accurate when used with Leiden-mod or with Leiden-CPM with small resolution values (Fig 5). Furthermore, the improvement in accuracy is sometimes very large.

Comparisons between FastEnsemble, FastConsensus, ECG, and Leiden-mod on the five LFR networks based on modularity are shown in Table 2. FastConsensus was unable to complete on three of the five networks given 48 h of analysis, while the other methods completed on all networks. Unsurprisingly, Leiden-mod was the fastest, completing in just a few minutes on each network. ECG and FastEnsemble had roughly similar runtimes, sometimes favoring ECG and sometimes favoring FastEnsemble. However, FastEnsemble was generally more accurate than ECG on these networks. Thus, this experiment establishes that FastEnsemble can run on very large networks, up to 3.8 million nodes, and provides an improvement in accuracy over both Leiden-mod and Leiden-CPM.

Table 2. Experiment 5: Clustering accuracy (ARI/NMI) and runtime of methods on 5 large modularity-based LFR networks. “n.d.” stands for no data due to method failing to return a clustering within 48 h. The cit_hepph network has 34,546 nodes, the wiki_topcats network has 1,791,489 nodes, and the other three networks have at least 3,000,000 nodes.

		ARI	NMI	runtime
LFR cit_hepph mod	FastEnsemble(default)	0.9992	0.9949	43 s
	FastConsensus	0.9812	0.9947	21 m 54 s
	ECG	1.0000	1.0000	15 s
	Leiden-mod	0.9991	0.9947	3 s
LFR wiki_topcats mod	FastEnsemble(default)	0.8486	0.9923	7 h 36 m 24 s
	FastConsensus	0.9540	0.9997	14 h 34 m 39 s
	ECG	0.5682	0.9474	6 h 20 m 36 s
	Leiden-mod	0.0990	0.8252	1 m 38 s
LFR cit_patents mod	FastEnsemble(default)	0.8511	0.9916	23 h 1 m 35 s
	FastConsensus	n.d.	n.d.	>2d
	ECG	0.6241	0.9086	1d 12 h 8 m
	Leiden-mod	0.1374	0.7749	2 m 48 s
LFR cen mod	FastEnsemble(default)	0.8820	0.9882	12 h 8 m 47 s
	FastConsensus	n.d.	n.d.	>2d
	ECG	0.9463	0.9803	12 h 38 m
	Leiden-mod	0.4141	0.8973	2 m 31 s
LFR oc mod	FastEnsemble(default)	0.8145	0.9889	1d 3 h 52 m 6 s
	FastConsensus	n.d.	n.d.	>2d
	ECG	0.6958	0.9479	21 h 59 m
	Leiden-mod	0.1502	0.8378	3 m 37 s

5 Discussion

This study reported results on synthetic networks for three consensus clustering methods: ECG, FastConsensus, and FastEnsemble. When optimizing for modularity, each of these reliably produced clusterings that were more accurate than Leiden-mod clusterings under many conditions and did not reduce accuracy. However, while both ECG and FastEnsemble could run on the very large networks, FastConsensus was slower and failed to complete within the allowed 48 hr time period on nearly all networks with 1M or more nodes.

The consensus clustering methods showed very different performance in terms of accuracy. While there were certainly some model conditions where there were often very little differences in accuracy, there were many conditions where FastEnsemble provided the best accuracy, and some conditions where ECG or FastConsensus provided the best accuracy. Experiments 3 and 4 focused on networks that are generally atypical of real-world networks and Experiment 5 only

examined FastEnsemble, and so we restrict this discussion to Experiments 1 and 2. Experiment 1 networks have a range of mixing parameters (and hence clustering difficulty), and this experiment suggests that FastEnsemble has an advantage when the mixing parameter is not too small (and conversely, ECG has an advantage for the smallest mixing parameters). Experiment 2 includes these training networks and also modularity-based networks from [17], which have very low mixing parameters (Supplementary Materials). For the Experiment 2 networks, when FastConsensus or ECG is more accurate than FastEnsemble, it is only by small amount, and the mixing parameter is small (Supplementary Materials). Thus, these two experiments suggest that *when* ECG or FastConsensus is more accurate than FastEnsemble, then the network is generally very easy to cluster, and the top methods have very high accuracy. Here we note that CPM-based clusterings of real-world networks reported in [17] typically have moderate to high mixing parameters (Supplementary Materials), suggesting that accuracy on networks with moderate or higher mixing parameters is the more important criterion.

Finally, we only tested the Strict Consensus variant of FastEnsemble in Experiments 3 and 4, which address performance on graphs that are largely unclusterable or that present the resolution limit challenge, respectively. As we expected, the Strict Consensus provides excellent results for those problems.

Thus, the tested consensus methods have different strengths: FastEnsemble is best suited to networks that have at least moderate mixing parameters, ECG and FastConsensus are better suited to networks with small mixing parameters, and the Strict Consensus is suited to the case where the goal is to avoid false discovery. However, we also observed that FastConsensus was much slower than both ECG and FastEnsemble, especially on the networks with more than 1,000,000 nodes.

6 Conclusions

Our study showed that FastEnsemble can provide very good results, matching or improving on both ECG and FastConsensus, two other consensus methods that use more sophisticated techniques, on many networks that are generally difficult to cluster, such as those with moderate to high mixing parameters. However, ECG and FastConsensus sometimes provide better results than FastEnsemble for networks with low mixing parameters, so that the three consensus clustering methods each have contexts where they have an advantage. We also found that ECG and FastEnsemble are both faster and more scalable than FastConsensus. Finally, we showed that FastEnsemble, used with Leiden-CPM provided improved accuracy compared to Leiden-CPM, but Leiden-CPM is not enabled for use within ECG and FastConsensus.

Given the similarity in algorithmic design between ECG and FastEnsemble, it is interesting that we saw differences in accuracy in some conditions – sometimes favoring ECG and sometimes favoring FastEnsemble. One possible explanation is the use of Leiden-mod within FastEnsemble and Louvain within ECG, but

another possibility is the weighting of edges in ECG requiring the endpoints to be in 2-cores (whereas this is not required for FastEnsemble). Future work is needed to better understand why we see these differences.

Although true communities are not known for real-world networks, future work should evaluate scalability to large real-world networks, such as the Open Citations network with $\sim 13M$ nodes [17], as well as new consensus clustering methods [7]. Our study was performed using synthetic networks generated using LFR, but other simulators, including ABCD [10], ABCD+o [11], Stochastic Block Models [19], and newer simulators [1], could be used.

Data and Code Availability. The code and scripts used in this study are available at <https://github.com/ytabatabae/fast-ensemble>. The data are available at <https://github.com/ytabatabae/ensemble-clustering-data>.

References

1. Anne, L., Vu-Le, T.A., Park, M., Warnow, T., Chacko, G.: Synthetic networks that preserve edge connectivity. In: Proceedings, International Conference on Complex Networks and their Applications (2024)
2. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**(10), P10,008 (2008)
3. Boyack, K.W., Klavans, R.: An improved practical approach to forecasting exceptional growth in research. *Quant. Sci. Stud.* **3**, 1–25 (2022)
4. Erdős, P., Rényi, A.: On random graphs. *Publicationes Mathematicae* **6**(3–4), 290–297 (1959)
5. Fortunato, S.: Community detection in graphs. *Phys. Rep.* **486**(3–5), 75–174 (2010)
6. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. *Proc. Natl. Acad. Sci.* **104**(1), 36–41 (2007)
7. Hussain, M.T., Halappanavar, M., Chatterjee, S., Radicchi, F., Fortunato, S., Azad, A.: Parallel median consensus clustering in complex networks. *Sci. Rep.* **15**(1), 3788 (2025)
8. Jeub, L.G., Sporns, O., Fortunato, S.: Multiresolution consensus clustering in networks. *Sci. Rep.* **8**(1), 1–16 (2018)
9. Jiang, H., Liu, Z., Liu, C., Su, Y., Zhang, X.: Community detection in complex networks with an ambiguous structure using central node based link prediction. *Knowl. Based Syst.* **195**, 105,626 (2020)
10. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (ABCD)-fast random graph model with community structure. *Netw. Sci.* **9**(2), 153–178 (2021)
11. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection with outliers (ABCD+o). *Appl. Netw. Sci.* **8**(1), 25 (2023)
12. Lancichinetti, A., Fortunato, S.: Limits of modularity maximization in community detection. *Phys. Rev. E-Statistical Nonlinear Soft Matter Phys.* **84**(6), 066,122 (2011)
13. Lancichinetti, A., Fortunato, S.: Consensus clustering in complex networks. *Sci. Rep.* **2**(1), 1–7 (2012)

14. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. *Phys. Rev. E* **78**(4), 046,110 (2008)
15. Newman, M.E.: Mixing patterns in networks. *Phys. Rev. E* **67**(2), 026,126 (2003)
16. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. *Phys. Rev. E* **69**(2), 026,113 (2004)
17. Park, M., et al.: Well-connectedness and community detection. *PLOS Complex Syst.* **1**(3), e0000009 (2024)
18. Pedregosa, F., et al.: Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
19. Peixoto, T.P.: The graph-tool python library. figshare (2014). <https://doi.org/10.6084/m9.figshare.1164194>
20. Poulin, V., Théberge, F.: Ensemble clustering for graphs: comparisons and applications. *Appl. Netw. Sci.* **4**(1), 51 (2019)
21. Ronhovde, P., Nussinov, Z.: Local resolution-limit-free potts model for community detection. *Phys. Rev. E* **81**(4), 046,114 (2010)
22. Tabatabae, Y., Wedell, E., Park, M., Warnow, T.: Supplementary materials for FastEnsemble: a new scalable ensemble clustering method (2024). <https://doi.org/10.5281/zenodo.13625629>. Zenodo
23. Tandon, A., Albeshri, A., Thayananthan, V., Alhalabi, W., Fortunato, S.: Fast consensus clustering in complex networks. *Phys. Rev. E* **99**(4), 042,301 (2019)
24. Traag, V.A., Van Dooren, P., Nesterov, Y.: Narrow scope for resolution-limit-free community detection. *Phys. Rev. E* **84**(1), 016,114 (2011)
25. Traag, V.A., Waltman, L., Van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 1–12 (2019)
26. Yang, Z., Algesheimer, R., Tessone, C.J.: A comparative analysis of community detection algorithms on artificial networks. *Sci. Rep.* **6**(1), 1–18 (2016)



Examining Different Research Communities: Authorship Network

Shrabani Ghosh^(✉) 

University of North Carolina at Charlotte, Charlotte, NC 28223, USA
sghosh15@uncc.edu

Abstract. Google Scholar is one of the top search engines to access research articles across multiple disciplines for scholarly literature. Google Scholar's advanced search option gives the privilege to extract articles based on phrases, publisher name, author name, time duration, etc. In this work, we collected Google Scholar data (2000–2021) for two different research domains in computer science: Data Mining and Software Engineering. The scholar database resources are powerful for network analysis, data mining, and identifying links between authors via co-authorship network. We examined co-authorship networks for each domain and studied their network structure. Extensive experiments are performed to analyze publication trends and identify influential authors and affiliated organizations for each domain. The network analysis shows that the network's features are distinct from one another and exhibit small communities within the influential authors of a particular domain.

Keywords: network analysis · graph theory · co-author network · research collaboration

1 Introduction

Social network analysis has become a popular technique in many fields such as social media networks, web networks, biological networks, academic social networks etc. The network analysis gives a broader picture of connections that has been maintained within a community over a long period of time. As academic papers are increasing exponentially, each academic area is segmented and specialized. To better understand the scientific community in general, it is important to study and model the factors of research communities, the growth and evaluation of a particular research domain. Therefore, modeling these factors efficiently is important to better understand the working culture of the scientific community. In that case, authorship network is very useful for analysis across different scientific domains. We are interested to look at the authorship network of research domains under computer science stream in recent era. The pattern and culture are quite a bit different in different research domains under same stream.

Thanks to my advisor Dr. Bojan Cukic.

Most of the authorship networks analysis finds communities and structures of sub-communities.

Many co-authorship networks have been studied to investigate collaboration among users [2, 4, 13]. Most of them used databases, for example Aminer, DBLP digital libraries to generate network. DBLP provides comprehensive access to conference proceedings and allows for filtering of publications by conference stream, offering similar functionality to parsing results from Google Scholar. Co-authorship network can describe publication rate and scientific productivity [1, 11]. Previous studies have shown that, there is a strong positive relation between co-authorship network and scientific productivity [5, 8]. Researchers from diverse geographical locations [6, 12] and multi-disciplinary backgrounds collaborate together which positively impacts scientific productivity. Social network studies have furthered the understanding of the relationship between co-authorship and productivity. Studies assessing the relationship between productivity and the position of authors in the co-author network have found that authors who publish with many different co-authors bridge communication and exhibits higher rates of publication records.

In this paper, we use data collected from Google Scholar on data mining and software engineering research domain. Using this dataset, our proposed work will follow the below research contributions:

- Investigate pattern of publication over past 20 years for different research domains.
- Identify the most influential authors based on published articles.
- Identify the most frequently affiliated organization appeared within a domain.
- Analyze and compare network features of two distinct research areas.

2 Related Work

Co-authorship networks have been widely used in the past to explore scientific collaboration among authors. An authorship network is an academic social network where individuals are connected based on their collaboration through research articles. In previous studies, co-authorship networks have been analyzed to explore different statistical characteristics. It has been extensively analyzed at network level and individual node level [14] and behavioral patterns in scientific collaborations [10]. Similar to authorship network, citation network also have been studied much. In citation network, nodes connection are created from paper citations, where papers are nodes and links generate when one paper is cited by another paper. Usually citation network shows the relation among papers. On the other hand, Co-authorship networks illustrate the relationships among researchers, typically reflecting collaborative bonds formed through joint publications. These networks are dynamic, evolving as researchers collaborate within or across different research domains over time. A single researcher may collaborate in multiple research areas, thereby participating in different co-authorship networks. While these networks may overlap due to the presence of common

researchers, they can still represent distinct collaborative groups based on the specific research topics or domains. Many research dimensions like finding rising stars [9], name disambiguation [7], domain experts [3] have been studied in academic social network.

In this paper, we analyze and compare the publication pattern, features of domain experts and collaboration culture of two research domains in computer science. The number of published papers per year shows how these two fields have emerged over time. From the google scholar profile, it is easy to capture authors information and affiliations to understand the most involved organizations/institutions in a domain. Also, the network analysis shows relation among authors and degree distribution of authors.

3 Data Collection

The dataset used in this work is collected from Google Scholar. The data is collected for two different research domains: Data Mining and Software Engineering. The focus on software engineering and data mining in this analysis was chosen to offer a comparative perspective between two distinct domains: one well-established and foundational (software engineering) and one that has experienced significant growth in recent years (data mining). Software engineering, being a more mature field, allows us to examine long-term trends and shifts in publication practices, while data mining, a relatively younger and increasingly popular research area, provides insights into how emerging fields evolve and attract academic and industrial interest.

For each research domain, three recognized and popular conferences have been chosen to extract using advanced search option. The three selected conferences for Data Mining are, 1. IEEE International Conference on Machine Learning and Applications (ICMLA) 2. IEEE International Conference on Data Mining (ICDM) 3. ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD). On the other hand, for Software Engineering the selected conferences are: 1. International Conference on Software Engineering (ICSE) 2. ACM International Symposium on Foundations of Software Engineering (SIGSOFT) 3. IEEE/ACM International Conference on Automated Software Engineering (ASE). For each conference, we have extracted 1000 published papers within time-frame 2000 to 2021. We do think that the sample In total, we have collected 3000 published papers for each area. For dataset interested researchers can contact the author via email¹. The source code used in this study will be made available in the following directory.²

A published paper typically involves one or more authors. We have listed authors' information from the papers. We can identify 2788 authors in Software Engineering conferences and 4245 authors in Data Mining conferences. We have collected the author's information: full name, citation, h-index, and affiliation

¹ sghosh15@uncc.edu.

² <https://github.com/srbnghosh99/Examining-Different-Research-Communities-Authorship-Network>.

While these trends are informative, it is important to consider alternative explanations for these differences. One factor could be changing acceptance rates, which may have remained constant or varied differently between the two fields. Also, higher number of researchers in data mining than software engineering could be another reason of decreased number of publications. It is also possible that researchers are getting more interested in data mining over time. Additionally, the emergence of competing conferences in software engineering might have split the pool of submissions, contributing to the observed decline. Conversely, the increasing focus on data mining, driven by the rise of big data and machine learning applications, may have spurred more publications in that field. Although this study does not examine these factors in detail, they offer plausible explanations for the publication trends observed in Fig. 2.

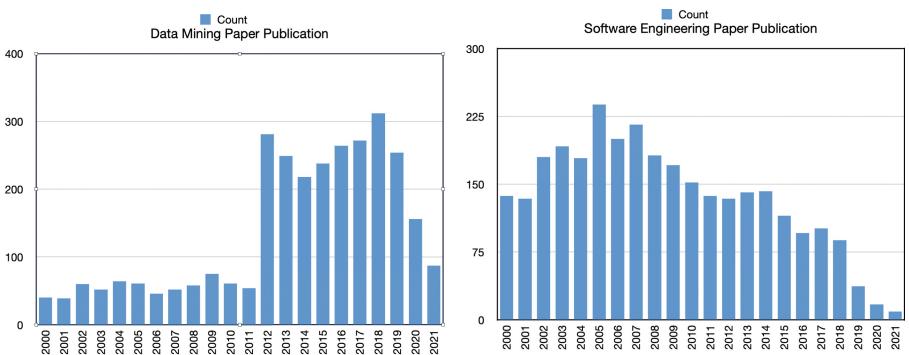


Fig. 2. Number of publications per year: The left image shows Data Mining and right image shows Software Engineering

4.2 Influential Authors

Authors tend to publish papers at the same conferences consistently, allowing us to identify the influential authors based on the number of papers published within their respective research domains. In the scholar data, each author is assigned a unique identifier. Our analysis is conducted based on these unique IDs, allowing us to subsequently map them back to their corresponding original names. In this study, we define influence by how frequently an author appears as either the first author or a co-author. Using this metric, we have identified the top 10 influential authors based on their publication records in our dataset. From Fig. 3, the top 5 influential authors in data mining are Jiawei Han, Huan Liu, Eamonn Keogh, Philip Yu, and Ryan Baker, with Jiawei Han appearing 32 times and Huan Liu 30 times between 2000 and 2021. In software engineering, Barbara Kitchenham, Thomas Zimmermann, Mark Harman, Gail Murphy, and Krysztof Czarnecki stand out, with Barbara Kitchenham appearing 35 times and Thomas

Zimmermann 26 times in the same period. The influence of these authors reflects not only their publication frequency but also their significant contributions to their fields. For instance, Jiawei Han's foundational work in data mining, particularly in developing algorithms for large-scale datasets, and Barbara Kitchenham's systematic approaches to evidence-based software engineering, have made them central figures in their domains. While it is important to recognize these key contributors, the analysis focuses on their impact and contributions rather than merely listing their names. This ensures that the recognition of influential researchers is grounded in their significant and consistent output over time.

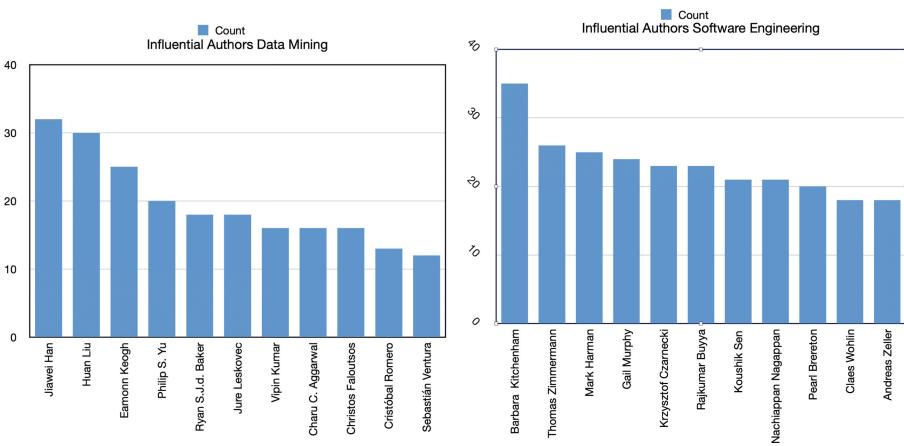


Fig. 3. The top 10 influential authors: The left image shows Data Mining and right image shows Software Engineering

4.3 Affiliated Organizations

We also looked at the authors Google Scholar profiles that are classified as valid profiles. The valid profiles usually have full name, research interest, citations, h-index, and affiliated organization information. We have captured the current or available affiliations of the authors as provided by Google Scholar. However, we do not have access to the authors' affiliations at the time of publication. This limitation may introduce distortions, as authors often change institutions over time. For example, an author like Barabási may have been affiliated with different organizations during the publication of various papers, yet only the most recent or selected affiliation might be displayed in databases like Google Scholar.

From the most current affiliation collected from Google Scholar, we have come up with the top most frequently appeared affiliated organizations of the authors for each research domain. In data mining domain, we can see that Google has been the topmost affiliated company to collaborated with, then University

of Minnesota, Microsoft, University of Illinois Urbana-Champaign, University of North California, Riverside so and so forth. In software engineering, Microsoft has been the topmost affiliated company, and then Keele University, Carnegie Mellon University, Facebook, Waterloo University so and so forth (Fig. 4).

These affiliations provide valuable insights into the institutions that have a significant influence on research in these fields. For instance, Google, Microsoft and Facebook have prominent roles in both domains reflect their ongoing investment in research and development, as well as their collaborations with leading academic institutions. The presence of Carnegie Mellon University further highlights the contribution of academic research to both data mining and software engineering. This analysis underscores how industry-academia partnerships shape research trends and suggests that institutions with strong affiliations in these domains are likely to remain influential in advancing research and innovation in the future.

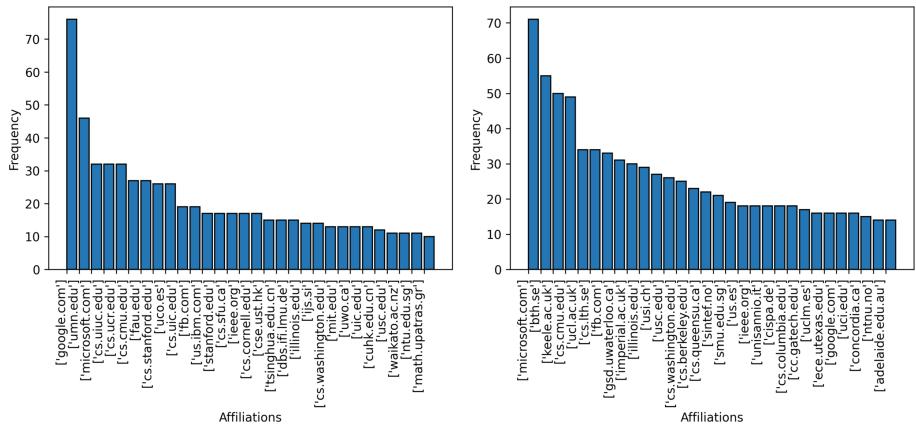


Fig. 4. Number of publications per year: The left image shows Data Mining and right image shows Software Engineering

5 Network Analysis

In this section, we perform a co-author network analysis to examine the relationships between the authors within domains. We have generated undirected network for both domains. Data mining co-author network has 4245 nodes and 4363 edges. There are 1523 connected components in the network and the giant connected component has 571 nodes involved within it. Figure 5 (a) shows the largest connected component on the top and degree distribution for all the nodes. The distribution shows that more than 1200 nodes have degree value of 2 and few nodes have degree value above 10. We found Huan Liu and Jiawei Han have highest degree value of 44 & 39 correspondingly.

On the other hand, the software engineering co-author net has 2788 nodes and 2870 edges. Many papers have only one author. Out of the whole network, there are 898 connected components. The giant connected component of the network consists of 708 nodes. The degree histogram (Fig. 5 (b)) of the whole network shows that more than 800 nodes have degree 2 and few nodes have degree 10. In this case, Mark Harman and David Lo have degree value of 21 and 20 correspondingly.

Compared to software engineering, data mining authors' highest degree value 40 is twice that of software engineering authors' highest degree value 20.

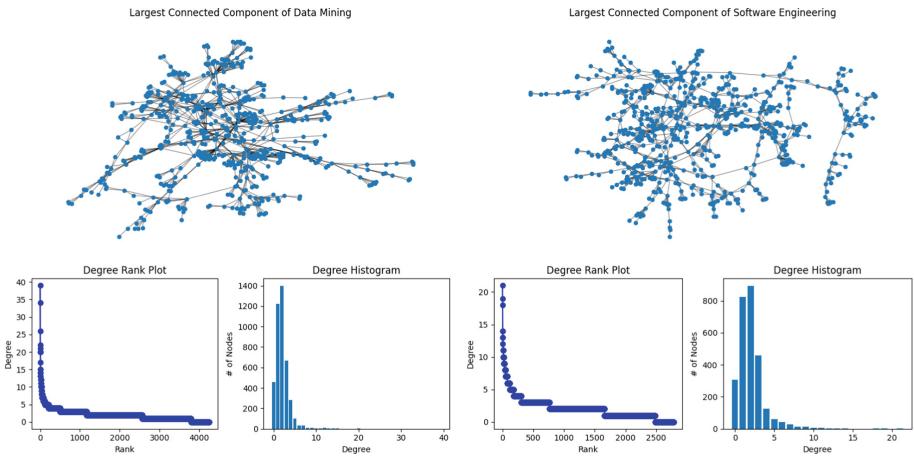


Fig. 5. Degree Distribution: (a) The left image shows Data Mining and (b) right image shows Software Engineering

5.1 Collaboration Teams

In today's research system, team collaboration is the most essential part. In a research collaboration team, collaborators interchange their ideas, thoughts, datasets, problem-solving approaches, writing, etc. Some of the effective teams work together many times. It is important to point out strong tied collaboration teams in the co-author network to understand collaboration culture in a research domain.

From the network analysis, we have brought out the paired collaborators who co-authored together several times. We found that in software engineering, Barbara Kitchenham has collaborated with Pearl Brereton highest 13 times and with David Budgen 12 times. Also, "Per Runeson and Martin Höst", "Nachappan Nagappan and Thomas Ball", "Andrea De Lucia and Rocco Oliveto" each paired collaborators worked together 8 times within the time-frame for the conferences mentioned early.

On the other hand, in data mining “Cristóbal Romero and Sebastián Ventura” collaborated together highest 12 times, “Jiliang Tang and Huan Liu” 7 times. “George Karypis and Joseph Konstan”, “Eamonn Keogh and Stefano Lonardi”, “Igor Santos and Xabier Ugarte-Pedrero” each paired collaborator worked together 5 times.

5.2 Connected Authors

At a high level, connected components exhibit several key characteristics, often representing small communities within the network. Our focus here is on closely examining the largest connected components from each domain. The size and color of the nodes are determined by their degree, while the weight of the edges reflects the number of times two authors have collaborated.

Figures 6 and 7 allow for a comparison of the connected components across these two domains. In software engineering, authors tend to be sparsely connected, with fewer instances of tightly knit collaborations. Conversely, in data mining, the nodes are more densely interconnected, indicating that several authors have frequently worked together. This is particularly evident in the purple-colored edges, which represent high levels of collaboration.



Fig. 6. Software Engineering Largest Connected Component



Fig. 7. Data Mining Largest Connected Component

6 Conclusion

We studied the characteristics of two different research communities: data mining and software engineering authorship network from google scholar. Analyzing network structure, collaboration and author profiles of these two communities allowed us to discover interesting features. We have observed that data mining research domain has flourished more than software engineering over past 10 years. Though we have seen that the influential authors have similar number of publication record in both research domain, however, in data mining have more researchers compared to software engineering. We have seen that from the network analysis that data mining authors have more knit group of collaboration than software engineering authors. The overall collaboration culture and number of authors helped to flourish data mining domain. Furthermore, the giant tech company for example Google collaborators seemed to be more interested in data mining domain and also Microsoft collaborators have higher publication record in data mining than software engineering tracks.

Although our findings are based solely on data from selective conferences, and we do not intend to generalize these results broadly, we believe they offer a meaningful perspective on the trends and developments within the particular domains covered by these conferences. We wanted to present findings and demonstrate it through our dataset. The dataset used was carefully selected to ensure that it is representative of the field. Nevertheless, we acknowledge that future work could benefit from exploring networks constructed from different or broader samples to evaluate whether the observed patterns hold. This process can be extended to many directions in future. Authorship networks can be useful for predicting future collaboration teams by analyzing historical collabor-

ration patterns among authors. By examining the connections between authors in the network, we can identify individuals who have previously collaborated, which often indicates shared research interests and complementary expertise. Additionally, network analysis techniques can reveal clusters of authors who are frequently connected, suggesting potential teams for future collaborations. This approach aligns with social network theory, which posits that individuals are more likely to collaborate with those within their existing networks. Authorship network can be useful to do further network analysis. For example, centrality measures to rank authors not just by publication count, but by their importance in the co-authorship network. Also, as a dynamic network we can study the evolution of the network for each domain over time: if there are any emerging trends or shifts in key players. We are highly interested to explore citations network of each domain and analyze citation authors and articles network. As a future direction, we propose to develop a comprehensive analysis pipeline that can be adapted for studying various research areas. This pipeline will involve customizing the selection of conferences or journals and applying similar methodologies to analyze the co-authorship and citation networks in different fields.

In addition to the primary findings of this study, we recognize that understanding authorship networks can illuminate broader issues, including gender and racial biases in research collaboration. Our analysis highlights the collaborative patterns among authors, which may reflect underlying biases in the academic community. Future work will aim to investigate how these biases manifest in co-authorship networks, potentially impacting the diversity of research teams and the representation of marginalized voices in scientific discourse. By exploring these aspects, we can gain a more comprehensive understanding of the factors that shape collaborative practices and promote equity in research.

References

1. Mark R Costa. The dynamics of social capital in scientific collaboration networks. In: Proceedings of the American Society for Information Science and Technology, vol. 51, no. 1, pp. 1–4 (2014)
2. Cunningham, S., Dillon, S.: Authorship patterns in information systems. *Scientometrics* **39**(1), 19–27 (1997)
3. Daud, A., Li, J., Zhou, L., Muhammad, F.: Temporal expert finding through generalized time topic modeling. *Knowl.-Based Syst.* **23**(6), 615–625 (2010)
4. Egghe, L., Rousseau, R., Van Hooydonk, G.: Methods for accrediting publications to authors or countries: consequences for evaluation studies. *J. Am. Soc. Inf. Sci.* **51**(2), 145–157 (2000)
5. Glänzel, W., Schubert, A.: Analysing scientific networks through co-authorship. In: *Handbook of Quantitative Science and Technology Research*, pp. 257–276. Springer (2004)
6. Goldfinch, S., Dale, T., DeRouen, K.: Science from the periphery: collaboration, networks and periphery effects in the citation of new Zealand crown research institutes articles, 1995–2000. *Scientometrics* **57**(3), 321–337 (2003)

7. Huang, S., Yang, B., Yan, S., Rousseau, R.: Institution name disambiguation for research assessment. *Scientometrics* **99**(3), 823–838 (2014)
8. Leydesdorff, L., Wagner, C., Park, H.W., Adams, J.: International collaboration in science: the global map and the network. arXiv preprint [arXiv:1301.0801](https://arxiv.org/abs/1301.0801) (2013)
9. Li, X.L., Foo, C.S., Tew, K.L., Ng, S.K.: Searching for rising stars in bibliography networks. In: International Conference on Database Systems for Advanced Applications, pp. 288–292. Springer (2009)
10. Liu, X., Bollen, J., Nelson, M.L., Van de Sompel, H.: Co-authorship networks in the digital library research community. *Info. Process. Manag.* **41**(6), 1462–1480 (2005)
11. Melin, G.: Pragmatism and self-organization: research collaboration on the individual level. *Res. Policy* **29**(1), 31–40 (2000)
12. Nemeth, C.J., Goncalo, J.A.: Creative collaborations from AFAR: the benefits of independent authors. *Creativity Res. J.* **17**(1), 1–8 (2005)
13. Newman, M.E.: Scientific collaboration networks. I. Network construction and fundamental results. *Phys. Rev. E* **64**(1), 016131 (2001)
14. Newman, M.E.J.: The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci.* **98**(2), 404–409 (2001)



Constructing Telegram Channels Digital Profiles

V. A. Popov and A. A. Chepovskiy

Higher School of Economics, Moscow, Russia
{vapopov,aacheponvsky}@hse.ru

Abstract. In this paper, the features of the popular messenger Telegram are described. After that, a model to construct a graph of interacting objects is considered, where the vertices are Telegram channels, and the edges are the interactions between them. The weight of the graph edges depends on the presence of reposts, mentions and common sources of information between pairs of vertices. Further, taking into account the features of the Telegram messenger and the capabilities of its API, the concept of a digital channel profile is introduced in the form of a normalized vector. Each vector coordinate is responsible for a separate characteristic of the channel in the network. Then, examples of constructing these digital profiles for channels and their further use are given. A cluster analysis of the digital profiles for 4 Telegram channel networks is carried out. After that, the characteristics of each for the resulting clusters were described. Thus, the authors presented a method for constructing Telegram channels digital profiles, which, together with the analysis of the network topology of channels, provides a broader toolkit for analyzing Telegram channels.

Keywords: scale-free networks · model of information impact · digital profiles · community detection · social networks analysis

1 Introduction

The concept of complex networks has been known in science for many years – there are many biological, technological, telecommunication and social networks around us [1]. These networks often have non-trivial topological characteristics, in particular, the distribution of vertex degrees may follow a power law. Such networks are referred to as so-called scale-free networks. The analysis of graphs characterizing scale-free networks has been the subject of research by many authors over the past two decades [2, 3].

In the context of practical research for telecommunication networks, it is important to import real data and construct graphs of interacting objects on this data – graphs that characterize the communication between objects of the original network and the degree of its intensity. Nowadays, one of the relevant networks for data analysis is the Telegram channels network. Telegram provides functionality for organizing public Telegram channels, which serve as information and news feeds. Many media, information communities and bloggers have their own Telegram channels and regularly publish content in them, also Telegram users can subscribe to these channels and receive information in the form of messages.

Telegram provides a wide range of tools for managing channels. In addition to unique text and photo-video content, the channel administrator can publish a post from another channel in their own (repost), quote or mention other Telegram channels, and make a link to external addresses on the Internet. The use of these features establishes connections between Telegram channels, which allows us to consider their network for importing data and constructing a weighted graph of interacting objects. In previous works, the authors proposed the (U, M, R)-model for graph construction, in which the resulting graph has the properties of scale-free networks [4, 5].

Furthermore, the construction of digital profiles for network objects is of interest for analysis. These profiles can be formed using publicly available information about the Telegram channel (such as its description and the number of subscribers), the texts of the channel's posts, comments and reactions from subscribers, and the channel's connections with others. Consequently, each Telegram-channel can be compared with its own specific characteristics to define its digital profile.

It is worth saying that nowadays there are many works devoted to the study of channels in the Telegram messenger. Some of them study the texts of posts, divide them by topics and style of narration [6]. Also popular are works related to the search for channels distributing fake information or engaged in illegal activities [7].

Presenting channels as digital profiles allows us to compare channels, find patterns and perform cluster analysis, thereby solving various problems in analyzing and searching for specific channels in Telegram.

2 Graph Construction Model

This study focuses on public channels in the Telegram messenger. The following key factors of interaction between channels can be identified: reposts between channels, mentions of one channel by another, and the presence of common external URLs in posts of two channels. Thus, Telegram-channels form a network that can be represented as a weighted graph.

To construct such models of interaction between channels, it is initially necessary to import data from the messenger over a selected time period. For this purpose, software using the official Telegram API was developed. This application is capable of importing channel information into a specialized format [4, 5], containing all the necessary components for further analysis and the identification of interactions between channels.

To construct a graph of interacting objects, we initially form the set of vertices V . This set contains all imported vertices corresponding to channels. Then we perform a transition to a weighted graph $G(V, E)$. If there are interactions between a pair of vertices from set V , then an edge is constructed between them. The data import and the construction of the set of vertices V are organized in such a way that the graph $G(V, E)$ represents one connected component. The weight of this edge determined based on the degree of interaction between vertices. The function F described below is used to calculate the specified weight of the edges.

Let us define a weight function w on the initial set of edges E , depending on the identified interaction factors $\delta_{e_{AB}}^U$, $\delta_{e_{AB}}^M$, $\delta_{e_{AB}}^R$:

$$w(e_{AB}) = F\left(\delta_{e_{AB}}^U, \delta_{e_{AB}}^M, \delta_{e_{AB}}^R\right) \quad (1)$$

where $\delta_{e_{AB}}^U$ – the number of common unique external links (URLs) in posts for channels A and B over the selected period;

$\delta_{e_{AB}}^M$ – the number of posts in which channel A mentioned channel plus the number of posts in which B mentioned A over the selected period (for each post, unique mentions are counted, i.e. if channel A mentioned channel B several times in one post, this mention is still taken into account once in this coefficient);

$\delta_{e_{AB}}^R$ – the number of reposts by channel A of messages from channel B, plus the number of reposts by channel B of messages from channel A over the selected period;

F – a function, depending on $\delta_{e_{AB}}^U$, $\delta_{e_{AB}}^M$, $\delta_{e_{AB}}^R$ which returns non-negative values.

In this work, we will use the function F of the following type:

$$F = 1 \cdot \delta_{e_{AB}}^U + 2 \cdot \delta_{e_{AB}}^M + 3 \cdot \delta_{e_{AB}}^R \quad (2)$$

Previously, in [8], several models to construct graphs of the Telegram network were investigated. The results show that the graphs constructed using the weight function (2) generally have a distribution of vertex weights close to a power law with degree parameter $\alpha \in [-3; -2]$ (Fig. 1). Here we consider the following graphs: G_1 with 600 vertices and 18009 edges; G_2 with 619 vertices and 2973 edges; G_3 with 773 vertices and 6611 edges; G_4 with 1252 vertices and 17841 edges.

Therefore, the graphs constructed using the model described above possess the properties of scale-free networks.

It is worth mentioning that in [6, 7] the authors construct some graphs of interactions between Telegram channels. Only their graphs are not weighted and do not take into account external common links and channel mentions.

3 Telegram Channel Digital Profile

Constructing the Telegram channels interaction graph information about channels interaction with each other, such as number of reposts and mentions of other channels is obtained. Also we can get the main characteristics for each channel-vertex (channel's name, description, number of subscribers). Additionally, information about the channel's posts can be considered, in particular the texts of the posts, user reactions, the number of views, and other data. Also, it is possible to calculate the centralities of nodes in the graph.

Thus, taking into account the capabilities of Telegram, the following characteristics can be used to define the Telegram channel digital profile:

- Participants_count – number of subscribers of Telegram channel;
- Reactions_avg_by_post – average number of user reactions on Telegram channel posts for the selected time period;
- Emojis_text_count_avg_by_post – average number of emojis used in the texts of Telegram channel posts for the selected time period;
- Betweenness_centrality – classical betweenness centrality of a node in a network of interacting objects.

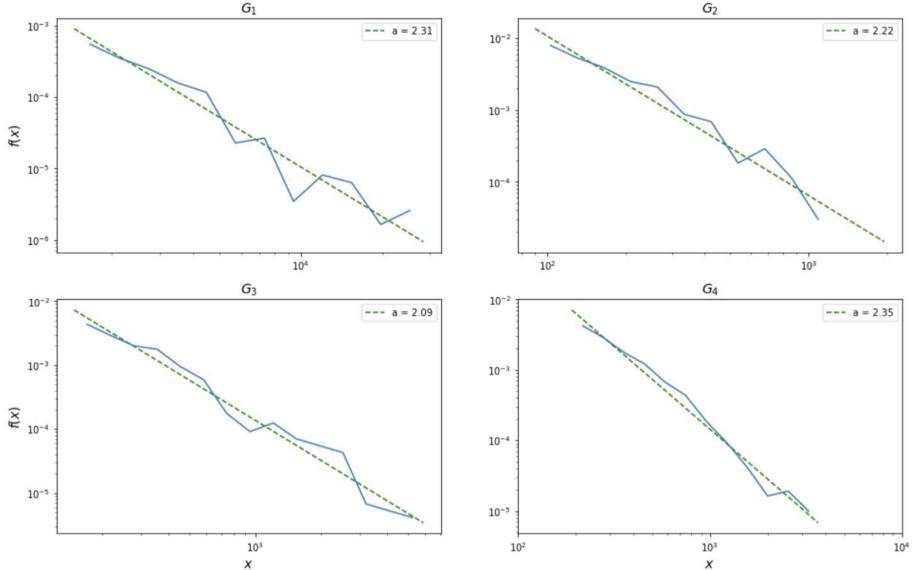


Fig. 1. Blue lines are the probability densities of the distribution of vertex weights for 4 graphs with the weight function $F = 1 \cdot \delta_{e_{AB}}^U + 2 \cdot \delta_{e_{AB}}^M + 3 \cdot \delta_{e_{AB}}^R$; Green dotted lines are probability densities of power law distributions with the specified parameters α .

The selected parameters characterize the Telegram channel from different sides. Participants_count indicates how popular the channel is; Reactions_avg_by_post characterizes the activity and engagement of the audience; Emojis_text_count_avg_by_post shows emoji usage in channel posts, this characteristic can indicate how formal, emotional and eye-catching the text of the channel's posts is; Betweenness_centrality determines the channel role in the interacting objects network. Such a vector representation of channel's characteristics enables comparative and cluster analysis of digital profiles.

Based on this, each Telegram channel in the interaction network corresponds to a vector of 4 values. Having normalized all these 4 characteristics, dividing each by the maximum corresponding value in the entire graph, we obtain a vector – the digital profile of the Telegram channel for the given network over a specified time period, where each characteristic takes a value from 0 to 1.

For example, let us consider a channels network related to sports (data imported from Telegram over two-week period of 2023). We will construct a digital profile for such popular channel ***sport*** (Table 1). As we can see this channel is popular among the users. But the amount of reactions to posts is low. Also, the channel posts have a moderate number of emoji, and the channel itself has an average betweenness centrality value in the network.

Table 1. Digital Profile of the Telegram channel ***sport***

Channel name	Participants count	Reactions avg by post	Emojis text count avg by post	Betweenness centrality
sport	0.959	0.03	0.32	0.573

4 Examples of Using Telegram Channels Digital Profiles

Digital profiles of Telegram channels can be used for many tasks, in particular, it is possible to conduct a comparative analysis of channels by finding the distance between vectors, or conduct cluster analysis, find patterns between different channels.

Let us consider 4 Telegram channels networks with different network sizes, channel content and imported over different time periods:

- ***Sport*** – 100 vertices, 357 edges, data over 2 weeks;
- ***Education*** – 302 vertices, 2641 edges, data over 5 months;
- ***Property*** – 464 vertices, 1921 edges, data over 1 month;
- ***Finance*** – 1592 vertices, 34877 edges, data over 1 month.

We construct a digital profile in the form of a vector of four values for each Telegram channel in these networks. Then we cluster the obtained vectors for each graph using the k-means algorithm and look at the centers of the resulting clusters (Table 2, 3, 4 and 5). Parameter k was chosen equal to 4 and 5 in order to divide the original sets of channels into large groups that have clearly expressed differences between them. As can be seen from the tables, all channels in the graphs are divided into 4–5 characteristic categories. Each group has its own characteristics and differs from the others, in particular, channels with high values of any of the four characteristics of the digital profile are singled out separately.

Table 2. Cluster centers of network ***Sport***

Cluster_id	Participants count	Reactions avg by post	Emojis text count avg by post	Betweenness centrality
0	0.047	0.038	0.618	0.011
1	0.253	0.107	0.429	0.555
2	0.308	0.453	0.314	0.002
3	0.081	0.039	0.217	0.030
4	0.849	0.133	0.367	0.132

In the ***Sport*** network, a cluster with channels having the largest audience (cluster 4) stood out separately; cluster 2 includes channels of average popularity, but with an active audience; cluster 0 channels have the largest number of emojis in the posts; and cluster 1 contains nodes with the highest betweenness centrality.

Table 3. Cluster centers of network ***Education***

Cluster_id	Participants count	Reactions avg by post	Emojis text count avg by post	Betweenness centrality
0	0.008	0.008	0.052	0.007
1	0.013	0.011	0.543	0.012
2	0.045	0.009	0.153	0.521
3	0.006	0.005	0.264	0.009
4	0.490	0.312	0.028	0.002

Table 4. Cluster centers of network ***Property***

Cluster_id	Participants count	Reactions avg by post	Emojis text count avg by post	Betweenness centrality
0	0.012	0.018	0.058	0.012
1	0.006	0.010	0.326	0.017
2	0.383	0.475	0.073	0.031
3	0.012	0.012	0.127	0.389

Table 5. Cluster centers of network ***Finance***

Cluster_id	Participants count	Reactions avg by post	Emojis text count avg by post	Betweenness centrality
0	0.017	0.014	0.022	0.005
1	0.020	0.017	0.431	0.003
2	0.018	0.014	0.176	0.004
3	0.228	0.398	0.048	0.009

In the ***Education*** network, a cluster with popular channels and an active audience was identified – cluster 4. Channels with the highest centrality value were included in cluster 3. Small channels fall into cluster 0. And the remaining channels were divided by the number of emoji in posts.

In the ***Property network***, a cluster of popular channels with an active audience (cluster 2) also stands out; in cluster 3, the nodes have the highest centrality and average emoji usage in texts.

In the ***Finance*** network, cluster 3 with popular channels is similarly distinguished, the remaining clusters are divided by the number of emojis use in posts. And channels with different centrality values are distributed evenly across the clusters.

Thus, it is possible to determine the type of channels, taking into account what characteristic cluster they belong to. Usually it is clearly possible to single out popular channels, channels with an active audience, small channels, and also by betweenness centrality it is possible to determine the channel role in the network.

It is worth mentioning that it is possible to analyze the Telegram channel network itself, but the described cluster analysis of profiles allows obtaining additional information. For example, using the Louvain algorithm, divide the ***Education*** and ***Finance*** channel networks into communities. In the ***Education*** graph, the algorithm identifies 5 communities, while in ***Finance***, it identifies 9 communities. Let us see how many channels of each digital profile type are presented in each community (Table 6 and 7):

Table 6. Number of channels of each type in ***Education*** network clusters.

Type of Cluster	Comm 0	Comm 1	Comm 2	Comm 3	Comm 4
Small	47	45	25	9	18
A lot of emoji in texts	5	17	3	5	5
High betweenness centrality	4	2	2	0	0
Small & emoji in texts	20	30	23	5	31
Popular & active audience	5	1	0	0	0

Table 7. Number of channels of each type in ***Finance*** network clusters.

Type of Cluster	Comm 0	Comm 1	Comm 2	Comm 3	Comm 4	Comm 5	Comm 6	Comm 7	Comm 8
Small	64	330	90	149	46	40	189	203	24
A lot of emoji in texts	2	7	10	6	14	2	5	26	0
Small & emoji in texts	20	44	54	41	30	9	56	77	1
Popular & active audience	2	25	1	9	2	0	8	5	1

As can be seen from the tables, each of the selected communities has representatives from different groups of digital profiles. Accordingly, it is possible to combine both of these network analysis methods: examining different vertices types within a single community or analyzing which communities a particular vertex type has been classified into. It also allows to analyze Telegram-channels from two sides: using the network

topology and using the channels data. The network topology can show what community or topic the selected channel belongs to, what function the node performs in the network, and the channel metadata characterizes it and its audience in more detail.

5 Conclusion

In this paper, the scheme to construct digital profiles of Telegram channels is shown. To determine the channel digital profile the set of characteristics is considered. Accordingly, each Telegram channel corresponds to a normalized vector of 4 values, which will be its digital profile.

Examples of using Telegram channels digital profiles for comparative and cluster analysis are also provided. Thus, various examples show how channels can be identified by type: popular or small channels, channels with active or inactive audience, channels with different level of emoji usage in posts, channels with different value of betweenness centrality. In combination with the identification of communities on the graph of Telegram channels interaction, this allows for a more detailed analysis of networks and Telegram channels in particular.

References

1. Newman, M.E.J.: The structure and function of complex networks. *SIAM Rev.* **45**, 167–256 (2003)
2. Barabasi, A.L.: Scale-Free Networks: A Decade and Beyond. *Science* **325**(5939), 412–413 (2009). <https://doi.org/10.1126/science.1173299>
3. Chepovskiy, A.A.: On Implicit Communities on the Graph of Interacting Objects. *Russian Journal of Cybernetics* **4**(1), 56–64 (2023)
4. Popov, V.A., Chepovskiy, A.A.: Telegram Messenger Data Import Models. *Vestnik Novosibirskogo gosudarstvennogo universiteta. Seriya: Informacionnye tekhnologii* **20**(2), 60–71 (2022)
5. Chepovskiy, A.A.: Analysis of Graphs of interacting objects. M.: National open university “INTUIT”, p. 270 (2022)
6. Willaert, T.: A computational analysis of Telegram’s narrative affordances. *PLoS ONE* **18**(11) (2023)
7. La Morgia, M., Mei, A., Mongardini, A.M., Wu, J.: Uncovering the Dark Side of Telegram: Fakes, Clones, Scams, and Conspiracy Movements. (2021). <https://arxiv.org/abs/2111.13530>.
8. Popov, V.A., Chepovskiy, A.A.: About models to construct a graph of interacting objects in a network of Telegram channels. *Voprosy kiberbezopasnosti* **3**(61), 105–112 (2024)



Fair-mod: Fair Modular Community Detection

Georgios Panayiotou^(✉) and Matteo Magnani

InfoLab, Department of Information Technology, Uppsala University,
Uppsala, Sweden
georgios.panayiotou@it.uu.se

Abstract. Despite the recent interest in fairness-aware graph clustering, most existing community detection approaches have not yet been extended to include measures of fairness in the community detection process. In this paper, we introduce Fair-mod, a group fairness-aware method for community detection optimizing for both modularity and fairness. We evaluate our method on real-world social network datasets, highlighting the trade-offs between modularity and fairness. We also compare our approach with state-of-the-art fair graph clustering based on spectral methods.

Keywords: fairness · community detection · modularity · graph clustering · social networks

1 Introduction

Given the increasing usage of automation in decision making systems, algorithmic fairness has emerged as an important way of studying algorithms, looking beyond features that have traditionally received more attention such as accuracy and execution time [6, 11, 25]. Concepts of algorithmic fairness have been discussed in the context of a variety of critical decision-making systems, such as pre-trial risk assessment [22], credit scoring [9], and education [18].

One emerging area where we expect algorithmic fairness to have an important societal impact is graph clustering, also known as community detection. This is due to the usage of graph data to represent social networks in very large platforms, including online social media and online retail systems. In these systems, community detection methods are used to group together similar users and provide similar recommendations to users belonging in the same community [12, 30]. In this context, the application of community detection algorithms only based on sub-graph density, as are the majority of state-of-the-art graph clustering algorithms, may have detrimental effects. For example, in online social media platforms such as Facebook and X [21, 27], individuals with extremely similar interests and political views would likely be grouped together, leading to filter bubbles and reduced access to the diversity of opinions and facts that is needed to support democratic societies [24]. Instead, we need methods to identify well-connected communities to ensure relevance of the provided services (e.g. recommendations), but at the same time maintain a diverse representation of

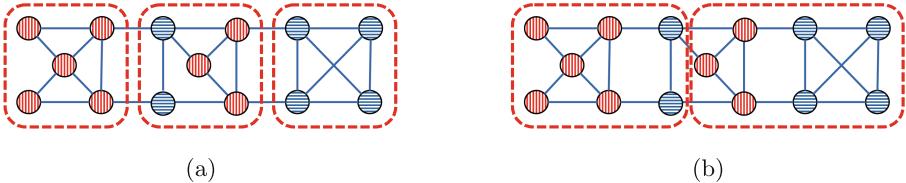


Fig. 1. Two clusterings of the same graph: (a) a highly modular clustering, with two completely non-diverse clusters, and (b) two more diverse clusters, still highly modular

demographics in the communities, so as to avoid bias stemming from disproportional representation.

Unfortunately, while fairness has been extensively examined in the field of clustering in general, it has received comparatively less attention in the context of graph clustering [10, 26]. So far, research on fair community detection has focused primarily on spectral clustering methods [16, 19, 29], while there is no fairness-aware community detection algorithm optimizing the concept of modularity, which is used significantly more often in practical applications [15]. Recent work on fairness has targeted modularity-based clustering, but only providing an assessment of the fairness of the results of modularity-based community detection, without designing algorithms able to identify fair and modular communities [20].

Modularity captures how well-connected individuals in the same group are, in comparison to individuals belonging to separate groups [23]. However, only optimizing modularity without considering fairness can lead to communities without proportional representation of sensitive demographics. Consider, for example, the graph in Fig. 1, with 14 nodes and a sensitive node attribute with two values (for example, male and female), represented respectively by vertical red lines and horizontal blue lines. A community detection method optimizing modularity alone identifies the three well-connected communities in Fig. 1a. However, two of the three communities are only containing one value (in this case, all males, or all females). A fair community detection algorithm should aim to keep a more balanced male-female ratio within the produced communities, while also maintaining a high modularity score, as is the case for the two communities in Fig. 1b.

In this paper, we present preliminary results concerning the incorporation of fairness constraints into modularity-based clustering. We provide the following contributions. We introduce Fair-mod, a modularity-based clustering method considering demographic fairness constraints. We also discuss alternative ways to incorporate fairness into the modularity objective function. We evaluate the proposed algorithm on various social network datasets, assessing the impact of different trade-offs between modularity and fairness on the resulting communities. Finally, we compare the proposed method with a state-of-the-art approach based on spectral clustering, showing that our method is significantly faster and can identify significantly more modular communities.

2 Related Work

The field of algorithmic fairness in clustering has attracted substantial attention recently [3, 7]. A prominent example is the seminal work of Chierichetti et al. [8] codifying the notion of disparate impact [14]. Other relevant works include methods for fair correlation- [1], hierarchical- [2] and probabilistic clustering [13].

Algorithmic fairness has also been explored within graph mining applications in general [10], and within the context of social network analysis [26]. The task of fairness-aware graph clustering, however, has received less attention compared to its non-graph counterpart [10, 26]. Most of the fair graph clustering algorithms introduced consider group (or demographic) fairness constraints. Recently, a spectral graph clustering algorithm considering group fairness constraints was introduced [19], which was later extended for scalability [29], with a fairness function based on sensitive group balance. Demographic fairness was also explored in the context of correlation clustering [5, 17]. Individual fairness constraints for graph clustering have been implemented by Ghodsi et al. introducing a nonnegative matrix tri-factorization solution [16].

Finally, a demographic fairness definition based on modularity has been proposed in [20]. While based on modularity, when applied to a community, this definition only quantifies fairness. Instead, in this paper, we evaluate the integration of fairness and modularity into an objective function, so that both can be considered by the clustering algorithm. The fairness measure proposed in [20] is used to evaluate the result of the Louvain algorithm [4] when applied to social networks. The study highlights how these results may include low-fairness clusters. This observation provides an important motivation for our work. However, the authors of [20] do not propose fairness-aware clustering algorithms, which we introduce and evaluate in this paper.

To our best knowledge, this paper is the first to provide a fairness-aware community detection algorithm also explicitly optimizing modularity.

3 Fair Modular Community Detection

In this work, we consider the following problem: given a feature-rich network $G = (V, E)$ with a sensitive node attribute S , and a weight α controlling the trade-off between modularity and fairness, we want to obtain a partitioning $C(G)$ of the graph's vertices maximizing a weighted sum of modularity and fairness. Therefore, we want to obtain a partitioning $C(G)$ that maximizes the following objective function:

$$Obj = \alpha * Q + (1 - \alpha) * F \quad (1)$$

where Q is the global modularity score for the partitioning $C(G)$ and F is the fairness score assigned to the current partition $C(G)$.

The formula above is a very common way to handle two objective functions, and allows us to use existing optimization algorithms. However, when used in this context, this approach raises two questions. The first is the definition of F , which must be usable inside the chosen optimization algorithm. This aspect is

discussed in the remainder of this section. The second question is empirical: the extent to which the two parts of the objective function can both be optimized depends again on the choice of the fairness function, but also on the choice of α and on the distribution of attribute values with respect to the position of the edges in the input graph. We address this second question through experiments on real data in the next section.

Fairness Score. In this work, we consider a group fairness definition aimed at achieving a proportional representation of the sensitive attribute groups. We use the notion of group fairness of [8], measuring the balance between nodes having different values for S in each cluster. For simplicity, we consider a single protected group with two colors, red and blue, although the definition can be extended for multiple protected groups, as in [19]. The fairness of community C_i is defined as:

$$F(C_i) = \min \left(\frac{|B(C_i)|}{|R(C_i)|}, \frac{|R(C_i)|}{|B(C_i)|} \right) \quad (2)$$

where $B(C_i)$ and $R(C_i)$ is the count of red and blue nodes in community C_i respectively.

In [8] the fairness for the whole clustering is defined as the minimum balance over all partitions. While this is a reasonable way to measure the fairness of a given clustering, it is not a good choice for our objective function: improvements in the fairness of one community may not result in improvements of the overall fairness score, if there is another community with an even lower score. Instead, we need an objective function that increases when the fairness of any of the communities improves.

A way to address this problem is to use the average fairness over all communities instead of the minimum. However, this would favor clusterings made of a large number of two-node communities, with the two nodes in the same cluster having different colors. Therefore, average fairness is also an imperfect choice for our objective function.

As a result of this discussion, we argue that an appropriate choice for the F function is the weighted average of $F(C_i)$ over all communities, considering both the ratio of the sensitive groups and the relative size of the community into our overall fairness score:

$$F = \frac{\sum_{C_i \in C(G)} (|C_i| * F(C_i))}{|V|} \quad (3)$$

In our experiments, we not only use this formula to assess the quality of the results, but also evaluate its appropriateness within the chosen optimization approach for the objective function in Eq. 1.

Algorithm. The algorithm tested in the next section uses the optimization approach of the Louvain algorithm [4] applied to our objective function, henceforth *Fair-mod*. The algorithm first assigns each node in the network to its own community. It then merges neighbouring communities if the move yields an overall

Table 1. Network characteristics

Network	#Nodes	#Edges	Attribute	#Blue nodes	#Red nodes
Facebook	4,039	88,234	Gender	2,503	1,536
Deezer	28,281	92,752	Gender	15,743	12,538
Twitch Gamers	168,114	6,797,557	Maturity	89,081	79,033
Pokec-a	1,138,314	10,794,057	Age	623,704	514,610
Pokec-g	1,632,640	22,301,602	Gender	828,304	804,336

gain in the objective function. Instead of using modularity as the objective function, as in the original Louvain algorithm, Fair-mod calculates the gain for the objective function in Eq. 1, using the group fairness definition of Eq. 3 instead.

4 Experimental Evaluation

In this section, we evaluate the Fair-mod algorithm on various social network datasets. We specifically examine the trade-off between modularity and fairness as the weight α increases, and the number of communities obtained with respect to the Louvain method. We also compare our findings with the recently proposed Scalable Fair Spectral Clustering (sFairSC) algorithm [29].

Datasets. For this illustration of the algorithm, we consider real datasets from the Stanford Large Network Dataset Collection, listed in Table 1. Specifically, we select the following networks:

- *Facebook*¹: The dataset includes Facebook friend lists of survey participants.
- *Deezer*²: The social network consists of European users of the Deezer online social network, where an edge between two users means they both follow the same artist online.
- *Twitch Gamers*³: The network includes Twitch users, and their mutual streamer following relationships.
- *Pokec*⁴: The dataset consists of users of the Pokec online social network, and friendships on the site between them. For this dataset, we consider both age (*Pokec-a*) and gender (*Pokec-g*) of the users as sensitive attributes. Following the paradigm of [20], for the age attribute we split the nodes into red and blue according to the median age, after removing nodes without an age value set in their profile.

¹ <http://snap.stanford.edu/data/ego-Facebook.html>.

² <https://snap.stanford.edu/data/feather-deezer-social.html>.

³ https://snap.stanford.edu/data/twitch_gamers.html.

⁴ <https://snap.stanford.edu/data/soc-Pokec.html>.

Settings. We implement Fair-mod as described above, by modifying the open source code for Louvain community detection in the NetworkX library⁵. We compare our results against the original NetworkX implementation of Louvain, as well as an implementation of sFairSC in Python, available online⁶.

Results. In Fig. 2, we report the values of modularity and fairness, along with the number of communities, for partitions obtained through Fair-mod for various values of the weight parameter α . Our baseline for comparison is the Louvain algorithm.

As we can observe from Fig. 2, Fair-mod produces a partition less modular but more fair than Louvain for all values of $\alpha \geq 0.6$ for all networks except for Pokec-g, where this is the case only for $\alpha = 0.9$. We note that these partitions always identify a larger amount of communities than Louvain, sometimes in orders of magnitude.

As expected, for $\alpha = 1.0$ the quality of Fair-mod’s partition is similar to the partition Louvain produces. Small differences in both the obtained modularity and fairness scores, as well as the number of communities obtained, are simply due to the node shuffling done to initialize the algorithm.

However, Fair-mod obtains lower fairness with low values of α (in particular, $\alpha = 0$) than what it can achieve with the best values of α . This is an unexpected result, also different from the well-behaved trend we observed on the small synthetic data on which we tested the correctness of the algorithm, such as the graph in Fig. 1 (not reported here for space reasons). In most experiments, $\alpha = 0$ produces a value of fairness even lower than the one produced by the Louvain algorithm. Here we observe the correlation between ability to improve fairness and number of communities, which we discuss in the next section.

Comparing our algorithm with the recently introduced sFairSC algorithm, we observe that Fair-mod performs better in both partition quality and algorithm performance. Notice in Fig. 3 that the partitions sFairSC yields receive a relatively similar group fairness score as the one produced by Fair-mod, but the modularity score is significantly lower. We also note that the performance of sFairSC under the default settings (≈ 2 minutes) is much slower than Fair-mod (≈ 0.7 seconds). As indicated in the figure, we tested sFairSC only on the Facebook data, which is the smallest of the real datasets we used, because of the limited scalability of the algorithm.

5 Discussion

As we see from the experiments, our algorithm is successful at producing communities that, while only slightly less modular than those produced by the Louvain algorithm, improve upon the balance between the sensitive groups. This is apparent for a range of α between 0.6-0.9, while the best effect is seen for $\alpha = 0.9$.

⁵ https://networkx.org/documentation/stable/_modules/networkx/algorithms/community/louvain.html.

⁶ https://github.com/uuinfolab/paper.24_ComplexNetworks_Fair-Mod.git.

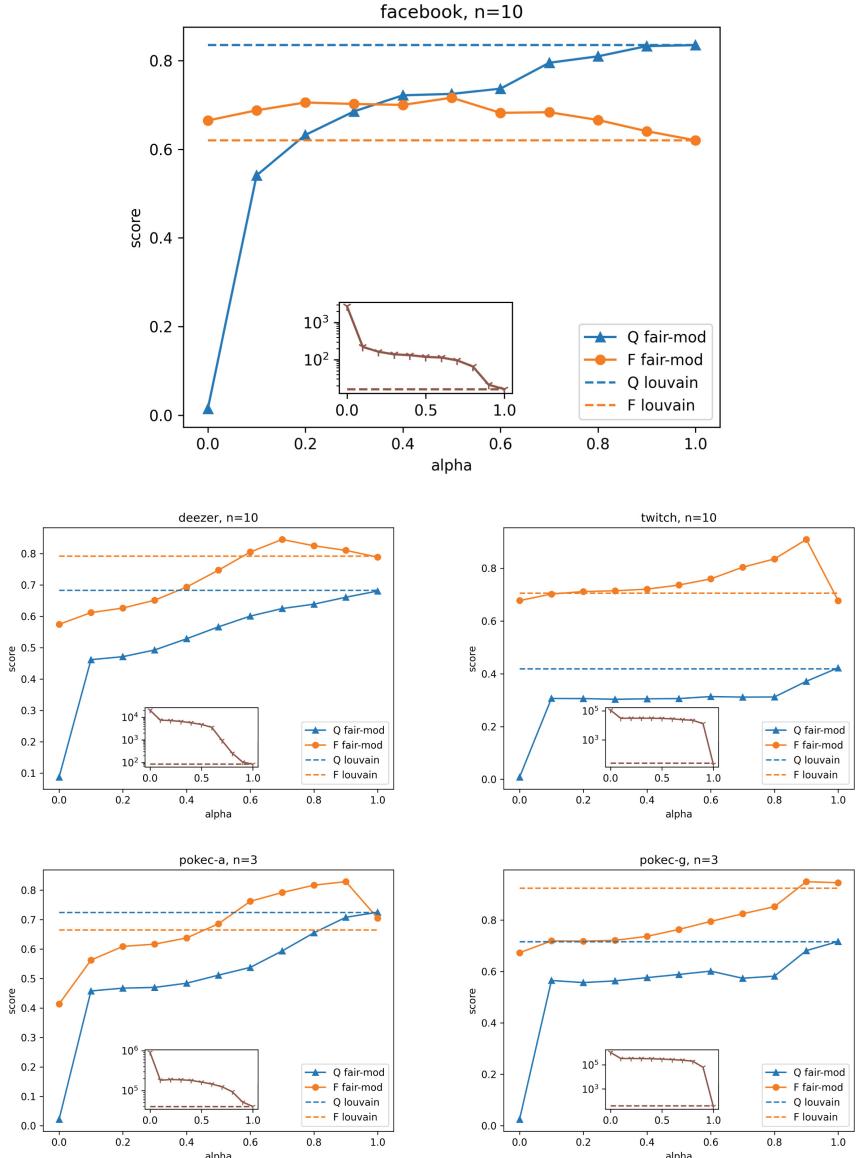


Fig. 2. Comparison of average values of modularity (blue line), fairness (orange line) and number of communities (inset figures – brown line) obtained by Fair-mod over different values of α , versus the respective partition scores for Louvain community detection (dashed lines).

For these values, the algorithm tends to generate a partitioning into a larger number of communities. Merging them together would generate a more modular partitioning, but the overall fairness score would decrease.

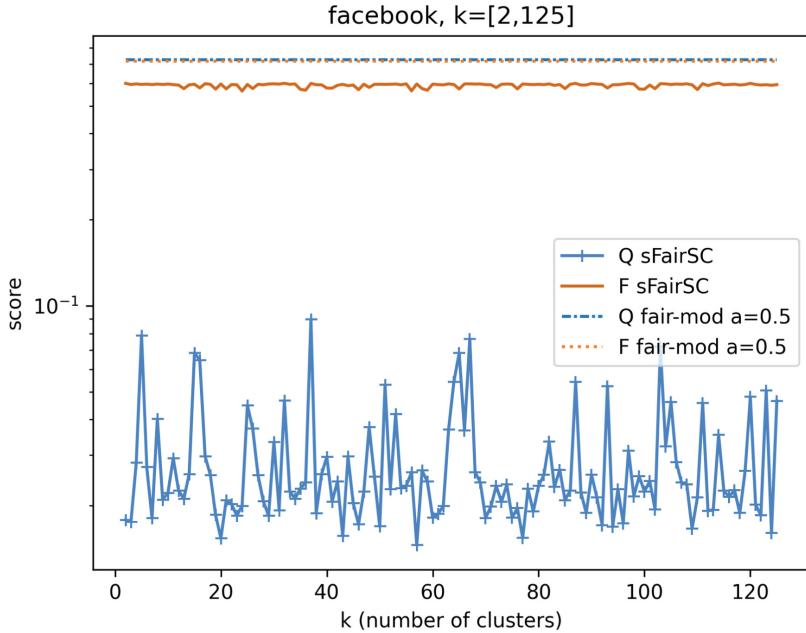


Fig. 3. Comparison of Fair-mod ($\alpha = 0.5$) and sFairSC (for a number of clusters $k \in [2, 125]$) on the Facebook network

Comparing our results with the state-of-the-art method sFairSC, we find that Fair-mod can achieve a much more modular partition for large values of α , while still maintaining a high fairness score. The performance of the greedy Fair-mod algorithm is also significantly better than spectral clustering. As a result, sFairSC cannot execute for larger networks; we also note that, due to the adjacency matrix format needed for spectral clustering, sFairSC cannot load Twitch Gamers, the third largest network in our list.

The experiments also highlight an interesting unwanted behaviour for low values of α . As we discuss in Sect. 3, our assumption when designing the algorithm was that some objective functions for fairness would favor results with a large number of small communities. While the function we chose does not, it still generates a large number of communities when used inside the Louvain optimization approach trying to mostly optimize fairness by setting a low value of α . This is due to the design of the algorithm, as the decision to merge two communities is based on calculating the overall gain on the objective function that the move can yield. When merging communities together at the early stages, the algorithm tries to merge nodes with different colors, easily succeeding especially for $\alpha = 0$ where no structural constraints are imposed apart from the two nodes being adjacent. Once a two-node cluster with two different colors (that is, optimal local fairness) has been produced, the addition of any node would reduce the fairness of the cluster, preventing the algorithm from performing the

move. This behavior can be prevented when structural information is given a sufficient weight, as we see in our experiments. This points towards the need for an improved algorithm, if our goal is to optimize primarily for fairness and not modularity. Other methods of normalizing the modularity and fairness scores should also be explored. This discussion and experimental results indicate the part of the algorithm that has to be modified.

Additionally, we observe that calibrating α to an appropriate value depends on the data. Therefore, while the behavior of α for large values is intuitive (one should set a larger alpha to increase modularity, at the potential expense of fairness), α should in general be treated as a hyperparameter. Being the only hyperparameter (with the possible addition of modularity resolution) and one with a smooth effect, it is also easy to set experimentally. Generally, the maximum scores for the objective function depend on modularity and the balance between the two sensitive groups, and therefore largely depend on the interplay between network structure and attribute values.

This discussion spotlights the need for alternative algorithmic approaches to the fair modular community detection problem. Such alternatives can, for example, integrate other notions of fairness into the objective function, which consider both nodes and edges in a community (e.g. modularity-based fairness in [20]), or consider constraining the produced communities to contain an amount of sensitive nodes each (e.g. constrained modularity community detection [28]).

6 Conclusions

In this paper, we delve into the relatively unexplored problem of fair modular community detection in graphs. We introduce Fair-mod, a group fairness-aware modification of the popular Louvain algorithm based on the balance of sensitive groups in each community. Finally, we evaluate Fair-mod against real social networks and an alternative approach for fair graph clustering, and discuss possible future steps and directions.

Our Fair-mod algorithm is able to produce a set of communities that are slightly less modular but fairer than partitions Louvain yields. We also show that our approach outperforms group-fair spectral clustering algorithms when tuned correctly, while it is also able to produce partitions for large networks because of its significantly lower execution time. The observation that the proposed algorithm fails to consistently yield more fair partitions when less emphasis is placed on maximizing fairness also generated knowledge about the choice of objective function for fairness, and how the Louvain optimization approach could be modified to avoid getting stuck in local maxima.

Future work includes exploring alternative fairness score integrations into the objective function of our algorithm, such as individual fairness constraints [16], or the recently introduced modularity-based fairness, which as mentioned in Sect. 2 cannot be used alone to obtain highly-modular communities [20]. Another potential direction is defining fairness metrics simultaneously considering multiple sensitive attributes and groups. Alternative algorithmic approaches to the

fair modular community detection problem should also be identified, considering the limitations of the Louvain-like approach in yielding partitions of optimal fairness for lower values of α . Finally, the fair modular community detection problem should also be extended to other types of feature-rich networks commonly used in social network analysis, such as multilayer networks.

Acknowledgements. This work has been partly funded by eSENCE, an e-Science collaboration funded as a strategic research area of Sweden. The authors would also like to thank Georgios Fakas for useful remarks on early versions of the manuscript.

Data Availability Statement. The implementation of the Fair-mod algorithm is available at https://github.com/uuinfolab/paper.24_ComplexNetworks.Fair-Mod.git. The datasets used for the experiments are available on the Stanford Large Network Dataset Collection (<https://snap.stanford.edu/data/>).

References

1. Ahmadian, S., et al.: Fair hierarchical clustering. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS 2020, pp. 21,050–21,060. Curran Associates Inc., Red Hook, NY, USA (2020)
2. Ahmadian, S., Epasto, A., Kumar, R., Mahdian, M.: Fair correlation clustering. In: Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, pp. 4195–4205. PMLR (2020). <https://proceedings.mlr.press/v108/ahmadian20a.html>. ISSN: 2640-3498
3. Bera, S., Chakrabarty, D., Flores, N., Negahbani, M.: Fair algorithms for clustering. In: Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019). https://proceedings.neurips.cc/paper_files/paper/2019/hash/fc192b0c0d270dbf41870a63a8c76c2f-Abstract.html
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**(10), P10,008 (2008). <https://doi.org/10.1088/1742-5468/2008/10/P10008>, <https://iopscience.iop.org/article/10.1088/1742-5468/2008/10/P10008>
5. Casel, K., Friedrich, T., Schirneck, M., Wietheger, S.: Fair correlation clustering in forests. In: DROPS-IDN/v2/document/10.4230/LIPIcs.FORC.2023.9. Schloss-Dagstuhl - Leibniz Zentrum für Informatik (2023). <https://doi.org/10.4230/LIPIcs.FORC.2023.9>, <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.FORC.2023.9>
6. Caton, S., Haas, C.: Fairness in machine learning: a survey. *ACM Comput. Surv.* **56**(7), 166:1–166:38 (2024). <https://doi.org/10.1145/3616865>, <https://dl.acm.org/doi/10.1145/3616865>
7. Chhabra, A., Masalkovaitė, K., Mohapatra, P.: An overview of fairness in clustering. *IEEE Access* **9**, 130,698–130,720 (2021). <https://doi.org/10.1109/ACCESS.2021.3114099>, <https://ieeexplore.ieee.org/document/9541160/?arnumber=9541160>. Conference Name: IEEE Access
8. Chierichetti, F., Kumar, R., Lattanzi, S., Vassilvitskii, S.: Fair clustering through fairlets. In: Proceedings of the 31st Conference on Neural Information Processing Systems, pp. 5029–5037 (2017)

9. Das, S., Stanton, R., Wallace, N.: Algorithmic fairness. *Ann. Rev. Finan. Econ.* **15**, 565–593 (2023)
10. Dong, Y., Ma, J., Wang, S., Chen, C., Li, J.: Fairness in graph mining: a survey (2023). <https://doi.org/10.48550/arXiv.2204.09888>, <http://arxiv.org/abs/2204.09888>, ArXiv:2204.09888
11. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ITCS 2012, pp. 214–226. Association for Computing Machinery, New York, NY, USA (2012). <https://doi.org/10.1145/2090236.2090255>, <https://dl.acm.org/doi/10.1145/2090236.2090255>
12. Eirinaki, M., Gao, J., Varlamis, I., Tserpes, K.: Recommender systems for large-scale social networks: a review of challenges and solutions. *Future Gener. Comput. Syst.* **78**, 413–418 (2018). <https://doi.org/10.1016/j.future.2017.09.015>, <https://www.sciencedirect.com/science/article/pii/S0167739X17319684>
13. Esmaeili, S., Brubach, B., Tsepenekas, L., Dickerson, J.: Probabilistic Fair Clustering. In: Advances in Neural Information Processing Systems, vol. 33, pp. 12,743–12,755. Curran Associates, Inc. (2020). https://proceedings.neurips.cc/paper_files/paper/2020/hash/95f2b84de5660ddf45c8a34933a2e66f-Abstract.html
14. Feldman, M., Friedler, S.A., Moeller, J., Scheidegger, C., Venkatasubramanian, S.: Certifying and removing disparate impact. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2015, pp. 259–268. Association for Computing Machinery, New York, NY, USA (2015). <https://doi.org/10.1145/2783258.2783311>
15. Fortunato, S.: Community detection in graphs. *Phys. Rep.* **486**(3), 75–174 (2010)
16. Ghodsi, S., Seyed, S.A., Ntoutsi, E.: Towards cohesion-fairness harmony: contrastive regularization in individual fair graph clustering (2024). <http://arxiv.org/abs/2402.10756>, ArXiv:2402.10756
17. Gullo, F., La Cava, L., Mandaglio, D., Tagarelli, A.: When correlation clustering meets fairness constraints. In: Pascal, P., Ienco, D. (eds.) *Discovery Science*, pp. 302–317. Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-18840-4_22
18. Kleinberg, J., Ludwig, J., Mullainathan, S., Rambachan, A.: Algorithmic Fairness. *AEA Papers Proc.* **108**, 22–27 (2018)
19. Kleindessner, M., Samadi, S., Awasthi, P., Morgenstern, J.: Guarantees for spectral clustering with fairness constraints. In: Proceedings of the 36th International Conference on Machine Learning, pp. 3458–3467. PMLR (2019). <https://proceedings.mlr.press/v97/kleindessner19b.html>. ISSN: 2640-3498
20. Manolis, K., Pitoura, E.: Modularity-based fairness in community detection. In: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining, pp. 126–130. ACM, Kusadasi Turkiye (2023). <https://doi.org/10.1145/3625007.3627518>, <https://dl.acm.org/doi/10.1145/3625007.3627518>
21. Meta: The AI behind unconnected content recommendations on Facebook and Instagram. <https://ai.meta.com/blog/ai-unconnected-content-recommendations-facebook-instagram/>
22. Mitchell, S., Potash, E., Barocas, S., D’Amour, A., Lum, K.: Algorithmic fairness: choices, assumptions, and definitions. *Ann. Rev. Stat. Appl.* **8**, 141–163 (2021)
23. Newman, M.E.J.: Modularity and community structure in networks. *Proc. Nat. Acad. Sci. United States Am.* **103**(23), 8577–8582 (2006). <https://doi.org/10.1073/pnas.0601602103>, <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1482622/>
24. Pariser, E.: *The Filter Bubble: What the Internet is Hiding from You*. Penguin Books (2012). Google-Books-ID: Qn2ZnjzCE3gC

25. Pessach, D., Shmueli, E.: Algorithmic Fairness. In: Rokach, L., Maimon, O., Shmueli, E. (eds.) *Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook*, pp. 867–886. Springer International Publishing, Cham (2023). https://doi.org/10.1007/978-3-031-24628-9_37. URL https://doi.org/10.1007/978-3-031-24628-9_37
26. Saxena, A., Fletcher, G., Pechenizkiy, M.: FairSNA: algorithmic fairness in social network analysis. *ACM Comput. Surv.* **56**(8), 213:1–213:45 (2024). <https://doi.org/10.1145/3653711>, <https://dl.acm.org/doi/10.1145/3653711>
27. Twitter: Twitter’s recommendation algorithm. https://blog.x.com/engineering/en_us/topics/open-source/2023/twitter-recommendation-algorithm
28. Viles, W.D., O’Malley, A.J.: Constrained community detection in social networks. *New England J. Stat. Data Sci.*, 1–12 (2023). <https://doi.org/10.51387/23-NEJSDS32>. URL <https://nejsds.nestat.org/journal/NEJSDS/article/36>. Publisher: New England Statistical Society
29. Wang, J., Lu, D., Davidson, I., Bai, Z.: Scalable spectral clustering with group fairness constraints. In: Proceedings of The 26th International Conference on Artificial Intelligence and Statistics, pp. 6613–6629. PMLR (2023). <https://proceedings.mlr.press/v206/wang23h.html>. ISSN: 2640-3498
30. Ying, J.C., Shi, B.N., Tseng, V.S., Tsai, H.W., Cheng, K.H., Lin, S.C.: Preference-aware community detection for item recommendation. In: 2013 Conference on Technologies and Applications of Artificial Intelligence, pp. 49–54 (2013). <https://doi.org/10.1109/TAAI.2013.23>, <https://ieeexplore.ieee.org/document/6783842>. ISSN: 2376-6824



Improved Community Detection Using Stochastic Block Models

Minhyuk Park, Daniel Wang Feng, Siya Digras, The-Anh Vu-Le, George Chacko, and Tandy Warnow

Siebel School of Computing and Data Science, University of Illinois
Urbana -Champaign, Urbana, IL 61801, USA
`{minhyuk2, chackoge, warnow}@illinois.edu`

Abstract. Community detection approaches resolve complex networks into smaller groups (communities) that are expected to be relatively edge-dense and well-connected. The stochastic block model (SBM) is one of several approaches used to uncover community structure in graphs. In this study, we demonstrate that SBM software applied to various real-world and synthetic networks produces poorly-connected to disconnected clusters. We present simple modifications to improve the connectivity of SBM clusters, and show that the modifications improve accuracy using simulated LFR networks.

Keywords: Connectivity · Stochastic Block Model · Clustering

1 Introduction

Community detection is a task in which nodes of a network are partitioned into subsets, each called a community or a cluster (the terms are interchangeable in this manuscript) [2, 12]. Communities do not have to cover an entire network [11], and may also be overlapping [5, 9, 23].

Many community detection methods are based on optimization criteria that reflect one or more of the following properties: preference for clusters that are dense and so have many intra-cluster edges; that are separated from the rest of the network and so have relatively fewer inter-cluster edges; and finally are well-connected, meaning that they do not have small edge cuts (i.e., sets of edges where deleting the edges but not the endpoints separates the cluster into two pieces) [15, 21].

Despite the natural expectation that clusters should be well-connected, this property does not result from some clustering methods [21, 25]. We have previously reported that the Leiden algorithm [21], Infomap [19], Iterative-K-core Clustering (IKC) [22] and Markov Clustering (MCL) [1] community detection algorithms produce clusters that are not well connected [15]. Moreover, [21] documented that the Louvain algorithm can produce disconnected clusters, i.e., clusters that have two or more connected components.

Modifying such clusters to improve connectivity is a logical remediation, and the Connectivity Modifier [15] is one such method that recursively modifies an input clustering to ensure that all final clusters meet a user-provided well-connectedness criterion that depends on the size of the smallest edge cut for the cluster. For the default setting for this criterion, [15] said that a cluster would be considered “well-connected” if the number of edges in the smallest edge cut was strictly greater than $\log_{10}(n)$, where n is the number of nodes in the cluster, and otherwise it was considered poorly connected. While other thresholds can be considered, we use the same setting in this study since it is a very slow-growing function, and thus provides a very mild constraint.

Here, we report on a study examining clustering using Stochastic Block Models (SBMs) [10] on both real-world and synthetic networks. On a collection of more than 100 real-world networks, we find that the SBM clustering methods in graph-tool [17] frequently produce disconnected clusters. We explore three techniques for modifying the clustering to improve the connectivity: simply returning the connected components (CC), repeatedly finding and removing small edge cuts until all clusters meet the default criterion to be considered well-connected (WCC), or applying the recursive Connectivity Modifier method to the clustering. We show that these modifications improve accuracy on LFR networks, with the greatest improvement obtained using WCC.

The rest of this manuscript is as follows. In Sect. 2, we describe the experimental study. We present the results of our experiments on real-world and synthetic LFR networks [8] in Sect. 3. We discuss these results in Sect. 4 and conclude in Sect. 5.

2 Materials and Methods

Due to space constraints, full details are provided in the Supplementary Materials [14].

2.1 Networks

Real-World Networks. We collected a set of 122 real-world networks that range in size from 11 to 13,989,436 nodes. Of these, 120 are from the Netzsleuder network catalogue [18] and we also include the Orkut network (3,072,441 nodes) and the Curated Exosome Network (13,989,436 nodes) [15]. The Netzsleuder network set includes 10 small networks with at most 1000 nodes, 103 medium-sized networks between 1000 and 1,000,000 nodes, and 7 networks with at least 1,000,000 nodes (see Supplementary Materials) [14] for the full list of networks). All real-world networks used in this study were pre-processed to remove self-loops and parallel edges and were treated as unweighted and undirected.

Synthetic Networks. We used synthetic networks that were generated using the LFR [8] software for a previous study [15]. These networks were generated

based on parameters obtained from clusterings computed on five real-world networks using the Leiden algorithm [20] optimizing either the modularity criterion [13] or the Constant Potts Model criterion [21] with five different resolution values (0.0001, 0.001, 0.01, 0.1, 0.5). These LFR networks range in size from 34,546 nodes to 3,774,768 nodes. As reported in [15], a few of these LFR networks had a high incidence of disconnected ground-truth clusters and were not suitable for analysis in this study.

2.2 Stochastic Block Models

We used SBM implemented in graph-tool [17] as a clustering method with the option of three different models: degree-corrected [7], non degree-corrected [4], and planted partition [24]. For each network we clustered using SBMs, we selected the model that had the best fit—i.e., the one with the lowest description length—as our preferred SBM model. We refer to that model as the “selected SBM”, and use it in subsequent post-processing treatments.

2.3 Post-processing Treatments to Improve Connectivity

The Connectivity Modifier (CM) [15] pipeline is designed to modify clusterings in order to ensure that all clusters are well-connected and that no cluster is too small. In prior work [15], we found that the CM pipeline typically improved Leiden clustering accuracy on synthetic networks, and that when it reduced accuracy this was due to removing small clusters. Hence, in this study, we have eliminated the filtering of small clusters, and restricted the CM pipeline to modifying clusterings in order to ensure edge-connectivity.

We evaluate the use of this simplified CM approach as well as two other post-processing treatments, each of which takes as input a clustering \mathcal{C} of a network N , and modifies it, if necessary, to ensure some standard for edge-connectivity. The three treatments we study are:

- **CC (Connected Components)**: If a cluster is disconnected, we return each of its connected components as a cluster.
- **WCC (Well-Connected Clusters)**: We modify clusters by repeatedly removing small edge cuts of size at most $\log_{10}(n)$ until each cluster is well-connected. To find small edge cuts, we use VieCut [3]. Each of these pieces is then examined for well-connectedness and further processed, if needed.
- **CM (Connectivity Modifier)**: We apply the inner loop of the Connectivity Modifier pipeline [15]. If a cluster C has a small edge cut, then removal of the edge cut divides C into two subsets, and each of these is then “re-clustered” using the same clustering method used to produce the input clustering \mathcal{C} . These clusters are then added back into the iterative algorithm, which checks each cluster for being well-connected. Each iteration finds and removes small edges cuts, and then reclusters the two sets. The iteration stops when the cluster satisfies the edge-connectivity criterion.

2.4 Evaluation

We report cluster statistics, including percent of clusters that are connected, percent well-connected, and percent poorly connected. We also report cluster size distributions and node coverage after restriction to clusters of size at least two. For synthetic networks, we report accuracy, measured using three standard criteria: Normalized Mutual Info (NMI), Adjusted Rand Index (ARI), and Adjusted Mutual Info (AMI). For all three accuracy criteria, we used the implementation provided by the Scikit-learn library [16].

3 Performance Study and Results

Here we describe the experiments we performed and the results we obtained.

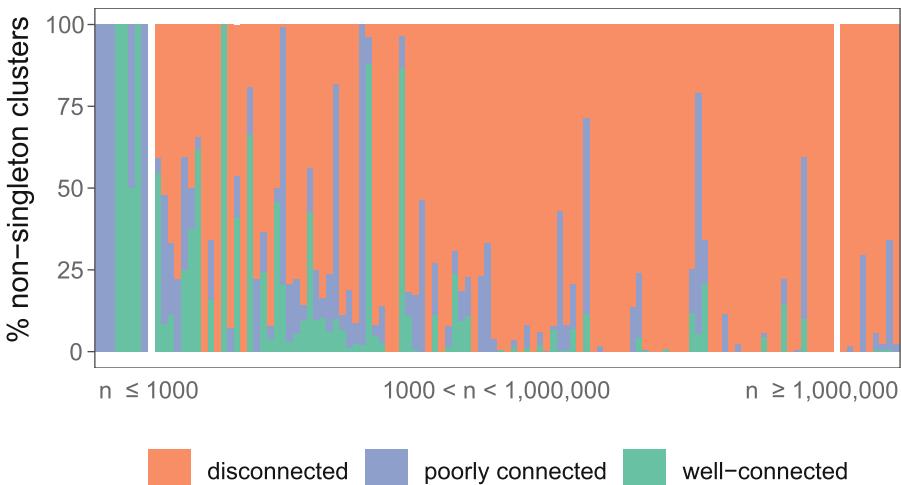


Fig. 1. Experiment 1: Cluster Connectivity of SBM on 120 Real-World Networks. Percentage of disconnected, poorly connected, and well-connected clusters are shown for the selected SBM clustering of 120 real world networks. Each colored bar represents a single network, white bars separate the network groups into small, medium, and large. This figure does not include results for two networks from the initial set of 122 networks, as the selected SBM model returned no non-singleton clusters. The networks are sorted by the number n of nodes

3.1 Experiments

We conducted three experiments:

- Experiment 1: We evaluate the connectivity profile of SBM clusterings on real-world networks.

- Experiment 2: We evaluate the impact of returning the connected components of the clusters of SBM clusterings on real-world networks.
- Experiment 3: We evaluate the impact of our three treatments on clustering accuracy on synthetic networks.

All experiments were performed on the Illinois Campus Cluster with maximum computational resources limit set to 72 h of runtime, 256 GB of RAM, and 16 cores of parallelism.

3.2 Experiment 1: Connectivity of SBMs

In this experiment, we examined the connectivity profile of clusters generated by SBM. Figure 1 shows the cluster connectivity status for the selected SBM model (Sect. 2.2) on each of the networks, which are sorted from left to right by the number of nodes, which range in size from 11 to 13,989,436 nodes. Here, red indicates that the cluster is disconnected, blue indicates poorly connected, and green indicates well-connected. For networks with at most 1000 nodes, clusters are connected, and often well-connected. Above 1000 nodes, however, the clusters in the selected SBM are very often disconnected, and most clusters are disconnected for most of the networks in the upper half of the size range.

3.3 Experiment 2: Impact of Treatments on Real-World Networks

Table 1. Impact of Treatment on Node Coverage on Real-World Networks
 For small, medium, and large network groups, the node coverage (i.e., percentage of nodes in non-singleton clusters) is shown for the selected SBM before and after treatment. On one of the medium networks, WCC ran into a memory error with 256GB of RAM, hence the results for that network are omitted from this calculation

clustering	Node Coverage		
	small	medium	large
Selected SBM	62%	100%	100%
Selected SBM - CC	62%	48%	45%
Selected SBM - WCC	55%	36%	25%
Selected SBM - CM	55%	25%	17%

The three post-clustering treatments we apply operate by breaking an input clustering into sub-clusters, and so will change the cluster size distribution, the number of clusters, and even node coverage (i.e., the percentage of nodes in non-singleton clusters). Specifically, if in the process singleton clusters are created, then the node coverage, which is calculated based on non-singleton clusters, can also reduce.

We first examine node coverage (Table 1). Reductions in node coverage are relatively small on the small networks, but all three treatments produce large reductions on the medium and large networks. The largest reductions are for CM, and the smallest are for CC, with WCC in between. However, even CC produces a large reduction in node coverage. Since node coverage is the percentage of nodes in non-singleton clusters, this reduction can only occur because enough nodes are placed in clusters where they do not have any neighbors.

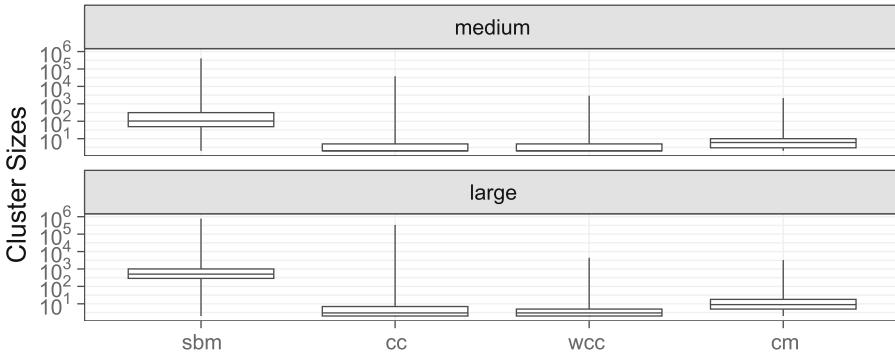


Fig. 2. Experiment 2: Impact of Treatment on Cluster Sizes of Medium and Large Real-World Networks. The distribution of non-singleton cluster sizes is shown as a boxplot for the selected SBM and its treatments. The y-axis is plotted on a log scale. The whiskers indicate the minimum and maximum cluster sizes in all of the networks in the group. Both groups and treatments have minimum cluster size of 2 for SBM clusterings whether treated or not, but differ in the medians and maxes, as follows. Medium group median/max: SBM: 103/403801, SBM+CC: 2/38539, SBM+WCC: 2/2966, SBM+CM: 6/2169. Large group median/max: SBM: 507/777770, SBM+CC: 3/337018, SBM+WCC: 3/4387, SBM+CM: 9/3258

We next examine the impact on cluster size distribution (Fig 2). For both medium-sized networks (top) and large networks (bottom), the median cluster size before treatment is much larger than the final median cluster size after treatment, and this holds for all three treatments. Moreover, the majority of clusters are dramatically reduced in size by the treatments. Even CC, which only modifies the clusters to return connected components, produces a large impact on the cluster size distribution. The cluster sizes seem to be impacted slightly less when CM treatment is applied compared to CC or WCC, both of which have similar impacts on the cluster sizes regardless of network size.

Finally, we examine the impact on the number of non-singleton clusters (Supplementary Materials). All three treatments increase the number of such clusters, and on average CM produced the smallest number of non-singleton clusters, CC produced the next smallest, and then WCC, which produced the most non-singleton clusters.

3.4 Experiment 3: Impact of Treatment on Synthetic Networks

In order to assess the impact of these treatments beyond empirical properties of clusterings, we use synthetic LFR networks with ground truth clusterings to capture the effect treatment has on clustering accuracy. We evaluated NMI, ARI, and AMI accuracies on the LFR networks tested. Recall that some LFR networks had disconnected clusters (i.e., the LFR networks based on CEN clustered using Leiden-CPM with $r = 0.1$ or $r = 0.5$, and the wiki_topcats clustered using Leiden-CPM with $r = 0.5$). On the LFR network for cit_patents with $r = 0.5$, WCC treatment could not produce a clustering within our runtime limit of 72 h when starting with the selected SBM clustering, and so had a “time-out”.

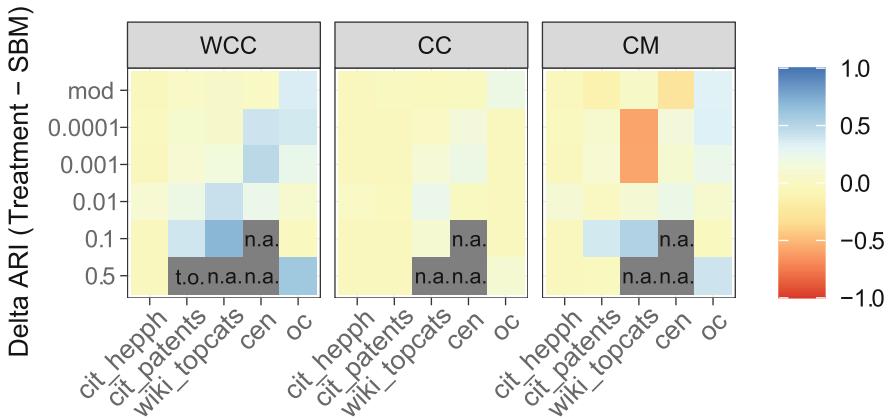


Fig. 3. Experiment 3: Impact of Treatment on ARI Scores of Selected SBM (heatmap). Each LFR network is based on a Leiden clustering of a real-world network, with the column indicating the real-world network and the row specifying the optimization problem (either modularity or CPM for a given resolution value). Blue indicates that post-processing using the corresponding treatment improves ARI accuracy for the clustering method, orange and red indicate that treatment hurts ARI accuracy, and yellow indicates neutral impact. We use “n.a.” to indicate that a network was either not used because of too many disconnected ground-truth clusters or that the LFR software failed to generate the network, and “t.o.” to indicate that WCC failed to complete within 72 h

Fig 3 shows that WCC and CC treatments range from neutral (yellow) to beneficial (blue) with respect to the ARI accuracy of SBM clusterings, but WCC improvements are both more frequent and larger than CC improvements. In comparison, CM can even be detrimental. The relative benefit of WCC over CC and CM holds as well for NMI and AMI (Supplementary Materials), but for those criteria the impact is generally lessened. Overall, therefore, WCC is the preferred treatment for SBM on these networks.

We explore the impact of WCC in greater detail, noting the ARI accuracy for SBM and the final accuracy for SBM-WCC (Fig 4). Note that results are

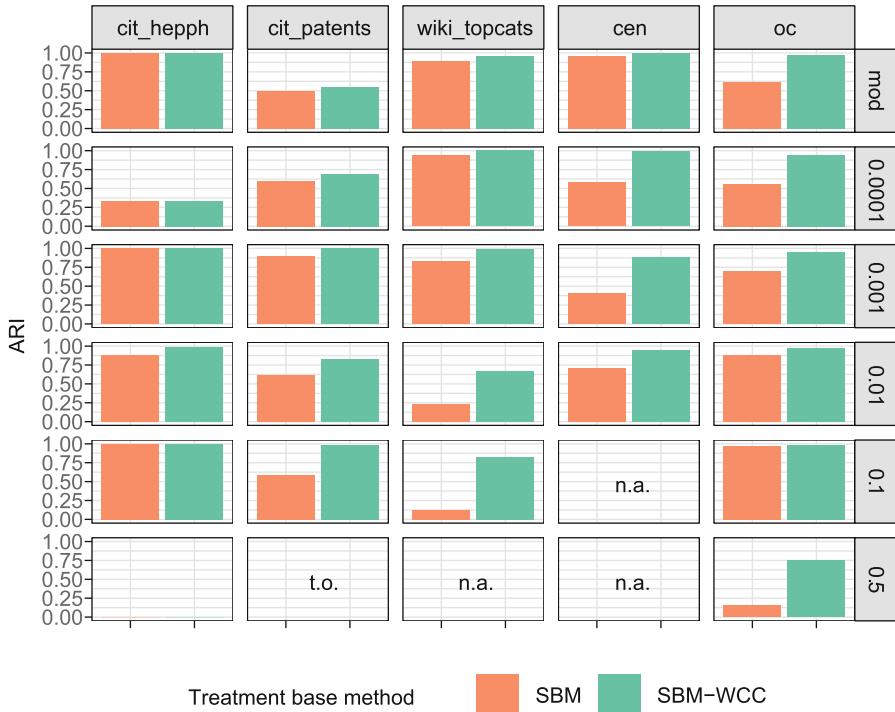


Fig. 4. Experiment 3: Impact of WCC Treatment on ARI Scores of Selected SBM (bar chart). Boxes marked as “n.a.” are for conditions where the LFR networks had too many disconnected ground truth clusters or failed to generate; “t.o.” indicates a failure to complete within 72 h. On the LFR network for cit_hepph with $r = 0.5$, both the selected SBM model and its follow-up WCC yielded 0.0 ARI accuracy

not shown (marked as “n.a.”) for some LFR networks (i.e., the 0.5 CEN, 0.1 CEN, and 0.5 wiki_topcats), because they have many disconnected ground truth clusters or failed to generate, as reported in [15]. On the 0.5 cit_patents network, WCC treatment could not produce a clustering within our runtime limit of 72 h when starting with the selected SBM clustering, and so is marked as a time-out (“t.o.”).

In every case, SBM-WCC is at least as accurate as SBM. Moreover, there are many cases where the benefit from WCC treatment is very large (e.g., Open Citations with resolution value 0.5, the Curated Exosome Network with resolution value 0.001, wiki_topcats with resolution values 0.01 and 0.1). Finally, most of the cases where WCC has at most a small positive impact are for the cases where SBM is already highly accurate, with close to 1.00 ARI accuracy, so that there is little room for improvement.

We also examined the models selected by SBM on the LFR networks. We found that on the LFR networks, the non degree corrected SBM never yielded the lowest description length, and that typically the planted partition model had

the lowest description length (Supplementary Materials). We also saw that the lowest description length model resulted in the best NMI/ARI accuracies on the LFR networks tested.

4 Discussion

Summary of Trends. As seen in Figs. 3 and 4, both CC and WCC post-processing treatments improve accuracy for SBM clusterings on simulated datasets, and WCC is particularly beneficial for accuracy. CM sometimes improves accuracy but sometimes reduces accuracy, and so is not recommended. However, the impact on cluster size and node coverage produced by even the simple CC technique is noteworthy. Our prior work [15] showed that CM improved Leiden clustering accuracy, which is different from what we observe here for SBM clusterings. While the reason for this is not clear, the tendency of SBM to produce disconnected clusters may be part of the explanation.

Impact of the Description Length Formula. Given that the construction of SBMs within graph-tool produces disconnected clusters, we now consider how the code operates. Recall that this approach seeks to generate SBMs that optimize the description length, and this is a minimization problem. Let

- A be the adjacency matrix,
- b be the cluster (block) assignment,
- k be the degree vector (induced by A),
- and e be the edge count matrix (induced by A and b).

In Eq. (1) we provide the formula for the description length of a clustering b of a network given by its adjacency matrix A (i.e., $\text{DL}(A,b)$) under the Degree Corrected (DC) model:

$$\text{DL}(A,b) = -\log p(A|b,e,k) - \log p(k|b,e) - \log p(b) - \log p(e) \quad (1)$$

Note that the description length is calculated as the sum of various components: the negative logarithm of the model likelihood (i.e., $-\log p(A|b,e,k)$) and the negative logarithm of each of the priors.

In our analyses (Supplementary Materials), we observed that the model likelihood without priors favors connected clusters returned by the CC treatment. In contrast, certain priors heavily penalize having many clusters, leading to a worse description length for the clustering returned by the CC treatment.

We provide an example of this phenomenon on a real-world network, `linux`, in Table 2; in the Supplementary Materials, we show that the trends observed on this network are also observed in the other real-world networks. One clustering is from SBM(DC) (i.e., the degree corrected SBM output) and the second clustering is from SBM(DC)-CC (i.e., the result of running the CC treatment on the degree corrected SBM output). The SBM(DC) clustering has a lower description length than SBM(DC)-CC, and hence the untreated SBM clustering

Table 2. Breakdown of Description Lengths on the linux real-world network
The last row is the sum of the values in the first four rows. The ratio is SBM(DC)-CC
SBM(DC), so that values less than 1.0 favor SBM treated by CC and values greater
than 1.0 favor untreated SBM. Bold text indicates the preferred clustering.

Quantity	SBM(DC)	SBM(DC)-CC	Ratio
$-\log p(A b, e, k)$	699228.26	315644.88	0.45
$-\log p(k b, e)$	95737.43	45066.47	0.47
$-\log p(b)$	147018.92	256817.11	1.75
$-\log p(e)$	50786.40	1584554.98	31.20
$\text{DL}(A, b)$	992771.01	2202083.44	2.22

is the preferable clustering with respect to the minimization of the description length under the degree corrected model. However, although $-\log p(A|b, e, k)$ and $-\log p(k|b, e)$ for the SBM(DC)-CC clustering are lower, $-\log p(b)$ and $-\log p(e)$ for SBM(DC)-CC clustering are higher, and by a larger magnitude, and hence offset the first two quantities. Moreover, between these two priors, the $-\log p(e)$ component has the bigger impact on this outcome. Furthermore, if this component had *not been included* in the formula, then SBM(DC)-CC would have a lower description length, and would have been favored.

We examined the other 102 networks that had selected DC as the model. For all of these, the $-\log p(e)$ component strongly favored the untreated SBM over the treated SBM. We then examined whether removing the $-\log p(e)$ component of the summation of the description length for both treated and untreated SBM models, to see which model would have been returned. We found that for 80 of the 103 networks in total, removing this component of the summation would have resulted in the treated SBM model having a lower description length than the untreated model. Thus, this specific component of the summation accounts for 77.7% of the cases where the untreated SBM is favored over the treated SBM.

We examine the formula for the negative logarithm of the prior for the edge count matrix, which is given by:

$$-\log p(e) = \log \left(\frac{B(B+1)/2 + E - 1}{E} \right) \quad (2)$$

where B and E are the number of blocks and edges, respectively. As B increases, this value will increase, and does so quickly. This explains why clusterings with a larger number of clusters (such as are produced by running CC, WCC, and CM) have larger description lengths, and hence are less favored.

5 Conclusion

Our study demonstrates that clustering using SBMs is prone to producing disconnected clusters, with the frequency of disconnected clusters increasing as the

network size grows. We show that two simple techniques—CC, which returns the connected components of the clusters, and WCC, which repeatedly removes small edge cuts (based on a mild criterion for “small” that depends on the size of the cluster)—can be used to modify an SBM clustering and tends to improve clustering accuracy on synthetic networks. Moreover, WCC has the strongest improvements in our simulation study. Interestingly, using the Connectivity Modifier [15] under the same mild criterion had variable impact, sometimes improving and sometimes reducing accuracy. Thus, our study provides two simple ways to modify clustering using SBMs that lead to improvements in accuracy, without requiring substantial computational resources.

This study focused on improving clustering quality for SBMs, but did so through essentially *ad hoc* techniques. Future work should investigate how to achieve these improvements and guarantees of connectivity within the model-based framework of SBMs. Other future work includes evaluation using other synthetic networks, such as ABCD [6], and exploring whether other post-processing approaches can lead to larger improvements in accuracy while maintaining scalability.

Funding Information. This work was supported in part by the Illinois-Insper partnership.

Software. See <https://github.com/MinhyukPark/constrained-clustering> for the CC and WCC codes.

References

1. Dongen, S.V.: Graph clustering via a discrete uncoupling process. SIAM J. Matrix Anal. Appl. **30**(1), 121–141 (2008)
2. Fortunato, S., Newman, M.E.J.: 20 years of network community detection. Nat. Phys. **18**(8), 848–850 (2022)
3. Henzinger, M., Noe, A., Schulz, C., Strash, D.: Practical minimum cut algorithms. ACM J. Exp. Algorithms **23**, 1–22 (2018)
4. Holland, P.W., Laskey, K.B., Leinhardt, S.: Stochastic blockmodels: first steps. Soc. Netw. **5**(2), 109–137 (1983)
5. Jakatdar, A., Liu, B., Warnow, T., Chacko, G.: AOC: assembling overlapping communities. Quant. Sci. Stud. **3**(4), 1079–1096 (2022)
6. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (ABCD)—Fast random graph model with community structure. Netw. Sci. **9**(2), 153–178 (2021). <https://doi.org/10.1017/nws.2020.45>
7. Karrer, B., Newman, M.E.: Stochastic blockmodels and community structure in networks. Phys. Rev. E-Statistical Nonlinear Soft Matter Phys. **83**(1), 016107 (2011)
8. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E **78**(4), 046110 (2008)
9. Lancichinetti, A., Radicchi, F., Ramasco, J.J., Fortunato, S.: Finding statistically significant communities in networks. PloS One **6**(4), e18961 (2011)

10. Lee, C., Wilkinson, D.J.: A review of stochastic block models and extensions for graph clustering. *Appl. Netw. Sci.* **4**(1) (2019). <https://doi.org/10.1007/s41109-019-0232-2>
11. Miasnikof, P., Shestopaloff, A.Y., Raigorodskii, A.: Statistical power, accuracy, reproducibility and robustness of a graph clusterability test. *Int. J. Data Sci. Anal.* **15**(4), 379–390 (2023)
12. Newman, M.E.J.: Detecting community structure in networks. *Eur. Phys. J. B* **38**(2), 321–330 (2004). <https://doi.org/10.1140/epjb/e2004-00124-y>
13. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. *Phys. Rev. E* **69**(2), 026113 (2004)
14. Park, M., Feng, D.W., Diga, S., Vu-Le, T.A., Chacko, G., Warnow, T.: Supplementary materials for improved community detection using stochastic block models (2024). <https://doi.org/10.5281/zenodo.1334515>
15. Park, M., et al.: Well-connectedness and community detection. *PLOS Complex Syst.* **1**(3), e0000009 (2024)
16. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
17. Peixoto, T.P.: The graph-tool python library. figshare (2014). <https://doi.org/10.6084/m9.figshare.1164194>, <http://figshare.com/articles/graphtool/1164194>
18. Peixoto, T.P.: The netzschleuder network catalogue and repository. Zenodo10 **5281** (2020). <https://doi.org/10.5281/zenodo.7839980>. <https://zenodo.org/records/7839981>
19. Rosvall, M., Axelsson, D., Bergstrom, C.T.: The map equation. *Eur. Phys. J. Special Top.* **178**(1), 13–23 (2009)
20. Traag, V.: Leiden algorithm: leidenalg. <https://github.com/vtraag/leidenalg> (2019)
21. Traag, V.A., Waltman, L., Van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 1–12 (2019)
22. Wedell, E., Park, M., Korobskiy, D., Warnow, T., Chacko, G.: Center-periphery structure in research communities. *Quant. Sci. Stud.* **3**(1), 289–314 (2022)
23. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: the state-of-the-art and comparative study. *ACM Comput. Surv.* **45**(4), 1–35 (2013)
24. Zhang, L., Peixoto, T.P.: Statistical inference of assortative community structures. *Phys. Rev. Res.* **2**(4), 043271 (2020)
25. Zhu, Z.A., Lattanzi, S., Mirrokni, V.: A local algorithm for finding well-connected clusters. In: International Conference on Machine Learning, pp. 396–404. PMLR (2013)



A Knowledge Graph for Monitoring and Controlling Public Policies

Álvaro Carvalho and Raimir Holanda Filho^(✉)

University of Fortaleza, Avenue Washington Soares - 1321, Fortaleza, Brazil
raimir@unifor.br

Abstract. Public Policies are aimed at the citizens of their respective countries. Instead of the traditional view, of the analysis of public policy in the sense of the State for the citizen, the present project sought to verify the feasibility of creating an information system with a different bias: from the citizen to the State, emphasizing how the citizen is being served by the State in all its aspects. If such a conception is viable, the project sought to answer what the necessary requirements are, how the database architecture should be and whether there would be benefits in this approach, such as, for example, the possibility of detecting fraud in the granting of social benefits. Preponderantly, the control systems of these public policies are developed in a relational model database (SQL). Relational tables have the potential to form powerful queries through joins between them, which, given the significant volume of data, can consume a lot of computational resources, and are usually not available online. This modeling does not allow a complete view of the interaction between the State and the citizen. For several technical reasons, analyzed in the project, the information system needs to be developed in a NoSQL graph database. The present work aims to verify whether a graph database allows a modeling with the citizen as the center of public policies. Through public data from three social income transfer programs in Brazil, a graph was created with 433 million nodes and edges. The study allowed us to conclude that the modeling of public policies x citizens is feasible and, through Knowledge graph, has numerous advantages in relation to the traditional model of relational tables and presents the possibility of fraud detection.

Keywords: knowledge graph · public policies · citizen

1 Introduction

Public Policies are aimed at the citizens of their respective countries. In Brazil, public policies are carried out by several Ministries/agencies, according to their competences. For example, the Ministry of Development and Social Assistance, Family and Fight against Hunger (MDS) [7] and the National Institute of Social Security (INSS), which were the databases consulted.

Preponderantly, the control systems of these public policies are developed in relational database (SQL) models. Huge repositories of databases are sent to

the public spending control agencies, coming from hundreds of different public policy systems, each system containing hundreds of tables (i.e., $100 \times 100 =$ thousands of tables), with tens of thousands of fields, which do not allow an immediate and integrated view of the citizen. The tables of relational databases are not connected, but they have the potential to form powerful queries by joining them. However, these queries, given the significant volume of data, can consume a lot of computational resources, and are usually not available online. In the formulation of joins between tables of different systems, there is always the risk of drawing inappropriate conclusions [10], which translates into an erroneous understanding of the semantics of certain fields, which if misinterpreted, can result in information with serious distortions.

In the relational database model, the citizen is not immediately envisioned in a holistic way.

2 Related Work

Erven's work [8] implemented a graph-based database of corporate links between companies and bids with the Federal Government. The work sought, in this case study, to detect evidence of fraud in bidding processes. The conclusion was that the methodology has numerous advantages over SQL models. He points out that the approach can be applied to various control, specialization, and intelligence agencies at the national level and that it allows for the extraction of foundations that can be extended to other problems with modeling relationships between entities.

The work of Erven et al. [9], which deals with an excerpt from [8], consider that several control institutions seek to identify signs of fraud in bids, such as queries in databases to detect relationships between companies that participate in the same purchase process. The authors consider, however, that many of these queries are performed in a limited way in information systems based on relational models of data storage (SQL). A case study was carried out with information from partners and companies and consultations were carried out of interest to the auditors of CGU - Office of the Comptroller General of the Union, one of the control bodies of the Federal Public Administration. In the end, the authors highlight the benefits of graph database technology, as opposed to the existing framework (relational databases), supporting the decision on its use not only by the CGU, but also by other control institutions.

In [5], in the conclusions, the authors mention that: the methodology for using the graph database has proven to be very adequate to detect fraud in public policies; and it is estimated that its use can bring significant benefits to public control agencies.

In a context where one have a large dataset and the relationship between the State and the citizen is the intended objective, such as the scenario of this project, as highlighted in [5,8,13], graph databases have some competitive advantages over relational models: performance is not related to the size of the database, which into a relational environment it degrades; flexibility in modeling data to the domain, without rigid structures planned in advance; agility in

the development of applications, due to the language of intuitive queries, which favors the crossing between nodes and edges.

In conclusion of [5], it was described that in order to combat fraud (in that case, in public tenders), it was necessary to have reliable databases available to carry out the necessary checks. It was highlighted that most databases are relational (SQL) and that the preparation of cross-references of data requires knowledge of different systems, the structure of tables, primary and foreign keys and the availability of common fields between systems. It was pointed out that the queries are complex and require a reasonable amount of time for development and testing.

The same authors [5] state that the identification of fraud essentially involves cross-referencing data under two aspects: first, the verification of the compliance necessary for the legality of a management act; second, check if there is a non-conformity, that is, prove that something that would be prohibited is in fact occurring. This involves verifying the legal connection or proving an illegal connection. The authors state that identifying fraud involves verifying connections or relationships.

3 Project Overview

Initially, some basic concepts about databases deserve to be highlighted [6]: SQL or NoSQL; relational and graph-based data models.

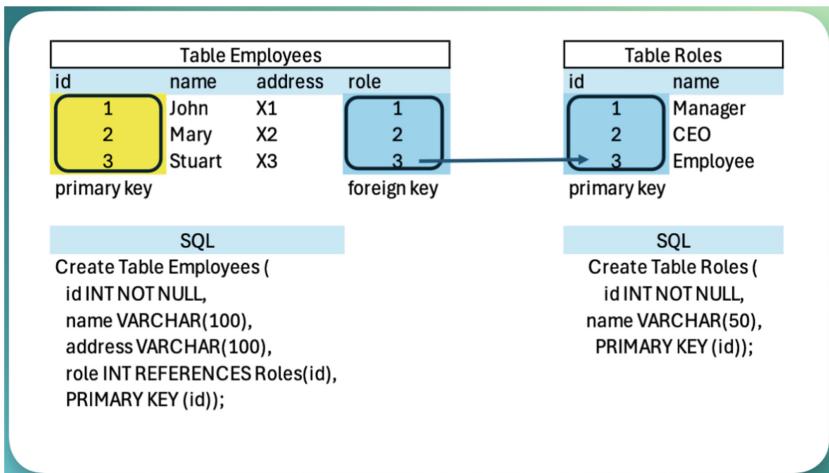
The basic concept of SQL databases (also known as RDBMS - Relational Data Base Management System) is that it is a repository of tables, each containing rows and columns, just like a spreadsheet. The information is stored in tables that relate to each other through common fields, called primary and foreign keys. Primary keys uniquely identify a record in a table, while the foreign key identifies a primary key in another table and is the common field that connects the two tables. The following Fig. 1 illustrates the basic structure of relational databases. RDBMS uses SQL - Structured Query Language as the query and system management language.

However, with the growth in the number of information generated in various media and unstructured formats (Big Data), the advantages of the SQL model (rigidity in standardization) and scalability limitation (capacity for growth) have given rise to the need for new standards, called NoSQL.

Models based on the NoSQL standard seek to overcome the limitations of the SQL model: they have flexibility in the schema (one record can have a different structure from the others), they have horizontal scalability (higher technical feasibility and low cost), overcoming the constraints of the SQL model.

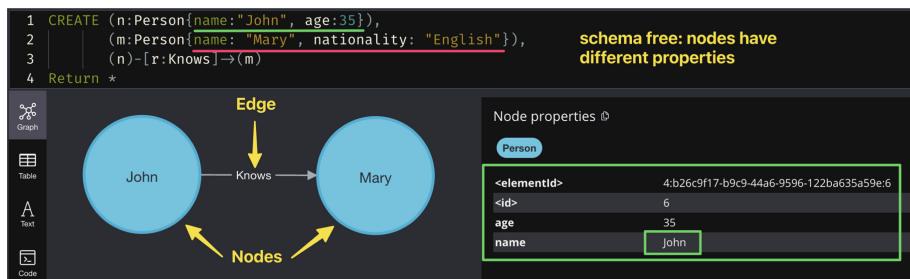
There are four NoSQL database models, each with a specific functionality ([6]):

- Document Databases: In this type of database, data is stored in JSON (JavaScript object notation) or BSON (Binary JSON)-like documents;
- Key-Value Databases: In this type of database, data is stored as key-value pairs, where the key is a unique identifier for the associated value;

**Fig. 1.** SQL - basic structure of relational database

- Column-Family Store: These databases organize data into column families, allowing each row to contain different columns;
- Graph Databases: Graph databases are optimized to store and retrieve interconnected data.

A graph is a collection of nodes (entities) and edges, which represent the relationship between them. Nodes and edges contain properties, sets of key:value pairs, as illustrated in Fig. 2. Graph-based database is a platform that specializes in managing and manipulating graphs [12].

**Fig. 2.** Graph-based database

The choice of a database model has many facets, since many aspects must be considered: scalability; flexibility; costs; performance; ease of use; code maintenance; quality of documentation; user community.

The requirements of the proposed information system, capable of observing public policies from the citizen's point of view, relating to the largest possible number of connections it has with the State, are the following:

- it must be a large system, as it intends to include records from a variety of public policy systems;
- it must have low implementation costs, both at the hardware level and at the level of the software development team;
- scalability must not be an issue;
- the data must be very connected;
- queries should provide performance in searches with very connected data, as many queries will be transversal, covering information from more than one public policy;
- the query operating environment must be online.

In view of the above technical reasons, the choice should fall on a NoSQL graph database model, since it: has horizontal scalability; has routines aimed at cross-sectional analysis; is easy to code; operating environment is online; queries are generic (high power of synthesis); solution meets the requirements for flexibility in schema and gradual deployment, with significantly lower financial, hardware and development team costs.

Through public data from three social income transfer programs in Brazil, a graph was created with the objective of verifying whether graph modeling is feasible and whether it presents advantages in relation to the relational database model. The approach considers that the citizen must be the center of convergence of public policies.

The methodology applied was quantitative: public data were obtained from three largest cash transfer assistance programs in Brazil, namely:

- Bolsa Família [7], which could be translated to “Family Allowance Program” (FAP, in the graph), the largest social program in Brazil, aimed at families with very low financial conditions (data from Mar/2023 to May/2024: 306,709,179 of edges) [1];
- Seguro-Defeso [14], which could be translated to “Fishing Off-Season Insurance” (FOSI, in the graph), intended for fishermen in periods when fishing is prohibited, in the reproduction periods of the species (data from Mar/2023 to Jul/2024: 6,628,058 of edges) [2];
- Benefício de Prestação Continuada [7], which could be translated to “Continuous Assistance Benefit” (CAB, in the graph), intended to guarantee a minimum wage per month to the elderly aged 65 or over or to the disabled person of any age (data from Mar/2023 to Jun/2024: 89,521,529 of edges) [3];

The work consisted of: performing the modeling of the *Knowledge graph*; make the necessary transformations in the data and insert the data into the graph (ETL - Extract, Transform and Load); carry out several management consultations and assess the feasibility and performance of the model.

The project used the Neo4J [11] database, the most popular among graph-based databases. The Neo4J query language is called Cypher. The graph was composed of 433 (four hundred and thirty-three) millions of nodes and edges, as shown in Fig. 3 below.

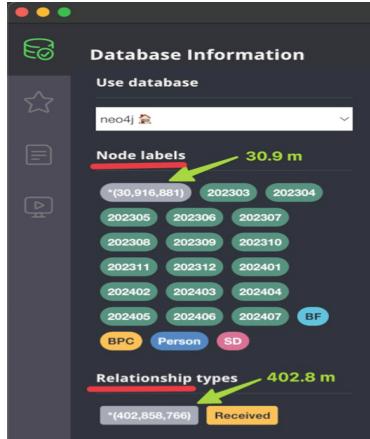


Fig. 3. Database information: numbers of nodes and edges

The graph allowed us to verify that there were 30.9 million beneficiaries of the social programs mentioned.

Graph-based database modeling presents, however, its own challenges: it is essential both the existence of professionals with expertise in graph databases, as well as the business rules, since the insertion of data in the graph must be absolutely in line with the modeled object. Once these difficulties are overcome, the benefits of using the graph-based database begin, which are incomparable to those of the relational model, with easy management consultations ([12]) and extremely satisfactory performance, as shown in the examples in Table 1. The modeling and insertion of nodes and edges, once these requirements are met, allows reducing the risks of drawing inappropriate conclusions [10] from the connections, since the nodes are satisfactorily connected, meeting the business rules, leaving the network available online for all other customers in the client-server computing environment.

4 The Proposed Methodology

The present study had some limitations in relation to the available data, which, however, do not compromise the conclusions obtained.

The data were obtained from the public source [4] and cannot be considered open. In Brazil, the federal agency that collects taxes, Federal Revenue Service (RFB), identifies people (individuals) by the document called CPF, which is a

unique key, consisting of eleven numerical digits. The CPF was obtained through masks, making it impossible to use it as a unique key, that is, the data is not primary. A solution was adopted to overcome this obstacle: a new primary key, with the combination of the name and CPF fields. Even so, many CPF fields are not filled in, having been replaced by “***.***.***-**”. This contributed to many homonyms and with the missing CPF being considered identical records, although they are different people, from different states.

In [5], it was highlighted that in a graph database, data is not stored in tables (rows and columns), but in nodes. There are no primary and foreign keys. Nodes are interconnected through edges (relationships). For example: a natural person has several characteristics (stored in the node properties; key-value pairs), such as name, CPF, address, gender. This node can have, in a large network, several connections (as proposed in this project): in civil terms, with other natural persons (child, father or mother, husband or wife); partner in companies; beneficiary of social programs; as a public servant or employee of private companies; owner of the vehicle, in the National Registry of Vehicle Ownership. In this example, it's easy to identify what connections an individual has: just identify him and make a query with just a command line, requesting his relationships. Such a simple query, in relational databases, would require a deep knowledge of the structures of numerous tables in different systems, the formulation of complex queries in SQL. When it comes to verifying relationships between nodes (records), as in the example above, the numerous advantages of graph databases can be seen immediately. The following Fig. 4 illustrates the above modeling and a simple query, **Listing**, in Cypher, demonstrates how easily to obtain all these connections of the citizen with specific “id”:

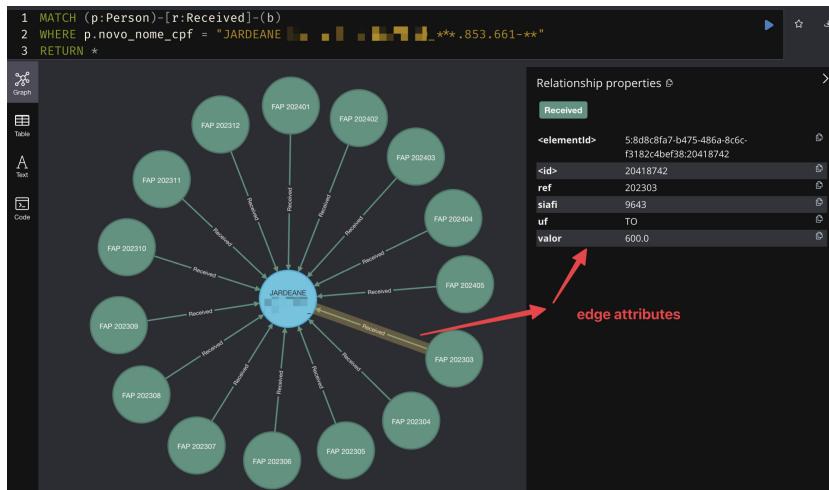


Fig. 4. Query to identify all benefits of a beneficiary

Listing 1.1. Cypher query to get all the relationship of the person with id equal to “150”

```
MATCH (p:Person)-[r:]-(c)
WHERE p.id = '150'
RETURN *
```

It was not possible to obtain the national register of deceased personnel. In order to verify the feasibility of detecting a possible fraud, such as a deceased person receiving one of the social benefits analyzed, nine people were randomly selected and a relationship was established with a fictitious register of deceased persons. The next query, Fig. 5, was able to identify the nine presumed deceased people by detecting the fraud.

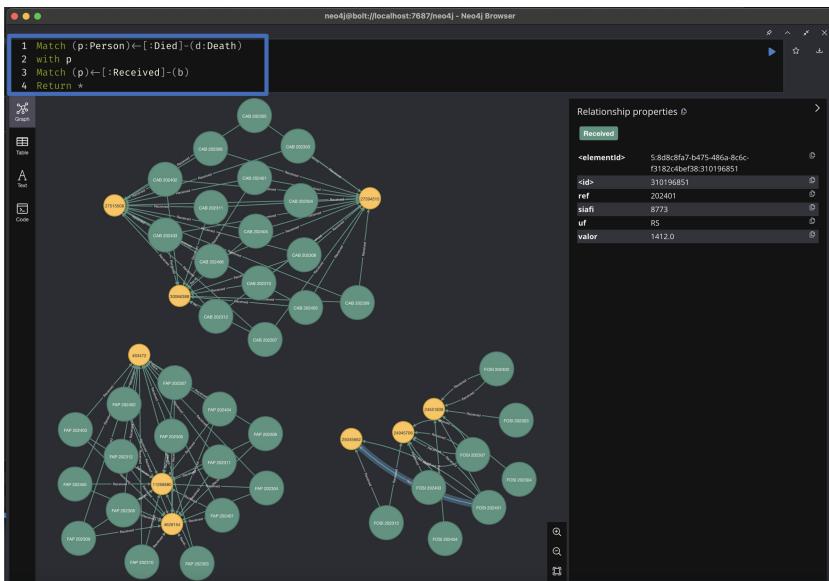


Fig. 5. Cypher query for deceased beneficiary fraud detector

In order to prove the flexibility of creating queries in the graph-based database, another simulation was performed. Three hypothetical entities were inserted into the graph that control the registrations of motor vehicles, boats and airplanes, respectively, Departament of Motor Vehicles (red), Cost Guard (green) and Federal Administration of Aviation (blue). Hypothetical connections between the beneficiaries (orange) of the social programs and the aforementioned control agencies were projected, which could be considered frauds, since people with low financial standards could not buy such properties. The objective was to create a query to identify these frauds. In the SQL model, if “n” were social programs and “m” the number of control bodies, “n x m” queries would be needed

(one check per social program in each control body), plus “n-1” union queries of the results. It is verified that the query in the Fig. 6 graph is of only one simple line, regardless of whether there are 3 or 30 social programs, or whether there are 3 or 30 control bodies. The example demonstrates the flexibility and agility in formulating queries in the *Knowledge graph*.

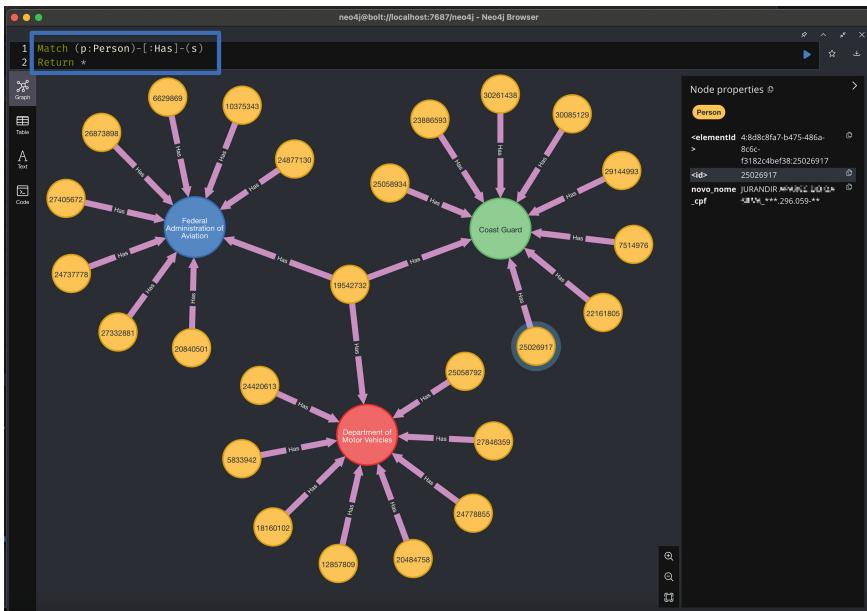


Fig. 6. Cypher query to detect fraud with benefits where the beneficiaries have properties of high values

5 Results

After the ETL (Extract, Transform, and Load) process, the graph is presented with 30.9 million nodes and 402.8 million edges. The size of the database on disk is approximately 133Gb.

The processing was carried out on a notebook, MacBook Pro, chip Apple M3 Pro, 12 cores, with 36Gb of RAM.

The study allowed us to conclude that the modeling of public policies x citizens, through graph-based databases, has numerous advantages in relation to the traditional model of relational tables, including:

- holistic view of individuals and public policies, allowing to connect numerous relationships to citizens, such as public policies, kinship, ownership of assets, such as vehicles, companies (Fig. 7);

- online environment;
- unique modeling, with immediate benefits to all clients of the database server;
- avoid risks of drawing inappropriate conclusions (inadequate understanding of data), inducing unnecessary audit risks [10];
- queries that are easy to develop and test in the Cypher language (**Listing**);
- rapid development of consultations, allowing the fight against fraud in the granting of social benefits (**Figs.5 and 6**), public bids [5];
- enables scalability up to billions of nodes and edges;
- good performance (Table 1);

The Table 1 contains the execution time of several queries performed on the Neo4J database. It can be seen that the queries have high performance.

Table 1. Performance of queries in Cypher

Task	Elapsed Time
read a specific node in 30,916,881 nodes and 15 edges in 402,858,766nodes	9 ms
read 1,465,299 edges	4.7 s
read 1,465,299 edges, group by municipality	4.3 s
read 20,419,605 edges, group by UF	6.2 s
read 20,419,605 edges, group by Program	12 s
read 306,709,179 edges, group by Program	146 s



Fig. 7. Proposed methodology: holistic view of the citizen in graph databases

By connecting to the Neo4J database server (which allows connections to various languages), data can be obtained from the server for more robust processing in successive steps, allowing deeper analysis.

6 Conclusions

Normally, the analysis of public policies are carried out from the perspective of public entities (Ministries/agencies/States/municipalities). The present project sought to verify the feasibility of creating an information system with a different bias: from the citizen to the State, emphasizing how the citizen is being served by the State in all its aspects, as shown in Fig. 7. After numerous researches and analyses, it was possible to prove the feasibility of this project. The project sought to answer what are the necessary requirements for the implementation of this system, what the database architecture should be like and whether there would be benefits in this approach, such as, for example, the possibility of detecting fraud in the granting of social benefits.

Obviously, the scope of the analysis is too broad. The present study used public data from three income distribution programs in Brazil. After technical analysis about the database model (SQL or NoSQL) suitable for the project, the NoSQL graph database solution was adopted. A Knowledge graph with approximately 433 million of nodes and edges was built. Several queries were carried out in the database and the response times were satisfactory. Hypothetical situations were created in the prototype that could be considered fraud in the granting of social benefits. The queries carried out in the database were able, in a significantly simple and objective way, to identify all the hypothetical situations of fraud created.

The solution proposed and proven in the present research work could be used by the public agencies that plan and execute public policies, or by the agencies that supervise them, with the possibility of generating numerous benefits, such as verification of the overlap of public policies, increased effectiveness of the application of public resources, detection of possible fraud in concessions of social benefits and, with the savings generated, expand the necessary services for the needy population.

The topology of this approach also allows for large-scale data analysis. The use of Artificial Intelligence (AI) can enable insights into non-explicit patterns, allowing for fraud detection or adjustments to existing public policies, all aimed at higher standards of economy, efficiency and effectiveness.

Public Administrations, whether federal, state or municipal, have numerous systems in custody as copies of the public policies implemented. The gradual integration of these systems into a Knowledge graph would allow for a holistic view not only of the systems, but of the outcomes of these policies in the lives of citizens.

In summary, in addition to the existing benefits with the traditional use of relational databases, the present work was able to satisfactorily demonstrate the feasibility of introducing graph-based databases (through *Knowledge graph*) in the monitoring of public policies.

Relational databases and graph-based databases are extraordinary and non-mutually exclusive platforms. The concomitant use of the two platforms can bring significant benefits in the monitoring of public policies.

References

1. Brazil: Office of the comptroller general of the union - transparency portal (2024). <https://portaldatransparencia.gov.br/download-de-dados/novo-bolsa-familia>. Accessed 27 2024
2. Brazil: Office of the comptroller general of the union - Transparency portal (2024). <https://portaldatransparencia.gov.br/download-de-dados/seguro-defeso>. Accessed 27 2024
3. Brazil: Office of the comptroller general of the union - Transparency portal (2024). <https://portaldatransparencia.gov.br/download-de-dados/bpc>. Accessed 27 2024
4. Brazil: Office of the comptroller general of the union - transparency portal (2024). <https://portaldatransparencia.gov.br/download-de-dados/>. Accessed 27 2024
5. Carvalho, A., Filho, H., Holanda Filho, R.: Detecting fraud in public acquisition of brazilian government with an analytical approach. In: The 15th International Conference on Ambient Systems, Networks and Technologies Networks (ANT), 23-25 April 2024, Hasselt University, Belgium, vol. 238, pp. 248–254. Procedia Computer Science (2024). <https://www.sciencedirect.com/science/article/pii/S1877050924012584>
6. DataCamp (2024). <https://www.datacamp.com/blog/sql-vs-nosql-databases>
7. Ministry of development and social assistance, family and fight against hunger (2024). <https://www.gov.br/mds/pt-br/acoes-e-programas>
8. Erven, G.C.G.V.: MDG-NoSQL: modelo de dados para bancos NoSQL baseados em grafos. Master's thesis, Universidade de Brasília, Brasília, Brasil (2015). <http://repositorio.unb.br/handle/10482/18992>
9. Erven, G.C.G.V., Holanda, M.T.d., Ralha, C.: Graph database: a case study for detecting fraud in acquisition of Brazilian government. In: 12th Iberian Conference on Information Systems and Technologies (CISTI), Lisbon, Portugal, pp. 1–6 (2017). <https://doi.org/10.23919/CISTI.2017.7975974>
10. Intosai: Issai 100 - Fundamental principles of public-sector auditing (2024). <https://www.issai.org/wp-content/uploads/2019/08/ISSAI-100-Fundamental-Principles-of-Public-Sector-Auditing-1.pdf>
11. Neo4J (2024). <https://neo4j.com>
12. Oracle (2024). <https://www.oracle.com/autonomous-database/what-is-graph-database/>
13. Robinson, I., Webber, J., Eifrem, E.: Graph Databases, 2nd edn. O'Reilly Media, Inc., Sebastopol, CA, USA (2015). ISBN: 9781491930892
14. Social Security INSS, B.N.I.: (2024). <https://www.gov.br/pt-br/servicos/solicitar-seguro-defeso-pescador-artesanal>



High-Performance Implementation of Louvain Algorithm with Representational Optimizations

Subhajit Sahu¹(✉), Kishore Kothapalli¹, and Dip Sankar Banerjee²

¹ IIIT Hyderabad, Professor CR Rao Rd, Gachibowli, Hyderabad, Telangana, India
subhajit.sahu@research.iiit.ac.in, kkishore@iiit.ac.in

² IIT Jodhpur, NH 62, Surpura Bypass Rd, Karwar, Rajasthan, India
dipsankarb@iitj.ac.in

Abstract. Community detection is the problem of identifying natural divisions in networks. Efficient parallel algorithms for identifying such divisions is critical in a number of applications, where the size of datasets have reached significant scales. This paper presents one of the most efficient multicore implementations of the Louvain algorithm, a high quality community detection method. On a server equipped with dual 16-core Intel Xeon Gold 6226R processors, our Louvain, which we term as GVE-Louvain, outperforms Vite, Grappolo, NetworKit Louvain, and cuGraph Louvain (running on NVIDIA A100 GPU) by 50×, 22×, 20×, and 5.8× faster respectively - achieving a processing rate of 560M edges/s on a 3.8B edge graph. In addition, GVE-Louvain improves performance at an average rate of 1.6× for every doubling of threads.

Keywords: Community detection · Parallel Louvain algorithm

1 Introduction

Community detection is the problem of uncovering the underlying structure of complex networks, i.e., identifying groups of vertices that exhibit dense internal connections but sparse connections with the rest of the network, in an unsupervised manner. It has numerous applications in domains such as drug discovery, protein annotation, topic discovery, anomaly detection, and criminal identification. Communities identified are intrinsic when based on network topology alone, and are disjoint when each vertex belongs to only one community [11]. The *Louvain* method [4] is a popular heuristic-based approach for community detection, with the modularity metric [18] being used to measure the quality of communities identified.

In recent years, the collection of data and the relationships among them, represented as graphs, have reached unmatched levels. This has necessitated the design of efficient parallel algorithms for community detection on large networks. Existing studies on Louvain propose several optimizations [10, 12, 17, 26] and parallelization techniques [2, 3, 6, 7, 10, 12, 17, 22, 27]. Further, significant research effort has been focused on developing efficient parallel implementations of Louvain algorithm for multicore CPUs [7, 12, 23, 24], GPUs [17], CPU-GPU hybrids

[3], multi-GPUs [6,8], and multi-node systems—CPU only [10] / CPU-GPU hybrids [2].

However, many of the aforementioned works concentrate on optimizing the local-moving phase of the Louvain algorithm—these optimization techniques are scattered over a number of papers, making it difficult for a reader to get a grip over them—but do not address optimization for the aggregation phase of the algorithm, which emerges as a bottleneck after the local-moving phase has been optimized. Some implementations also fail to adequately parallelize the algorithm. Moreover, much attention has been directed towards GPU-based solutions. However, developing algorithms that efficiently utilize GPUs can be challenging both in terms of initial implementation and ongoing maintenance. Further, the soaring prices of GPUs present hurdles. The multicore/shared memory environment holds significance for community detection, owing to its energy efficiency and the prevalence of hardware with ample DRAM capacities. Through our implementation of the Louvain algorithm¹, we aim to underscore that CPUs remain adept at irregular computation , especially for algorithms where workload diminishes progressively with each iteration. Additionally, we show that achieving optimal performance necessitates a focus on the data structures.

2 Related Work

The *Louvain* method is a greedy modularity-optimization based community detection algorithm, and is introduced by Blondel et al. from the University of Louvain [4]. It identifies communities with resulting high modularity, and is thus widely favored [15]. Algorithmic improvements proposed for the original algorithm include early pruning of non-promising candidates (leaf vertices) [12], attempting local move only on likely vertices [22], ordering of vertices based on node importance [1], moving nodes to a random neighbor community [25], threshold scaling [12,17], threshold cycling [10], subnetwork refinement [26], multilevel refinement [22], and early termination [10].

To parallelize the Louvain algorithm, a number of strategies have been attempted. These include using heuristics to break the sequential barrier [16], ordering vertices via graph coloring [12], performing iterations asynchronously [22], using adaptive parallel thread assignment [17], parallelizing the costly first iteration [27], using vector based hashtables [12], and using sort-reduce instead of hashing [6].

We now discuss about a number of state-of-the-art implementation of Parallel Louvain. Ghosh et al. [9] propose Vite, a distributed memory parallel implementation of the Louvain method that incorporates several heuristics to enhance performance while maintaining solution quality, while Grappolo, by Halappanavar et al. [12], is a shared memory parallel implementation. Qie et al. [20] present a graph partitioning algorithm that divides the graph into sets of partitions, aiming to minimize inter-partition communication delay and avoid community swaps, akin to the graph coloring approach proposed by Halappanavar et al.

¹ <https://github.com/puzzlef/louvain-communities-openmp>.

[12]. We do not observe the community swap issue on multicore CPUs (it likely resolves itself), but do observe it on GPUs (likely due to lockstep execution). NetworkKit [24] is a software package designed for analyzing the structural aspects of graph data sets with billions of connections. It is implemented as a hybrid with C++ kernels and a Python frontend, and includes a parallel implementation of the Louvain algorithm. Finally, cuGraph [13] is a GPU-accelerated graph analytics library that is part of the RAPIDS suite of data science and machine learning tools. It harnesses the power of NVIDIA GPUs to significantly speed up graph analytics compared to traditional CPU-based methods. cuGraph’s core is written in C++ with CUDA and is accessed through a Python interface.

However, most existing works only focus on optimizing the local-moving phase of the Louvain algorithm, and lack effective parallelization. For instance, the implementation of NetworkKit Louvain exhibits several shortcomings. It employs plain OpenMP parallelization for certain operations, utilizing a static schedule with a chunk size of 1, which may not be optimal when threads are writing to adjacent memory addresses. Additionally, NetworkKit Louvain employs guided scheduling for the local-moving phase, whereas we utilize dynamic scheduling for better performance. NetworkKit Louvain also generates a new graph for each coarsening step, leading to repeated memory allocation and preprocessing due to recursive calls. The coarsening involves several sequential operations, and adding each edge to the coarsened graph requires $O(D)$ operations, where D represents the average degree of a vertex, which is suboptimal for parallelism. Lastly, NetworkKit Louvain lacks parallelization for flattening the dendrogram, potentially hindering its performance. Our parallel implementation of Louvain addresses these issues.

3 Preliminaries

Consider an undirected graph $G(V, E, w)$ with V representing the set of vertices, E the set of edges, and $w_{ij} = w_{ji}$ denoting the weight associated with each edge. In the case of an unweighted graph, we assume unit weight for each edge ($w_{ij} = 1$). The neighbors of a vertex i are denoted as $J_i = \{j \mid (i, j) \in E\}$, the weighted degree of each vertex as $K_i = \sum_{j \in J_i} w_{ij}$, the total number of vertices as $N = |V|$, the total number of edges as $M = |E|$, and the sum of edge weights in the undirected graph as $m = \sum_{i,j \in V} w_{ij}/2$.

3.1 Community Detection

Disjoint community detection is the process of identifying a community membership mapping, $C : V \rightarrow \Gamma$, where each vertex $i \in V$ is assigned a community-id $c \in \Gamma$, where Γ is the set of community-ids. We denote the vertices of a community $c \in \Gamma$ as V_c , and the community that a vertex i belongs to as C_i . Further, we denote the neighbors of vertex i belonging to a community c as $J_{i \rightarrow c} = \{j \mid j \in J_i \text{ and } C_j = c\}$, the sum of those edge weights as $K_{i \rightarrow c} = \sum_{j \in J_{i \rightarrow c}} w_{ij}$, the sum of weights of edges within a community c as

$\sigma_c = \sum_{(i,j) \in E \text{ and } C_i = C_j = c} w_{ij}$, and the total edge weight of a community c as $\Sigma_c = \sum_{(i,j) \in E \text{ and } C_i = c} w_{ij}$.

3.2 Modularity

Modularity serves as a metric for assessing the quality of communities identified by heuristic-based community detection algorithms. It is computed as the difference between the fraction of edges within communities and the expected fraction if edges were randomly distributed, yielding a range of $[-0.5, 1]$ where higher values indicate better results [5]. The modularity Q of identified communities is determined using Eq. 1, where δ represents the Kronecker delta function ($\delta(x, y) = 1$ if $x = y$, 0 otherwise). The *delta modularity* of moving a vertex i from community d to community c , denoted as $\Delta Q_{i:d \rightarrow c}$, can be computed using Eq. 2.

$$Q = \frac{1}{2m} \sum_{(i,j) \in E} \left[w_{ij} - \frac{K_i K_j}{2m} \right] \delta(C_i, C_j) = \sum_{c \in \Gamma} \left[\frac{\sigma_c}{2m} - \left(\frac{\Sigma_c}{2m} \right)^2 \right] \quad (1)$$

$$\Delta Q_{i:d \rightarrow c} = \frac{1}{m} (K_{i \rightarrow c} - K_{i \rightarrow d}) - \frac{K_i}{2m^2} (K_i + \Sigma_c - \Sigma_d) \quad (2)$$

3.3 Louvain Algorithm

The Louvain method [4] is a modularity optimization based agglomerative algorithm for identifying high quality disjoint communities in large networks. It has a time complexity of $O(L|E|)$ (with L being the total number of iterations performed), and a space complexity of $O(|V| + |E|)$ [15]. The algorithm consists of two phases: the *local-moving phase*, where each vertex i greedily decides to move to the community of one of its neighbors $j \in J_i$ that gives the greatest increase in modularity $\Delta Q_{i:C_i \rightarrow C_j}$ (using Eq. 2), and the *aggregation phase*, where all the vertices in a community are collapsed into a single super-vertex. These two phases make up one pass, which repeats until there is no further increase in modularity [4].

4 Approach

4.1 Optimizations for Louvain Algorithm

We use a parallel implementation of the Louvain method to determine suitable parameter settings and optimize the original algorithm through experimentation with a variety of techniques. We use *asynchronous* version of Louvain, where threads work independently on different parts of the graph. This allows for faster convergence but can also lead to more variability in the final result [4, 12].

For each optimization, we test a number of relevant alternatives, and show the relative time and the relative modularity of communities obtained by each

alternative in Fig. 3. This result is obtained by running the tests on each graph in the dataset (see Table 1), 5 times on each graph to reduce the impact of noise, taking their geometric mean and arithmetic mean for the runtime and modularity respectively, and representing them as a ratio within each optimization category.

Adjusting OpenMP Loop Schedule. We attempt *static*, *dynamic*, *guided*, and *auto* loop scheduling approaches of OpenMP (each with a chunk size of 2048) to parallelize the local-moving and aggregation phases of the Louvain algorithm. Results indicate that the scheduling behavior can have small impact on the quality of obtained communities. We consider OpenMP’s *dynamic* loop schedule to be the best choice, as it helps achieve better load balancing when the degree distribution of vertices is non-uniform, and offers a 7% reduction in runtime with respect to OpenMP’s *auto* loop schedule, with only a 0.4% reduction in the modularity of communities obtained (likely to be just noise).

Limiting the Number of Iterations per Pass. Restricting the number of iterations of the local-moving phase ensures its termination within a reasonable number of iterations, which helps minimize runtime. This can be important since the local-moving phase performed in the first pass is the most expensive step of the algorithm. However, choosing too small a limit may worsen convergence rate. Our results indicate that limiting the maximum number of iterations to 20 allows for 13% faster convergence, when compared to a maximum iterations of 100.

Adjusting Tolerance Drop Rate (Threshold Scaling). Tolerance is used to detect convergence in the local-moving phase of the Louvain algorithm, i.e., when the total delta-modularity in a given iteration is below or equal to the specified tolerance, the local-moving phase is considered to have converged. Instead of using a fixed tolerance across all passes of the Louvain algorithm, we can start with an initial high tolerance and then gradually reduce it. This is known as threshold scaling [12, 17], and it helps minimize runtime of the first pass of the algorithm (which is usually the most expensive). Based on our findings, a tolerance drop rate of 10 yields 4% faster convergence, with respect to a tolerance drop rate of 1 (threshold scaling disabled), with no reduction in quality.

Adjusting Initial Tolerance. Starting with a smaller initial tolerance allows the algorithm to explore broader possibilities for community assignments in the early stage, but comes at the cost of increased runtime. We find an initial tolerance of 0.01 provides a 14% reduction in runtime of the algorithm with no reduction in the quality of identified communities, when compared to an initial tolerance of 10^{-6} .

Adjusting Aggregation Tolerance. The aggregation tolerance determines the point at which communities are considered to have converged based on the

number of community merges. In other words, if too few communities merged this pass we should stop here, i.e., if $|V_{\text{aggregated}}|/|V| \geq$ aggregation tolerance, we consider the algorithm to have converged. Adjusting aggregation tolerance allows the algorithm to stop earlier when further merges have minimal impact on the final result. According to our observations, an aggregation tolerance of 0.8 appears to be the best choice, as it presents a 14% reduction in runtime, when compared to the aggregation tolerance being disabled (1), while identifying final communities of equivalent quality.

Vertex Pruning. Vertex pruning is used to minimize unnecessary computation [22]. Here, when a vertex changes its community, its marks its neighbors to be processed. Once a vertex has been processed, it is marked as not to be processed. However, it comes with the added overhead of marking/unmarking of vertices. Based on our results, vertex pruning justifies this overhead, and should be enabled for 11% performance gain.

Finding Community Vertices for Aggregation Phase. In the aggregation phase of the Louvain algorithm, the communities obtained in the previous local-moving phase of the algorithm are combined into super-vertices in the aggregated graph, with the edges between two super-vertices being equal to the total weight of edges between the respective communities. This requires one to obtain the list of vertices belonging to each community, instead of the mapping of community membership of each vertex that we have after the local-moving phase ends. A straight-forward implementation of this would make use of two-dimensional arrays for storing vertices belonging to each community, with the index in the first dimension representing the community id c , and the index in the second dimension pointing to the n^{th} vertex in the given community c . However, this requires memory allocation during the algorithm, which is expensive. We employ a parallel prefix sum technique along with a preallocated Compressed Sparse Row (CSR) data structure, eliminating repeated memory allocation and deallocation, and enhancing performance. Indeed, our findings indicate that using parallel prefix sum along with a preallocated CSR is $2.2\times$ faster than using 2D arrays for aggregating vertices.

Storing Aggregated Communities (Super-Vertex Graph). After the list of vertices belonging to each community have been obtained, the communities need to be aggregated (or compressed) into super-vertices, such that edges between two super-vertices being equal to the total weight of edges between the respective communities. This is generally called the super-vertex graph, or the compressed graph. It is then used as an input to the local-moving phase of the next pass of the Louvain algorithm. A simple data structure to store the super-vertex graph in the adjacency list format would be a two-dimensional array. Again, this requires memory allocation during the algorithm, which is bad for performance. Utilizing two preallocated CSRs, one for the source graph and the

other for the target graph (except the first pass, where the dynamic graph may be stored in any desired format suitable for dynamic batch updates), along with parallel prefix sum can help here. We observe that using parallel prefix sum along with preallocated CSRs for maintaining the super-vertex graph is again $2.2\times$ faster than using 2D arrays.

Hashtable Design for Local-Moving/Aggregation Phases. One can use C++’s inbuilt maps as per-thread (independent) hashtables for the Louvain algorithm—but this has poor performance. We use a key-list and a full-size values array (collision-free) to dramatically improve performance. However, if memory addresses of the hashtables are nearby (*Close-KV*)—as with NetworkKit Louvain [24], performance is not as high, even if each thread uses its own hashtable exclusively. This is possibly due to false cache-sharing. Alternatively, if we ensure that the memory address of each hashtable are farther away (*Far-KV*), the performance improves. Our results indicate that *Far-KV* has the best performance and is $4.4\times$ faster than *Map*, and $1.3\times$ faster than *Close-KV*.

4.2 Our Optimized Louvain Implementation

A flow diagram illustrating the first pass of GVE-Louvain for a Weighted 2D-vector based or a Weighted CSR with degree based input graph, is shown in Fig. 4. In the local-moving phase, vertex community memberships are updated until the total change in delta-modularity across all vertices reaches a specified threshold. Community memberships are then counted and renumbered. In the aggregation phase, community vertices in a CSR are first obtained. This is used to create the super-vertex graph stored in a Weighted Holey CSR with degree. In subsequent passes, the input is a Weighted Holey CSR with degree and initial membership for super-vertices from the previous pass. The detailed algorithm of GVE-Louvain is included in our extended report [21].

We aim to incorporate GVE-Louvain into our upcoming command-line graph processing tool named “GVE”, derived from “Graph(Vertices, Edges)”. GVE-Louvain exhibits a time complexity of $O(KM)$, where K signifies the total iterations conducted. Its space complexity is $O(TN + M)$, where T denotes the

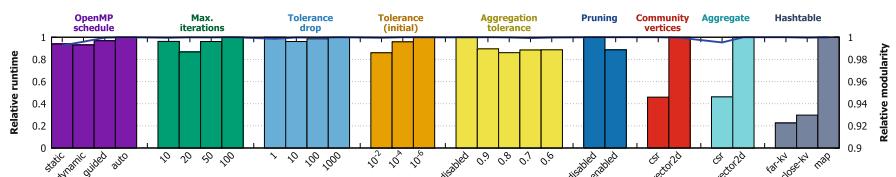


Fig. 1. Impact of various parameter controls and optimizations on the runtime and modularity of the *Louvain* algorithm. The impact upon relative runtime is shown as colored bars on the left y-axis, and upon relative modularity as a blue line on the right y-axis.

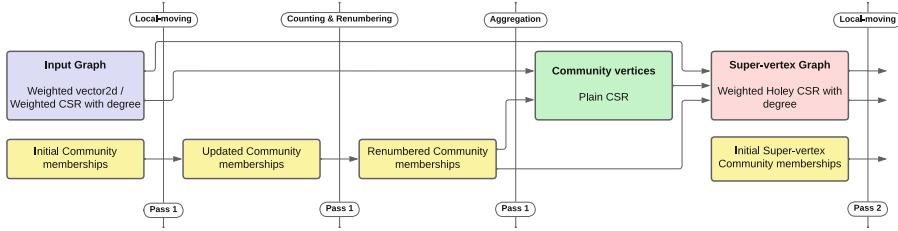


Fig. 2. A flow diagram depicting the first pass of GVE-Louvain.

number of threads utilized, and TN represents the collision-free hash tables H_t employed per thread.

5 Evaluation

5.1 Experimental Setup

System Used. We use a server that has two 16-core x86-based Intel Xeon Gold 6226R processors running at 2.90 GHz. Each core has an L1 cache of 1 MB, an L2 cache of 16 MB, and a shared L3 cache of 22 MB. The machine has 376 GB of system memory and runs on CentOS Stream 8. For our GPU experiments, we employ a system featuring an NVIDIA A100 GPU (108 SMs, 64 CUDA cores per SM, 80 GB global memory) paired with an AMD EPYC-7742 processor (64 cores, 2.25 GHz). This server is equipped with 512 GB of DDR4 RAM and operates on Ubuntu 20.04.

Configuration. We employ 32-bit integers for vertex IDs and 32-bit floats for edge weights, but use 64-bit floats for both computations and hashtable values. Our implementation leverages 64 threads to align with the number of cores on the system, unless otherwise specified. For compilation, we use GCC 8.5 and OpenMP 4.5 on the CPU system, while on the GPU system, we use GCC 9.4, OpenMP 5.0, and CUDA 11.4.

Dataset. The graphs used in our experiments are given in Table 1. These are sourced from the SuiteSparse Matrix Collection [14]. The graphs have vertex counts ranging from 3.07 to 214 million and edge counts ranging from 25.4 million to 3.80 billion. We ensure that the edges are undirected and weighted, with a default weight of 1. We avoid using SNAP datasets with ground-truth communities because they are non-disjoint, whereas our work focuses on disjoint communities. Importantly, ground truth communities may represent different or unrelated aspects of the network structure—relying solely on this correlation could overlook other meaningful structures [19].

Table 1. List of 13 graphs from the SuiteSparse Matrix Collection [14] (\Rightarrow directed). Here, $|V|$ is the vertex count, $|E|$ the edge count (including reverse edges), D_{avg} the average degree, and $|\Gamma|$ the number of communities obtained using GVE-Louvain.

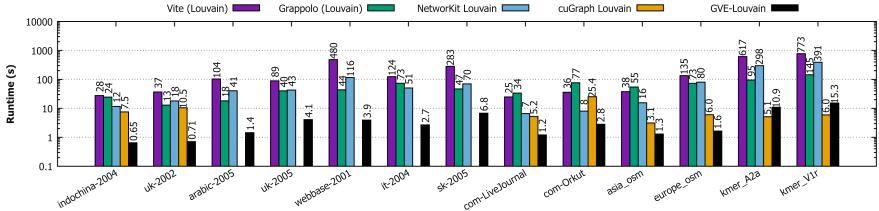
Graph	$ V $	$ E $	D_{avg}	$ \Gamma $	Graph	$ V $	$ E $	D_{avg}	$ \Gamma $
Web Graphs (LAW)					Social Networks (SNAP)				
indochina-2004*	7.41M	341M	41.0	4.24K	com-LiveJournal	4.00M	69.4M	17.4	2.54K
uk-2002*	18.5M	567M	16.1	42.8K	com-Orkut	3.07M	234M	76.2	29
arabic-2005*	22.7M	1.21B	28.2	3.66K	Road Networks (DIMACS10)				
uk-2005*	39.5M	1.73B	23.7	20.8K	asia.osm	12.0M	25.4M	2.1	2.38K
webbase-2001*	118M	1.89B	8.6	2.76M	europe.osm	50.9M	108M	2.1	3.05K
it-2004*	41.3M	2.19B	27.9	5.28K	Protein k-mer Graphs (GenBank)				
sk-2005*	50.6M	3.80B	38.5	3.47K	kmer_A2a	171M	361M	2.1	21.2K
					kmer_V1r	214M	465M	2.2	6.17K

5.2 Comparing Performance of GVE-Louvain

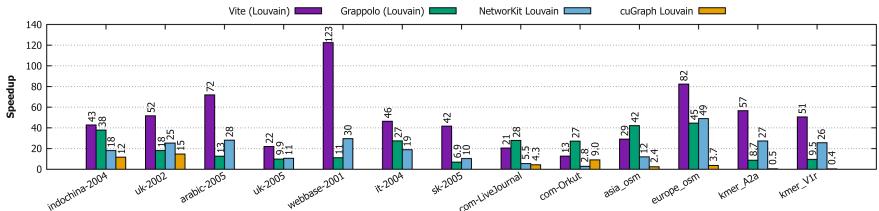
We now compare the performance of GVE-Louvain with Vite (Louvain) [9], Grappolo (Louvain) [12], NetworKit Louvain [24], and cuGraph Louvain [13]. For Vite, we convert the graph datasets to Vite’s binary graph format, run it on a single node with threshold cycling/scaling optimization, and measure the reported average total time. For Grappolo, we measure the run it on the same system, and measure the reported total time. For NetworKit Louvain, we use a Python script to invoke PLM (Parallel Louvain Method), and measure the total time reported with `getTiming()`. To test cuGraph’s Louvain algorithm, we write a Python script that configures the Rapids Memory Manager (RMM) to use a pool allocator for fast memory allocations. We then execute `cugraph.louvain()` on the loaded graph. For each graph, we measure the runtime and the modularity of the obtained communities (as reported by each implementation), performing five runs to calculate an average. When using cuGraph, we discard the runtime of the first run to ensure that subsequent measurements accurately reflect RMM’s pool usage without the overhead of initial CUDA memory allocation.

Figure 3(a) shows the runtimes of Vite (Louvain), Grappolo (Louvain), NetworKit Louvain, cuGraph Louvain, and GVE-Louvain on each graph in the dataset. cuGraph’s Louvain algorithm fails to run on *arabic-2005*, *uk-2005*, *webbase-2001*, *it-2004*, and *sk-2005* graphs because of out-of-memory issues. On the *sk-2005* graph, GVE-Louvain finds communities in 6.8 seconds, and thus achieve a processing rate of 560 million edges/s. Figure 3(b) shows the speedup of GVE-Louvain with respect to each implementation mentioned above. GVE-Louvain is on average 50 \times , 22 \times , 20 \times , and 5.8 \times faster than Vite, Grappolo, NetworKit Louvain, and cuGraph Louvain respectively. Figure 3(c) shows the modularity of communities obtained with each implementation. GVE-Louvain on average obtains 3.1% higher modularity than Vite (especially on web graphs), and 0.6% lower modularity than Grappolo and NetworKit (especially on social

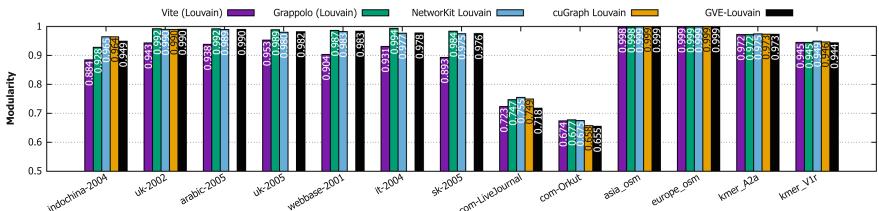
networks with poor clustering), and 2.6% higher modularity than cuGraph Louvain (primarily because cuGraph Louvain failed to run on graphs that are well-clusterable).



(a) Runtime in seconds (logarithmic scale) with *Vite (Louvain)*, *Grappolo (Louvain)*, *NetworkKit Louvain*, *cuGraph Louvain*, and *GVE-Louvain*



(b) Speedup of *GVE-Louvain* with respect to *Vite (Louvain)*, *Grappolo (Louvain)*, *NetworkKit Louvain*, and *cuGraph Louvain*.



(c) Modularity of communities obtained with *Vite (Louvain)*, *Grappolo (Louvain)*, *NetworkKit Louvain*, *cuGraph Louvain*, and *GVE-Louvain*.

Fig. 3. Runtime in seconds (logarithmic scale), speedup, and modularity of communities obtained with *Vite (Louvain)*, *Grappolo (Louvain)*, *NetworkKit Louvain*, *cuGraph Louvain*, and *GVE-Louvain* for each graph in the dataset.

5.3 Analyzing Performance of GVE-Louvain

The phase-wise and pass-wise split of GVE-Louvain is shown in Figs. 4(a) and 4(b) respectively. Figure 4(a) indicates that GVE-Louvain spends most of the runtime in the local-moving phase on *web graphs*, *road networks*, and *protein*

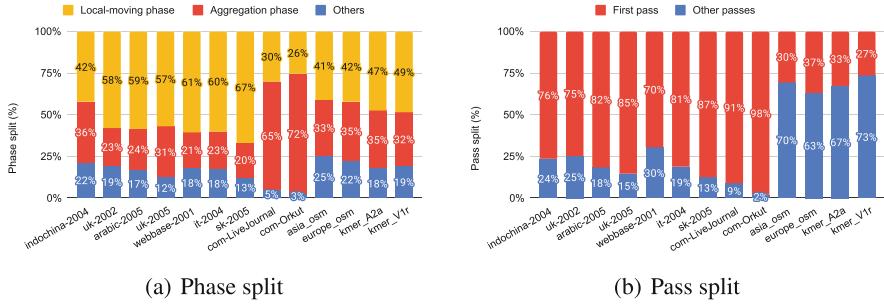


Fig. 4. Phase split of *GVE-Louvain* shown on the left, and pass split shown on the right for each graph in the dataset.

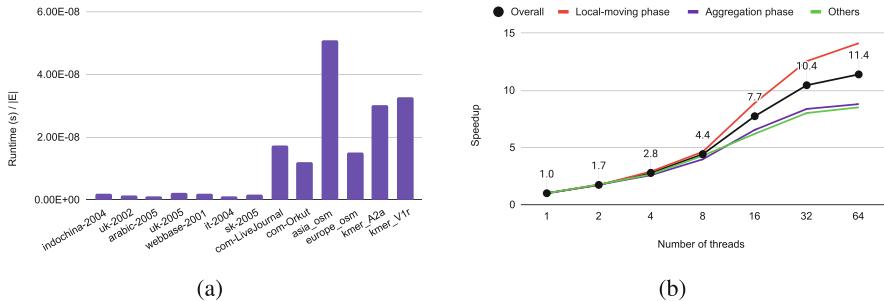


Fig. 5. (a) Runtime $/|E|$ factor with *GVE-Louvain* for each graph in the dataset. (b) Overall speedup of *GVE-Louvain*, and its various phases (local-moving, aggregation, others), with increasing number of threads (in multiples of 2).

k-mer graphs, while it devotes majority of the runtime in the aggregation phase on *social networks*. The pass-wise split (Fig. 4(b)) indicates that the first pass dominates runtime on high-degree graphs (*web graphs* and *social networks*), while subsequent passes prevail in execution time on low-degree graphs (*road networks* and *protein k-mer graphs*).

On average, 49% of GVE-Louvain's runtime is spent in the local-moving phase, 35% is spent in the aggregation phase, and 16% is spent in other steps (initialization, renumbering communities, looking up dendrogram, and resetting communities) of the algorithm. Further, 67% of the runtime is spent in the first pass of the algorithm, which is the most expensive pass due to the size of the original graph (later passes work on super-vertex graphs) [27].

We also observe that graphs with lower average degree (*road networks* and *protein k-mer graphs*) and graphs with poor community structure (such as *com-LiveJournal* and *com-Orkut*) have a larger runtime/ $|E|$ factor, as shown in Fig. 5(a).

5.4 Strong Scaling of GVE-Louvain

Finally, we measure the strong scaling performance of GVE-Louvain. To this end, we adjust the number of threads from 1 to 64 in multiples of 2 for each input graph, and measure the overall time taken for finding communities with GVE-Louvain, as well as its phase splits (local-moving, aggregation, others), five times for averaging. The results are shown in Fig. 5(b). With 32 threads, GVE-Louvain obtains an average speedup of $10.4\times$ compared to running with a single thread, i.e., its performance increases by $1.6\times$ for every doubling of threads. Scaling is limited due to the various sequential steps/phases in the algorithm. At 64 threads, GVE-Louvain is impacted by NUMA effects, and offers speedup of only $11.4\times$.

6 Conclusion

This paper presented our parallel multicore implementation of the Louvain algorithm—a high quality community detection method, which, as far as we are aware, stands as the most efficient implementation of the algorithm on multicore CPUs. Here, we explored 9 different optimizations, including 4 novel ones aimed at enhancing the aggregation phase, which collectively, significantly improve the performance of both the local-moving and the aggregation phases of the algorithm. Future work could focus of designing fast community detection algorithms that enable interactive updation of community memberships of vertices in large dynamic graphs.

References

1. Aldabobi, A., Sharieh, A., Jabri, R.: An improved Louvain algorithm based on node importance for community detection. *JATIT* **100**(23), 1–14 (2022)
2. Bhowmick, A., Vadhiyar, S., Varun, P.V.: Scalable multi-node multi-GPU Louvain community detection algorithm for heterogeneous architectures. *CCPE* **34**(17), 1–18 (2022)
3. Bhowmik, A., Vadhiyar, S.: HyDetect: a hybrid CPU-GPU algorithm for community detection. In: IEEE HiPC, pp. 2–11. Goa, India (2019)
4. Blondel, V., Guillaume, J., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**(10), P10008 (2008)
5. Brandes, U., et al.: IEEE TKDE **20**(2), 172–188 (2007)
6. Cheong, C., Huynh, H., Lo, D., Goh, R.: Hierarchical parallel algorithm for modularity-based community detection using GPUs. In: Proceedings of the 19th Euro-Par, pp. 775–787. Springer, Berlin, Heidelberg (2013)
7. Fazlali, M., Moradi, E., Malazi, H.: Adaptive parallel Louvain community detection on a multicore platform. *Microprocess. Microsyst.* **54**, 26–34 (2017)
8. Gawande, N., Ghosh, S., Halappanavar, M., Tumeo, A., Kalyanaraman, A.: Towards scaling community detection on distributed-memory heterogeneous systems. *Parallel Comput.* **111**, 102898 (2022)
9. Ghosh, S., Halappanavar, M., Tumeo, A., Kalyanaraman, A., Gebremedhin, A.H.: Scalable distributed memory community detection using Vite. In: 2018 IEEE HPEC, pp. 1–7 (2018)

10. Ghosh, S., et al.: Distributed Louvain algorithm for graph community detection. In: IEEE IPDPS, pp. 885–895. Vancouver, British Columbia, Canada (2018)
11. Gregory, S.: Finding overlapping communities in networks by label propagation. *New J. Phys.* **12**(10), 103018 (2010)
12. Halappanavar, M., Lu, H., Kalyanaraman, A., Tumeo, A.: Scalable static and dynamic community detection using Grappolo. In: IEEE HPEC, pp. 1–6. Waltham, MA, USA (2017)
13. Kang, S., Hastings, C., Eaton, J., Rees, B.: cuGraph C++ primitives: vertex/edge-centric building blocks for parallel graph computing. In: IEEE IPDPS Workshops, pp. 226–229 (2023)
14. Kolodziej, S., et al.: The suite sparse matrix collection website interface. *J. Open Source Softw.* **4**(35), 1244 (2019)
15. Lancichinetti, A., Fortunato, S.: Community detection algorithms: a comparative analysis. *Phys. Rev. E Stat. Nonlinear Soft Matter Phys.* **80**(5), 056117 (2009)
16. Lu, H., Halappanavar, M., Kalyanaraman, A.: Parallel heuristics for scalable community detection. *Parallel Comput.* **47**, 19–37 (2015)
17. Naim, M., Manne, F., Halappanavar, M., Tumeo, A.: Community detection on the GPU. In: IEEE IPDPS, pp. 625–634. Orlando, Florida, USA (2017)
18. Newman, M.: Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* **74**(3), 036104 (2006)
19. Peel, L., Larremore, D.B., Clauset, A.: The ground truth about metadata and community detection in networks. *Sci. Adv.* **3**(5), e1602548 (2017)
20. Qie, H., Li, S., Dou, Y., Xu, J., Xiong, Y., Gao, Z.: Isolate sets partition benefits community detection of parallel Louvain method. *Sci. Rep.* **12**(1), 8248 (2022)
21. Sahu, S.: GVE-Louvain: fast Louvain algorithm for community detection in shared memory setting. arXiv preprint [arXiv:2312.04876](https://arxiv.org/abs/2312.04876) (2023)
22. Shi, J., Dhulipala, L., Eisenstat, D., Lacki, J., Mirrokni, V.: Scalable community detection via parallel correlation clustering (2021)
23. Staudt, C.L., Meyerhenke, H.: Engineering parallel algorithms for community detection in massive networks. *IEEE TPDS* **27**(1), 171–184 (2015)
24. Staudt, C.L., Sazonovs, A., Meyerhenke, H.: NetworKit: a tool suite for large-scale complex network analysis. *Netw. Sci.* **4**(4), 508–530 (2016)
25. Traag, V.: Faster unfolding of communities: speeding up the Louvain algorithm. *Phys. Rev. E* **92**(3), 032801 (2015)
26. Waltman, L., van Eck, N.J.: A smart local moving algorithm for large-scale modularity-based community detection. *Eur. Phys. J. B* **86**(11), 1–14 (2013). <https://doi.org/10.1140/epjb/e2013-40829-0>
27. Wickramaarachchi, C., Frincu, M., Small, P., Prasanna, V.: Fast parallel algorithm for unfolding of communities in large graphs. In: IEEE HPEC, pp. 1–6. Waltham, MA, USA (2014)



Branching Out: How Academic Genealogy Networks Shape Academic Groups

Gustavo Moraes¹(✉) , Angelo Brayner¹ , and Ronaldo Menezes^{1,2}

¹ Department of Computer Science, Federal University of Ceará Fortaleza,
Fortaleza, Brazil

gustavo.moraes@alu.ufc.br, brayner@dc.ufc.br

² BioComplex Laboratory, Department of Computer Science, University of Exeter,
Exeter, UK

r.menezes@exeter.ac.uk

Abstract. Academic grouping and collaboration are fundamental to advancing scientific knowledge, as their composition directly influences productivity. However, accurately identifying the structure of these groups can be challenging because academics' stated fields of study often do not align with their actual research output. While these discrepancies may seem minor, they can significantly impact an institution's ability to make informed decisions, implement effective interventions, or provide appropriate incentives for the development of groups in emerging fields. Identifying these groups is essential for better assessing an institution's performance and uncovering knowledge gaps. In this study, we employ an approach rooted in Network Science to analyse group composition, using an institution in Brazil as a case study. We further investigate the relationships between these groups and the academics' research interests, examine the diversity of the groups in relation to their areas of expertise, and compare our findings with self-declared academic groups.

Keywords: Academic genealogy · Network science · Community

1 Introduction

The dissemination of knowledge through scientific publications plays a crucial role in the advancement of science. Academics and students often collaborate to produce publications under the guidance of their supervisors. The interaction between supervisors and students establishes an academic genealogy that can be comprehensively analysed using network science, data science, and statistics.

Academic genealogy serves as a valuable framework for evaluating an academic's impact, especially within their affiliated institutions. Insights from the Science of Science (SciSci) [5] highlight the importance of relationships within research teams, many of which originate from supervisor-student interactions. In this context, examining collaborations through co-supervision relationships is crucial, as these relationships foster multidisciplinary research. This micro-level perspective contrasts with the extensive body of literature focusing on macro-level collaborations [4, 7, 10, 13].

The connection between supervisors and students in an academic genealogy network (AGN) reveals the evolution of local research within an institution. By analysing these relationships, we can identify academic groups and track their growth in number and diversity over time, which can later be contrasted with performance indicators such as grants secured and publications produced. Genealogy provides insights into how new researchers (students, postdocs) adopt areas of study or integrate into collaboration networks through their supervisors' connections. This information is valuable for institutions to understand research success, impact, and the promotion of a multidisciplinary culture.

Recognising the potential of academic genealogy and the significance of academic groups, this study proposes an analysis of academic genealogy using a case study from a postgraduate programme in Computer Science at a university in South America. For privacy reasons, the specific institution is not disclosed. By applying network science techniques, we aim to identify key connections between academics and academic groups to enhance our understanding of academic institutions. Our analysis is based on two AGNs: one constructed from the co-supervision of MSc and PhD work (CoAGN) and another from committee/viva participation (vAGN), as illustrated in Fig. 1.

2 Related Work

Acquiring information about supervisor-student relationships in academic settings can require significant time and effort. Various studies have explored methods to identify these connections by utilising networks of contributions among authors in scientific publications. Wang et al. [15] introduced a probabilistic model (TPFG) designed to detect mentor-mentee relationships with high accuracy. More recently, Zhao et al. [17] presented a novel deep learning model called tARMM, which enhances the detection process through advanced neural network techniques. Liu et al. [9] proposed Shifu2, a sophisticated Network Representation Learning (NRL) model that uniquely considers both the structural and semantic aspects of the network, offering a comprehensive analysis of supervisor-student dynamics.

Academic genealogy and the training of researchers have also garnered considerable interest in Brazil, particularly through the exploration of data from the government platform Lattes¹, a comprehensive research database managed by the National Council for Scientific and Technological Development (CNPq). The data available on this platform enables the construction of academic genealogy structures that explore connections between supervisors, students, and their research contributions. Several studies [3, 11] have enhanced the understanding of academic evolution in Brazil.

Among the notable studies, Cota et al. [11] introduced a platform named Science Tree, specifically designed to visualise and explore the academic genealogy of Brazilian researchers. Science Tree facilitates the exploration of various aspects of a researcher's mentoring efforts within their academic genealogy tree.

¹ Plataforma Lattes. <http://lattes.cnpq.br>. Accessed: August 2024.

The metrics extracted from this tree provide a comprehensive understanding of the structure and contributions within researchers' academic genealogy.

Studies such as Science Tree collect data from researchers across various fields, offering a broad view through their metrics. In contrast, Fu et al. [6] focus on Computer Science, examining the evolution of knowledge within this field. The study emphasises the occurrence of incremental evolution and knowledge mutation, particularly in specific areas of the domain. Kumar et al. [8] propose three methods to predict researchers' influence using academic genealogy networks, validated with data from the Mathematics Genealogy Project (MGP), incorporating both structural and temporal information.

Regarding the identification of academic groups, Perianes-Rodríguez et al. [14] propose a methodology to detect, identify, and visualise groups using a co-authorship network. Their work emphasises the importance of structural network analysis without dismantling connections or separating elements. This approach results in sets of groups, each with unique characteristics.

Using network science techniques, we investigate the academic structures within an institution in Brazil, mapping their academic genealogy to identify and understand academic groups and how they relate to declared fields of study. This targeted approach offers valuable insights into their interrelationships and demonstrate that often groups of individuals with similar declared fields of study are in fact quite diverse in composition.

3 Data and Methods

The dataset was obtained from a postgraduate programme in a STEM department at a large university in Brazil. For privacy reasons, the specific name of the institution is not disclosed, as the data includes information about academics from other institutions, making it impossible to obtain authorisation from everyone involved. Consequently, the names of academics, students, and the institutions themselves have been omitted. However, anonymised data are available for download from the repository². The programme primarily focuses on seven key areas as declared by the academics: high-performance computing, software engineering, computer graphics, databases, logic and artificial intelligence, algorithms and optimisation, and computer networks.

We have used a dataset encompassing MSc and PhD works dating from December 1996 to January 2023, totalling 374 MSc and 92 PhD works. Each work provides information, including the student's name, title, year of completion, examining committee, and supervisor and co-supervisor (if applicable). After some data wrangling to correct for typos and inconsistencies, we constructed a network illustrated in Fig. 1. In this network, two types of individuals are combined in an Academic Genealogy Network (AGN) which captures relationships between students and academics (supervisors) across three possible layers: PhD supervision, MSc supervision, and participation in the students' committees. As

² Dataset. <https://gitlab.com/ggustavo/academic-genealogy-networks>.

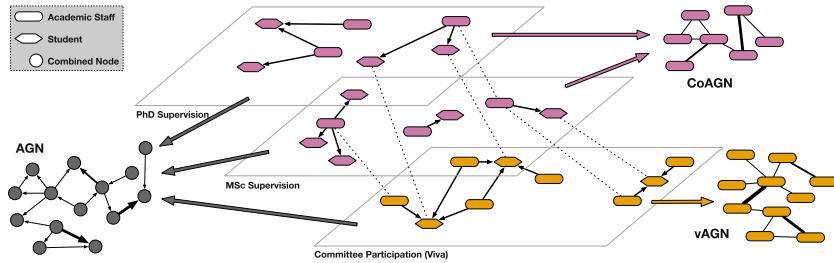


Fig. 1. Academic Genealogy Network (AGN). The dataset contains several relationships that can be seen as a multi-layer network. These relationships can be combined in one bipartite network (the AGN), where nodes are students or academic staff. However, the layers can also be projected in a unipartite network of academic staff only where relations arise from co-supervision of MSc and PhD students (CoAGN) or participation of committeees/vivas (vAGN). Note that in our case, the vAGN projection is a super-set of the CoAGN projection given the assumption that supervisors participate in the student committee (which may not be true in all countries).

shown in the figure, the AGN is an amalgamation of three individual layers, each representing the relationships separately.

Although the separate layers are not directly used in this initial work, the layered modelling provides a detailed view of the mentoring and collaborative interactions within the academic community, and also allows for unipartite projections giving us a AGN from co-advising (CoAGN) and committee/viva participation (vAGN). It is worth noting that our analysis focus most on academics; students are included to represent relationship but could also be included as academic if they also have committee and advising participation. Through these network, we can analyse academic collaborations via co-advising (CoAGN) and via involvement in examining committees (vAGN).

As we mentioned before, the CoAGN captures the PhD and MSc layers from Fig. 1 and projects them onto a unipartite network of academics. In this network, the nodes represent only academics, and the links represent the master's and doctorate co-supervision connections between them. To further understand the relationships among academics, we projected the vAGN the nodes exclusively represent academics (students only if they are also academics), and the links, similar to the previous network, are weighted to reflect the number of committees in which two academics participated together. The vAGN allows for the identification of further collaborators derived from examination committees, shedding light on the academic relationships formed through these interactions. Table 1 details the characteristics of the three previously presented networks.

The three studied networks exhibit low density, as shown in Table 1, indicating that they are sparse, with nodes having varying degrees of connectivity. This sparsity suggests potential for isolation and cluster formation within the networks. The clustering coefficient (C) is particularly low across all networks, with the CoAGN having the lowest value, indicating minimal clustering and less

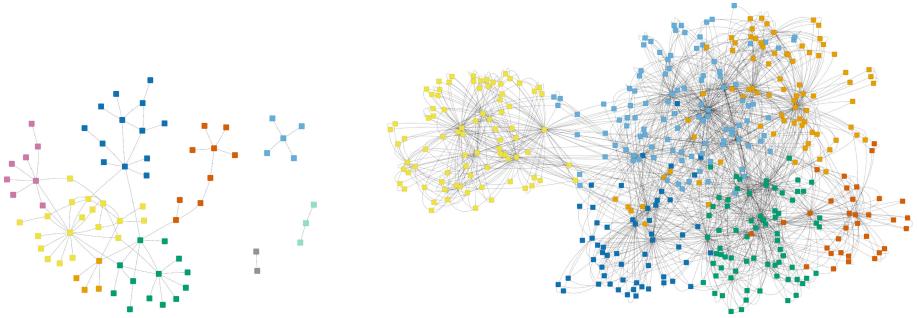


Fig. 2. Networks from Co-supervision and Committee Participation. The figure on the left shows the network built between academics from co-supervision (CoAGN), while the figure on the right illustrates the academic relations built from participation in committee members (vAGN). Note that these are projections from the layers in Fig. 1, with students serving as the source of the relationships. The colours represent distinct communities in each network, which will be discussed in Sect. 4.1.

cohesive subgroups within this network. The density is highest in the CoAGN, despite its lower number of nodes, suggesting a more concentrated network of relationships compared to the AGN and vAGN. The table also provides the exponent (α) values for the degree distributions, reflecting the scale-free nature of these networks, with the vAGN showing the lowest α value, indicating a broader range of node degrees. Additionally, the weighted degree exponent (α_w) high-

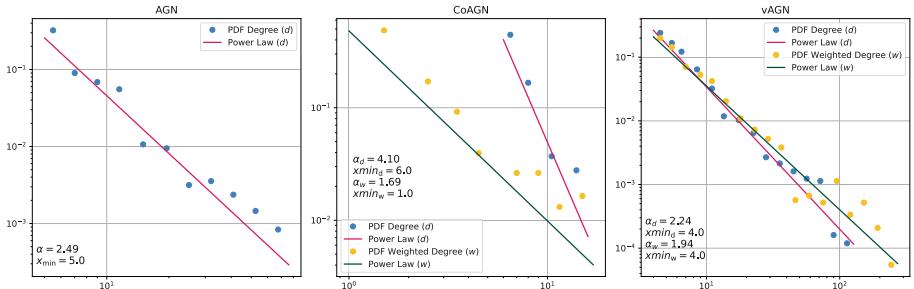


Fig. 3. Degree Distributions. This figure shows the degree distributions for three networks derived from the academic genealogy dataset. On the left is the AGN, a bipartite network where the degree is the sum of in-degree and out-degree for each node. The middle panel displays the CoAGN, a projection focused on academics, where students are included only if they have also supervised other students. This panel presents both the degree and weighted degree distributions. The right panel shows the vAGN, another academic projection similar to CoAGN, with degree and weighted degree distributions representing connections formed through participation in examination committees. All distributions are presented using the Probability Density Function (PDF) and are analysed on a log-log scale.

Table 1. Characteristics of CoAGN and vAGN Networks. This table summarises the main characteristics of the CoAGN and vAGN networks, with the AGN included as a general reference network that is not analysed in depth. The focus is on CoAGN and vAGN, as these are the primary networks studied. Notably, the vAGN network exhibits exponent values (α and α_w) closer to those expected from power-law distributions, likely due to its larger size compared to the smaller CoAGN.

Metric	AGN	CoAGN	vAGN
Nodes (n)	782	76	430
Links (m)	1,932	89	1,951
Density	0.003	0.031	0.021
Clustering Coeff (C)	0.016	0.007	0.028
Exponent (α)	2.49	4.10	2.24
Exponent (α_w)	–	1.69	1.94
Modularity	–	0.727	0.592

lights the weighted connections, with the vAGN network displaying a lower α_w , suggesting more variability in the strength of collaborations. The modularity values indicate the extent of community structure, with the CoAGN network showing a higher modularity, suggesting more distinct communities within this network.

Figure 3 shows the degree distribution across the three networks: AGN, CoAGN, and vAGN. Each graph illustrates the variability in node connectivity, with academics (and students in the case of the AGN) represented along the x-axis according to the number of connections (degree) and the y-axis indicating their frequency. The distributions demonstrate that the networks follow a power-law distribution, which is common in network science and suggests the presence of highly connected nodes, or hubs. The use of the Probability Density Function (PDF) for these distributions is appropriate as it allows us to visualise the proportion of nodes with different degrees, providing a clear understanding of the network’s structure. The combination of degree distributions, clustering coefficients, and modularity values indicate the existence of groups within these networks that could be effectively identified using standard community detection methods, as explored further in Sect. 4.1.

4 Results

4.1 Community in Academic Networks and Topics of Interest

Community detection is crucial for understanding the underlying structure and dynamics within academic networks, as it helps to identify closely connected groups of researchers. In this study, we applied community detection to both the CoAGN and vAGN networks to reveal the natural groupings of academics

based on their collaborative relationships. By analysing these communities, we can compare and verify the alignment of self-declared areas, such as topics of interest, with the actual collaborative patterns observed in the network. The Louvain Community Detection algorithm [1], because it has been used for similar tasks before [2, 16].

Figure 2 illustrates the application of community detection in both the CoAGN and vAGN networks. Nine distinct communities were identified in the CoAGN, while six were found in the vAGN. It is noteworthy that vAGN, being a superset of CoAGN, includes additional academics who were not part of the CoAGN. The inclusion of these additional members in vAGN appears to have aggregated the communities further, leading to fewer but potentially more cohesive groups. While these communities offer valuable insights, the communities in CoAGN are primarily used for alignment with self-declared topics of interest, as many of the academics in vAGN are not from the studied department, and therefore, we lack information about their declared topics of interest. Note that if an academic never co-supervised a student they will not be shown in the CoAGN, and similarly not in the vAGN if they have never participated in committees.

The term “academic group” can be interpreted in various ways. Generally, it refers to a community of scientists who collaborate to conduct research while sharing material and financial resources. For an institution, clearly defining academic groups is crucial, as they represent the institution’s identity and reflect its level of multidisciplinarity, which are important factors for securing funding and enhancing its reputation. However, identifying these groups can be challenging, as the information is often based on self-reporting and tends to be static. As researchers change their areas of focus, these static representations can quickly become outdated. Although formalised groups are common, studies suggest that academic groups can also be inferred from research relationships [14]. The use of community detection methods, as discussed earlier, can help better identify or validate these self-reported groups, offering a more dynamic and accurate representation of the institution’s research landscape.

Community detection and self-reporting represent two complementary approaches for evaluating the structure of an institution and its level of multidisciplinarity. Figure 4 presents a visual comparison between the communities identified in the co-supervision network (see Sect. 4.1) and the self-reported groups within the studied university department. While there are clear similarities between the two, differences also emerge that can lead to inconsistencies when assessing the strengths and weaknesses of the institution. In this study, we use the inferred communities from the CoAGN as a benchmark against the self-reported groups, given that the self-reported information for the vAGN is incomplete due to the involvement of several external academics.

Academics with similar topics of interest are generally more likely to collaborate. However, this does not exclude the possibility of productive collaborations between academics with differing topics of interest, as seen in Fig. 4 (bottom). Identifying academic groups that encompass a diverse range of topics is advanta-

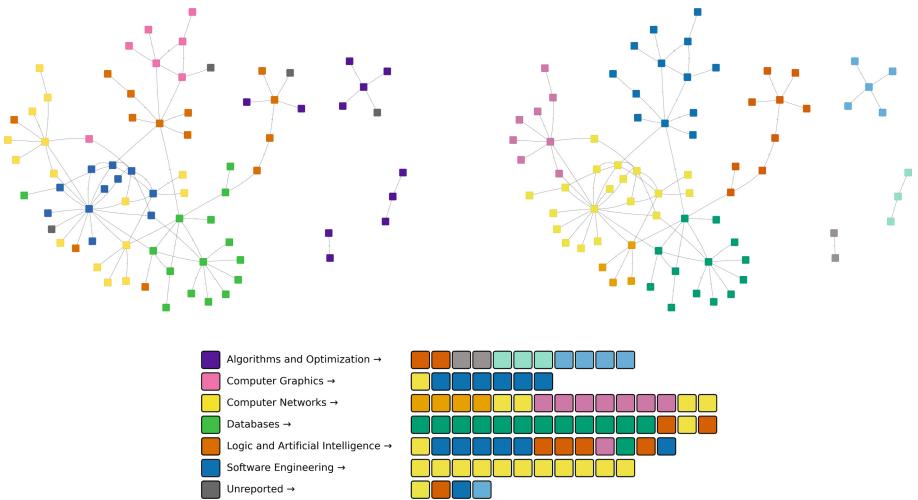


Fig. 4. Comparison of communities and topics of interest for the co-supervisor network. The top-left figure shows the CoAGN with self-reported groups, while the top-right network illustrates the CoAGN with communities identified through the detection algorithm (same one shown in Fig. 2(left)). The bottom figure compares the identified communities with the self-reported groups, highlighting the similarities and discrepancies between the two approaches.

geous, as it fosters multidisciplinary activities. By contrasting community detection results with self-reported groups, we can gain a deeper understanding of how well these groups align with actual collaborative practices, offering insights into the institution's true multidisciplinary nature.

The findings in Fig. 4 reveal that not all self-reported groups align perfectly with the inferred communities, indicating potential discrepancies in how natural academic groups are identified. For instance, the Software Engineering group shows a high level of cohesion, with all its members belonging to the same inferred community. In stark contrast, the Computer Networks group is dispersed across various communities, suggesting a lack of uniformity in collaboration patterns. These differences may arise from certain fields being more inclined towards working in silos, whereas others naturally promote multidisciplinary interactions. Alternatively, it could be that individuals are not always the best at accurately expressing their true areas of interest based on their actual academic activities, as captured through academic genealogy. The visual comparison in Fig. 4 underscores the importance of combining self-reported data with community detection analysis to obtain a more accurate and nuanced understanding of academic group dynamics within an institution.

4.2 Diversity in Academic Collaborations

Diversity in academic collaborations is essential for fostering multidisciplinary research and innovation. By examining the diversity of academics' collaborations, we gain valuable insights into the breadth and depth of their academic networks. This analysis leverages ego graphs, which represent a network centred on a specific node (the 'ego') and its direct connections with other nodes. Ego graphs offer a detailed view of a node's immediate neighbours and their interactions within the network [12]. Such visualisations are instrumental in understanding how individual researchers contribute to and are influenced by their collaborative environments, ultimately revealing the multidimensional nature of academic networks.

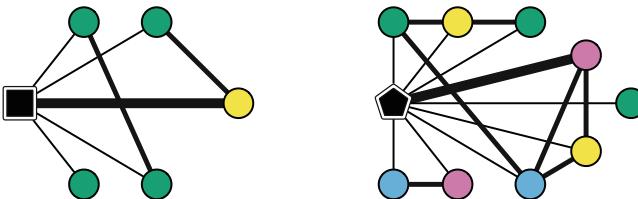


Fig. 5. Sample Ego Networks of Two Academics. This figure illustrates two ego networks representing the immediate collaboration networks of two different academics. The example demonstrates how individual academics can exhibit significant differences in their collaboration patterns. These differences are reflected both in the diversity of areas their collaborators are associated with (as identified by community detection) and the degree of clustering within their networks.

Figure 5 illustrates two ego graphs extracted from the vAGN, centred on a specific academic. This example demonstrates that ego networks can differ significantly in terms of their composition, with varying areas of collaboration (as identified by community detection) and the degree of interconnectedness among collaborators. The link weights in the graph represent the strength of these collaborations within the vAGN, highlighting how individual academics can have distinct patterns of collaboration, both in the diversity of their collaborators' areas and in the cohesiveness of their networks.

To analyse the structure of connections within an ego graph, we use two key metrics for the vAGN: the clustering coefficient and Shannon Entropy. The clustering coefficient measures the level of interconnectedness within an academic's immediate network, indicating how cohesive their collaborations are. Shannon Entropy, on the other hand, captures the diversity of connections across different groups, offering insights into the multidisciplinary nature of the network. These metrics help determine whether collaborations are concentrated within specific areas or spread across multiple disciplines, providing a nuanced understanding of academic networks.

Regarding entropy, a higher value signifies greater diversity in an academic's collaborators, reflecting a broader range of community affiliations within their ego network. In this context, we focus on groups identified through community detection rather than self-declared topics of interest, as the vAGN includes many academics for whom declared topics are not available. Relying on self-declared data could lead to misleading conclusions about low entropy, as academics without declared topics might appear as part of a single group, thereby underrepresenting the actual diversity of their collaborations.

We aim to examine the correlation between the clustering coefficient and Shannon entropy within academic networks. High entropy may indicate involvement in multidisciplinary collaborations, while a high clustering coefficient suggests concentrated, cohesive collaboration within specific subgroups. Understanding these relationships helps institutions assess their collaboration structures and identify areas for intervention to enhance their reputation and internationalisation efforts.

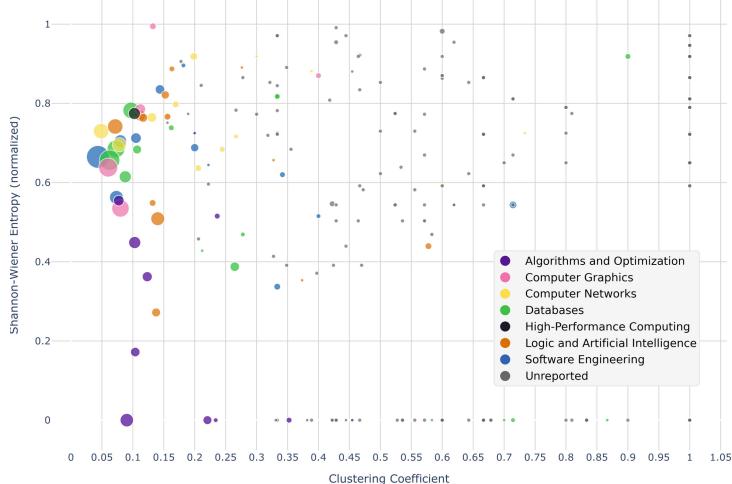


Fig. 6. Correlation Between Clustering Coefficient and Shannon Entropy. This figure shows the relationship between the clustering coefficient and Shannon entropy for individual academics. The entropy values are based on the community detection analysis of their collaborators. Circle sizes represent the number of MSc and PhD students supervised, and colours denote the academics' topics of interest.

Figure 6 highlights several important trends in the relationship between clustering coefficient and Shannon entropy in academic collaboration networks. Academics with higher productivity, measured by the number of supervised master's and doctoral students, tend to exhibit moderate to high entropy, indicating a broad range of multidisciplinary collaborations. These academics generally have lower clustering coefficients, suggesting that their collaborations are diverse but

not tightly interconnected. Conversely, academics with higher clustering coefficients, who also show moderate to high entropy, often have lower supervision productivity or belong to the “unreported” category, typically referring to external members with limited known networks. The analysis also shows that the most dispersed academics, primarily external members with fewer supervised students, contribute significantly to the diversity of the collaboration network despite their lower overall academic output.

5 Discussion

This study has demonstrated the value of academic genealogy as a novel approach to understanding the internal dynamics of academic groups within academic institutions. By focusing on the micro-level relationships between academics—such as co-supervision and participation in examination committees—rather than traditional metrics like publication output or grant acquisition, we offer a different lens through which to view academic structures. This perspective can provide institutions with valuable insights into how they organise themselves and inform decisions related to hiring, promotions, and incentive strategies.

The methodologies employed in this study, particularly the construction of specialised networks like the CoAGN and vAGN, reveal how academics naturally cluster into communities based on shared research interests. The application of network science techniques, such as community detection, allows us to identify discrepancies between these inferred communities and self-reported academic groups, highlighting potential biases in self-reporting. By incorporating network-based analyses, institutions can achieve a more accurate understanding of their multidisciplinary landscape, aiding in strategic decisions about how they present themselves and how they nurture collaborative environments.

While this study is based on a single institution, the approach has broader implications. It serves as an encouragement for other institutions to adopt similar methods, enabling comparisons and alignments across different academic settings. In countries like Brazil, where there is a well-defined system for evaluating university performance, this approach could be particularly effective. However, applying these methods internationally might present challenges due to varying evaluation criteria and academic structures across countries.

In conclusion, this work highlights the potential for academic genealogy networks to reshape how institutions understand and manage their internal research dynamics. By expanding this approach to more institutions and contexts, we can contribute to a more nuanced and strategic understanding of academic collaboration, ultimately enhancing the ability of institutions to make informed decisions that align with their goals and values.

Declaration. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) licence to any accepted manuscript version arising from this submission.

References

1. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**(10), P10008 (2008)
2. Chen, J.M., Tang, Y., Li, J.G., Mao, C.J., Xiao, J.: Community-based scholar recommendation modeling in academic social network sites. In: Huang, Z., Liu, C., He, J., Huang, G. (eds.) WISE 2013. LNCS, vol. 8182, pp. 325–334. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54370-8_28
3. Damaceno, R.J., Rossi, L., Mugnaini, R., Mena-Chalco, J.P.: The Brazilian academic genealogy: evidence of advisor-advisee relationships through quantitative analysis. *Scientometrics* **119**, 303–333 (2019)
4. Divakarmurthy, P., Biswas, P., Menezes, R.: A temporal analysis of geographical distances in computer science collaborations. In: 2011 IEEE Third International Conference on Social Computing, pp. 657–660. IEEE (2011)
5. Fortunato, S., et al.: Science of science. *Science* **359**(6379), eaao0185 (2018)
6. Fu, Z., Cao, Y., Zhao, Y.: Identifying knowledge evolution in computer science from the perspective of academic genealogy. Available at SSRN 4610412 (2023)
7. Jaramillo, A.M., Williams, H.T., Perra, N., Menezes, R.: The structure of segregation in co-authorship networks and its impact on scientific production. *EPJ Data Sci.* **12**(1), 47 (2023)
8. Kumar, D., Bhowmick, P.K., Paik, J.H.: Researcher influence prediction (ResIP) using academic genealogy network. *J. Inf.* **17**(2), 101392 (2023)
9. Liu, J., et al.: Shifu2: a network representation learning based model for advisor-advisee relationship mining. *arXiv* (abs/2008.07097) (2020)
10. Martinez, M., Sá, C.: Highly cited in the south: international collaboration and research recognition among brazil's highly cited researchers. *J. Stud. Int. Educ.* **24**(1), 39–58 (2020)
11. M.C., J.M., Laender, A.H., Prates, R.O.: Science tree: a platform for exploring the Brazilian academic genealogy. *J. Braz. Comput. Soc.* **27**(1), 1–20 (2021)
12. McAuley, J., Leskovec, J.: Learning to discover social circles in ego networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, p. 539–547 (2012)
13. Newman, M.E.: The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci.* **98**(2), 404–409 (2001)
14. Perianes-Rodríguez, A., Olmeda-Gómez, C., Moya-Anegón, F.: Detecting, identifying and visualizing research groups in co-authorship networks. *Scientometrics* **82**(2), 307–319 (2010)
15. Wang, C., et al.: Mining advisor-advisee relationships from research publication networks. In: Proceedings of the 16th ACM SIGKDD, p. 203–212. ACM (2010)
16. Zhao, W., Luo, J., Fan, T., Ren, Y., Xia, Y.: Analyzing and visualizing scientific research collaboration network with core node evaluation and community detection based on network embedding. *Pattern Recogn. Lett.* **144**, 54–60 (2021)
17. Zhao, Z., Liu, W., Qian, Y., Nie, L., Yin, Y., Zhang, Y.: Identifying advisor-advisee relationships from co-author networks via a novel deep model. *Inf. Sci.* **466**, 258–269 (2018)



Entropy-Based Attack on Community Detection in Directed Networks

Saif Aldeen Madi¹(✉) and Giuseppe Pirrò²

¹ American University of Madaba, Madaba, Jordan

s.madi@aum.edu.jo

² University of Calabria, Arcavacata, Italy

giuseppe.pirro@unical.it

Abstract. Community detection algorithms, which cluster nodes into community structures, can inadvertently compromise user privacy. To counter this, community deception algorithms have been developed, strategically altering edges to influence the outcome of community detection. However, existing approaches have notable limitations, including inadequate privacy protection across various dimensions and a lack of consideration for edge directionality. In this paper, we introduce a novel attack on community detection algorithms aimed at obfuscating entire community structures. Our method utilizes the variation in structural entropy of the directed network caused by the presence of the target community structure, employing it as the basis for optimizing our attack strategy. We compared our approach against state-of-the-art methods, yielding promising results.

Keywords: community detection · community structure deception · directed networks · structural entropy

1 Introduction

Community detection, a process of clustering network nodes into communities, is commonly performed by third-party entities like research organizations and data analysts [8]. While serving various purposes, this practice can lead to substantial privacy breaches at multiple levels. At the *entire community structure* level, detection algorithms might expose the infrastructure of a network, such as IoT systems [5]. At the *community* level, they could reveal sensitive group dynamics, like the activities of social network activists [26]. In response to these privacy challenges, the field of *community deception* [7, 29] has emerged. Yet, existing approaches for hiding a community structure have overlooked edge direction, despite the fact that real-world social connections are frequently directed. This research focuses on transforming an initial network G into a new, modified network G' in which the true community structure is obscured, thus making it more challenging for detection algorithms to identify. This paper introduces EBAD (Entropy-Based Attack on Community Detection), a novel approach targeting

community detection in *directed* networks. The fundamental principle of EBAD is leveraging the entropy differential between a graph and its partitioned community structure. By exploiting this entropy gap, EBAD aims to disrupt the modular structure that community detection algorithms typically rely on. This modular structure is characterized by higher prevalence of intra-community edges than inter-community edges, which intuitively captures the notion of community.

Related Work. Table 1 compares EBAD with state-of-the-art approaches.

Table 1. Comparison of EBAD with work most related.

Strategy	NAGARAJA	DICE	SAF	Q-ATTACK	MOD	NEURAL	DSAF	DMOD	DPER	iDEC	REM	EPA	CGN	PROHICO	LSHA	COMDEC	EBAD
Addition	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Deletion		✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Directed						✓	✓	✓	✓								✓

Focusing on *single community* deception, Nagaraja (2010) [21] explored hiding a community by adding edges towards nodes of high centrality. Both DICE [29] and MOD [7] operate on the heuristic of deleting intra-community edges and adding inter-community ones, under the assumption that these updates minimize modularity. SAF [7], an approach based on node safeness, has been further developed by Chen et al. [4]. NEURAL [20] leverages permanence minimization, while COMDEC. [32] refines DICE’s heuristic. In the realm of *community structure* deception, we consider Q-ATTACK [2], REM [16], CGN [14], and PROHICO [15]. Q-ATTACK struggles with scalability due to the combinatorial nature of genetic algorithms and has only been tested on networks with around 100 nodes. REM [16] minimizes residual entropy but is limited to undirected edge additions. CGN [14] uses NMI as an optimization criterion, tying it to the output of a specific detection algorithm. LSHA [31] is founded on structural ideas. Most existing approaches are designed for specific types or levels of attacks, lacking versatility. For example, REM [16] employs structural entropy for entire community structure deception, whereas safeness [7] targets single communities. EPA [3] supports a multilevel strategy but suffers from scalability issues in larger networks. Notably, none of these approaches consider edge directionality, a critical aspect in various fields like social/information networks and biology. These fields often feature asymmetric relationships that are oversimplified in undirected network models, leading to loss of vital information [19]. Although specific algorithms like DSAF, DMOD, DPER [6], and iDEC [17, 18] show improved performance in directed networks, their scope is limited to hiding single communities.

Contributions. This paper makes several key contributions: (I) We introduce EBAD, a novel community deception approach, which uniquely utilizes the network’s directed *structural* entropy [12], making it independent of the specific quality functions of detection algorithms; (II) We propose a greedy algorithm with a complexity of $O(k|V|)$, guiding nodes on how to strategically manipulate their connections, where k is the number of communities and V is the number

Table 2. Notation table

Symbol	Meaning	Formula
\mathbf{d}_u^\uparrow	Out-degree of node u	$ \{(u, v) : v \in V\} $
\mathbf{v}_C^\uparrow	Out-volume of a community C	$\sum_{u \in C} \mathbf{d}_u^\uparrow$
$\tilde{\mathbf{m}}_C^\uparrow$	Num. of edges outgoing from C	$ \{(u, v) : u \in C \wedge v \notin C\} $

of nodes. We evaluated EBAD across various real-world networks and detection algorithms with encouraging results.

Practical Usage. EBAD offers versatile applications in different real-world scenarios. Indeed, since community detection can be used to de-anonymize particular network users [30], social network operators can apply EBAD to safeguard their network prior to releasing it to the public (say, for research purposes) [1].

Outline. This paper is structured as follows: In §2, we provide the necessary background. Theoretical insights into hiding entire community structures are presented in §3. We present EBAD and its complexity in §3.1. The evaluation of our approach is detailed in §4. We conclude and outline future work in §5.

2 Background and Problem Statement

We assume a directed unweighted network $G = (V, E)$, where V is a set of \mathbf{n} nodes, and E is a set of \mathbf{m} edges. An edge connecting a source node u to destination node v is denoted with (u, v) . We denote with $G \ominus (u, v)$ (resp., $G \oplus (u, v)$) the graph G' when deleting (resp., adding) the edge (u, v) from G . We call *new edge* an edge (i, j) , $i, j \in V \wedge (i, j) \notin E$. Given G , a community detection algorithm \mathcal{A}_D produces a community structure $\mathcal{P} = \{C_1, C_2, C_3, \dots, C_k\}$ consisting of k communities. Moreover, we assume that $C_i \cap C_j = \emptyset$, that is, communities do not overlap. Given a community $\bar{V} \subseteq V$, an intra- \bar{V} edge (or intra-community edge) has the form $(u, v) : \{u, v\} \subseteq \bar{V}$ while inter- \bar{V} edge (or inter-community edge) has the form $(u, v) : u \in \bar{V}, v \notin \bar{V}$. Table 2 summarizes the primary notation.

2.1 Normalized Directed Residual Entropy

We present *Normalized Directed Residual Entropy (NDRE)* [12] – the objective function at the core of EBAD. We chose NDRE because it provides a theoretically-founded framework to quantify structural information in complex networks, treating them as communication systems. Moreover, entropy minimization at *different levels* offers a principled way of understanding real-world networks.

Let G be a network of communicating nodes with edges being *directed* communication channels. At any moment, if we collect messages *en route* uniformly and randomly, we can say that the probability of a given node u being a message sender is $\mathbf{d}_u^\uparrow / \mathbf{m}$; we can use this probability distribution to define a *structural entropy* of the network:

Definition 1 (Structural Entropy for Directed Networks). *The structural entropy of a directed network G captures the average number of bits required to encode message senders and equals Shannon's entropy of the distribution $(\mathbf{d}_i^\uparrow / \mathbf{m})_{i \in V}$:*

$$\mathcal{H}(G) := - \sum_{i=1}^n \frac{\mathbf{d}_i^\uparrow}{\mathbf{m}} \log_2 \frac{\mathbf{d}_i^\uparrow}{\mathbf{m}} \quad (1)$$

where $-\log_2 \frac{\mathbf{d}_i^\uparrow}{\mathbf{m}}$ is the number of bits necessary to encode some node i . Now, consider the presence of some community structure of G , say $\mathcal{P} = \{C_1, C_2, C_3, \dots, C_k\}$. The structural information of each community $C_j \in \mathcal{P}$ as a message sender consists of two levels: (a) the level of a single node $i \in C_j$; and (b) the level of the whole community C_j . This bi-level hierarchy of structural information can be connected with Eq. (1) through the following equation:

$$-\log_2 \frac{\mathbf{d}_i^\uparrow}{\mathbf{m}} = \underbrace{-\log_2 \frac{\mathbf{d}_i^\uparrow}{\mathbf{v}_{C_j}^\uparrow}}_{(a)} - \underbrace{\log_2 \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}}}_{(b)} \quad (2)$$

Here, (a) captures the number of bits to encode i as a message sender within community C_j , and (b) is the number of bits to encode community C_j as a message sender within G . Now, we want to define the bi-level structural entropy of G , taking into account the presence of \mathcal{P} . Just as in Eq. (1), we express the structural information on the node-level by $\mathcal{H}(G|C_j) = -\sum_{i \in C_j} \frac{\mathbf{d}_i^\uparrow}{\mathbf{v}_{C_j}^\uparrow} \log_2 \frac{\mathbf{d}_i^\uparrow}{\mathbf{v}_{C_j}^\uparrow}$.

However, for the community-level, if the sending and receiving nodes are both members of C_j , we don't need the community-level codeword, and we express the community-level structural information as $-\frac{\tilde{\mathbf{m}}_{C_j}^\uparrow}{\mathbf{m}} \log_2 \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}}$. Thus, we can express the structural entropy of G relative to \mathcal{P} by:

$$\mathcal{H}_{\mathcal{P}}(G) = \sum_{j=1}^k \left[\frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \mathcal{H}(G|C_j) - \frac{\tilde{\mathbf{m}}_{C_j}^\uparrow}{\mathbf{m}} \log_2 \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \right] \quad (3)$$

Note that the weight assigned to the community-level codeword is expressed only with *inter-community* edges $\frac{\tilde{\mathbf{m}}_{C_j}^\uparrow}{\mathbf{m}}$. In fact, for any non-trivial \mathcal{P} , we have $\mathcal{H}(G) > \mathcal{H}_{\mathcal{P}}(G)$, since number of bits is expressed by the second term of Eq. (3) is based only on *inter-community* edges. This difference between Eq. (1) and Eq. (3) is the basis of NDRE:

Definition 2 (Normalized Directed Residual Entropy). *Given a directed network G having structural entropy $\mathcal{H}(G)$ and a relative entropy $\mathcal{H}_{\mathcal{P}}(G)$, the Normalized Directed Residual Entropy is defined as follows:*

$$\rho_{\mathcal{P}}(G) = \frac{\mathcal{H}(G) - \mathcal{H}_{\mathcal{P}}(G)}{\mathcal{H}(G)} \quad (4)$$

A community detection algorithm \mathcal{A}_D typically search for modular [9] structures with dense *intra-community* and sparse *inter-community* connections. From an information theoretical perspective, most message sending and receiving in this setting will occur *within* the same community because of the high edge density. Thus, the second term of (3) will equal zero more frequently, and $\mathcal{H}_{\mathcal{P}}(G)$ will be lower. In other words, community detection algorithms strive to find a community structure \mathcal{P} that maximizes $\mathcal{H}(G) - \mathcal{H}_{\mathcal{P}}(G)$, and hence $\rho_{\mathcal{P}}(G)$. On the contrary, from a *deception* perspective, it makes sense to develop networks where the NDRE relative to the reference structure \mathcal{P} will be *minimized*. This will disrupt the network's modular structure, thus effectively acting in the reverse way that detection algorithms strive to do.

3 Hiding Community Structures with EBAD

This section presents a Community Structure Deception (CSD) attack strategy in directed graphs. We formally define CSD as an optimization problem in Problem 1. CSD takes the entire network G and the true *target* community structure \mathcal{P} as input. Network owners could perform CSD to avoid, for instance, intellectual property theft [10].

Problem 1 (Community Structure Deception). *Given a directed network $G = (V, E)$, a community structure \mathcal{P} and an update budget β , community structure deception is defined as:*

$$\begin{aligned} & \text{MINIMIZE} && \rho_{\mathcal{P}}(G') \\ & \text{where} && G' = (V, E') \\ & && E' = (E \cup E^+) \setminus E^- \\ & && E^+ \subseteq \{(u, v) : u, v \in V, (u, v) \notin E\} \\ & && E^- \subseteq \{(u, v) : u, v \in V, (u, v) \in E\} \\ & && |E^+| + |E^-| \leq \beta \end{aligned} \tag{5}$$

The challenge is determining the best set of edges, within a budget β , for minimizing NDRE. Intuitively, searching across all possible edge additions and deletions (and their combinations) is impractical for most real-world applications. However, we can devise an efficient strategy that explores only the most promising operations (deception-wise). Like in similar scenarios (e.g., collapsed k-core [33]) we will *significantly reduce the search space* by setting theoretical conditions for choosing deleted or added edges. To do so, we need to evaluate the impact of edge updates on the NDRE.

Edge Deletion: We now show that given any two arbitrary edges $(u, v), (x, y) \in E$, deleting (u, v) will yield higher entropy than deleting (x, y) , if and only if $\mathbf{d}_u^\uparrow \geq \mathbf{d}_x^\uparrow$. Theorem 1 shows that an attack strategy for CSD should *select edges having source nodes with the highest out-degree*.

Theorem 1. Given two arbitrary edges $(u, v), (x, y) \in E$, $\mathcal{H}(G \ominus (u, v)) \geq \mathcal{H}(G \ominus (x, y))$ if and only if $\mathbf{d}_u^\uparrow \geq \mathbf{d}_x^\uparrow$.

Proof.

$$\begin{aligned} \Delta &= \mathcal{H}(G \ominus (u, v)) - \mathcal{H}(G \ominus (x, y)) \\ &= \frac{1}{m-1} \left[(\mathbf{d}_x^\uparrow - 1) \log_2 (\mathbf{d}_x^\uparrow - 1) + \mathbf{d}_u^\uparrow \log_2 \mathbf{d}_u^\uparrow - (\mathbf{d}_u^\uparrow - 1) \log_2 (\mathbf{d}_u^\uparrow - 1) - \mathbf{d}_x^\uparrow \log_2 \mathbf{d}_x^\uparrow \right] \end{aligned} \quad (6)$$

We observe that $\Delta \geq 0$ if and only if:

$$\left[(\mathbf{d}_x^\uparrow - 1) \log_2 (\mathbf{d}_x^\uparrow - 1) - \mathbf{d}_x^\uparrow \log_2 \mathbf{d}_x^\uparrow \right] \geq \left[(\mathbf{d}_u^\uparrow - 1) \log_2 (\mathbf{d}_u^\uparrow - 1) - \mathbf{d}_u^\uparrow \log_2 \mathbf{d}_u^\uparrow \right] \quad (7)$$

Which reduces to:

$$\left[\mathbf{d}_x^\uparrow \log_2 \frac{\mathbf{d}_x^\uparrow - 1}{\mathbf{d}_x^\uparrow} - \log_2 (\mathbf{d}_x^\uparrow - 1) \right] \geq \left[\mathbf{d}_u^\uparrow \log_2 \frac{\mathbf{d}_u^\uparrow - 1}{\mathbf{d}_u^\uparrow} - \log_2 (\mathbf{d}_u^\uparrow - 1) \right] \quad (8)$$

Let $f(\gamma) = \gamma \log_2 \frac{\gamma - c}{\gamma} - c \log_2 (\gamma - c)$, where c is a constant. Taking the derivative of f w.r.t γ , we can show that it is monotonically decreasing for $\gamma > 0$. Since both sides of Eq. (8) resemble f , and $\mathbf{d}_u^\uparrow \geq \mathbf{d}_x^\uparrow$, the inequality holds.

Edge Addition: Next, Theorems 2 and 3 study the effect of edge additions on NDRE.

Theorem 2. Given two arbitrary new edges $(u, v), (x, y) \notin E$, $\mathcal{H}(G \oplus (u, v)) \geq \mathcal{H}(G \oplus (x, y))$ if and only if $\mathbf{d}_u^\uparrow \leq \mathbf{d}_x^\uparrow$.

Proof (Sketch). We start by setting $\Delta = \mathcal{H}(G \oplus (u, v)) - \mathcal{H}(G \oplus (x, y))$, and then substituting each term using Eq. (1), we can algebraically show that

$$\Delta = \frac{1}{m+1} \left[\mathbf{d}_x^\uparrow \log_2 \frac{\mathbf{d}_x^\uparrow + 1}{\mathbf{d}_x^\uparrow} - \mathbf{d}_u^\uparrow \log_2 \frac{\mathbf{d}_u^\uparrow + 1}{\mathbf{d}_u^\uparrow} + \log_2 \frac{\mathbf{d}_x^\uparrow + 1}{\mathbf{d}_u^\uparrow + 1} \right] \quad (9)$$

Finally, we conclude that Δ is positive whenever $\mathbf{d}_u^\uparrow \leq \mathbf{d}_x^\uparrow$, by showing that the function $f(\gamma) = \gamma \log_2 \frac{\gamma + 1}{\gamma}$ is increasing for $\gamma \geq 0$.

The results in Theorem 1 (deletion) and Theorem 2 (addition) suggest that a sensible attack strategy should search for maximum and minimum out-degree source nodes, respectively. To further refine the attack strategy, Theorem 3 shows that for any pair of new edges (edges not already in the graph), the effect of adding any of them would be identical on the numerator of Eq. (4) in any of the following cases: (i) a pair of *inter-community* new edges, where both *source nodes* belong to the same community C_i ; (ii) a pair of *intra-community* new edges belonging to the same community C_k .

Theorem 3. Given two new edges (u, v) , (x, y) and some community $C \in \mathcal{P}$, where $u, x \in C$, then $\mathcal{H}(G \oplus (u, v)) - \mathcal{H}_{\mathcal{P}}(G \oplus (u, v)) = \mathcal{H}(G \oplus (x, y)) - \mathcal{H}_{\mathcal{P}}(G \oplus (x, y))$.

Proof. Let X, Y be two probability distributions such that $X \sim (\frac{\mathbf{v}_{C_1}^\uparrow}{\mathbf{m}}, \dots, \frac{\mathbf{v}_{C_k}^\uparrow}{\mathbf{m}})$, and $Y \sim (\frac{c_{1,1}}{\mathbf{v}_{C_1}^\uparrow}, \frac{c_{2,1}}{\mathbf{v}_{C_1}^\uparrow}, \dots, \frac{c_{n,1}}{\mathbf{v}_{C_1}^\uparrow}, \dots, \frac{c_{1,k}}{\mathbf{v}_{C_k}^\uparrow}, \dots, \frac{c_{n,k}}{\mathbf{v}_{C_k}^\uparrow})$, where $c_{i,j} = \mathbf{d}_i^\uparrow$ if $i \in C_j$, and $c_{i,j} = 0$ otherwise. Note that the entropy of the joint probability distribution of X and Y is:

$$\mathcal{H}(X, Y) = - \sum_{i=1}^{|V|} \frac{\mathbf{d}_i^\uparrow}{\mathbf{m}} \log_2 \frac{\mathbf{d}_i^\uparrow}{\mathbf{m}} = \mathcal{H}(G) \quad (10)$$

Hence, we can rewrite Eq. (10) as:

$$\begin{aligned} \mathcal{H}(X, Y) &= \mathcal{H}(Y|X) + \mathcal{H}(X) \\ &= \sum_{j=1}^k \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \mathcal{H}(Y|X = j) + \mathcal{H}(X) = \sum_{j=1}^k \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \mathcal{H}(G|C_j) - \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \log_2 \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \end{aligned} \quad (11)$$

At this point, using eqs. (11) and (3) we have:

$$\mathcal{H}(G) - \mathcal{H}_{\mathcal{P}}(G) = \sum_{j=1}^k \frac{\tilde{\mathbf{m}}_{C_j}^\uparrow - \mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \log_2 \frac{\mathbf{v}_{C_j}^\uparrow}{\mathbf{m}} \quad (12)$$

since (12) and $u, x \in C$, we have $\mathcal{H}(G \oplus (u, v)) - \mathcal{H}_{\mathcal{P}}(G \oplus (u, v)) = \mathcal{H}(G \oplus (x, y)) - \mathcal{H}_{\mathcal{P}}(G \oplus (x, y))$.

Theorem 3 states that the directed residual entropy would be the same after adding a new edge, regardless of its incident nodes, as long as the source node (sender) is in the community C_i , and both edges are either inter-community or intra-community. This implies that new edges incident to lower-degree source nodes have higher directed structural entropy (i.e., lower NDRE) than other new edges.

3.1 Algorithm and Complexity

Building on the theoretical results above, we present EBAD, an effective multi-level attack algorithm that finds the best network updates for minimizing NDRE, and thus, achieving deception on different levels. The pseudo-code of EBAD is shown in Algorithm 1. Given an attack type (structure, community, or node) and a budget update β , the algorithm selects the best candidate edge addition and deletion (lines 3-4); then, it applies on the graph the update having the lowest NDRE. The result is the updated graph G' .

Algorithm 1 The EBAD Attack Algorithm

Require: $G = \{V, E\}$, \mathcal{P} , update budget β
Ensure: updated network G'

```

1:  $G' \leftarrow G$ 
2: while  $\beta > 0$  do
3:    $\alpha^-, \beta^- \leftarrow getBestDel(G', \mathcal{P})$  /*Theorem 1*/
4:    $\alpha^+, \beta^+ \leftarrow getBestAdd(G', \mathcal{P})$  /*Theorem 2 & Theorem 3*/
5:    $\rho^- \leftarrow \rho_{\mathcal{P}}(G' \ominus (\alpha^-, \beta^-))$  // NDRE variation
6:    $\rho^+ \leftarrow \rho_{\mathcal{P}}(G' \oplus (\alpha^+, \beta^+))$  // NDRE variation
7:   if  $\rho^- < \rho^+$  then
8:      $G' \leftarrow \{V, E \setminus \{(\alpha^-, \beta^-)\}\}$ 
9:   else
10:     $G' \leftarrow \{V, E \cup \{(\alpha^+, \beta^+)\}\}$ 
11:   end if
12:    $\beta = \beta - 1$ 
13: end while
14: return  $G'$ 
```

Theorem 4. *Algorithm 1 runs in $O(k|V|)$ time, where k is the number of communities.*

Proof. Assuming four data structures to keep communities ordered in ascending (resp., descending) by the lowest (resp., highest) out-degree, two additional structures keep nodes sorted by in-degree and out-degree, respectively. To pick an edge for deletion, the algorithm iterates over all the k communities and for each community C_i : (i) gets the highest out-degree node u with at least one outgoing edge; (ii) among the neighbors of u gets the destination node randomly. This costs $O(|C_i|)$ for a total cost of $O(k|V|)$. For edge additions, the code loops over the k communities; for each community C_i : (i) gets the lowest out-degree node u ; (ii) picks a random community and a random node that is not a neighbor. Even in this case, the cost is $O(k|V|)$.

4 Experiments

We evaluate the multi-level performance of EBAD in real-world directed networks. The algorithms have been implemented in Python.

Datasets. We focused on directed social networks from a wide range of domains¹ whose *size* and *variety* align with (and exceeds) the evaluation of community detection algorithms (see, e.g., [19] for a survey). Table 3 reports, for each network, the number of communities found by the detection algorithms considered.

Detectors. We considered detection algorithms for *directed* networks² acting as adversaries to EBAD: (i) Leiden [28] (LEI); (ii) Directed modularity [11] (NM) [22]; (iii) Surprise [27] (SUR); (iv) InfoMap [24] (INF); (v) Gemsec [25] (GEM); (vii) Directed Label Propagation (DLP) [13]; (iv) Scalable Community Detection (SCD) [23] a state-of-the-art approach for large undirected networks.

¹ <https://snap.stanford.edu>.

² Implemented in the cdlib library <https://cdlib.readthedocs.io>.

Table 3. Datasets and number of communities found by each Detector.

ID	Network	V	E	Number of communities						
				LEI	NM	SUR	INF	GEM	DLP	SCD
FREE	Freeman	~50	~500	5	5	7	6	5	8	7
EM	Email	~1K	~25K	28	32	21	12	16	13	14
ANY	AnyBeat	~12K	~67K	129	81	143	156	112	128	137
WIKI	WikiVote	~7K	~103K	30	34	43	51	49	38	27
SOC	SocialNet	~900	~142K	6	5	6	7	5	4	6
EP	Epinions	~75K	~508K	795	986	689	876	546	645	587
ACA	Academia	~200K	~1.4M	89	93	2134	213	324	198	214
LIVEJ	LiveJournal	~4M	~34M	5457	5786	4356	1887	6578	4745	3794

Deceptors. Since no competitors work on directed networks and can hide entire community structures as EBAD, we considered the following undirected community structure deceptrors: REM, PROHICO (PROH). We could not compare EBAD with EPA since the code is not available³. We observe that EPA was only tested on three relatively small real networks due to the inherent complexity of the approach as stated in [3]. For PROH, we report the results of the best variant (DCSBM).

Evaluation Methodology. For each detection algorithm \mathcal{A}_D and network G , we compute the initial community structure \mathcal{P} on G then apply the updates suggested by the deceptor \mathcal{D}_A , giving a new network G' , and then recompute the new community structure with the same \mathcal{A}_D on G' . To measure performance, we refer to existing methodologies and use Jaccard, NMI, and Recall, where lower scores correspond to better performance [12]. Experiments have been conducted on an M2 Ultra server with 192 GB RAM. Results are the average (95% confidence interval) of 10 runs.

4.1 Experimental Results

We compare EBAD with REM, PROHICO and a strategy (RND) that randomly performs edge updates considering two budget update β values set as percentages of the number of edges in G following the methodology in [16]. Figure 1 shows the comparison of EBAD with its competitors in the four smallest networks. We note that EBAD achieves lower (NMI, Recall, Jaccard) scores almost universally compared to the three competitors. Considering NMI, for instance, EBAD outperforms PROH and REM by more than 0.25 (e.g. Anybeat network with NM). Here, baseline deceptrors seem to have no effect at all. This highlights the fact that EBAD is more effective against directed community detectors like NM. We see also that the random strategy RND sometimes (e.g. WIKI and INF detection algorithm) performed better than PROH and REM. This comes as a surprise since this behavior was not observed in the case of undirected networks.

³ We asked the authors to share the code and received no reply.

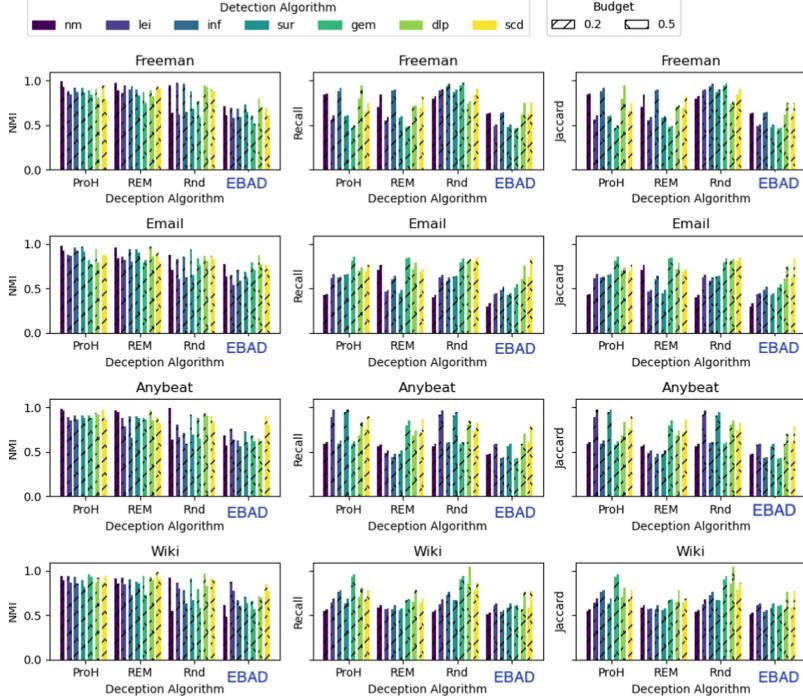


Fig. 1. Detailed experiments on community structure deception in small networks.

We note again that considering edge directions negatively impacts the two approaches. In larger networks (see Fig. 2), EBAD still outperforms three competitors, though the margins are smaller. Results are ranked and summarized in Table 4 using normalized scores, where the best deceptor achieves a composite score of 7. EBAD consistently outperforms PROH across all scenarios and metrics, while REM generally exceeds PROH and sometimes approaches EBAD (e.g., on WIKI with the INF detector for Recall and Jaccard). EBAD's superior performance is likely due to its refined strategies involving edge directions and deletions. Increasing the budget improves rankings in some cases, such as RND outperforming REM on WIKI for NMI at lower budgets. However, on LIVEJ, EBAD occasionally scores below 7, slightly underperforming PROH with the LEI detector.

Running Time. While details are not reported here for sake of space, we note that EBAD performed similar to REM, (e.g., 25 secs in SOC with ~ 1000 communities) with REM being faster on networks with a few communities. This is because EBAD considers both edge additions and deletions while REM only edge additions, which limits the effectiveness of deception where one can both add and delete edges. Moreover, in light of the theoretical results in Sect. 3, EBAD does not impose restrictions on the destination node of an edge deletion or addition, simplifying the choice.

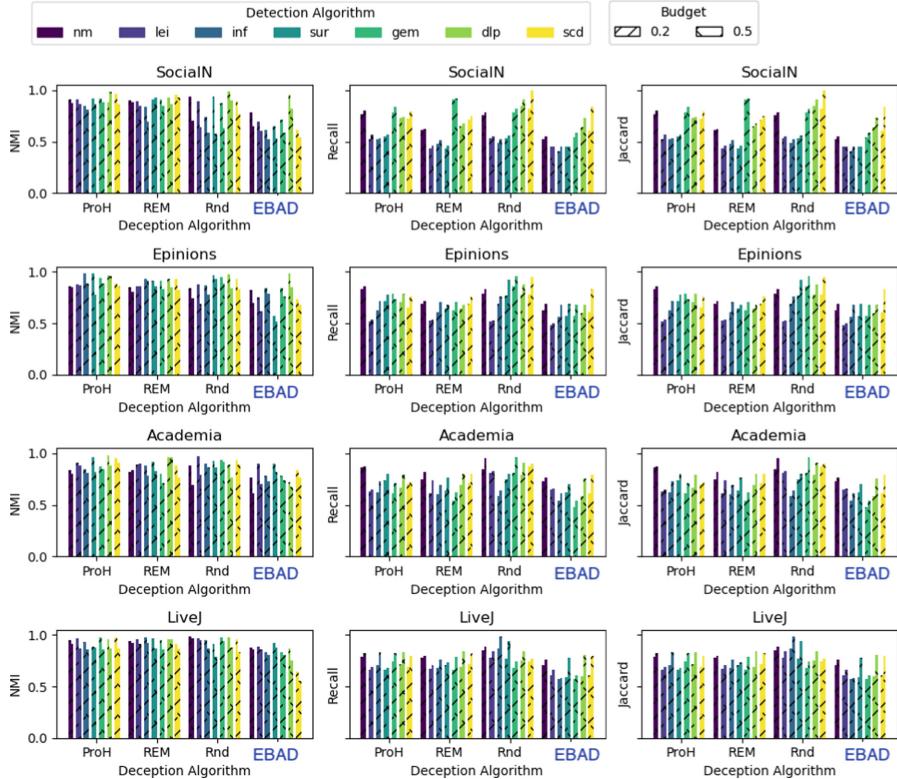
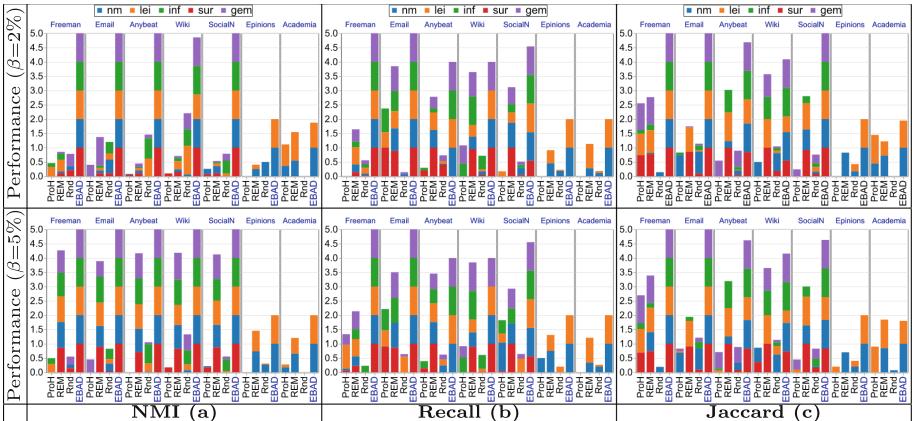


Fig. 2. Detailed experiments on community structure deception in large networks.

Table 4. y-axis shows composite performance. Results were min-max normalized, such that the winning deceptor (originally with lowest score), gets a composite score of 7.



5 Concluding Remarks and Future Work

We studied the vulnerabilities of community detection algorithms to adversarial attacks. Our motivation stemmed from the imperative to shield the intricate web of connections within networks from malicious entities that could exploit network knowledge to breach privacy. A notable gap in the current research landscape, which our study begins to bridge, is the treatment of directed networks. These networks, exemplified by platforms like Twitter, offer a rich tapestry of social dynamics where influence and information flow are not merely bilateral. Understanding attacks on such networks is crucial since they can have asymmetric properties that traditional undirected models fail to capture. Our research provides new insights into the vulnerabilities and defense mechanisms pertinent to the complex structures of directed networks.

In future work, we plan to study attacks from the perspective of node updates [17]. Furthermore, we will delve into the defense mechanisms that networks can employ to counteract such attacks. This includes real-time detection of anomalous node behavior, reinforcement of node-level security protocols, and developing resilient community detection algorithms that can withstand perturbations brought about by node updates.

References

1. Abawajy, J.H., Ninggal, M.I.H., Herawan, T.: Privacy preserving social network data publication. *IEEE Commun. Surv. Tutorials* **18**(3), 1974–1997 (2016)
2. Chen, J., et al.: Ga-based q-attack on community detection. *IEEE Trans. Comput. Soc. Syst.* **6**(3), 491–503 (2019). <https://doi.org/10.1109/TCSS.2019.2912801>
3. Chen, J., Chen, Y., Chen, L., Zhao, M., Xuan, Q.: Multiscale evolutionary perturbation attack on community detection. *IEEE Trans. Comput. Soc. Syst.* **8**, 62–75 (2020)
4. Chen, X., Jiang, Z., Li, H., Ma, J., Philip, S.Y.: Community hiding by link perturbation in social networks. *IEEE Trans. Comput. Soc. Syst.* **8**, 704–715 (2021)
5. Dhelim, S., Aung, N., Kechadi, M.T., Ning, H., Chen, L., Lakas, A.: Trust2Vec: Large-scale IoT trust management system based on signed network embeddings. *IEEE Internet Things J.* **10**(1), 553–562 (2023). <https://doi.org/10.1109/JIOT.2022.3201772>
6. Fionda, V., Madi, S.A., Pirrò, G.: Community deception: from undirected to directed networks. *Soc. Netw. Anal. Min.* **12**(1), 74 (2022)
7. Fionda, V., Pirrò, G.: Community deception or: how to stop fearing community detection algorithms. *IEEE Trans. Knowl. Data Eng.* **30**(4), 660–673 (2018)
8. Fortunato, S., Hric, D.: Community detection in networks: a user guide. *Phys. Rep.* **659**, 1–44 (2016)
9. Girvan, M., Newman, M.E.: Community structure in social and biological networks. *PNAS* **99**(12), 7821–7826 (2002)
10. Kang, S., Molinaro, C., Pugliese, A., Subrahmanian, V.S.: Randomized generation of adversary-aware fake knowledge graphs to combat intellectual property theft. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4155–4163 (2021)

11. Leicht, E.A., Newman, M.E.: Community structure in directed networks. *Phys. Rev. Letters* **100**(11), 118703 (2008)
12. Li, A., Pan, Y.: Structural information and dynamical complexity of networks. *IEEE Trans. Inf. Theory* **62**(6), 3290–3339 (2016)
13. Li, X.: Directed LPA: propagating labels in directed networks. *Phys. Lett. A* **383**(8), 732–737 (2019)
14. Liu, D., Chang, Z., Yang, G., Chen, E.: Hiding ourselves from community detection through genetic algorithms. *Inf. Sci.* **614**, 123–137 (2022)
15. Liu, X., Fu, L., Wang, X., Hopcroft, J.E.: ProHiCo: a probabilistic framework to hide communities in large networks. In: IEEE INFOCOM 2021–IEEE Conference on Computer Communications, pp. 1–10. IEEE (2021)
16. Liu, Y., Liu, J., Zhang, Z., Zhu, L., Li, A.: Rem: from structural entropy to community structure deception. *Adv. Neural. Inf. Process. Syst.* **32**, 12938–12948 (2019)
17. Madi, S.A., Pirrò, G.: Community deception in directed influence networks. *Soc. Netw. Anal. Min.* **13**, 122 (2023). <https://doi.org/10.1007/s13278-023-01122-8>
18. Madi, S.A., Pirrò, G.: Influence-based community deception. In: Cherifi, H., Mantegna, R.N., Rocha, L.M., Cherifi, C., Micciche, S. (eds.) *Complex Networks and Their Applications XI*, pp. 175–187. Springer International Publishing, Cham (2023). https://doi.org/10.1007/978-3-031-21131-7_14
19. Malliaros, F.D., Vazirgiannis, M.: Clustering and community detection in directed networks: a survey. *Phys. Rep.* **533**(4), 95–142 (2013)
20. Mittal, S., Sengupta, D., Chakraborty, T.: Hide and seek: outwitting community detection algorithms. *IEEE Trans. Comput. Soc. Syst.* **8**, 799–808 (2021)
21. Nagaraja, S.: The impact of unlinkability on adversarial community detection: effects and countermeasures. In: PETS, pp. 253–272 (2010)
22. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. *Phys. Rev. E* **69**, 066133 (2004). <https://doi.org/10.1103/PhysRevE.69.066133>
23. Prat-Pérez, A., Dominguez-Sal, D., Larriba-Pey, J.L.: High quality, scalable and parallel community detection for large real graphs. In: WWW, pp. 225–236. ACM (2014)
24. Rosvall, M., Bergstrom, C.T.: Maps of random walks on complex networks reveal community structure. *PNAS* **105**(4), 1118–1123 (2008)
25. Rozemberczki, B., Davies, R., Sarkar, R., Sutton, C.: GEMSEC: graph embedding with self clustering. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 65–72 (2019)
26. Stoltenberg, D., Maier, D., Waldherr, A.: Community detection in civil society online networks: theoretical guide and empirical assessment. *Soc. Netw.* **59**, 120–133 (2019)
27. Traag, V.A., Aldecoa, R., Delvenne, J.C.: Detecting communities using asymptotical surprise. *Phys. Rev. E* **92**(2), 022816 (2015)
28. Traag, V.A., Waltman, L., Van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 1–12 (2019)
29. Waniek, M., Michalak, T.P., Wooldridge, M.J., Rahwan, T.: Hiding individuals and communities in a social network. *Nat. Hum. Behav.* **2**(2), 139–147 (2018)
30. Wondracek, G., Holz, T., Kirda, E., Kruegel, C.: A practical attack to de-anonymize social network users. In: 2010 IEEE Symposium on Security and Privacy, pp. 223–238. IEEE (2010)

31. Yang, H., Chen, L., Cheng, F., Qiu, J., Zhang, L.: LSHA: a local structure-based community detection attack heuristic approach. *IEEE Trans. Comput. Soc. Syst.* **11**, 1–13 (2023). <https://doi.org/10.1109/TCSS.2023.3312394>
32. Zhang, C., et al.: Community deception in large networks: through the lens of Laplacian spectrum. *IEEE Trans. Comput. Soc. Syst.* **11**, 2057–2069 (2023)
33. Zhang, F., Zhang, Y., Qin, L., Zhang, W., Lin, X.: Finding critical users for social network engagement: the collapsed k-core problem. In: Proceedings of the AAAI Conference on Artificial Intelligence (2017)



Synthetic Networks That Preserve Edge Connectivity

Lahari Anne[✉], The-Anh Vu-Le[✉], Minhyuk Park[✉], Tandy Warnow[✉],
and George Chacko^(✉)[✉]

Siebel School of Computing and Data Science, Grainger College of Engineering,
University of Illinois Urbana-Champaign, Urbana, IL 61801, USA
{warnow,chackoge}@cs.illinois.edu

Abstract. Since true communities within real-world networks are rarely known, synthetic networks with planted ground truths are an alternative for evaluating community detection methods. Of several available synthetic network generators, Stochastic Block Models (SBMs) produce networks with ground truth clusters that well approximate input parameters from real-world networks and clusterings. However, SBMs can produce disconnected ground truth clusters, even when provided parameters from clusterings where all clusters are connected. Here we describe the REalistic Cluster Connectivity Simulator (RECCS), a technique that creates and then modifies an SBM synthetic network to improve the fit to a given clustered real-world network. Using real-world networks up to 13.9 million nodes in size, we show that RECCS results in synthetic networks that have a better fit to cluster edge connectivity than their starting SBMs, while providing roughly the same quality fit for other network and clustering parameters as unmodified SBMs.

Keywords: synthetic networks · community detection

1 Introduction

Community detection methods resolve networks at the meso-scale by identifying clusters of nodes that exhibit desired properties, such as density, edge connectivity, and separability from the remainder of the network [3–5, 7, 13, 18]. An understanding of when a method produces clusters of good quality is of some importance given the diverse applications of community detection (clustering) methods.

Since ground truth communities are not reliably known in real-world networks, evaluation using synthetically generated networks with planted ground truth communities provides a useful alternative [14]. Several synthetic network generators are in use, including Stochastic Block Models (SBM) [15, 16], the

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-82435-7_14.

LFR (Lancichinetti-Fortunato-Radicchi) generator [10], ABCD and ABCD+o (Artificial Benchmark for Community Detection) [8, 9], and nPSO (nonuniform popularity similarity optimization) [12].

Synthetic network generators should produce networks with ground truth clusterings with properties similar to those of real-world networks [12, 19]. It has been shown that SBMs produced using graph-tool [15] have a good fit to many real-world *network properties*, such as degree sequence, local and global clustering coefficients, and diameter [19]. However, whether the ground-truth *clusterings* in the synthetic networks resembled the clusterings of the real-world networks and the fit between outlier nodes in the clustered real-world networks and the outlier nodes in the synthetic network was not examined. However, in [20], the importance of having ground truth clusters that are at least connected is recognized, along with an approach to modify a synthetic network with ground truth clustering to ensure connectedness.

Here, we present RECCS, the REalistic Cluster Connectivity Simulator, which addresses the goal of approximating the edge connectivity of clusters in a given clustered real-world network, while not worsening the fit with respect to other network and clustering properties. The input to RECCS is a real-world network and a set of parameters obtained from clustering it. RECCS first computes and then modifies an SBM for the clustered subnetwork in order to improve the fit to the edge connectivity in the real-world clustering. In the second step the remaining “outlier” nodes are added. We show that, compared to SBM alone, this two-step approach produces synthetic networks that have excellent fit for the edge connectivity of the real-world clustered network, while maintaining the fit for other empirical statistics of the clustered real-world network.

2 Materials and Methods

2.1 New Synthetic Network Generation Pipelines

High-Level Description. The input is a real-world network G and its clustering. Note that the clustering may include clusters containing only a single node (singleton clusters). We refer to nodes in singleton clusters as being “unclustered” or “outliers”, and all other nodes are considered “clustered”. The subnetwork of G induced by the non-singleton clusters is called the “clustered subnetwork” and is denoted by G_c . From this network and clustering, we extract the parameters required for degree-corrected SBM network generation, which includes the assignment of nodes to clusters, number of edges within each cluster and between each pair of clusters, and the node degree sequence. We also calculate the edge connectivity of each of the clusters, which we now define. For a given non-singleton cluster C , an edge cut is a set of edges such that deleting those edges, but not the endpoints, disconnects the cluster C . The size of the smallest edge cut for C is its edge connectivity, and is denoted $k(C)$.

We then produce a synthetic network N using the degree-corrected SBM network generation methods in graph-tool to model the clustered subnetwork G_c . If G has any unclustered nodes, then the clustered subnetwork G_c will not be

the entire network. Next, we make the SBM network a *simple graph* by removing self-loops and replacing each set of parallel edges with a single edge.

We call this modified network N_c , noting that it is a reduced version of the original SBM network N , and may have fewer edges than in the real-world network G_c ; this provides us the opportunity to strategically add edges to ensure the required edge connectivity within the clusters while maintaining the integrity of other network properties. This is Step 1 of the network generation procedure.

We also add outlier nodes back into the network, and determine the edges that are incident with these outlier nodes; this is Step 2 of the network generation process, which produces a network only containing edges involving at least one outlier node. Finally, the two synthetic networks are *merged* into one synthetic network. For additional details, see the Supplementary Materials [1].

Step 1: Improving Edge Connectivity. We refer to the set of parameters we calculate from the real-world network G_c and its clustering \mathcal{C} as $Param(G_c, \mathcal{C})$. The problem we seek to solve can be described as follows:

- Input: Simple graph N_c with clustering \mathcal{C} and $Param(G_c, \mathcal{C})$, where \mathcal{C} does not have singleton clusters
- Output: Network N with the same clustering \mathcal{C} formed by adding edges to N_c , with the objective of having a good fit to $Param(G_c, \mathcal{C})$.

The set $Param(G_c, \mathcal{C})$ includes the edge connectivity values $k(C)$ for every cluster in \mathcal{C} . Since SBMs in general provide a good fit to many network parameters but not to edge connectivity in clusters, we seek to improve the fit to the $k(C)$ values without hurting the fit to the other parameters.

In our experiments, we explored techniques to solve this problem that operate in two phases. In the first phase, we add edges to ensure that every cluster has edge connectivity at least $k(C)$, where $\{k(C) : C \in \mathcal{C}\}$ is part of the input parameter set. In the second phase we add additional edges to improve the fit with respect to the degree sequence. The following two-phase approach comprises Step 1 of the “RECCS” workflow, shown in Fig. 1.

Step 2: Adding in Outliers. Given the network G and its clustering \mathcal{C} , we now focus on creating a synthetic version of the subnetwork G^* of G containing the same nodes but only those edges where at least one endpoint is an outlier node, i.e., the “outlier edges”, which can connect two outlier nodes or one outlier node and one clustered node. Note that G^* includes clustered nodes as well as outlier nodes, and has the same set of clusters as in \mathcal{C} ; however, there are no edges within any non-singleton cluster, and no edges between any two non-singleton clusters.

We propose three strategies for this problem, ranging from Strategy 1, which has the least randomness, to Strategy 3, which has the most randomness. Each takes the input parameters computed from G^* and \mathcal{C} and then returns a synthetic network N^* consisting of outlier edges.

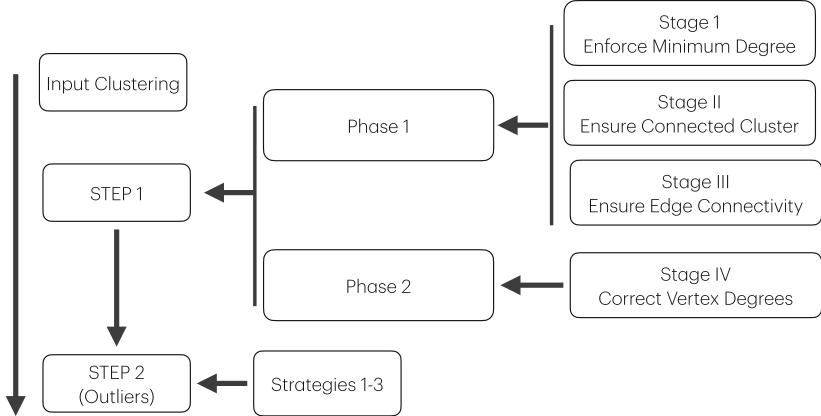


Fig. 1. RECCS Workflow. The input to RECCS is a collection of parameters computed on a real-world network N and an estimated clustering. Step 1 of RECCS produces a synthetic network corresponding to the clustered subnetwork of N , and Step 2 adds in the “outlier” nodes. Phase 1 of Step 1 computes an initial synthetic network using the graph-tool SBM software, and then adds edges within the clusters and between clusters to achieve a good fit to the input parameters, focusing on achieving the cluster connectivity (Stages 1, 2, and 3); Phase 2 then modifies the network to improve the fit with respect to the degree sequence. See Sect. 3 for additional details.

Strategy 1: This approach has each outlier in its own cluster, and then passes all the parameters computed for G^* to degree-corrected SBM to generate a network.

Note that this approach reproduces exactly the edges between outlier nodes as well as the number of edges between each outlier node and non-singleton cluster. However, there is still randomness in the assignment of edges between outlier nodes and clustered nodes.

Strategy 2: All the outliers are placed in a single cluster. The set of parameters restricted to the outlier cluster (i.e., the within-cluster degree sequence and number of edges) is used to generate the edges for the outlier cluster.

The remaining edges, between outliers and clustered nodes, are added at random for each outlier node and non-singleton cluster from \mathcal{C} in turn. This strategy is more random than Strategy 1 for how it places edges between outlier nodes, but handles edges between outliers and clustered nodes identically as Strategy 1.

Strategy 3: All the outliers are placed in a single cluster, and the parameters from G^* are used to generate a degree-corrected SBM.

Postprocessing: We postprocess N^* as follows: (1) if any self-loops or parallel edges have been created, the excess edges are removed so that the network becomes a simple graph, and (2) if the outlier nodes had been placed in a single cluster during the generation process (Strategies 2 and 3), then this artificial

treatment is ignored, and the outlier nodes are each considered to be singleton clusters.

Merging: Finally, the random networks generated in Step 1 and Step 2 are merged together; this is straightforward, since the node labelings for each random network are drawn from G .

2.2 Datasets

The datasets used comprise clustered networks, with some specifically used for the algorithm design (training) phase, and others used for the evaluation (testing) phase.

Real-world networks: We used 110 real-world networks that ranged from 1,000 to 13.9 million nodes that were taken from Netzschleuder [17] and the SNAP [11] repositories; see Supplementary Materials [1] for the list of datasets.

Clustering Methods: We clustered all the real-world networks by using a standard clustering method followed by the Connectivity Modifier (CM) [13]. For clustering, we used the Leiden algorithm [18] optimizing either the Constant Potts Model (CPM) with a range of resolution parameters or modularity, and the Iterative k-core method [21].

The Connectivity Modifier operates as follows. First, it removes clusters below size 11. Then, for all clusters that are considered to be poorly connected (because the size of its minimum edge cut is at most $\log_{10}(n)$ where n is the number of nodes in the cluster), CM removes the edge cut (thus breaking the cluster into two pieces), reclusters each piece, and repeats, until all clusters are well-connected. Finally, the clusters below the minimum size (11 by default) are removed. This last step was omitted in this study.

Training Data: We clustered all 110 networks using Leiden optimizing the Constant Potts Model (CPM) with resolution parameter $r = 0.001$, followed CM as described above.

Testing Data: We selected six large real-world networks from the set of 110 networks—Cit_hepph, Cit_patents, the Curated Exosome Network (CEN), Orkut, Wiki_talk, and Wiki_topcats—and clustered these networks using Leiden optimizing CPM at two different resolutions ($r = 0.01$ and $r = 0.1$), Leiden optimizing modularity, and Iterative k-Core (IKC) with $k = 10$, each followed by CM. Because the clusterings are different, these testing data are disjoint from the training data. Additional results for these 110 networks using a different clustering pipeline are included in the Supplementary Materials [1].

2.3 Evaluation Criteria

We evaluate the similarity between synthetic and real-world networks using various network properties. The key network properties include the edge connectivity of the clusters (minimum cuts), network diameter, mixing parameter, degree sequence, ratio of disconnected clusters, and clustering coefficients (both global and local). We use (1) a simple difference, calculated as $(s - s')$, where s is

the statistic value of the real-world network and s' is the statistic value of the synthetic network, for scalar properties bounded between 0 and 1, (2) relative difference calculated as $(s - s')/s$ for unbounded scalar statistics, and (3) Root Mean Square Error (RMSE), where $\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (s_i - s'_i)^2}$, for comparing sequences. For outlier modeling strategies, we examine the outlier degree sequence and the number of edges involving outlier nodes. Finally, we report the normalized edit distance between the synthetic network and the real-world network, given the bijection between the node sets.

2.4 Experiments

Experiment 1 - Preliminary analysis of SBM: We compute synthetic networks using SBM given parameters on the training dataset. The parameters that are given include the node-to-cluster assignment, number of edges within each cluster and between each pair of clusters, and the degree of every node in the network. The main focus is evaluating the frequency of disconnected ground truth clusters.

Experiment 2 - Algorithm Design and Development: We explore the algorithm design for the two steps of the synthetic network generation strategy, where the first step modifies the starting network with respect to edge connectivity within clusters and the second step focuses on outlier modeling. We use training data for this experiment, and compare our algorithms to SBM generation in graph-tool, post-processed to remove excess edges.

Experiment 3 - Evaluation on Test Data: We evaluate the pipelines we pick in Experiment 2 on testing datasets, in comparison to the generation of SBMs in graph-tool, post-processed to remove excess edges.

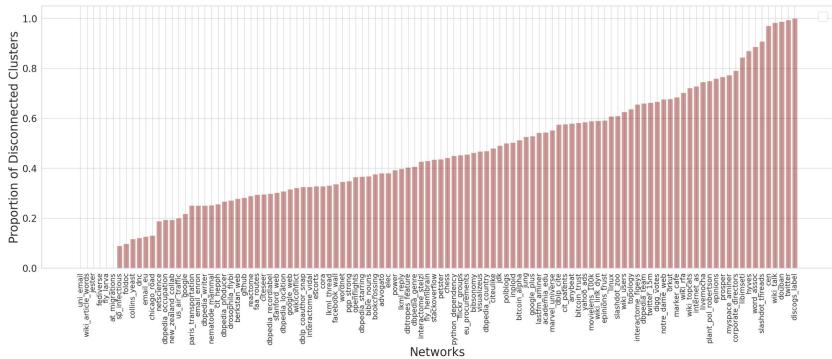


Fig. 2. Proportion of disconnected clusters in SBM generated networks. The x-axis shows 110 SBM networks generated using parameters from real world networks clustered with the Leiden+CM (Connectivity Modifier) pipeline (training data). The SBM method failed to reproduce the guaranteed connectivity of Leiden+CM clusters.

3 Results and Discussion

3.1 Experiment 1 Results: Preliminary Evaluation of SBM

Figure 2 shows the proportion of disconnected clusters in the 110 synthetic networks generated by SBM with inputs from training data as detailed in Sect. 2. Note that approximately half the networks have more than 40% of their clusters disconnected. Since the input clusterings were based on Leiden+CM and these are guaranteed to be connected, the input clustering parameters given to the SBM generation method in graph-tool are achievable with connected clusters, showing that SBM fails to recover this basic feature of the input clustering. This trend motivates our study.

3.2 Experiment 2 Results: Design of RECCS

The algorithmic structure of RECCS, provided in Fig. 1, is the result of a sequence of experiments on training data that we now describe. RECCS begins by computing a synthetic network N_c based on the set $\text{Param}(G_c, \mathcal{C})$ of parameter values computed from the clustered subnetwork G_c of a given real-world network G and its clustering \mathcal{C} , as described in Sect. 2.1. Recall that N_c is produced by removing excess edges from the SBM network, but it has the same ground truth clustering. RECCS operates by adding edges to N_c in two phases. Phase 1 adds edges to ensure that every cluster C has at least the target edge connectivity $k(C)$ (given in the input), and Phase 2 adds edges to improve the fit of the resultant degree sequence.

In our design of RECCS we initially explored techniques that omitted the second phase. These approaches had excellent fit for the edge connectivity but did not do as well for degree sequence. We then modified the design of first phase to also consider the degree of the nodes when adding edges to the network. However, these modifications did not fully address the edge deficit. We then introduced the second phase, which adds more edges to improve the degree sequence fit, but noticed that it slightly worsened the mixing parameter fit. To counter this effect, we developed a second version of the second phase. This resulted in two versions of RECCS that differ only in Phase 2, which is when edges are added to improve the fit of the degree sequence. RECCS operates in two phases, as follows.

- **Phase 1 (Ensure Edge Connectivity):** Edges are added to each cluster C to ensure edge connectivity at least $k(C)$, as follows:
 - **Stage 1: Enforce minimum degree:** Here we add edges to ensure that every node has at least $k(C)$ neighbors in the cluster. Therefore, if a node v has $d < k(C)$ neighbors in the cluster, we add $k(C) - d$ edges between v and other nodes in the cluster to which it is not adjacent.
 - **Stage 2: Ensure connected clusters.** If a cluster C is disconnected, we add $k(C)$ edges at random between its largest component and each of the other components.

- **Stage 3: Ensure edge connectivity.** This stage is an iterative method that ensures that the cluster C has edge connectivity at least $k(C)$. Specifically, we use VieCut [6] to calculate the size of a minimum edge cut; if this size is at least $k(C)$, then the cluster meets the desired minimum connectivity. Otherwise, the edge cut defines a partition of the cluster into two parts, and we add the required number of edges between the two parts. We repeat the process until the mincut of the cluster is at least $k(C)$.
- **Phase 2 (Correct vertex degrees):** We add edges to increase the degree of nodes with available degree (i.e., nodes whose current degree is below their target values) using two different techniques:
 - $v1$: for each node with available degree, we add edges to other nodes with available degree, following Algorithm 1 in the Supplementary Materials [1].
 - $v2$: edges are strategically added to nodes with available degrees, taking into account the number of inter-cluster and intra-cluster edges in the input subgraph G_c . This approach, detailed in Algorithm 2 in the Supplementary Materials [1], restricts the addition of edges to not exceed the expected number of inter-cluster edges.

Adding Edges. Recall that we are given a target degree for every node from G_c , but the synthetic network N may not achieve this degree for some nodes. Those nodes whose current degree is less than the target degree are said to be “nodes with available degree”. At each stage of Phase 1 of the algorithm, when adding an edge, we first randomly select nodes within the cluster with available degrees and update their available degree status accordingly. If no suitable nodes with available degree are found, we then randomly choose other nodes within the cluster, even if they have no available degree, to add the edge. We do not add parallel edges and self-loops.

3.3 Experiment 3 Results: Evaluation on Test Data

Here we show results of the two-step pipelines on the test data. There are six versions of the two-step pipeline, each formed by using RECCSv1 or RECCSv2 for the first step, and then followed by adding in the outlier nodes in three different ways in the second step. A comparison between all six pipelines on the full set of test networks and clusterings is shown in the Supplementary Materials [1], and reveals that Strategy 1 for Step 2 has the best accuracy. Due to space limitations, we present results here just for the two pipelines using outlier Strategy 1.

We begin with the results when clustering using Leiden-CPM with $r = 0.01$ followed by CM, in each case using Strategy 1 for outlier modeling. Both RECCSv1 and RECCSv2 improve the fit for the minimum edge cut size compared to SBM (Fig. 3, top panel). We also see an improvement in fit for both versions for degree sequence compared to SBM (with a larger improvement for

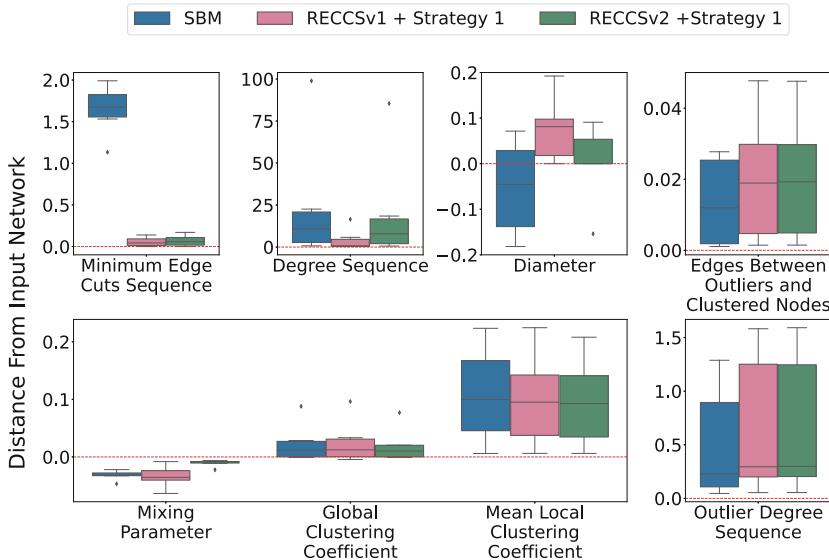


Fig. 3. Comparing SBM to the RECCS pipelines on the test networks using Leiden-CPM(0.01)+CM. We compare SBM networks to networks produced using the two pipelines, RECCSv1+Strategy 1 and RECCSv2+Strategy 1, for different network and clustering statistics. The y-axis shows different distance metrics for various network properties. Error is reported using RMSE for degree sequence, outlier degree sequence, and minimum edge cuts sequence; scalar difference for clustering coefficients and mixing parameter; and relative difference for the diameter, number of edges between outliers, and between outliers and clustered nodes. The test networks contain six real-world networks, each clustered using Leiden-CPM with $r = 0.01$ followed by CM.

RECCSv1). For diameter, RECCSv2 improves on the fit compared to SBM, but RECCSv1 is slightly worse. For the other criteria, the new pipelines have approximately the same accuracy as SBM.

Thus, the two pipelines—RECCSv1+Strategy 1 and RECCSv2+Strategy 1—both improve on SBM for edge cut sizes, with one clearly better suited for degree sequence and the other clearly better suited for diameter, and are nearly indistinguishable for the other properties. Results for the other test datasets (Fig. 4) show the same trends.

Finally, we compared the two pipelines and unmodified SBMs with respect to the *normalized edit distance* between the real-world network and the synthetic network they produce. For this distance, we use the number of edges that need to be added or removed from the real-world network, to produce the synthetic network, then normalized by the number of edges in the real-world network. Across the six networks in this test dataset, SBM and RECCSv2 are very close, while RECCSv2 produced synthetic networks that had a smaller normalized edit distance to the real-world network than RECCSv1 (Fig. 5). Furthermore,

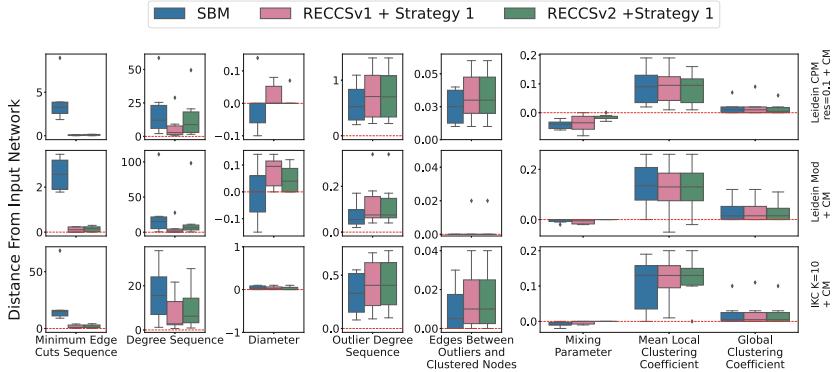


Fig. 4. Accuracy of SBM and Two RECCS pipelines on Test Data, using Three Additional Clusterings. The three additional clusterings are Leiden-CPM with $r = 0.1$ (top row), Leiden-modularity (middle row), and the Iterative k-core (IKC) method (bottom row). The y-axis shows different distance metrics for various network properties. Error is reported using RMSE for degree sequence, outlier degree sequence, and minimum edge cuts sequence; scalar difference is shown for clustering coefficients and mixing parameter; relative difference is shown for the diameter, number of edges between outliers, and between outliers and clustered nodes.

since the maximum normalized edit distance is 2.0, this shows that in all but one network, all three synthetic networks have a large enough distance to the real-world network to not simply replicate the real-world network. Thus, all three strategies—unmodified SBM and the two ways of post-processing the SBM—produce networks that are different from the real-world network and have good

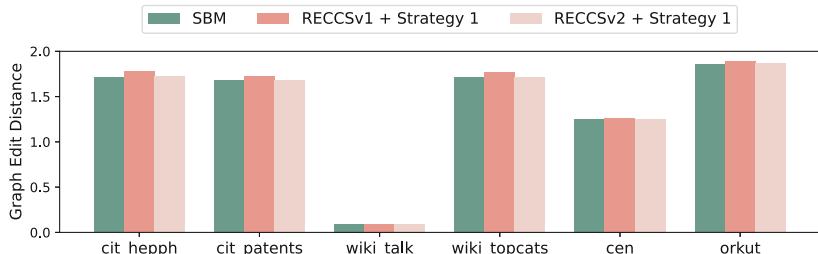


Fig. 5. Comparing SBM, RECCSv1, and RECCSv2 with respect to the normalized edit distance between synthetic and real world networks. The normalized edit distance between the edge sets of the true network G and the synthetic network N , i.e., $\frac{|E(G) \Delta E(N)|}{|E(G)|}$, where Δ denotes the symmetric difference, and so the maximum possible value is 2.0. Each real-world network is clustered using Leiden-CPM, with $r = 0.01$. Here, RECCSv2+Strategy 1 produces synthetic networks that are closer to the real-world network than RECCSv1+Strategy 1, and about as close as SBM networks.

fit for the network parameters we explored, while the two RECCS-pipelines also have a good fit for the cluster edge connectivity values but SBM does not.

4 Conclusion

Motivated by the need for synthetic networks that reproduce features of clustered real-world networks, we introduced the REalistic Cluster Connectivity Simulator (RECCS), which allows for variants of its basic two-phase approach and includes two strategies for adding in outlier nodes. We showed, using a diverse set of clustered real-world networks, that the RECCS pipelines that use outlier Strategy 1 produce synthetic networks that match or improve on the fit to empirical statistics of the clustered real-world networks compared to SBMs. Furthermore, the two versions of RECCS that we explore have different strengths, allowing for a range of synthetic networks to be developed.

In future work, we will explore a range of clustering methods on these synthetic networks in order to better characterize the conditions under which each method provides accuracy advantages over the other methods.

Funding Information. This work was supported in part by the Illinois-Inspier Partnership.

Software. The software for the RECCS pipeline, including the different outlier strategies we explore, are available from GitHub [2].

References

1. Anne, L., Le-Vu, T.A., Park, M., Warnow, T., Chacko, G.: Supplementary materials (2024). <https://doi.org/10.5281/zenodo.13367965>
2. Anne, L., Warnow, T., Chacko, G.: Github page for RECCS (2024). https://github.com/illinois-or-research-analytics/lanne2_networks
3. Coscia, M., Giannotti, F., Pedreschi, D.: A classification for community discovery methods in complex networks. *Stat. Anal. Data Min. ASA Data Sci. J.* **4**(5), 512–546 (2011). <https://doi.org/10.1002/sam.10133>
4. El-Moussaoui, M., Agouti, T., Tikiniouine, A., Adnani, M.E.: A comprehensive literature review on community detection: approaches and applications. *Procedia Comput. Sci.* **151**, 295–302 (2019). <https://doi.org/10.1016/j.procs.2019.04.042>
5. Fortunato, S., Newman, M.E.J.: 20 years of network community detection. *Nat. Phys.* **18**(8), 848–850 (2022). <https://doi.org/10.1038/s41567-022-01716-7>
6. Henzinger, M., Noe, A., Schulz, C., Strash, D.: Practical minimum cut algorithms. *J. Exp. Algorithmics (JEA)* **23**, 1–22 (2018)
7. Javed, M.A., Younis, M.S., Latif, S., Qadir, J., Baig, A.: Community detection in networks: a multidisciplinary review. *J. Netw. Comput. Appl.* **108**, 87–111 (2018). <https://doi.org/10.1016/j.jnca.2018.02.011>
8. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection (ABCD)-Fast random graph model with community structure. *Netw. Sci.* **9**(2), 153–178 (2021)

9. Kamiński, B., Prałat, P., Théberge, F.: Artificial benchmark for community detection with outliers (ABCD+ o). *Appl. Netw. Sci.* **8**(1), 25 (2023)
10. Lancichinetti, A., Fortunato, S.: Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E* **80**(1) (2009). <https://doi.org/10.1103/physreve.80.016118>
11. Leskovec, J., Sosić, R.: SNAP a general-purpose network analysis and graph-mining library. *ACM Trans. Intell. Syst. Technol. (TIST)* **8**(1), 1–20 (2016)
12. Muscoloni, A., Cannistraci, C.V.: A nonuniform popularity-similarity optimization (nPSO) model to efficiently generate realistic complex networks with communities. *New J. Phys.* **20**(5), 052002 (2018). <https://doi.org/10.1088/1367-2630/aac06f>
13. Park, M., et al.: Identifying well-connected communities in real-world and synthetic networks, pp. 3–14. Springer Nature Switzerland (2024). https://doi.org/10.1007/978-3-031-53499-7_1
14. Peel, L., Larremore, D.B., Clauset, A.: The ground truth about metadata and community detection in networks. *Sci. Adv.* **3**(5) (2017). <https://doi.org/10.1126/sciadv.1602548>
15. Peixoto, T.P.: The graph-tool python library. Figshare (2014). <https://doi.org/10.6084/m9.figshare.1164194>
16. Peixoto, T.P.: Bayesian stochastic blockmodeling. In: Doreian, P., Batagelj, V., Ferligoj, A. (eds.) *Advances in Network Clustering and Blockmodeling*, pp. 289–332. Wiley Online Library (2019)
17. Peixoto, T.P.: The Netzschleuder network catalogue and repository (2020). <https://doi.org/10.5281/zenodo.7839980>. Zenodo, 5201
18. Traag, V.A., Waltman, L., Van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**(1), 1–12 (2019)
19. Vaca-Ramírez, F., Peixoto, T.P.: Systematic assessment of the quality of fit of the stochastic block model for empirical networks. *Phys. Rev. E* **105**(5), 054311 (2022)
20. Viger, F., Latapy, M.: Efficient and simple generation of random simple connected graphs with prescribed degree sequence. *J. Complex Netw.* **4**(1), 15–37 (2016)
21. Wedell, E., Park, M., Korobskiy, D., Warnow, T., Chacko, G.: Center-periphery structure in research communities. *Quant. Sci. Stud.* **3**(1), 289–314 (2022)

Human Behavior



Criminal Network Construction and Analysis from Italian Civil Cases: A Case Study

Claudia Licari¹(✉) Gabriele Rinaldi², and Annamaria Ficara³

¹ Department of Mathematics and Computer Sciences, Physical Sciences and Earth Sciences, University of Messina, Messina, Italy

claudia.licari@studenti.unime.it

² Port System Authority of Stretto, Messina, Italy

gabrielerinaldi@pm.me

³ Department of Cognitive Sciences, Psychology, Education, and Cultural Studies, University of Messina, Messina, Italy

aficara@unime.it

Abstract. Criminal networks are complex systems that exploit social connections among their members. Analyzing these networks presents challenges, especially concerning data management. Researchers typically use either micro or macro approaches for criminal network analysis. The micro approach involves examining small criminal groups or organizations using judicial documents, intelligence reports, or investigation files. Conversely, the macro approach deals with larger networks at regional, national, or international levels, utilizing intelligence databases or archives. This paper employs institutional databases and open source intelligence to generate two networks: a large network from civil cases in the Court of a Sicilian city (in the period 2010–2014), and a smaller criminal network involved in a fraud case against the Public Administration. A preliminary analysis is conducted to identify key figures and communities in the criminal network.

Keywords: Criminal network · Fraud network · Social Network Analysis · Centrality

1 Introduction

Organized crime refers to a category of groups that operate secretly and unlawfully beyond the limits of legal frameworks, potentially causing severe harm to both societal structures and economic stability [12]. Criminal relationships of an associative nature can be considered complex systems, with typical dynamics and behaviors, which exploit social connections among their members. As a result of the increasing complexity of such criminal activities, networks have emerged that link not only different criminal organizations to each other and to common criminal activities, but also to institutions and professional consultants

within the legal economy. These connections can be analyzed in terms of network theory by representing them mathematically as graphs.

More specifically, Social Network Analysis (SNA) can be employed to investigate criminal phenomena [8]. From the 1990s through the early 2000s, the application of SNA in the study of organized crime experienced significant advancements. Interest in the field expanded considerably, leading to the emergence of new research directions [21]. One of the main goals of SNA is to gather and analyze data to identify the patterns of relationships and interactions among entities that play significant roles within a network [15, 16]. This approach is particularly significant in the study of criminal networks [19, 20]. Many studies in SNA applied to criminal investigations emphasize network metrics such as density, clustering, and centrality. Measures like degree, betweenness, closeness, and eigenvector centralities are especially useful for pinpointing key individuals within criminal organizations, whose removal could significantly disrupt the network [11]. These metrics can also be useful in developing crime prevention systems [11]. Sparrow [21] was the first researcher to pave the way for the use of criminal network analysis. It took nearly a decade to overcome the skepticism of many in the audience, who initially suspected he was working with law enforcement agencies (LEAs), during the presentation of his preliminary studies on drug trafficking networks. In the end, the critical approach adopted in data analysis gradually convinced most of the audience that there was more to the research than just identifying targets for arrest [5].

Today in Italy, Information and Communication Technologies (ICTs) for big data analysis are not typically included in the investigations of prosecutors and judicial police except in cases where the expert opinion of judicial assistants and technical advisors is required to assist during the search for evidence. Therefore, it is reasonable to assert that the availability of large volumes of data could slow down rather than help the work of LEAs and judicial authorities, especially when this data comes from heterogeneous sources. An example can be represented by the challenges of integrating data from environmental wiretaps, phone traffic, banking transactions, institutional databases, and multimedia content from seized fixed and mobile devices. Another challenge in the computational analysis of criminal networks is the issue of data quality [11]. In most cases, researchers rely on one of the few existing datasets, but a standardized data collection protocol has yet to be established. The existing criminal network datasets primarily concentrate on drug distribution networks and Mafia organizations, with comparatively less emphasis on gangs or human trafficking networks [11].

We can identify two main approaches for data management in criminal networks [11]: micro and macro. The micro approach [6, 7, 18] focuses on analyzing small criminal groups or single organizations, such as drug-trafficking organizations or mafia activities, to understand their structure and operations. In this case, criminal networks are generated from judicial documents, intelligence reports or investigation files. Problems related to these kinds of networks are (i) misunderstanding or misjudgment of the specific node relevance, and (ii) willing omission by the police. The macro approach [4, 7, 17] examines larger, more com-

plex networks at regional, national, or international levels, aiming to understand the interactions between different networks. In this case, criminal networks are generated from intelligence databases collected by LEAs or databases created through archival analyses. These networks suffer of issues related to validity, and reliability, the database quality and possible missing data.

In this paper, we used a combined approach to generate two real networks: the larger one (Net1) is a network of civil cases of the Courthouse of a Sicilian city in the period 2010–2014, and the smaller one (Net2), which is derived from the former, is a real criminal network involved in a fraud case against the Public Administration. The two networks have been generated respectively from using institutional databases and Open Source Intelligence (OSINT). In particular, journalistic investigations, hearing at criminal trial, interviews with legal professionals involved in a specific legal proceedings have been used to construct Net2. Our criminal dataset derives in fact from the Pathology Operation, i.e., a trial concluded in 2024. The Pathology Operation investigated on suspected fraud against the National Institute for Social Security (in italian, Istituto nazionale della previdenza sociale, INPS) for the recognition of disability pensions and other welfare benefits. The investigation involved doctors, lawyers, and other specialists, started after a judge begins to notice too many anomalies in the system and notify them to the local Prosecutor's Office. After a detailed description of our data collection and processing for the generation of the two real networks, this study focus on a preliminary analysis of the criminal network in order to highlight the full criminal potential of the criminal organization. In our analysis, Degree Centrality, Betweenness Centrality, Closeness Centrality and Eigenvector Centrality are used to identify the leaders of the criminal network, and the Louvain Algorithm to identify the communities within the criminal network itself.

The paper is organized as follows. Section 2 describes the tools used for managing and extracting data from civil cases, the generation of the two networks, and defines the fundamentals of SNA and centrality measures. Section 3 presents the obtained results, and Sect. 4 offers the research summary and future works.

2 Methodology

This paper aims to provide a new dataset along with insights into the methodology to be employed in judicial investigations involving a significant amount of data to analyze, in order to construct a first network of all civil cases of the Court of a Sicilian city in the period 2010–2014 (Net1), from which a criminal network (Net2) is extracted. In accordance with Freeman [13], the methodology used can be summarized in the following phases: (i) collection of data from investigative sources, newspapers, and databases; (ii) dataset processing; (iii) loading of data in an anonymous form and according to predefined rules; (iv) construction of the graph and calculation of centrality measures; (v) quantitative-relational analysis and selection of effective graphical representations.

2.1 Data Collection and Processing

The data extraction was performed using the Database (DB) from the Civil Information System (in italian, Sistema Informativo Civile, SIC) in the judicial district of the considered Sicilian city. SIC is a digital platform created to support the management of civil proceedings within the Italian judicial system. It uses a district-based Oracle Server DB, being entirely based on the VMware virtualization platform.

Our first activity involved the creation of a clone of the DB allowing tests to be conducted on experimental data without the risk of compromising the originals. Subsequently, it was possible to proceed with studying the DB according to the approach suggested by Bertino *et al.* [1]. It consists of two structures: the DB and the instance. The DB refers to all the physical files where the data is stored. The instance refers to the set of memory areas (System Global Area) and background processes necessary to access the data. The memory areas of an Oracle server are used to contain data, data dictionary information, SQL commands or Entity/Relationship (E/R) schema, and provide the ability to perform SQL queries, PL/SQL code, etc. The physical organization of the DB data complies with office and registry schemes (which collect topics according to the matter in dispute between the parties). Meanwhile, the conceptual organization of the registries is different, meaning the end-user only has a view of two subsystems for each office.

Our first challenge was understanding how to approach data extraction, given that there is no official documentation on the data model. To this end, a client machine was created, on which a tool for the DB administrator was installed. This tool was configured for access to the database of interest. Using this tool, the entire E/R schema of the DB was automatically reconstructed. Furthermore, queries were executed from this environment to obtain the data of interest. To do this, we used views, which are special programming constructs (i.e., queries stored in the DB) used to display a set of data aggregated from multiple tables or to display only a portion of the data contained in a table. These tables can contain many fields, and this phase required careful analysis. Therefore, through four partial views joined using the *join* clause, it was possible to extract all the data on civil cases useful for our experimentation. In particular, the data sample used was limited to civil disputes managed by the Court of the considered Sicilian city during the first half of 2009, amounting to approximately 1800 records, and the queries involved only 13 of the 132 tables in the District Civil Cognition Information System (in italian, Sistema Informativo Cognizione Civile Distrettuale, SICID). SICID is an ICT platform used in the Italian judicial system for the computerized management of civil proceedings in courts and courts of appeal. All relevant data was exported to an Excel file containing the following information: (i) case file identification code, (ii) annual sequential number, (iii) year of case registration, (iv) case file status (description and corresponding code), (v) case subject (description and corresponding code), (vi) data on judges (section affiliation, identification code), (vii) data on the parties (specific role in

the case, identification code), (viii) data on lawyers (identification code), (ix) data on technical advisors (identification code).

The central entity in the Italian civil judiciary is the case file, which is assigned to a court (or a single judge) and is associated with a dispute between two or more parties. Each party is represented by one or more lawyers. Judges may appoint experts for technical expertise in fields such as ICT, finance, ballistics, genetics, and so on. Unlike social networks, where connections typically link entities of the same type, namely individuals, in this study, we have created a semantic network only for the first network (Net1) that connects the aforementioned actors shown in Fig. 1.

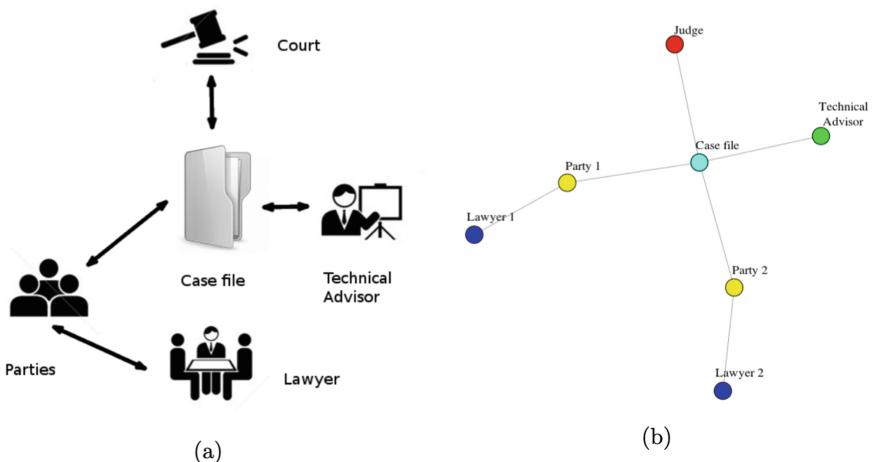


Fig. 1. The semantic model used for Net1 of the Italian Civil Judiciary (a) and its representation as a network (b)

In our model, there are several assumptions: there are no direct interactions between judges, nor between technical advisors. We must also assume that there is no interaction between lawyers, even if they represent the same parties. Then, the database was read four times to create the relationships. It required a thorough analysis from the database tables, as the central element of the constructed relationships is the case files. The data processing consisted in the following four steps: (i) reading connected judges with case files, (ii) linking case files with technical advisors, (iii) connecting case files with parties, (iv) reading linked parties with lawyers.

However, data related to the criminal activities from the Pathology operation mentioned in Sect. 1 are processed as follows. We focused on 58 case files which had been put under investigation. Many citizens, who had been denied welfare benefits by INPS, approached some firms for help. These firms directed them to a criminal organization promising a favorable outcome in appeals filed with the Labor Court. The success of the fraudulent operation was ensured by the

support and complicity of professional figures. In this way, the citizens supported by the criminal association always managed to win the cases and get reimbursed by INPS.

2.2 Data to Graph Transformation

As mentioned in Sect. 1, a challenge faced by SNA in the criminal domain involves the transformation of data into meaningful graphs. There is no standard approach for this task, and the process relies heavily on the analyst subjective judgment. For example, determining which nodes and thus which people involved in criminal activities to include in the network can be difficult.

A graph $G = \langle V, E \rangle$ is a structure consisting of two finite sets V and E . The set $V = \{1, \dots, N\}$ consists of the nodes (vertices or actors), where N is called the size of the graph. The set $E \subseteq N \times N$ consists of the edges (links or connections) between the vertices. If all the edges in a graph are bidirectional, then the graph is considered undirected. On the other hand, if the edges are represented by ordered pairs of vertices, the graph is directed. From a mathematical perspective, the data contained in the sets V and E can be represented as a matrix, known as the adjacency matrix A [23].

The associations between the interactions of network actors are generally analyzed using centrality measures. Centrality quantifies the importance and role of each node based on its position in the network. Nodes with high centrality are considered more significant concerning the property of interest [14]. However, the utility and practical meaning of centrality often depends on the specific criminal network being examined [9]. These indicators, referred to as global measures, characterize the network as a whole, and to evaluate the criminal network under investigation, we have selected the ones specified below.

Degree centrality [14] of a node is defined as the number of edges adjacent to it. It is a simple yet meaningful way to measure the importance of nodes within a network. For an undirected graph $G = (V, E)$ with n nodes, the degree centrality measure can be defined as:

$$C_D(v) = \frac{d(v)}{n - 1} \quad (1)$$

where $d(v)$ is the number of edges adjacent to node v . Since a node can at most be adjacent to $n - 1$ other nodes, $n - 1$ is the normalization factor introduced to make the definition independent of the network size, ensuring that $0 \leq C_D(v) \leq 1$. A node with a high degree can be considered a hub, an active node, and an important communication channel.

Closeness centrality is defined as the inverse of the sum of the shortest paths between a node and all other nodes in the graph [14]. It measures how central a node is concerning the spread of information and how far it is from all other nodes in the graph. Closeness centrality can be expressed as:

$$C_C(v) = \left[\sum_{u=1}^N d(v, u) \right]^{-1} \quad (2)$$

Actors with higher closeness centrality values are generally those more involved in the network and, therefore, “closer” to other nodes.

Betweenness centrality [14] is a centrality measure of a node, given by the number of shortest paths between each pair of nodes in the graph that pass through a specific node. The idea is that the more central a node is, the higher the number of shortest paths that pass through it. Betweenness centrality can be expressed as:

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \quad (3)$$

where σ_{st} is the number of shortest paths between s and t , and $\sigma_{st}(v)$ is the number of such paths that pass through v .

Eigenvector centrality [3] is defined as follows. Given a graph G and its adjacency matrix $A = (a_{v,t})$, the relative centrality x of the node v is computed as:

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in G} a_{v,t} x_t, \quad (4)$$

with $M(v)$ being the set of neighbors of the node v and λ a constant.

Finally, a network exhibits a community structure if its nodes can be grouped into collections such that the number of connections within each collection is significantly higher than the number of connections between collections. To detect communities, we used the concept based on modularity optimization [2] adopting the Louvain method [10]. The algorithm operates in two steps. In the first step, it assigns each node to its own community and then evaluates the potential modularity gain by moving each node into each of its neighboring communities, seeking the maximum positive gain. If no positive gain is found, the node stays in its initial community. The modularity gain from moving an isolated node into a community can be easily calculated using the following formula (derived from a combination of [2, 22], and some algebra):

$$\Delta Q = \frac{k_{i,\text{in}}}{2m} - \gamma \cdot \frac{\Sigma_{\text{tot}} \cdot k_i}{2m^2} \quad (5)$$

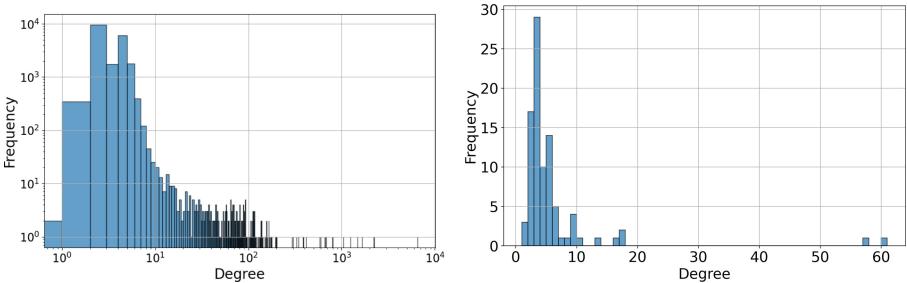
In this context, m represents the network size. The term $k_{i,\text{in}}$ refers to the total weight of the links from node i to nodes within community C . The value k_i denotes the sum of the weights of the links connected to node i , while Σ_{tot} is the cumulative weight of the links attached to nodes in C . Lastly, γ is the resolution parameter.

2.3 Network Generation

After data processing, two graphs have been generated at different times: (i) *Net1*, which represents a snapshot of the overall network of civil cases of the Court of the considered Sicilian city; (ii) *Net2*, which is a subgraph of *Net1* and a real criminal network. Some statistics related to the two networks are displayed in Table 1. The degree distribution of the two networks is shown in Fig. 2. *Net1* is

Table 1. Statistics of Net1 and Net2.

Network	Net1	Net2
Number of nodes	20,523	91
Number of edges	49,951	253
Average Degree	4.867806850850266	5.56043956043956
Average Shortest Path Length	3.9626985189718464	2.248107448107448
Largest Connected Component Size	20517	91
Network Diameter	10	4
Max Degree	6482	61
Density	0.00023719943723078966	0.061782661782661785

**Fig. 2.** Degree distributions in Net1 (Left Panel) and Net2 (Right Panel)

an undirected graph with 20,523 nodes and 49,951 edges, representing a snapshot of the actors and relationships within the Civil Court of the considered Sicilian city from 2010 to 2014. It follows the semantic model depicted in Fig. 1. Net2 is a subnet of Net1, representing the criminal network as a weighted undirected graph with 91 nodes and 253 edges. In fact, we extracted from Net1 the majority of the actors with specific roles within the criminal organization (i.e., lawyer, party and technical advisor). In Net1, we do not find those actors who promised the recognition of benefits in exchange for financial gain (i.e., INPS employee, business agent, legal secretary, Ministry of Justice employee). An edge between two nodes exists if they have committed a crime between them. For example, if a lawyer took a party to a doctor to get a false attestation of a medical condition, the lawyer, the party and the doctor are connected through an edge. Therefore, the reconstruction of the relationships for a single charge consists of a complete graph connecting all the nodes.

3 Results

In this Section, we focus on the analysis of the criminal network Net2.

For our analysis, Python Programming Language was used with NetworkX, i.e., a package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

SNA, in addition to considering the specific attributes of individuals, evaluates the set of relationships and exchanges within a social group to understand its power structure, control, degree of connection, information flow, and so on. In our network analysis, we used four centrality measures: Degree Centrality, Betweenness Centrality, Closeness Centrality and Eigenvector Centrality. For community detection, in order to identify groups of nodes within the network that are more densely connected to each other than to the rest of the network, we used the Louvain Algorithm.

Figure 3 shows the results of the aforementioned measures on Net2 in form of box plots to clarify the comparison between actor types. Our results reveal the presence of two particularly central actors according to the computation of all the centrality measures who exhibit an unusual behavior compared to the rest of the actors in their group. These actors correspond to a lawyer (Node 1) and a technical advisor (Node 2). More specifically, the lawyer group in the box models shows an upper whisker with the largest value of centrality for an actor with role “Lawyer”. The group of technical advisors shows an outlier that differ significantly from the rest of the group and is plotted as an individual point beyond the upper whisker on the box plot. These actors appear to be the leaders of the criminal organization, i.e., they are at the top of the criminal organization and hold leadership and decision-making roles in the group’s daily criminal activities and in the realization of its illicit goals. The other actors are just participants in the criminal organization, i.e., they provide a contribution to the criminal association without having decision-making power or autonomy.

To strengthen our findings on the search for leaders of the criminal organization, we used the Louvain algorithm to find communities in Net2. Our results are shown in Fig. 4. We identified six communities, each colored differently, with a well-organized distribution of roles and clear divisions of functions. The size of the communities is equal to 6, 11, 21, 25, 7 and 21, respectively. The largest communities are Community 3, Community 2, and Community 5. In particular, Community 3 is centered around Node 2 and Community 2 is centered around Node 1. These results confirm our findings on the leaders of the criminal organization which correspond to these two nodes. Community 5 reveals the centrality of another lawyer (Node 3), while from the smaller communities appears the importance of two other figures which are an INPS employee (Node 12) and a Business agent (Node 4). The role of “Party” is widely distributed across all

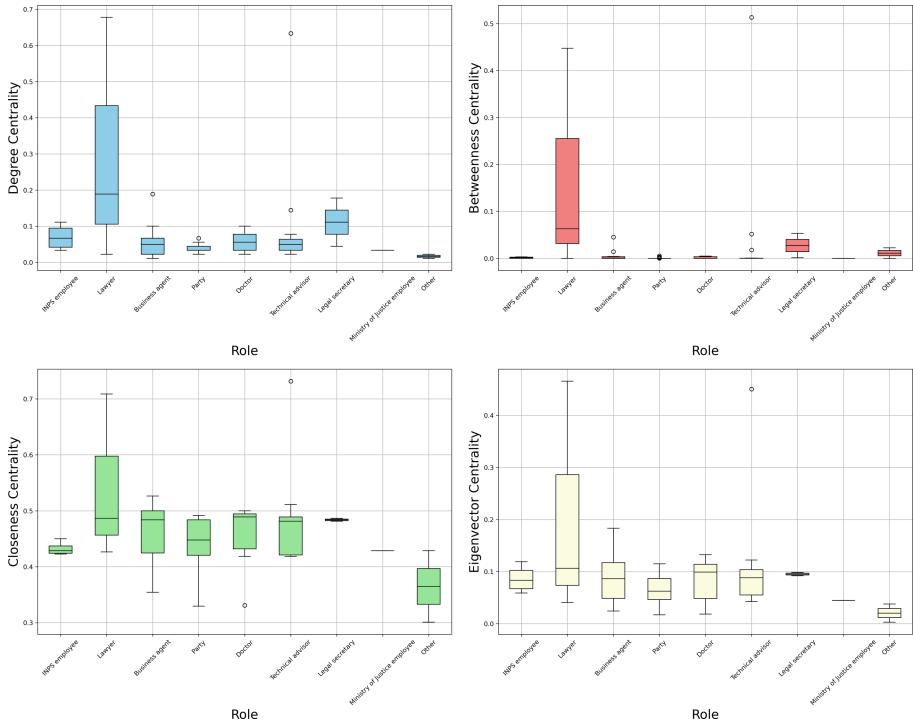


Fig. 3. Centrality distributions in Net2. Upper Left Panel: degree centrality distribution. Upper Right Panel: betweenness centrality distribution. Lower Left Panel: closeness centrality distribution. Lower Right Panel: eigenvector centrality distribution.

communities, suggesting that members with this role have a central operational function within the network. The role of “Doctor” is present in three different communities, suggesting his involvement in the criminal activity. Nodes with the roles “Other” and “Ministry of Justice employee” are positioned at the margins of the network, with few connections. Some nodes, such as those with the role of “Legal secretary” are present in small, tightly connected groups, suggesting a specialized support function with the exception of Community 5.

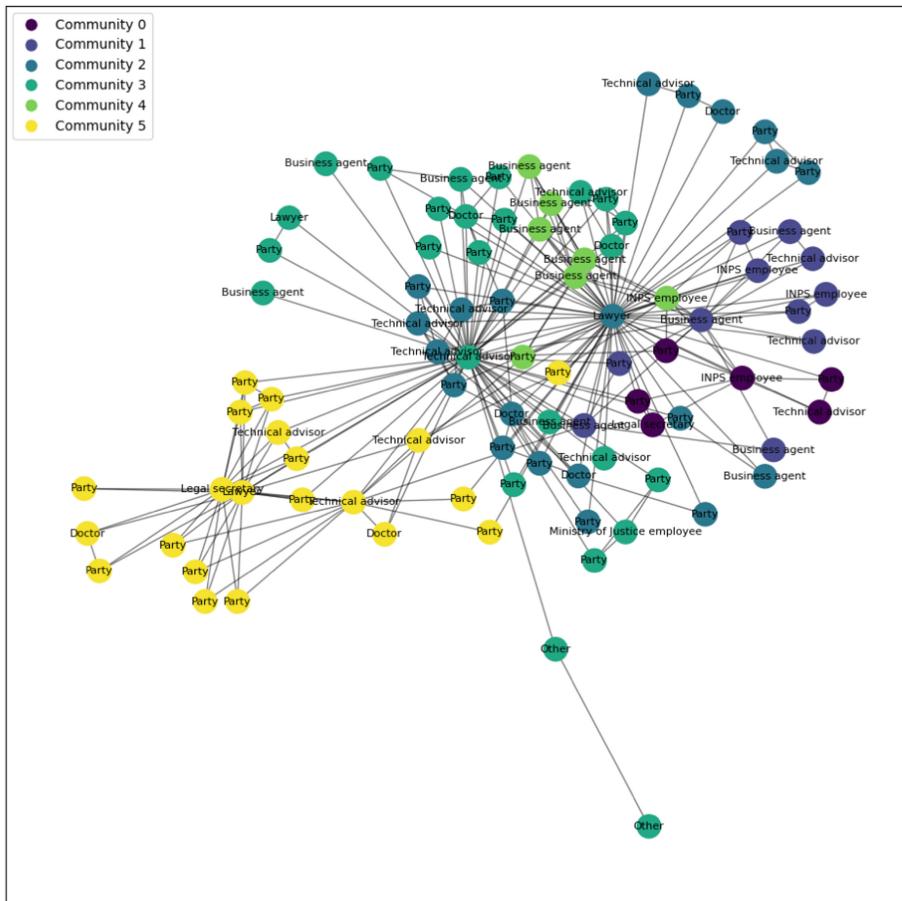


Fig. 4. Louvain Community Detection in Net2

4 Conclusion

In this article, we presented two real networks. The former (Net1) represents a network of civil cases of the Court of a Sicilian city in the period 2010-2014. The second one (Net2) is a real criminal network (with the names of the individuals anonymized for privacy reasons) constructed from 58 case files of Net1 put under an investigation called Pathology operation which led to precautionary measures against 33 key figures involved in the case. This network dataset has been enriched by the evidentiary elements presented during the investigators' press conference and by newspaper articles that reconstructed the events. In addition to the presentation of the new networks, we focus on the criminal network and analyzed it to find the main actors who held the reins of the criminal organization emerged. The quality of our analytical results has been confirmed by the survey results, which are consistent with the application of the

methodology presented here. The names of the main defendants in operation “Pathology”, published in newspapers, coincided exactly with what was revealed by our study. The methodology presented, further developed and verified, could be a valuable tool to assist investigators in identifying criminal networks operating in different fields, not immediately identifiable due to the large amount of data to analyze. Future improvements to our study may include a deeper comparison between Net2 and Net1, and a link prediction analysis in Net2 starting from the links existing in Net1. Moreover, we plan to transform our simple graphs into hypergraphs generalizing the concept of edges to hyperedges.

References

1. Bertino, E., Catania, B., Zarri, G.: Intelligent Database Systems. ACM Press books, Addison-Wesley (2001)
2. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *J. Stat. Mech. theory Exp.* **2008**(10), P10008 (2008)
3. Bonacich, P.: Power and centrality: a family of measures. *Am. J. Sociol.* **92**(5), 1170–1182 (1987)
4. Bouchard, M., Ouellet, F.: Is small beautiful? The link between risks and size in illegal drug markets. *Global Crime* **12**(1), 70–86 (2011)
5. Bright, D., Brewer, R., Morselli, C.: Using social network analysis to study crime: navigating the challenges of criminal justice records. *Soc. Netw.* **66**, 50–64 (2021)
6. Calderoni, F.: The structure of drug trafficking mafias: the ‘ndrangheta and cocaine. *Crime Law Soc. Chang.* **58**(3), 321–349 (2012)
7. Calderoni, F.: Social network analysis of organized criminal groups. In: Bruinsma, G., Weisburd, D. (eds.) *Encyclopedia of Criminology and Criminal Justice*, pp. 4972–4981. Springer, New York, NY, USA (2014). https://doi.org/10.1007/978-1-4614-5690-2_239
8. Campana, P.: Explaining criminal networks: strategies and potential pitfalls. *Methodol. Innov.* **9**, 2059799115622748 (2016)
9. De Andrade, R.L., Rêgo, L.C., Coelho da Silva, T.L., de Macêdo, J.A.F., Silva, W.C.: Energy disruptive centrality with an application to criminal network. *Commun. Nonlinear Sci. Num. Simul.* **99**, 105834 (2021)
10. De Meo, P., Ferrara, E., Fiumara, G., Provetti, A.: Generalized louvain method for community detection in large networks. In: 2011 11th International Conference on Intelligent Systems Design and Applications, pp. 88–93 (2011)
11. Ficara, A., Curreri, F., Fiumara, G., De Meo, P., Liotta, A.: Covert network construction, disruption, and resilience: a survey. *Mathematics* **10**(16), 2929 (2022)
12. Finckenauer, J.O.: Problems of definition: what is organized crime? *Trends Organized Crime* **8**, 63–83 (2005)
13. Freeman, L.: The development of social network analysis (2004)
14. Freeman, L.C.: Centrality in social networks conceptual clarification. *Soc. Netw.* **1**(3), 215–239 (1978)
15. Gamper, M.: Social network theories: an overview. In: *Social Networks and Health Inequalities*, pp. 35–48 (2022). https://doi.org/10.1007/978-3-030-97722-1_3
16. van der Hulst, R.C.: Introduction to social network analysis (SNA) as an investigative tool. *Trends Organized Crime* **12**, 101–121 (2009)
17. Malm, A., Bichler, G., Van De Walle, S.: Comparing the ties that bind criminal networks: is blood thicker than water? *Secur. J.* **23**, 52–74 (2010)

18. Morselli, C.: Inside Criminal Networks, vol. 8. Springer, New York, NY, USA (2009). <https://doi.org/10.1007/978-0-387-09526-4>
19. Morselli, C.: Assessing vulnerable and strategic positions in a criminal network. *J. Contemp. Crim. Justice* **26**, 382–392 (2010)
20. Rostami, A., Mondani, H.: The complexity of crime network data: a case study of its consequences for crime control and the study of networks. *PLoS ONE* **10**, e0119309 (2015)
21. Sparrow, M.K.: The application of network analysis to criminal intelligence: an assessment of the prospects. *Soc. Netw.* **13**, 251–274 (1991)
22. Traag, V., Waltman, L., van Eck, N.J.: From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**, 5233 (2019)
23. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Structural Analysis in the Social Sciences, Cambridge University Press (1994)



Improving Accuracy of Image Geolocalization Based on Region Segmentation Using Proximity Graph Partitioning

Rinto Koike^(✉), Takumu Toyama, and Takayasu Fushimi

Tokyo University of Technology, Tokyo, Japan
`{c0b21194df,c0b211173f,fushimity}@edu.teu.ac.jp`

Abstract. In this study, we tackle the geolocalization problem of identifying the location where a photograph was taken. Conventional methods often involve multiclass classification using area meshes, which means that the correlation between the photographs and the area classes is weak, limiting the performance of the extracted features. In this study, we propose a method to divide areas by constructing a proximity graph by connecting photography points of interest (POIs) and extracting communities. This strengthens the correlation between the image feature and the area class, making it possible to extract meaningful features. Through evaluation experiments using real data, we compare the differences between proximity graphs, and the differences between the proposed method and existing area segmentation methods such as administrative divisions.

Keywords: proximity graph · community structure · geolocalization

1 Introduction

With the spread of digital cameras and smartphones, an enormous number of photos are uploaded to the Internet every day. From the perspective of the personal information protection, many of these photos do not include location information, limiting their value. However, by improving technology that can automatically estimate the location of a photo, it will be possible to maximize the value of these photos. For example, in digital marketing and reviews, location information can be attached to photos to collect location-related word-of-mouth and reviews. By attaching location information to photos that customers have submitted, it is possible to promote a realistic experience to other customers. In social media, adding location information to posted photos enables discovery and sharing based on location. This allows for greater interaction with people who have been to the same location or who actually want to go there in the future. Given the above, estimating the location of a photo is an important research issue.

Geolocation or geolocalization is the task of identifying locations from photos and texts. Geolocalization has been studied frequently in recent years, and many methods attempt to solve the problem as a multiclass classification task by dividing the target area into several subareas and treating the areas as classes. Planet [12] and CPlanet [10] divide the target area into meshes, and StreetCLIP [5] divides the area into multiple hierarchical levels. In these methods, a Multi-Layer Perceptron for multiclass classification is added as a head on top of a backbone model specialized for image recognition of photos, such as VGGNet [11] and Vision Transformer [2], and finetuned. Therefore, in geolocalization using multiclass classification, the area class needs to be a meaningful set of photos. For example, if the area class is set by regional mesh or administrative division, the correlation between the area class and the photos taken will be weak, and the performance of the extracted features will be limited. On the other hand, photos posted on SNS (Social Networking Service) tend to be similar photos taken in the vicinity. For example, in the area surrounding a certain tourist spot, many photos with that spot as the subject will be posted. The existing geolocalization techniques mentioned above have been successful in identifying the area in which a photograph was taken by devising an area segmentation method.

In this study, we propose a new area segmentation method that uses photos posted on SNS, aiming to improve the accuracy of geolocalization. The proposed method constructs a proximity graph by connecting photo shooting points (POIs). Then it divides the proximity graph using community extraction to extract subareas where similar POIs are densely concentrated. The aim of this method is to enable the extraction of meaningful features through finetuning a CNN model based on a multiclass classification task of subareas.

2 Related Work

2.1 Geolocalization

This section summarizes the latest studies on geolocalization, which estimate locations from images.

Weyand et al. proposed a method called PlaNet, which treats the geolocation problem as a classification task by hierarchically splitting the locations where many photos are taken and estimating the location [12]. This method uses convolutional neural networks (CNNs) to extract photo features, leveraging CNN's translational invariance to recognize objects in the image regardless of their position and size. Seo et al. proposed a method they call CPlaNet, which treats the geolocation problem as a classification task and improves estimation accuracy through combinatorial partitioning of maps [10]. The method generates fine-grained classifications by intersecting multiple coarse-grained geographic partitions, enabling accurate location prediction at fine scales while maintaining sufficient training data for each class. Haas et al. proposed a method called StreetCLIP, which uses zero-shot learning for estimation, to address the problem in conventional geolocalization that the photos used for training data are biased toward specific regions where many photos are taken, such as urban areas,

making it difficult to estimate photos taken in regions where few people live, such as rural areas [5].

The commonality between these studies and ours is that they segment regions and estimate the location where the photo was taken. However, region classes based on regional meshes and administrative divisions have a low correlation with the photos taken. We differ in that we focus on the tendency of photos posted on SNS to be similar to photos taken in the surrounding area and create subarea classes that have a high correlation with the images.

2.2 Area Segmentation by Graph Partitioning

Studies on dividing a given geographical area into functional or uniform sub-areas have been conducted in geography. They are similar in terms of extracting certain areas with common characteristics. To extract these areas, multivariate and network analysis methods have been proposed. As a network analysis-based approach, Zhang et al. analyzed the topological structure of road networks and discriminated them into patterns [15]. Though the study was based on Traffic Analysis Zone (TAZ) delineation, Zhang et al. described the importance of deciding the analysis units and argued that they should be studied in the future. Farmer and Fotheringham applied a community detection method proposed by Newman [7] to the networks of travel-to-work flows and identified internally well connected and relatively cohesive sub-areas [3]. To delineate urban boundaries based on human movements, Yin et al. adopted a community detection method called Infomap [9] to a directed weighted network, where nodes and weighted links respectively represent underlying urban regions and Twitter users' displacements on them [13]. Another method treats urban road structures as a network, and based solely on the topological structure of road networks [4], extracts such functionally similar zones as mountainous areas, residential areas, and commercial districts. Thus, dividing a area into areas with common features has long been an important research subject that has been conducted across fields, but many of these researches use mesh grids based on latitude and longitude or administrative divisions as the minimum units, and target graphs that represent their adjacency relationships. On the other hand, in our study, the POI obtained from an SNS is used as the minimum unit, which is advantageous in that the shape and size of the area obtained by grouping POIs are flexible.

Recently, there exists a study to predict the category label of POIs using the hierarchical structure of categories [14]. Since the method proposed in [14] assumes that the POI labels can be inferred from the spatial arrangement of neighboring facilities, initializes the feature vector by multi-neighborhood analysis, and predicts the class label of POIs by using the feature vectors and the hierarchical structure of categories. In this way, although few studies have efficiently divided areas based on the feature and location information of a large number of photos posted on SNSs.

3 Proposed Method

We propose a method of area segmentation using proximity graphs and community extraction for geolocalization. Specifically, a proximity graph is created based on the latitude and longitude of the photo's shooting location. Community extraction is performed on the created proximity graph to partition the area.

Formally, it deals with the problem of predicting subarea label (location information) $\hat{\mathbf{c}} = [\hat{c}_i]_{i \in \mathcal{I}}$ for each of a given set of L pictures \mathcal{I} . The overview of our proposed method is shown below:

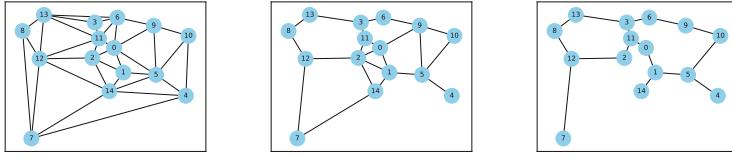
1. Input: A set of geo-tagged photos $\mathcal{I} = \{i_1, \dots, i_L\}$;
2. STEP0: Build a set of POIs by grouping photos with the same latitude and longitude $\mathcal{Q} = \{q_1, \dots, q_M\} \leftarrow \mathcal{I}$;
3. STEP1: Construct a proximity graph $G(\mathcal{V}, \mathcal{E})$;
4. STEP2: Divide into communities $G = \{G_1, \dots, G_K\}$ and assign subarea number (class label) $\mathbf{c} = [c_i]_{i \in \mathcal{I}}$;
5. STEP3: Perform multiclass classification;
 - (a) Input layer: Image vector \mathbf{y} of a photograph;
 - (b) Feature extraction layer: $\mathbf{h} \leftarrow \text{ViT}(\mathbf{y})$;
 - (c) Classification head:
 - i. Dropout layer: $\tilde{\mathbf{h}} \leftarrow \text{DropOut}(\mathbf{h}; 0.4)$;
 - ii. Fully-connected layer: $\mathbf{h}^{(1)} \leftarrow \text{ReLU}(\text{Dense}(\mathbf{h}; 768 \rightarrow 500))$;
 - iii. Dropout layer: $\tilde{\mathbf{h}}^{(1)} \leftarrow \text{DropOut}(\mathbf{h}^{(1)}; 0.3)$;
 - iv. Fully-connected layer: $\mathbf{h}^{(2)} \leftarrow \text{ReLU}(\text{Dense}(\tilde{\mathbf{h}}^{(1)}; 500 \rightarrow 200))$;
 - v. Dropout layer: $\tilde{\mathbf{h}}^{(2)} \leftarrow \text{DropOut}(\mathbf{h}^{(2)}; 0.2)$;
 - (d) Output layer: $\mathbf{p} \leftarrow \text{Softmax}(\text{Dense}(\tilde{\mathbf{h}}^{(2)}; 200 \rightarrow K))$;
 - (e) Loss: $\text{loss} \leftarrow \text{Xent}(\mathbf{p}; \mathbf{z})$;

When a user posts multiple photos from the same location, the photos will have the exact same latitude and longitude. In STEP 0, they are grouped and a set of POIs is constructed. The feature extraction layer uses the Vision Transformer [2] to obtain the feature vectors \mathbf{h} of the photos. In finetuning our model, parameters are updated to minimize the cross-entropy error with \mathbf{z}_i in order to predict the subarea number c_i (correct class label) from the feature vector \mathbf{h}_i of the photo i , where \mathbf{z}_i is a one-hot vector corresponding to the region number c_i to which it belongs.

3.1 STEP1: Proximity Graph Construction

For coordinate vectors $\mathbf{X} = [\mathbf{x}_q]_{q \in \mathcal{Q}}$ of the POIs \mathcal{Q} , the distance $d(i, j) = \sqrt{(x_{i,1} - x_{j,1})^2 + (x_{i,2} - x_{j,2})^2}$ defines the geographic neighborhood and constructs three types of proximity graphs: Delaunay Triangulation, Gabriel Graph, and Relative Neighborhood Graph.

A Delaunay Triangle graph (DT) $G^{(\text{DT})} = (\mathcal{V}, \mathcal{E}^{(\text{DT})})$ is a graph consisting of edges connecting three or more points if no other points are inside a circle inscribed in those points (see Fig. 1(a)).



(a) Delaunay Triangle (DT) (b) Gabriel Graph (GG) (c) Relative Neighborhood Graph (RNG)

Fig. 1. Proximity graph

A Gabriel graph (GG) $G^{(\text{GG})} = (\mathcal{V}, \mathcal{E}^{(\text{GG})})$ is a graph consisting of edges connecting two points v_i, v_j when no other points are inside a circle $\mathcal{S}(v_i, v_j)$ whose diameter is the line segment connecting the two points (See Fig. 1(b)):

$$\begin{aligned}\mathcal{E}^{(\text{GG})} &= \{(v_i, v_j); \mathcal{S}(v_i, v_j) \cap \mathcal{V} = \emptyset\} \\ &= \left\{ (v_i, v_j); \forall u \in \mathcal{V}: d(v_i, v_j) \leq \sqrt{d(v_i, u)^2 + d(v_j, u)^2} \right\}.\end{aligned}$$

A relative neighborhood graph (RNG) $G^{(\text{RNG})} = (\mathcal{V}, \mathcal{E}^{(\text{RNG})})$ is a graph consisting of edges connecting two points v_i, v_j when no other points are inside the overlapping portion of a circle $\mathcal{R}(v_i, v_j)$ whose radius is the line segment connecting the two points (See Fig. 1(c)):

$$\begin{aligned}\mathcal{E}^{(\text{RNG})} &= \{(v_i, v_j); \mathcal{R}(v_i, v_j) \cap \mathcal{V} = \emptyset\} \\ &= \{(v_i, v_j); \#u \in \mathcal{V}: d(v_i, v_j) \leq \max\{d(v_i, u), d(v_j, u)\}\}.\end{aligned}$$

Note that the relation $\mathcal{E}^{(\text{RNG})} \subset \mathcal{E}^{(\text{GG})} \subset \mathcal{E}^{(\text{DT})}$ holds for the edge sets of DT, GG, and RNG.

3.2 STEP2: Community Extraction

In STEP2, the proximity graph $G = (\mathcal{V}, \mathcal{E})$ constructed in STEP1 is partitioned into K densely connected subgraphs $G \rightarrow \{G_1, \dots, G_K\}$ of nodes using community extraction methods. This paper employs the Kernighan-Lin (KL) [6], Clauset-Newman-Moore (CNM) [1], and Async Fluid (AF) [8] methods, which allow the number of subareas to be specified.

For each node v , the Kernighan-Lin (KL) method calculates the difference between the number of edges $d_{\text{ext}}(v)$ entering and leaving the community to which it belongs and the number of edges $d_{\text{int}}(v)$ in the community: $D(v) = d_{\text{ext}}(v) - d_{\text{int}}(v)$. One node a from the community A is selected and one node b from the community B is selected and exchanged. The benefit (amount of change in cut size) from this exchange is calculated as $g(a, b) = D(a) + D(b) - 2 \cdot E(a, b)$, where $E(a, b)$ indicates the presence or absence of an edge between nodes a and b (1 if the edge exists, 0 otherwise). Find a pair (a, b) that maximizes $g(a, b)$, exchange them, and keep the state after the exchange. If the benefit from this

exchange is positive, it means that the cut size is reduced. This is repeated until all nodes are locked. The exchanged nodes are locked and will not be included in the exchange again.

The Clauset-Newman-Moore (CNM) method partitions the node set to maximize modularity $Q = \sum_{k=1}^K (e_{kk} - a_k^2)$ and extracts communities. Here, e_{kk} represents the ratio of the number of edges in the community k to the total number of edges on the graph, and $a_k = \sum_{h=1}^K e_{kh}$ represents the ratio of the number of edges that the nodes in the community have. ΔQ is maximized to achieve higher speed.

The Async Fluid (AF) method finds the community $G'(v)$ that maximizes the total density among node v and its neighbors $\Gamma(v)$:

$G(v) = \arg \max_{G_k \in G} \sum_{u \in \{v, \Gamma(v)\}} d(G_k) \times \delta(G(u), k)$, where $d(G_k)$ is the density of the community G_k , $G(u)$ is the community to which the node u belongs, and $\delta(G(u), k)$ is Kronecker's delta. If node v 's current community is not in $G'(v)$, randomly update it to a community in $G'(v)$. For a given number of communities k , each community is assigned to a different randomly chosen node. Each community has an initial density $d = 1.0$.

Each community extraction method assigns all nodes to one of K communities (subareas). In the proposed method, c_v is treated as the subarea number (correct label) of node v . That is, the proximity graph G can be expressed as $G = \{G_1, \dots, G_K\}$, where $G_k = (\mathcal{V}_k, \mathcal{E}_k)$ and $\mathcal{V}_k = \{v \in \mathcal{V}; c_v = k\}$.

3.3 STEP3: ViT Finetuning for Multiclass Classification

In STEP 3, the subarea numbers assigned to each node in STEP2 are used as class labels, and the Vision Transformer is finetuned through a multiclass classification task. The pretrained ViT is denoted as $\text{ViT}()$, and the finetuned ViT is denoted as $\text{ViT}^*()$. Define a classification layer that combines a fully-connected layer and a dropout layer that randomly reduces the number of features to prevent overfitting, denoted as $\text{MLP}()$. Finally, the classification probability vector \mathbf{p} is output after passing through an activation function using Softmax.

The image vectors $\mathbf{Y}^{(train)}$ of the training data are given as input, and ViT is finetuned so that $\hat{\mathbf{c}} \leftarrow \arg \max(\text{Softmax}(\text{MLP}(\text{ViT}(\mathbf{Y}^{(train)}))))$ and the correct label $\mathbf{c}^{(train)}$ match. Since the number of samples contained in the subareas obtained by graph partitioning can vary greatly, we use SMOTE (Synthetic Minority Over-sampling Technique) to oversample and eliminate imbalances between subareas. Specifically, the parameters of ViT and MLP are trained so that the cross-entropy error $\text{Xent}(\mathbf{p}, \mathbf{z})$ between the predicted probability $\mathbf{p} \leftarrow \text{Softmax}(\text{MLP}(\text{ViT}(\mathbf{Y})))$ and the one-hot vector \mathbf{z} of the correct label is minimized:

$$\text{loss} \leftarrow \sum_{i \in \mathcal{I}} \text{Xent}(\mathbf{p}_i; \mathbf{z}_i) = - \sum_{i \in \mathcal{I}} \sum_{k=1}^K z_{i,k} \log p_{i,k}.$$

Using the finetuned ViT obtained in this way, we predict the subarea labels of the test data $\mathbf{Y}^{(test)}$: $\hat{\mathbf{c}}^{(test)} \leftarrow \arg \max(\text{Softmax}(\text{MLP}^*(\text{ViT}^*(\mathbf{Y}^{(test)}))))$. Here,

MLP^{*}() means a classification head that has been trained by finetuning or transfer learning.

Table 1. Basic statistics

Target prefecture	Ibaraki	Kanagawa
#municipalities	44	58
#photos $ \mathcal{P} $	51,138	474,142
#POIs $ \mathcal{Q} $	22,363	221,805
#nodes $ \mathcal{V} $	21,760	209,257
#edges in DT $ \mathcal{E}^{\text{DT}} $	130,526	1,255,486
#edges in GG $ \mathcal{E}^{\text{GG}} $	71,270	744,164
#edges in RNG $ \mathcal{E}^{\text{RNG}} $	49,372	497,202

4 Experiments

4.1 Dataset

In this study, we utilize a dataset consisting of 2,158,026 entries collected from the photo-sharing website Flickr¹. The data covers the period from 2005 to 2019 and includes photographs posted in one metropolitan area and nine prefectures in Japan. The dataset contains information such as user names, user IDs, titles, tags, capture dates, posting locations (latitude, longitude), posting times, and the uploaded photos. We utilize the posted photos as well as the geotag information associated with them to conduct our experiments. Due to space limitations, we only show the results for Ibaraki prefecture and Kanagawa prefecture, but similar results were obtained for other regions. Table 1 shows basic statistics such as the number of photos, number of POIs, and number of nodes in the target area. The reason that the number of POIs does not match the number of nodes is because some nodes are deleted for calculation purposes when constructing the DT.

4.2 Settings

In our experiments, we consider DT, GG, and RNG as graph construction methods, KL, CNM, and AF as community extraction methods, and the number of divisions is set to $K \in \{2, 4, 8, 16, 32, 64\}$. We also compare the results with administrative divisions (Ibaraki has 44, Kanagawa has 58).

We quantitatively compare methods in terms of the size of the subareas, the features of each subarea, and the accuracy of subarea prediction. The more the

¹ <https://flickr.com>.

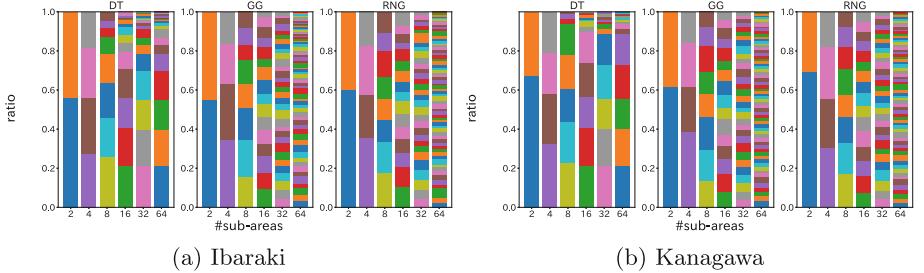


Fig. 2. Area size (#nodes) by CNM division

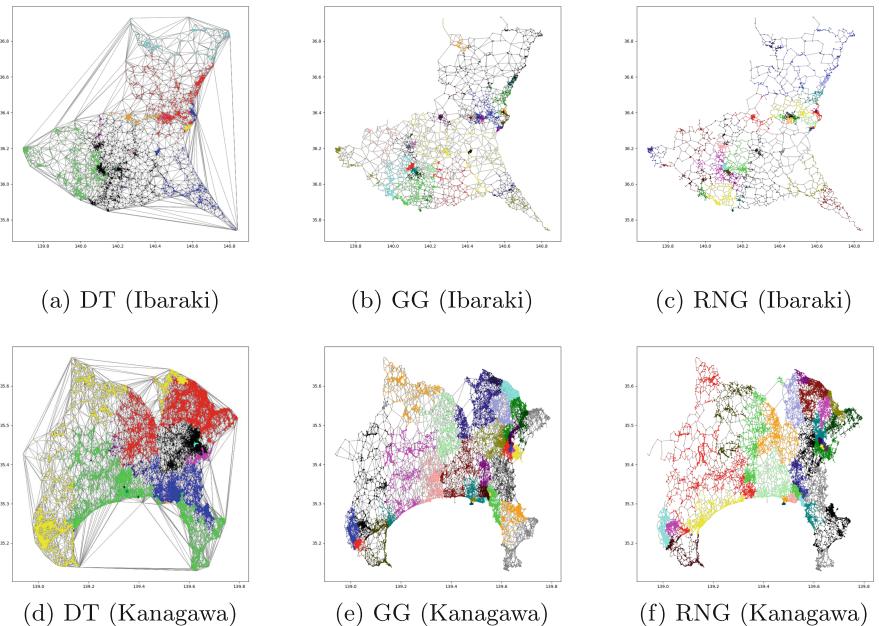


Fig. 3. CNM partitioned graph (32 partitions)

number of areas, the fewer the number of types of photos posted in each area and the higher the purity, making it possible to set area classes with semantic cohesion. On the other hand, if the number of regions is too large, a region of images with a certain characteristic will be divided into multiple regions, and regions with similar characteristics will be scattered. We therefore evaluate the features of each area and the similarity between areas when the number of areas is changed.

5 Results

5.1 Comparison of Graphs

In this subsection, we compare the differences in graph construction methods. Figure 2 shows the ratio of the number of nodes contained in each region when the CNM method is used to divide DT, GG, and RNG by $K \in \{2, 4, 8, 16, 32, 64\}$. Figure 3 shows the graph visualization results when the CNM method is used to divide DT, GG, and RNG by $K = 32$. Due to space limitations, only the results of 32 divisions using the CNM method are shown. From the results of Fig. 2 and Fig. 3, we can see that when extracting communities for DT, there is a bias in the size of the regions obtained, and even when divided finely, more than 90% of the regions are occupied by a very small number of regions. In other words, even if the number of divisions is increased, the size of each region is large, and even if it is possible to identify which region a photo belongs to, it is difficult to identify where within the region it is, so it can be said that this method is not suitable for the purposes of this study. On the other hand, it can be seen that with GG and RNG, regions of relatively equal size are obtained even when the number of divisions is increased. From these results, it can be said that the division using the CNM method for GG and RNG is suitable for the purposes of this study.

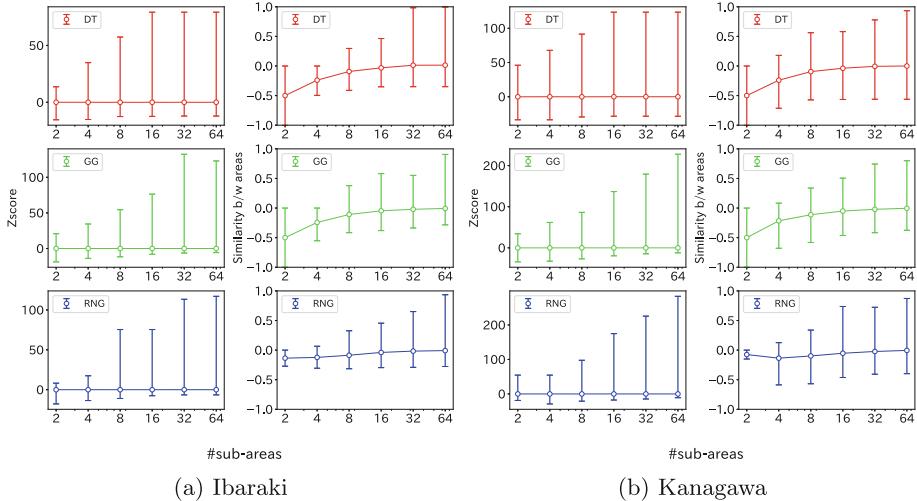


Fig. 4. Changes in Z-score and region-to-region similarity (CNM)

5.2 Comparison of Number of Subareas

In geolocalization tasks, it is necessary to have a correlation between subareas and images, that is, it is important to divide subareas so that each subarea

contains unique images. Therefore, for each divided subarea, we construct a feature vector whose elements are the uniqueness of the image features that appear, and calculate the similarity between subareas. In this paper, we use the $H = 1000$ categories of ImageNet obtained from VGG16 [11] as image features. The frequency of occurrence of feature h in community k is $f_{k,h}$, the frequency of occurrence of all features in community k is $F_k = \sum_{h=1}^H f_{k,h}$, the frequency of occurrence of feature h in all regions is $\bar{f}_h = \sum_{k=1}^K f_{k,h}$, the frequency of occurrence of all features in all regions is $F = \sum_{h=1}^H \bar{f}_h$. Assuming that feature h is uniformly and randomly distributed throughout the entire region, the expected frequency of occurrence of feature h in community k is $\hat{f}_{k,h} = \frac{F_k \cdot \bar{f}_h}{F}$ by the product of marginal distributions. If the actual value $f_{k,h}$ is significantly larger than the expected value $\hat{f}_{k,h}$ under the uniform random null hypothesis, it can be said that feature h is a unique feature that is distributed in a statistically significant manner in region k . Using the probability $\Pr(h) = \bar{f}_h/F$, the standard deviation of the multinomial distribution is $s_{k,h} = \sqrt{F_k \cdot \Pr(h) \cdot (1 - \Pr(h))}$, and using this, the Z-score of the occurrence frequency of feature h in community k is expressed as $z_{k,h} = \frac{f_{k,h} - \hat{f}_{k,h}}{s_{k,h}}$. If the Z-score $z_{k,h}$ is large, feature h is distributed in the region k with statistical significance. Conversely, if it is negative and the absolute value is large, it can be said that feature h is distributed significantly less.

Figure 4 shows the Z-score and the similarity between Z-score vectors when communities are extracted for each graph by changing the number of divisions using the CNM method. In the figure, the left side shows the Z-score, and the right side shows the similarity between the Z-score vectors, with the maximum, average, and minimum plotted with error bars. The Z-score shows a large absolute value when the feature value is significantly more or significantly less distributed in the region compared to the overall distribution. Therefore, a larger value is more likely to be obtained by increasing the number of divisions. This is because the number of photos in the region decreases and the purity increases as the region is divided into smaller regions. On the other hand, regions with similar feature vectors are scattered, and the similarity between regions increases, which hinders the calculation of specificity. In other words, it is desirable to have a high Z-score but not a high similarity between regions. From Fig. 4, it can be said that in Ibaraki Prefecture, it is desirable to have a value of about $30 \leq K \leq 50$ for GG and about $20 \leq K \leq 30$ for RNG. In Kanagawa Prefecture, it would be desirable to have around $40 \leq K \leq 50$ for GG and around $30 \leq K \leq 40$ for RNG.

5.3 Comparison of Prediction Accuracy

Table 2 shows the results of evaluating the classification accuracy for each proximity graph using the averages of four indices: accuracy (Acc), precision (Prec), recall (Rec), and F-measure (F1). PT is a model MLP*(ViT) that is created by transfer learning by adding a classification head to a pretrained ViT, and FT is a model MLP*(ViT*) that is created by finetuning ViT. The numbers in the

Table 2. classification accuracy

	DT				GG				RNG			
	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1	Acc	Prec	Rec	F1
PT02	0.67	0.67	0.67	0.67	0.69	0.69	0.69	0.69	0.67	0.67	0.67	0.67
FT02	0.96	0.96	0.96	0.96	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
PT04	0.50	0.50	0.51	0.50	0.53	0.53	0.53	0.53	0.34	0.34	0.34	0.34
FT04	0.93	0.93	0.93	0.93	0.93	0.94	0.93	0.93	0.93	0.93	0.93	0.93
PT08	0.50	0.49	0.50	0.46	0.43	0.43	0.43	0.41	0.43	0.44	0.43	0.42
FT08	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.92
PT16	0.60	0.46	0.60	0.51	0.27	0.35	0.27	0.19	0.31	0.31	0.31	0.25
FT16	0.94	0.94	0.94	0.94	0.91	0.91	0.91	0.91	0.92	0.92	0.92	0.92
PT32	0.28	0.08	0.05	0.03	0.05	0.05	0.05	0.04	0.14	0.19	0.14	0.07
FT32	0.80	0.30	0.25	0.26	0.91	0.91	0.91	0.91	0.90	0.91	0.90	0.90
PT64	0.20	0.03	0.02	0.01	0.02	0.02	0.02	0.02	0.09	0.10	0.09	0.05
FT64	0.74	0.18	0.11	0.11	0.62	0.81	0.62	0.57	0.53	0.74	0.53	0.46

first column of the table represent the number of subareas. Training was performed using 80% of the randomly selected samples, and accuracy was evaluated using the remaining 20% of the samples. Due to space limitations, we only show Ibaraki's results.

It can be seen that learning the ViT parameters through finetuning (FT) achieves higher accuracy than learning only the classification head through transfer learning (PT). In addition, finetuning is considered useful because the difference in accuracy with pretrained models is large when the number of subareas is large. In the results for administrative divisions (44 divisions), the accuracy for PT was 0.30 and the accuracy for FT was 0.33. From this, it can be said that the region segmentation by the proposed method achieves overwhelmingly higher accuracy. For DT with 32 and 64 subareas, the number of samples in some subareas was too small and SMOTE could not be performed, resulting in a high accuracy and low precision, recall, and F-measure. When comparing proximity graphs, the accuracy is higher than that of RNG, so the GG construction method is considered to be highly useful when dividing regions.

6 Conclusion

In this study, we proposed a method for the geolocalization problem of estimating where photos were taken, by creating a proximity graph from the latitude and longitude of the photo's location, extracting communities from the proximity graph, and dividing areas in order to strengthen the correlation between area classes and the photos taken and improve the performance of the extracted features. From the experiments using real dataset, we confirmed the following

observations. When extracting communities from DT, the sizes of the subareas are biased, and even when divided finely, more than 90% of the subareas are occupied by a very small number of subareas. On the other hand, as for GG and RNG, subareas of relatively equal size were obtained even when the number of divisions was increased. Finetuning is useful because the difference in accuracy with pretrained models is large when the number of region divisions is large. The proposed method achieved overwhelmingly higher accuracy in area segmentation than administrative divisions. When comparing proximity graphs, the accuracy rate is higher than that of RNG, so the GG construction method is highly useful when dividing regions.

As a future challenge, it is thought that by using image features when extracting communities, it will be possible to segment areas that have a stronger correlation with submitted photos, so one option is to expand the method to use technologies such as Graph Convolutional Neural Networks.

Acknowledgement. The third author is grateful for the financial support from JSPS Grant-in-Aid for Scientific Research (C) (No.22K12279).

References

1. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. *Phys. Rev. E* **70**, 066111 (2004)
2. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. In: International Conference on Learning Representations (2021)
3. Farmer, C.J.Q., Fotheringham, A.S.: Network-based functional regions. *J. Environ. Plann. A Econ. Space* **43**(11), 2723–2741 (2011)
4. Fushimi, T., Saito, K., Ikeda, T., Kazama, K.: Improving approximate extraction of functional similar regions from large-scale spatial networks based on greedy selection of representative nodes of different areas. *Appl. Netw. Sci.* **3**(1), 1–14 (2018). <https://doi.org/10.1007/s41109-018-0075-2>
5. Haas, L., Alberti, S., Skretta, M.: Learning generalized zero-shot learners for open-domain image geolocalization (2023)
6. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. *Bell Syst. Tech. J.* **49**, 291–307 (1970)
7. Newman, M.E.J.: Detecting Community Structure in Networks. *Eur. Phys. J. B - Condensed Matter Complex Syst.* **38**(2), 321–330 (2004). <https://doi.org/10.1140/epjb/e2004-00124-y>
8. Parés, F., et al.: Fluid communities: a competitive, scalable and diverse community detection algorithm. In: Cherifi, C., Cherifi, H., Karsai, M., Musolesi, M. (eds.) *COMPLEX NETWORKS 2017* 2017. SCI, vol. 689, pp. 229–240. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-72150-7_19
9. Rosvall, M., Bergstrom, C.T.: Mapping change in large networks. *PLoS ONE* **5**(1), e8694 (2010)
10. Seo, P.H., Weyand, T., Sim, J., Han, B.: CPlaNet: enhancing image geolocalization by combinatorial partitioning of maps. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) *ECCV 2018*. LNCS, vol. 11214, pp. 544–560. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01249-6_33

11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: Bengio, Y., LeCun, Y. (ed.) Proceedings of the 3rd International Conference on Learning Representations (ICLR2015) (2015)
12. Weyand, T., Kostrikov, I., Philbin, J.: PlaNet - Photo geolocation with convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 37–55. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_3
13. Yin, J., Soliman, A., Yin, D., Wang, S.: Depicting urban boundaries from a mobility network of spatial interactions: a case study of great britain with geo-located twitter data. *Int. J. Geogr. Inf. Sci.* **31**, 1293–1313 (2017)
14. Yu, W., Chen, J., Wei, C.: A hierarchical learning model for inferring the labels of points of interest with unbalanced data distribution. *Int. J. Appl. Earth Observ. Geoinf.* **108**, 102751 (2022)
15. Zhang, Y., Wang, X., Zeng, P., Chen, X.: Centrality characteristics of road network patterns of traffic analysis zones. *Transp. Res. Record J. Transp. Res. Board* **2256**, 16–24 (2011)



Harmony in Diagnosis: Exploring Consensus and Variability in Clinical Judgement

Sanad Satel¹(✉), George Kour², Eyal Zinger¹, Yoav Ganzach¹,
and Sarel Cohen¹

¹ School of Computer Science, The Academic College of Tel Aviv-Yaffo,
Tel Aviv, Israel

{sanadst,eyalzi,yoavg2,sarelco}@mta.ac.il
² IBM Research, Haifa, Israel
gkour@ibm.com

Abstract. This research explores the variability in decision-making among clinical psychologists by applying machine learning techniques to reverse-engineer their judgments. Using a dataset of 861 patients evaluated by 29 judges (both experienced psychologists and trainees), we trained decision tree and linear regression models for each judge to capture their decision patterns. We employed methods such as Jaccard similarity and pairwise Mean Squared Error (MSE) to quantify the distance between judges' models, providing a clear metric of variability. Additionally, we applied SMOTE for data augmentation to enhance model training and improve robustness in comparisons. To gain deeper insights into the decision-making process, we used Large Language Models (LLMs) to explain and compare individual and pairwise models, highlighting key differences in clinical judgment. Our findings reveal significant variability in how judges weigh psychological traits, offering valuable insights into the potential for improving consistency and understanding judgment discrepancies in clinical practice. You can also find the related code in the following [GitHub Repository](#).

Keywords: medical decision making · medical ML · relations between judges

1 Introduction

Clinical judgment is important in diagnosing and treating mental health disorders, but variability among expert decisions remains a concern. Even experienced psychologists can provide differing diagnoses for the same patient, leading to inconsistent treatment paths. Understanding the causes of this variability is crucial for improving diagnostic reliability.

Meehl's dataset [4], a key resource for studying clinical judgment variability, contains evaluations from multiple judges, yet the underlying decision-making processes remain unclear. Recent advances in machine learning offer new methods for analyzing this variability by modeling and comparing judges' decisions.

This research applies machine learning techniques, including decision trees and linear regression, to quantify judgment variability among clinical psychologists. By measuring differences using different metrics, and leveraging Large Language Models (LLMs) for model explainability, we aim to better understand and explain the key factors driving judgment variability.

Our findings contribute to the growing field of machine learning in mental health by offering insights into judgment consistency and proposing methods to improve clinical decision-making.

2 Related Work

The study of clinical judgment variability, particularly in mental health diagnoses, has its roots in Paul Meehl's work. Meehl [4] demonstrated that statistical models often outperform human clinicians in making consistent diagnoses, using a dataset that continues to provide insights into judgment patterns and variability.

Building on Meehl's findings, Yoav Ganzach [3] showed that non-linear models, like decision trees, better capture clinicians' complex decision-making processes than linear models. His research supports our use of decision tree models in this study. Additionally, the growing role of machine learning in clinical decision-making has been recognized for its ability to process complex data and improve consistency, as discussed by Adlung et al. [1]. However, integrating machine learning into clinical practice remains a challenge, particularly for high-level decisions.

Recent work by Miller et al. [5] emphasized the importance of explainability in machine learning models for healthcare. Our use of Large Language Models (LLMs) to explain judgment variability aligns with this focus on transparency. By combining decision trees, linear regression, and LLM-based explanations, we aim to provide a deeper understanding of variability among clinical psychologists, enhancing both accuracy and interpretability.

3 The Dataset

The dataset used in this research comes from Paul Meehl's studies on clinical judgment. It includes evaluations of 861 patients diagnosed as either neurotic or psychotic based on the Minnesota Multiphasic Personality Inventory (MMPI). The dataset captures patients' scores on eight clinical and three validity scales, with judgments rated from 0 (psychotic) to 11 (neurotic).

Judgments were provided by 29 clinical experts, including 13 psychologists and 16 trainees. The dataset has two parts:

MEELMMPI: Features representing the 11 psychological traits assessed by the MMPI. MEELJUD: Target variables representing each expert's judgment for each patient. The traits include lie, eccentricity, defensiveness, hypochondriasis, depression, hysteria, psychopathic deviate, paranoia, psychasthenia, schizophrenia, and hypomania. These characteristics are essential for evaluating personality.

4 Methodology

4.1 Experimental Overview and Key Findings

To explore the decision-making patterns among clinical judges, we employed various machine learning techniques. This section details the approaches used, including model training, cross-judge predictions, and the assessment of feature importance. By applying these methods, we aimed to quantify the similarities and differences in judgments and identify key factors influencing decision accuracy.

4.2 Estimate the Distance Between Judges

We employed several approaches to measure the distance between the judges' decision-making processes. Initially, we estimated the distance directly from the dataset. Next, we trained a decision tree model for each judge and measured the distance between the models using Jaccard similarity. Finally, we trained both decision tree and linear regression models for each judge, applied SMOTE for data augmentation, and calculated the pairwise Mean Squared Error (MSE) between the predictions of these models on the augmented dataset.

4.3 Explain Models and the Differences Between Models Using LLM

We further analyze the decision-making processes by leveraging Large Language Models (LLMs):

- **Model Explanation:** Train a decision tree model for each judge and use an LLM to explain the decision-making process of each model, focusing on individual judges.
- **Comparative Analysis:** Provide the LLM with pairs of models (representing two judges) and request an identification and explanation of the main differences in their decision-making processes.
- **Textual Descriptions:** Describe the models textually, allowing the LLM to articulate the decision-making patterns and variations between judges.

This approach provides a qualitative understanding of the similarities and differences in clinical judgments among the judges, enhancing the interpretability of the machine learning models.

5 Research Outline

The research conducted so far has provided important insights into the decision-making processes of clinical judges.

5.1 Experimental Overview and Key Findings

In this subsection, we provide a summary of the experiments conducted to analyze the decision-making processes of clinical judges. These experiments include model training, cross-judge predictions, and an analysis of feature importance, all aimed at understanding the variability and consistency in judgments. Below, we outline the key experiments and their findings, providing insights into the effectiveness of different machine learning models and the influence of patient characteristics on decision outcomes.

5.1.1 Single Judge Prediction

We trained models (Random Forest, Logistic Regression, Linear Regression, XGBoost) to predict a single judge's decisions. This experiment was aimed at understanding how well models could replicate individual judgment patterns. The results, presented in Table 1, show that non-linear models such as Random Forest and XGBoost performed better than linear models, reflecting the complexity of clinical decision-making.

5.1.2 Cross-Judge Prediction

We trained models on one judge's decisions to predict another's. This cross-prediction analysis allowed us to study the correlation between judges' decision patterns. We observed variability in prediction accuracy, indicating significant differences in judgment approaches. The results of the correlations are presented in heatmap Fig. 1.

5.1.3 Aggregated Decision Models

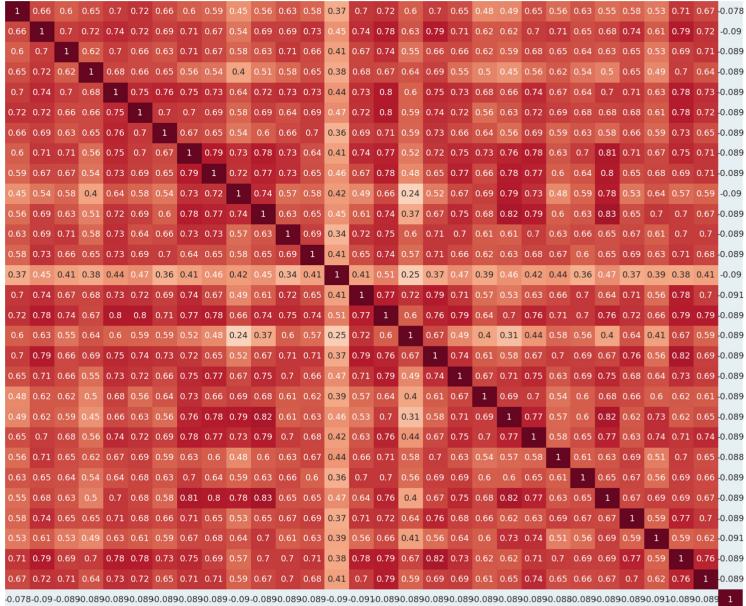
We trained models on the average and median decisions of the judges to predict test set outcomes. This experiment sought to capture the consensus judgment across multiple judges. Boxplot Fig. 2 illustrates the variability in prediction performance, with the median decision providing more consistent results.

5.1.4 Feature Importance

We analyzed the influence of patient characteristics on decision outcomes, using SHAP (SHapley Additive exPlanations) to quantify feature importance. SHAP allowed us to identify which features contributed the most to the model's predictions, providing a clear understanding of how each characteristic influenced the decisions. The SHAP values calculated across the test dataset quantify how much each feature influences the model's predictions on average. The higher the SHAP value for a feature, the more important it is in the decision-making process. Features like schizophrenia showed the highest influence on decisions across judges, as demonstrated in Figs. 3 and 4. This analysis, enhanced by SHAP, provided insights into which traits judges prioritize in their assessments.

Table 1. A model trained on a single judge, predicting the judge's own decisions on the test set. A warmup to comparing between judge's models.

Model	Train MSE	Test MSE	Accuracy	Precision	Recall	F1
Random Forest	0.117733	0.699422	0.300578	0.284878	0.300578	0.283050
Logistic Regression	0.569767	0.682081	0.317919	0.315165	0.317919	0.303902
Linear Regression	1.212758	1.153373	0.724793	NaN	NaN	NaN
XGBoost	0.187500	0.687861	0.312139	0.306110	0.312139	0.298034

**Fig. 1.** Correlation between the judges presented as a heatmap.

5.2 Estimate the Distance Between Judges

Measuring how judges make different decisions is important for finding inconsistencies and improving their overall decision quality. By understanding these differences, we can identify biases, assess the effectiveness of training programs, and enhance teamwork in decision-making. Analyzing judges' decisions using models helps us see patterns in their judgment and find areas that need improvement.

5.2.1 Naive Distance Estimation

Initially, we estimated the distance directly from the dataset. The results are visualized in a heatmap (Fig. 5), where we analyzed the raw differences between judgments.

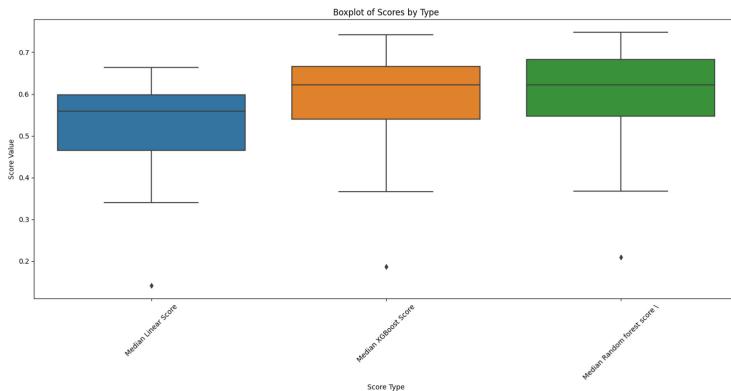


Fig. 2. Boxplot of score distributions for different methods across judges, showing variability and central tendencies.

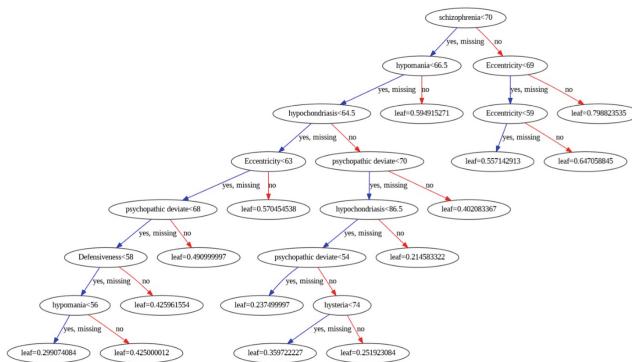


Fig. 3. Feature importance, highlighting schizophrenia as the most influential.

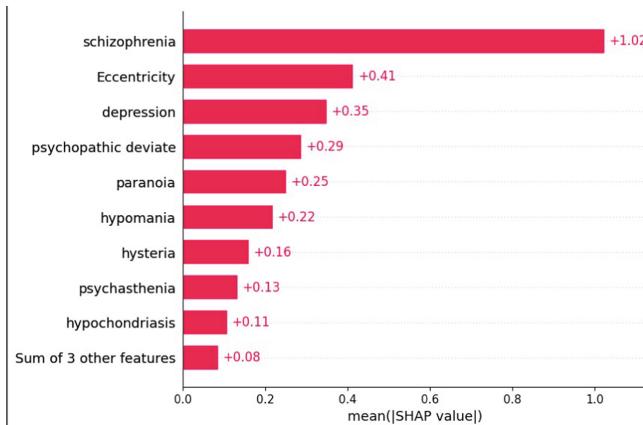


Fig. 4. Results showing schizophrenia with the highest mean feature value.

5.2.2 Decision Tree Model Distance

In our analysis, we measured the distance between the decision-making processes of different judges using the Jaccard similarity index. Specifically, we trained a decision tree model for each judge and then compared the sets of features used by each tree. The Jaccard similarity index, calculated as the ratio of the intersection over the union of the feature sets, provided a measure of similarity between the trees. The tree distance was then defined as $1 - \text{jaccard_similarity}$, with lower distances indicating more similar decision-making processes between judges. Figure 6 visualizes the dissimilarity matrix, showing how judges vary in their use of patient features.

5.2.3 Distance Estimation on Models with SMOTE

To address potential class imbalances, we applied SMOTE augmentation to the dataset and recalculated the distances. This allowed us to improve model robustness and better reflect judgment similarities and differences.

This approach was implemented using decision tree and linear regression models. The resulting distances are shown in heatmaps for decision trees (Fig. 7) and linear regression models (Fig. 8).

Data Augmentation and Model Training

For each judge $j \in \{1, 2, \dots, 29\}$, we trained a decision tree classifier f_j on their respective datasets $\mathcal{D}_j = \{(\mathbf{x}_i, y_{ij})\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^d$ represents the feature vector of the i -th patient, and $y_{ij} \in \{1, 2, \dots, 11\}$ denotes the decision made by judge j for patient i .

To ensure robust evaluation, we employed a data augmentation method called *Minority Over-sampling Technique* [2] (SMOTE) to generate augmented datasets $\mathcal{D}_j^{\text{aug}} = \{(\mathbf{x}_{jk}^{\text{aug}}, y_{jk}^{\text{aug}})\}_{k=1}^M$, with $M = 5500$. Separate augmented datasets were created for each judge to preserve the integrity of the decision-making patterns.

Distance Calculation

The distance between the decision processes of two judges j and j' was quantified by the Mean Squared Error (MSE) between their predictions on the augmented datasets:

$$\text{MSE}(j, j') = \frac{1}{M} \sum_{k=1}^{2M} (\hat{y}_{kj}^{\text{aug}} - \hat{y}_{kj'}^{\text{aug}})^2$$

where $\hat{y}_{kj}^{\text{aug}}$ and $\hat{y}_{kj'}^{\text{aug}}$ are the predicted decisions for the k -th patient in the augmented dataset by judges j and j' , respectively.

A symmetric 29×29 distance matrix \mathbf{D} was constructed, with each element $D_{jj'}$ representing the MSE between the predictions of judges j and j' :

$$\mathbf{D} = [D_{jj'}] = [\text{MSE}(j, j')]$$

Results and Discussion

The pairwise MSE heatmaps (Figs. 7 and 8) reveal the extent of similarity or dissimilarity in the decision processes of different judges. Lower MSE values indicate higher similarity in decision-making, while higher values suggest greater dissimilarity.

These visualizations provide valuable insights into the consistency and variability of medical decisions across different practitioners. For instance, judges with similar decision-making patterns cluster together, highlighting potential opportunities for peer learning and standardization of best practices.

A notable observation is the consistently high MSE values associated with **Judge 14**. In the linear regression model (Fig. 8), **Judge 14** shows the highest MSE of 4.22 when compared to Judge 17, indicating a significant dissimilarity in decision-making. Similarly, in the decision tree model (Fig. 7), **Judge 14** again shows a relatively high MSE of 3.31 when compared to Judge 17. This suggests that Judge 14's decision-making process is markedly different from that of their peers across both models. Such findings highlight the need for further investigation into the underlying reasons for this variability, which may point to differences in judgment criteria, experience, or other factors influencing medical decision-making.

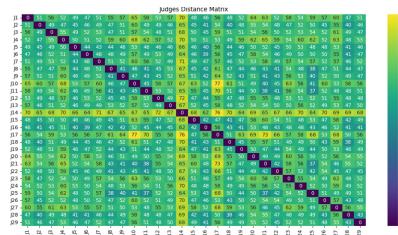


Fig. 5. Euclidean distance of the dataset between the judges

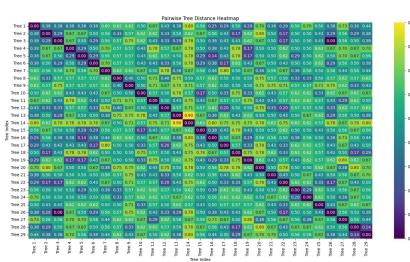


Fig. 6. Heatmap of distances between the judges

5.3 Explainable AI

- **Comparison Using LLMs:** Expand the use of Large Language Models (LLMs) to provide more detailed comparative analyses between judges' decision-making models, with a focus on identifying underlying patterns and differences.
- **Model Evaluation and Interpretation:** Further develop the models for each judge and enhance the interpretability of these models through detailed LLM-based explanations, aiming to make the decision-making processes more transparent and understandable.
- **Application to Clinical Practice:** Consider how these findings can be applied to real-world clinical settings, improving the consistency and reliability of medical judgments through the integration of machine learning tools.

5.3.1 Example of Comparative Analysis

- Decision Tree Comparisons

To illustrate the differences in decision-making processes, we analyzed the decision trees of two judges. Each model aims to classify a patient's condition (Psychotic/Neurotic) using the features from the MMPI dataset.

- Non-Technical Description

****First Judge:****

- **Main Focus:** This judge starts by looking at how defensive the patient is. Then, they pay close attention to traits like hysteria, paranoia, and depression. They go through a more complex evaluation, considering a wide range of symptoms to make their decision.
- **Approach:** This judge's approach is thorough, involving many different angles and more in-depth analysis of each symptom.

****Second Judge:****

- **Main Focus:** Similarly, this judge also begins by evaluating defensiveness but quickly moves to consider schizophrenia and psychopathic traits. They have a more structured approach, focusing on specific symptoms like hysteria and hypomania.

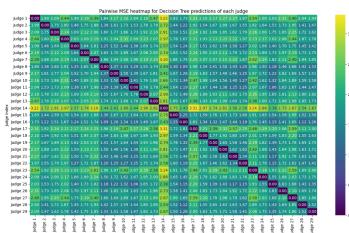


Fig. 7. MSE heatmap for predictions of each decision tree model

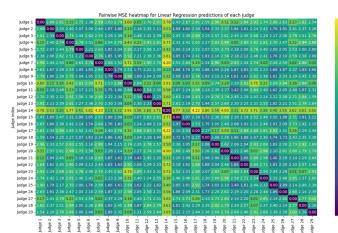


Fig. 8. MSE heatmap for predictions of each linear regression model

- **Approach:** This judge's method is more straightforward and balanced. They rely on a consistent set of symptoms to form their diagnosis.
Differences Between the Two Judges:
- **First Judge:** Uses a broader variety of symptoms and follows a more intricate path in their diagnostic process, reflecting a detailed and perhaps more cautious approach.
- **Second Judge:** Uses a more consistent and straightforward set of symptoms to make their diagnosis, indicating a balanced and possibly more efficient approach.

– **Conclusion: Who is the More Experienced Judge?**

Based on the differences in the decision trees, the more experienced judge is likely represented by the first tree. Here's why:

- **Complexity and Depth:** The first tree shows a more intricate and varied path in the decision-making process, suggesting that the judge is considering a wider range of symptoms and is taking a more nuanced approach to the diagnosis. An experienced judge is likely to recognize and evaluate subtle patterns and interrelations among different features (symptoms).
- **Thorough Evaluation:** The first tree's thoroughness and attention to multiple features (Defensiveness, Hysteria, Paranoia, Depression, Hypomania) suggest a deep understanding of the condition. An experienced judge is likely to delve deeper into various aspects of the patient's condition, reflecting their comprehensive knowledge and careful consideration.

In contrast, the second tree, while efficient and balanced, indicates a more straightforward approach, which could be indicative of a judge who relies on well-established patterns and might prioritize efficiency over depth.

Thus, the judge represented by the first tree likely has more experience, as they demonstrate a deeper and more nuanced approach to diagnosing the condition.

6 Conclusion

This study explored variability in clinical decision-making among judges using machine learning models, yielding several important insights:

6.1 Key Findings

Non-linear models, such as Random Forest and XGBoost, more effectively captured individual judgment complexities compared to linear models. Cross-judge predictions showed significant variability in judgment approaches, highlighting clusters of similar decision patterns, while Judge 14 stood out as an outlier.

6.2 Variability in Clinical Judgments

Variability stemmed from differing interpretations of patient features, experience levels, and biases. Experienced judges tended to apply more nuanced evaluations, whereas less experienced judges used more straightforward approaches.

6.3 Role of Explainability

Using Large Language Models (LLMs) provided transparent explanations of decision-making patterns, offering valuable insights into how expertise and judgment style contribute to variability.

6.4 Implications for Practice

Understanding judgment variability can inform improvements in consistency, training, and standardization in clinical practice. Future research could automate these analyses for scalability.

6.5 Future Work

Future research should focus on automating analysis, applying this methodology to broader datasets, conducting longitudinal studies, integrating additional machine learning models, and providing feedback mechanisms to support clinician learning.

References

1. Adlung, L., Cohen, Y., Mor, U., Elinav, E.: Machine learning in clinical decision making. *Front. Med.* **8** (2022). <https://doi.org/10.3389/fmed.2021.765693>
2. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002)
3. Ganzach, Y.: Nonlinear models of clinical judgment: communal nonlinearity and nonlinear accuracy. *Psychol. Sci.* **12**(5), 403–407 (2001). PMID: 11554674. arXiv:<https://doi.org/10.1111/1467-9280.00374>
4. Meehl, P.E.: Clinical versus statistical prediction : a theoretical analysis and a review of the evidence. University of Minnesota Press, Minneapolis. <https://doi.org/10.1037/11281-000>
5. Miller, T., et al. Explanation in artificial intelligence: insights from the social sciences. *Artif. Intell.* **267**, 1–38 (2017). <https://doi.org/10.1016/j.artint.2018.07.007>

Information Spreading in Social Media



Characterize Fake News Diffusion Patterns Using Temporal Logic Rules

Valeria Fionda^(✉)

Department of Mathematics and Computer Science, via Pietro Bucci 30B,
87036 Rende, Italy
valeria.fionda@unical.it
<https://www.mat.unical.it/fionda/>

Abstract. The use of social media pervades everyday life making them powerful tools for conveying and disseminating information. Their pervasive use has raised significant concerns about the spread of disinformation, which can undermine public trust and destabilize societies. Accurately identifying fake news remains a critical challenge. This paper introduces a novel framework based on a variant of Linear Temporal Logic (LTL) designed to characterize the diffusion patterns of both fake and true news within disinformation networks. The framework uses logic formula templates to formally describe the temporal propagation behaviors that differentiate misinformation from authentic information. This framework lays the groundwork for future experimental studies to validate the proposed logical models in practical scenarios.

Keywords: Disinformation Networks · Fake News Detection · Temporal Logic · News Diffusion Analysis

1 Introduction

The rapid growth of social media has transformed global information dissemination and consumption. As of January 2024, there were about 5.04 billion social media users, representing 62.3% of the global population. User numbers grew by 266 million in the past year, an annual rate of 5.6% or 8.4 new users per second. According to GWI¹, the average user engages with 6.7 social platforms monthly and spends 2 h and 23 min per day on social media, amounting to about 15% of their waking hours.

Thus, social media platforms like Twitter and Facebook wield considerable influence in disseminating information, but with this power comes a growing concern about the spread of misleading and harmful content, commonly referred to as disinformation [1]. Disinformation, defined as false or misleading information spread deliberately to deceive, can have profound impacts on civil society. It can manipulate public opinion, influence political outcomes, and even incite violence,

¹ <https://www.gwi.com>.

as evidenced by incidents like the Pizzagate conspiracy and the surge of misleading information during the COVID-19 pandemic [1]. This is why detecting fake news quickly and accurately has become a critical challenge.

Current approaches to tackling disinformation often rely on artificial intelligence and deep learning techniques to detect fake news by analyzing the semantic content of news stories [2, 3, 18]. However, these methods have limitations, as they fail to account for the unique diffusion dynamics of fake news compared to true news. Fake news often spread more rapidly and widely, exploiting the social networks' structure and users' tendencies to share sensational content [28]. Therefore, understanding and modeling the diffusion patterns of disinformation is crucial for developing more effective detection and mitigation strategies. This is why some recent proposals have started to investigate the differential diffusion of true and false news stories by analyzing propagation networks [21, 23].

This paper takes an initial step towards defining a new logic-based framework for analyzing the diffusion of true and fake news in dynamic disinformation networks. By using a variation of Linear Temporal Logic (LTL) [7, 8, 10], the framework models the propagation patterns of news stories over time. This approach allows for a more detailed understanding of how disinformation spreads and offers strategies for countering it. The proposed framework aims to analyze the diffusion of true and fake news stories and automatically generate temporal logic formulas that describe their unique propagation patterns. These formulas capture the distinct characteristics that differentiate the spread of true news from fake news. By analyzing positive (true news) and negative (fake news) examples, the framework infer temporal logical models that accurately describe the diffusion mechanisms of each news category. These models can then be applied to new (partial) news propagation data to assess the likelihood of a story being true or false, enhancing early detection capabilities in disinformation networks.

The paper is organized as follows. Section 2 reviews related work. Section 3 outlines the structural, temporal, and engagement dynamics distinguishing fake from true news. Section 4 formalizes the problem, and Sect. 5 introduces News-Diffusion Temporal Logic (NDTL), detailing its syntax and semantics. Section 6 describes the overall framework and Sect. 7 concludes the paper.

2 Related Work

The spread of social media has raised concerns about their use in large-scale disinformation campaigns, promoting the dissemination of misleading and harmful information. This has led to new challenges in detecting and countering hostile influence operations. Numerous approaches have been proposed for fake news detection, leveraging features such as content, writing style, propagation patterns, and source credibility (see [31] for a recent survey). These methods use techniques including natural language processing [18], sentiment analysis [2, 3], deep learning [13–16, 19, 21, 24–26, 29], and graph mining [9]. Recent classifications categorize these techniques into content-based, context-based, and hybrid approaches [1], with our work aligning closely with context-based methods.

Context-based studies have focused on analyzing the differential diffusion of true and false news by examining propagation networks. Vosoughi et al. [28] explored the diffusion dynamics of news from temporal and structural perspectives, while Shu et al. [23] extended this analysis using hierarchical models of diffusion networks. Liu and Wu [16] encoded propagation paths as multivariate time series, and Wu and Liu [29] developed models that classify messages by simulating their diffusion through networks. Other hybrid approaches, such as those by Hamdi et al. [13] and Jiang et al. [15], combine user features with graph embeddings to improve fake news detection. Some recent works model evolving networks using graph snapshots or dynamic diffusion networks [25, 26], while others like Paraschiv et al. [19] and Salamanos et al. [21] use meta-graph structures and hypergraph convolution layers to represent complex relationships within social networks. Unlike deep learning methods, our approach focuses on a declarative framework using linear temporal logic (LTL), which characterizes diffusion networks through interpretable logical rules, offering explanations for why news is classified as fake, enhancing transparency.

Several logical frameworks have been developed for modeling information diffusion in social networks [4, 12, 17, 20]. Christoff and Hansen [4] extended modal logics with dynamic components to model information spread, and Grandi et al. [12] used LTL to model opinion diffusion with strategic agent behavior. Machado and Braga [17] introduced LTL-SN for evolving social networks, and Prandi and Pavese [20] proposed a logic for modeling paranoid agents in disinformation spread. None of the discussed approaches specifically model diffusion patterns in social media to distinguish between fake and true news by utilizing positive and negative examples of information diffusion.

Various extensions of LTL, including CTL [6], TTL [27], and GTL [30], have been developed to handle complex data structures. In particular, GTL [30] extends LTL for spatial-temporal properties of dynamic graphs, but unlike our approach, it focuses on fixed structures with dynamic labels, while our framework applies temporal logic to static graphs with temporal information.

3 Characterizing Fake News Diffusion

The diffusion of fake news on social media shows distinct characteristics compared to true news, particularly in structural features, user engagement, and temporal dynamics. Structurally, fake news networks are generally deeper and wider, featuring longer retweet chains and involving more users, which indicates broader reach [28]. These networks often exhibit higher maximum outdegree, meaning that tweets within fake news networks are reshared more frequently. Fake news also generates a larger number of cascades, leading to multiple simultaneous diffusion paths [28].

Regarding user engagement, fake news tends to generate more intense but short-lived interactions, with higher engagement levels in conversation threads that diminish rapidly as the news loses relevance [5]. Fake news also triggers strong emotional responses, such as anger, fear, and skepticism, which further

fuel its rapid spread since emotionally charged content is more likely to be widely shared [28].

Temporal dynamics, which are most relevant to this paper, highlight several key patterns. Fake news exhibits a notably faster initial spread compared to true news, with shorter times between the first tweet and the most influential retweet [28]. Despite this rapid onset, fake news has a shorter overall lifespan on social media, indicating that it fades out quickly as network activity subsides [28]. Additionally, fake news often spread in short, intense bursts driven by sudden spikes in engagement or coordinated bot activity [5]. Finally, fake news reaches its maximum impact-measured by peak retweet counts or maximum outdegree-much faster than true news, underscoring its ability to gain quick, but brief, visibility [28].

4 Problem Definition

A diffusion network for news stories on social media can be modeled as an attributed graph, where nodes represent events such as tweets, retweets, and replies, and directed edges represent the actions or relationships connecting these events, like retweeting or replying. In this paper, we focus on macro-level propagation networks that include news nodes, tweet nodes, and retweet nodes, while excluding detailed conversational trees of replies. The graph incorporates temporal information, as each event occurs at a specific point in time. The source tweet (news node) is assigned a time of 0, and each edge is labeled with the time elapsed between the connected events. Formally, a news diffusion graph is defined as $G = (N, E, \lambda, t, \delta, \sigma)$ where:

- N is the set of nodes representing news posts, tweets, and retweets.
- $E \subseteq N \times N$ is the set of directed edges between nodes.
- $\lambda : N \rightarrow \{\text{news, tweet, retweet}\}$ assigns to each node its event type (news post, tweet, or retweet).
- $t : N \rightarrow \mathbb{R}$ assigns to each node n the time at which the event occurs.
- $\delta : N \rightarrow \mathbb{R}$ assigns to each node n the time elapsed since the previous event n_s , where $(n_s, n) \in E$.
- $\sigma : N \rightarrow \mathbb{R}$ assigns to each node n the time elapsed since the originating event in the cascade (i.e., the news post for tweet nodes and the original tweet for retweet nodes). For a node n_j , if the originating event is node n_0 , then $\sigma(n) = \sum_{i=1}^j \delta(n_i)$, where n_i represents each event on the path connecting n_0 to n_j .

Example 1. Fig. 1 illustrates a propagation graph of a news story, showing the hierarchical structure of news diffusion. The initial **news** node (green) starts the process at $t = 2024-08-25 14:45$ with $\delta = 0$ and $\sigma = 0$. The red **tweet** node appears at $t = 2024-08-26 08:12$, with $\delta = 0d17h27m$ and $\sigma = 0d17h27m$, indicating the time since the initial news post. The first orange **retweet** node occurs at $t = 2024-08-26 14:49$, with $\delta = 0d6h37m$ and $\sigma = 0d6h37m$. The second orange **retweet** node appears at $t = 2024-08-28 14:51$, with $\delta = 2d0h2m$ and cumulative $\sigma = 2d6h39m = 2d0h2m + 0d6h37m$.

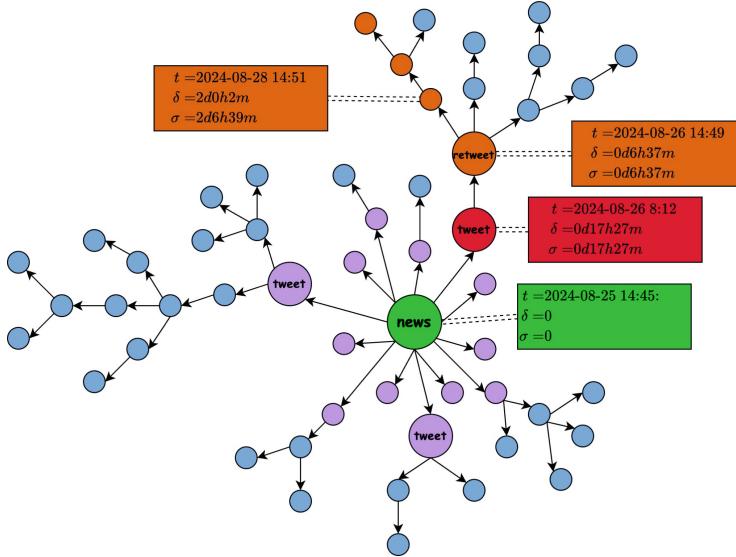


Fig. 1. An example of the propagation graph of a news story containing three types of nodes (news, tweet, retweet). Each node has associated the attributes related to the time at which the event happened (t), the time elapsed between the event and its parent (δ), and the time elapsed between the event and the originating event in the cascade (σ).

Let A be a set of news stories, each labeled by a partial function $f : A \rightarrow \{0, 1\}$ that indicates its truthfulness: $f(a) = 0$ if the news story a is true, and $f(a) = 1$ if a is fake. The *propagation graph* of a news story a is denoted by G_a . A collection of news stories is represented by a corresponding set of propagation graphs, referred to as a *propagation forest*. The objective is to develop a model \hat{f} that predicts the label of a given news story a_i using its (partial) propagation graph G_{a_i} , such that $\hat{f}(a_i) = f(G_{a_i})$.

5 News Diffusion Temporal Logic

In this paper, we introduce News-Diffusion Temporal Logic (NDTL), a temporal logic inspired by Linear Temporal Logic (LTL) [8, 10], tailored for analyzing the temporal dynamics of information diffusion in social media. Unlike traditional LTL, which focuses on event ordering in linear sequences, NDTL extends this to propagation graphs, enabling the measurement and comparison of time intervals between events.

Given a propagation graph $G = (N, E, \lambda, t, \delta, \sigma)$, an event condition is a predicate that checks if a node's label matches a type in $\{\text{post}, \text{tweet}, \text{retweet}\}$. This Boolean function $N \rightarrow \{0, 1\}$ returns 1 if the node's label corresponds to the specified event type. Specifically, G satisfies an event condition π at node v ,

denoted $(G, v) \models \pi$, if $\lambda(v) = \pi$. For a graph G and a node $v_0 \in N$, the neighbor operation $\Gamma^i : V \rightarrow 2^V$ is defined as:

$$\Gamma^i(v_0) = \{v_i \mid \exists v_1, \dots, v_{i-1} \in V, \text{ s.t. } (v_j, v_{j+1}) \in E \text{ for } j \in \{0, \dots, i-1\}\}$$

Intuitively, $\Gamma^i(v_0)$ consists of nodes that can be reached from v_0 through paths of length i . If i is omitted, it defaults to $i = 1$. The special value $i = \infty$ indicates nodes in the farthest neighborhood of v_0 .

An NDTL formula φ is constructed according to the following grammar:

$$\varphi ::= \pi \mid \rho \otimes \epsilon \mid (\rho + \rho) \otimes \epsilon \mid (\rho - \rho) \otimes \epsilon \mid \neg \varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathbf{X} \varphi \mid \mathbf{G} \varphi \mid \mathbf{F} \varphi$$

$$\rho ::= \max_{\oplus}(\Gamma^i \varphi) \mid \min_{\oplus}(\Gamma^i \varphi) \mid \text{avg}_{\oplus}(\Gamma^i \varphi)$$

where i is a positive integer, ϵ is a real value, $\pi \in \{\text{post, tweet, retweet}\}$, $\otimes \in \{<, >, \leq, \geq\}$, and $\oplus \in \{\delta, \sigma, t\}$. The symbols \neg, \wedge, \vee represent Boolean negation, conjunction, and disjunction, respectively. Temporal operators include \mathbf{X} (*next*), \mathbf{F} (*eventually*), and \mathbf{G} (*always*), which express temporal relationships within the news diffusion graph. Specifically, \mathbf{X} asserts that the formula holds in the next step, \mathbf{F} indicates that it will eventually hold at some point in the future, and \mathbf{G} requires that the condition holds at all future points in the propagation.

The satisfaction relation $(G, v) \models \varphi$ for a propagation graph G at node v with respect to an NDTL formula φ is recursively defined as follows:

$$\begin{aligned} (G, v) \models \pi &\iff \lambda(v) = \pi \\ (G, v) \models \rho \otimes \epsilon &\iff \rho(v) \otimes \epsilon \\ (G, v) \models (\rho_1 + \rho_2) \otimes \epsilon &\iff (\rho_1(v) + \rho_2) \otimes \epsilon \\ (G, v) \models (\rho_1 - \rho_2) \otimes \epsilon &\iff (\rho_1(v) - \rho_2) \otimes \epsilon \\ (G, v) \models \neg \varphi &\iff (g, v) \not\models \varphi \\ (G, v) \models \varphi_1 \wedge \varphi_2 &\iff (G, v) \models \varphi_1 \text{ and } (g, v) \models \varphi_2 \\ (G, v) \models \varphi_1 \vee \varphi_2 &\iff (G, v) \models \varphi_1 \text{ or } (g, v) \models \varphi_2 \\ (G, v) \models \mathbf{X} \varphi &\iff (G, v') \models \varphi, \text{ for all } v' \in \Gamma(v) \\ (G, v) \models \mathbf{G} \varphi &\iff (G, v') \models \varphi, \text{ for all } v' \in \{v\} \cup V' \text{ s.t. } V' = \bigcup_{i=1}^{\infty} \Gamma^i(v) \\ (G, v) \models \mathbf{F} \varphi &\iff \exists v_1, \dots, v_m \text{ s.t. } v_i \in \Gamma^\infty(v), v_{i+1} \in \Gamma(v_i) \forall i \in \{1, \dots, m-1\} \text{ and } \Gamma(v_m) = \emptyset \text{ such that } (G, v_i) \models \varphi \text{ for all } i \in \{1, \dots, m\} \end{aligned}$$

The semantics of the function ρ is defined as follows:

$$\max_{\oplus}(\Gamma^i \varphi)(v) = \max_{v' \in \Gamma^i(v)} \max_{(G, v_i) \models \varphi} (\oplus(v'))$$

$$\min_{\oplus}(\Gamma^i \varphi)(v) = \min_{v' \in \Gamma^i(v)} \min_{(G, v_i) \models \varphi} (\oplus(v'))$$

$$\text{avg}_{\oplus}(\Gamma^i \varphi)(v) = \text{avg}_{v' \in \Gamma^i(v)} \max_{(G, v_i) \models \varphi} (\oplus(v'))$$

Whenever $(G, v_0) \models \varphi$ holds, then we say that G is a *model* of φ and we just write $G \models \varphi$.

Example 2. Consider the NDTL formula $\varphi = \mathbf{F}(\text{tweet} \rightarrow \mathbf{X}(\max_{\sigma} \Gamma^\infty \leq 2d))$. Intuitively, this formula states that there exists a tweet node such that, for every retweet originating from it, the longest chain of retweets terminates within 2 d.

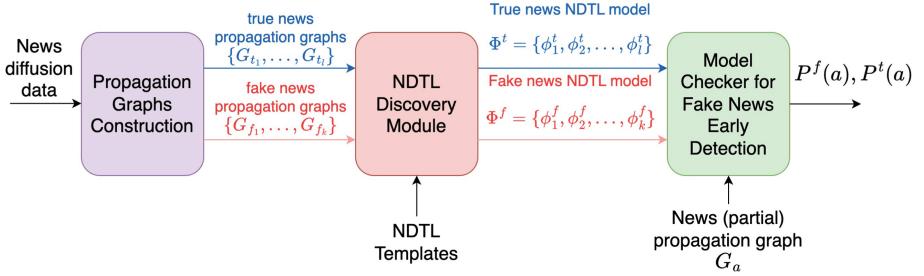


Fig. 2. Proposed fake news detection framework. News diffusion data is used to construct propagation graphs, which are classified into true (G_t) and fake (G_f) news graphs. The NDTL Discovery Module generates models (Φ^t and Φ^f), which are used by the Model Checker to evaluate partial propagation graphs (G_a) and determine the likelihood of news being fake ($P^f(a)$) or true ($P^t(a)$).

Applying this formula to the graph in Fig. 1, we find that the propagation graph does not satisfy the condition. This is because there exists at least one tweet (the red node) with a retweet chain (the orange path) that exceeds the specified time bound on σ .

6 Framework

The proposed fake news detection framework is composed of three key components: propagation graph construction, NDTL discovery module, model checker for early fake news detection. As discussed in Sect. 3, certain diffusion characteristics - such as rapid spread, short lifespan, bursty propagation, and quick time to maximum impact - can distinguish fake news from true news. In this section, we leverage NDTL to encode and identify these distinctive diffusion patterns, aiming to improve the early detection of fake news circulating within networks. To achieve this, we define NDTL templates that will be instantiated using propagation logs of fake news (positive examples) and true news (negative examples) in the NDTL discovery module. These instantiated formulas will then be applied to new (partial) propagation graphs in the model checker module to assess their compliance, providing a probabilistic estimation of whether the propagating news is fake or real (Fig. 2).

6.1 Propagation Graphs Construction

Given a news story propagating on social media, we begin by constructing its propagation graph. Fake news data repositories, such as those described in Shu et al. [22], provide news data that includes both the content of the news and associated temporal information. News content comprises the body text and social context, such as user interactions (e.g., posting, sharing, or commenting

on news on Twitter). Temporal information captures the timestamps of these user engagements.

To create the propagation graph, we add a news node for the original news post and a node for each tweet and retweet associated with it. Edges are then created from the news node to each tweet node. Additionally, an edge is created from node u to node v when a tweet u is retweeted by a user x , resulting in the creation of node v . On Twitter, both tweets and retweets can be retweeted. However, retweet data does not specify whether the retweeted content is an original tweet or another retweet. To address this ambiguity, the social network of users, as suggested by Goel et al. [11] can be used. This approach infers the source of a retweet by analyzing the user’s friends who have previously retweeted the content. If the timestamp of a user’s retweet is later than that of a friend’s retweet, it is likely that the user saw and retweeted the content from that friend. If no immediate friend’s retweet is found, it is assumed that the retweet originated from the original tweet rather than another retweet.

6.2 NDTL Discovery Module

The NDTL Discovery Module is an essential component of our fake news detection framework, designed to capture and formalize the distinct temporal patterns associated with the diffusion of fake news. By leveraging News-Diffusion Temporal Logic (NDTL), this module systematically identifies and encodes some temporal characteristics that differentiate fake news from true news. The process starts with the analysis of propagation graphs of both fake and true news stories. These graphs serve as the basis for constructing NDTL formulas that capture the distinctive diffusion dynamics of each news type. By instantiating predefined NDTL templates with real-world data, the module generates a set of logical rules that can be applied to new, unseen propagation graphs. The primary aim of the NDTL Discovery Module is to build a comprehensive repository of temporal logic formulas that can effectively distinguish between fake and true news. Table 1 summarizes several key templates that can be used for news categorization, providing their corresponding formulas, and their descriptions. The discovery process is inherently flexible, allowing for the addition of new templates and the re-execution of the module to continuously expand the formula repository. The instantiation of an NDTL template corresponds to determining the values of the time parameters (in the table the values of t_1, \dots, t_6) in the associated formulas. These parameters are crucial as they define the specific temporal thresholds that distinguish the diffusion patterns of fake news from those of true news. For each template, instantiating the time parameters involves analyzing the available propagation data from both fake and true news stories and setting the thresholds accordingly. This process customizes the template to accurately capture the temporal dynamics observed in real-world scenarios. Moreover, a model for characterizing fake news (and similarly for true news) is constructed as a set of instantiated templates that are considered to hold simultaneously. This is represented by a large conjunction (*AND*) among the formulas, meaning that all the instantiated templates must be satisfied at the same time. Thus, the

model comprehensively captures the distinctive diffusion patterns by combining multiple temporal criteria.

Table 1. NDTL Templates, Formulas, and Their Descriptions

Formula	Description
$G(\text{retweet} \rightarrow (\text{avg}_\delta(\Gamma(\text{retweet})) \otimes t_1))$	Average time interval between consecutive retweets. It captures the temporal spread rate of retweets, which is often shorter in fake news.
$G(\text{tweet} \rightarrow F(\text{retweet} \rightarrow \text{avg}_\delta(\Gamma(\text{retweet})) \otimes t_2))$	Average time between a tweet and its first subsequent retweet. Fake news often shows a rapid engagement pattern, reflected by a shorter time to the first retweet compared to true news.
$G(\text{post} \rightarrow (\max_t(\Gamma^\infty(\text{retweet})) - \min_t(\Gamma(\text{tweet}))) \otimes t_3)$	Time elapsed between the first tweet and the last retweet. Fake news typically has a shorter overall lifespan.
$G(\text{post} \rightarrow (\max_t(\Gamma(\text{tweet})) - \min_t(\Gamma(\text{tweet}))) \otimes t_4)$	Time span between the earliest and latest tweets. Fake news often shows a more concentrated burst of activity, resulting in a shorter time window.
$G(\text{post} \rightarrow (\text{avg}_\sigma(\Gamma(\text{tweet})) \otimes t_5))$	Average time at which tweets appear. It helps assess the regularity of tweet activity, which tends to be more irregular for fake news.
$G(\text{tweet} \rightarrow (\min_\delta(\Gamma(\text{retweet})) \otimes t_6))$	Time difference between when tweet is posted and when first retweet occurs. Shorter intervals indicate rapid diffusion characterizing fake news

Example 3. Consider Template T1, which measures the average time difference between adjacent retweet nodes. In particular, the set of templates contains it in two forms: $\otimes = \leq$ corresponding to $G(\text{retweet} \rightarrow (\text{avg}_\delta(\Gamma(\text{retweet})) \leq t_h))$, and $\otimes = \geq$ corresponding to $G(\text{retweet} \rightarrow (\text{avg}_\delta(\Gamma(\text{retweet})) \geq t_l))$. Instantiating these templates involves setting the parameters t_l and t_h such that fake news model holds for fake propagation graphs but not for true ones (and vice versa). For instance, if fake news retweets average 10 min apart and true news 20 min, the module identifies the models $G(\text{retweet} \rightarrow (\text{avg}_\delta(\Gamma(\text{retweet})) \leq 10))$ for fake news and $G(\text{retweet} \rightarrow (\text{avg}_\delta(\Gamma(\text{retweet})) \geq 20))$ for true news. The template

with $t_h = 10$ captures the rapid spread of fake news, while $t_l = 20$ reflects the slower spread of true news.

6.3 Model Checker for Fake News Early Detection

The Model Checker Module leverages the temporal formulas discovered by the NDTL Discovery Module to assess the likelihood that a given news story is fake. This module operates by systematically evaluating the (partial) propagation behavior of news stories against the predefined temporal patterns encoded in the NDTL formulas. The primary objective is to determine how closely the observed propagation behavior of a news item aligns with typical diffusion patterns associated with fake news, thereby providing an early indication of its authenticity. The evaluation process begins with a partial propagation graph G_a of a news story a , which represents the observed diffusion up to a given time point. The Model Checker uses the two sets of NDTL formulas $\Phi^f = \{\phi_1^f, \phi_2^f, \dots, \phi_k^f\}$ and $\Phi^t = \{\phi_1^t, \phi_2^t, \dots, \phi_l^t\}$, previously identified as distinctive markers of fake and true news respectively, to perform a logical evaluation. Recall that each formula in Φ^f (resp. Φ^t), encapsulates a specific temporal diffusion pattern which are characteristic of fake news and not of true news (resp., of true news and not of fake news). The Model Checker checks whether G_a satisfies each formula ϕ_j using a temporal logic satisfaction relation $(G_a, v) \models \phi_j$, where v is the news post node of G_a . The outcome of this evaluation is binary for each formula: either the propagation graph complies with the formula ($\models \phi_j$) or it does not ($\not\models \phi_j$). To quantify the likelihood that the news story a is fake (resp, true), the Model Checker aggregates the results of the individual formula evaluations. This is achieved by calculating two probability scores $P^f(a)$ and $P^t(a)$, reflecting the proportion of formulas that are satisfied by the observed propagation behavior. The scores are computed as follows:

$$P^\circ(a) = \frac{\sum_{\phi \in \Phi^\circ} I((G_a, v) \models \phi)}{|\Phi^\circ|} \text{ with } \circ = \{f, t\}$$

where $I(\cdot)$ is an indicator function that returns 1 if the condition is true (i.e., the formula is satisfied) and 0 otherwise. A high probability score $P^f(a)$ and a low probability score $P^t(a)$ indicates a closer alignment of the propagation behavior with patterns typical of fake news. On the other hand, a high probability score $P^t(a)$ and a low probability score $P^f(a)$ indicates a closer alignment of the propagation behavior with patterns typical of true news. In particular, given two thresholds τ^h and τ^l , if $P^f(a) \geq \tau^h$ and $P^t(a) \leq \tau^l$, the Model Checker flags the news story as likely to be fake, allowing for early intervention measures, such as alerting fact-checkers or limiting the spread of the content on the platform. This threshold-based decision-making provides a flexible mechanism for controlling the sensitivity of the detection system. Furthermore, the Model Checker can continuously update its evaluations as more propagation data becomes available, refining the probability scores $P^f(a)$ and $P^t(a)$ in real-time.

7 Concluding Remarks

In this paper, we proposed a framework using a variant of Linear Temporal Logic (LTL), called News-Diffusion Temporal Logic (NDTL), to characterize the diffusion patterns of true and fake news in disinformation networks. The approach models news propagation dynamics with temporal logic formulas that capture the distinct behaviors of each category and provides a declarative and interpretable method for fake news detection. This framework lays the groundwork for using temporal logic in disinformation analysis.

Future work will involve validating the framework with real-world data, refining logical templates, and incorporating additional contextual factors like network structure. Overall, this research opens new avenues for leveraging formal methods in the fight against disinformation.

Acknowledgments. This work was supported by the MUR Project HypeKG within PRIN 2022 Program (CUP: H53D23003710006); by European Union - Next Generation EU through the MUR PRIN 2022-PNRR project DISTORT (CUP: H53D23008170001) under the Italian PNRR Mission 4 Component 1; by the PNRR project FAIR - Future AI Research (PE00000013), Spoke 9 - Green-aware AI, under the NRRP MUR program funded by the NextGenerationEU.

References

1. Aïmour, E., Amri, S., Brassard, G.: Fake news, disinformation and misinformation in social media: a review. *Soc. Netw. Anal. Min.* **13**(1), 30 (2023)
2. Balshetwar, S.V., RS, A., R, D.J.: Fake news detection in social media based on sentiment analysis using classifier techniques. *Multim. Tools Appl.* **82**(23), 35,781–35,811 (2023)
3. Bhutani, B., Rastogi, N., Sehgal, P., Purwar, A.: Fake news detection using sentiment analysis. In: IC3, pp. 1–5. IEEE (2019)
4. Christoff, Z., Hansen, J.U.: A logic for diffusion in social networks. *J. Appl. Log.* **13**(1), 48–77 (2015)
5. Del Vicario, M., et al.: The spreading of misinformation online. *Proc. Natl. Acad. Sci.* **113**(3), 554–559 (2016)
6. Emerson, E.A., Clarke, E.M.: Using branching time temporal logic to synthesize synchronization skeletons. *Sci. Comput. Program.* **2**(3), 241–266 (1982)
7. Fionda, V., Greco, G.: The complexity of LTL on finite traces: Hard and easy fragments. In: AAAI, pp. 971–977 (2016)
8. Fionda, V., Greco, G.: LTL on finite and process traces: complexity results and a practical reasoner. *J. Artif. Intell. Res.* **63**, 557–623 (2018)
9. Fionda, V., Pirrò, G.: Fact checking via evidence patterns. In: IJCAI, pp. 3755–3761. ijcai.org (2018)
10. Giacomo, G.D., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: IJCAI, pp. 854–860. IJCAI/AAAI (2013)
11. Goel, S., Anderson, A., Hofman, J.M., Watts, D.J.: The structural virality of online diffusion. *Manag. Sci.* **62**(1), 180–196 (2016)

12. Grandi, U., Lorini, E., Novaro, A., Perrussel, L.: Strategic disclosure of opinions on a social network. In: AAMAS, pp. 1196–1204. ACM (2017)
13. Hamdi, T., Slimi, H., Bounhas, I., Slimani, Y.: A hybrid approach for fake news detection in twitter based on user features and graph embedding. In: Hung, D.V., D’Souza, M. (eds.) ICDCIT 2020. LNCS, vol. 11969, pp. 266–280. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-36987-3_17
14. Hu, L., Wei, S., Zhao, Z., Wu, B.: Deep learning for fake news detection: a comprehensive survey. *AI Open* **3**, 133–155 (2022)
15. Jiang, S., Chen, X., Zhang, L., Chen, S., Liu, H.: User-characteristic enhanced model for fake news detection in social media. In: Tang, J., Kan, M.-Y., Zhao, D., Li, S., Zan, H. (eds.) NLPCC 2019. LNCS (LNAI), vol. 11838, pp. 634–646. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-32233-5_49
16. Liu, Y., Wu, Y.B.: Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In: AAAI, pp. 354–361. AAAI Press (2018)
17. Machado, V., Benevides, M.R.F.: Temporal logic for social networks. *J. Log. Comput.* **32**(6), 1088–1108 (2022)
18. Oshikawa, R., Qian, J., Wang, W.Y.: A survey on natural language processing for fake news detection. In: LREC, pp. 6086–6093. European Language Resources Association (2020)
19. Paraschiv, M., Salamanos, N., Iordanou, C., Laoutaris, N., Sirivianos, M.: A unified graph-based approach to disinformation detection using contextual and semantic relations. In: ICWSM, pp. 747–758. AAAI Press (2022)
20. Prandi, L., Primiero, G.: A logic for biased information diffusion by paranoid agents in social networks. *J. Log. Comput.* **32**(6), 1292–1315 (2022)
21. Salamanos, N., Leonidou, P., Laoutaris, N., Sirivianos, M., Aspri, M., Paraschiv, M.: HyperGraphDis: leveraging hypergraphs for contextual and social-based disinformation detection. In: ICWSM, pp. 1381–1394. AAAI Press (2024)
22. Shu, K., Mahudeswaran, D., Wang, S., Lee, D., Liu, H.: FakeNewsnet: a data repository with news content, social context and dynamic information for studying fake news on social media. arXiv preprint [arXiv:1809.01286](https://arxiv.org/abs/1809.01286) (2018)
23. Shu, K., Mahudeswaran, D., Wang, S., Liu, H.: Hierarchical propagation networks for fake news detection: investigation and exploitation. In: ICWSM, pp. 626–637. AAAI Press (2020)
24. Silva, A., Han, Y., Luo, L., Karunasekera, S., Leckie, C.: Propagation2Vec: embedding partial propagation networks for explainable fake news early detection. *Inf. Process. Manag.* **58**(5), 102618 (2021)
25. Song, C., Shu, K., Wu, B.: Temporally evolving graph neural network for fake news detection. *Inf. Process. Manag.* **58**(6), 102712 (2021)
26. Song, C., Teng, Y., Zhu, Y., Wei, S., Wu, B.: Dynamic graph neural network for fake news detection. *Neurocomputing* **505**, 362–374 (2022)
27. Vardi, M.Y.: A temporal fixpoint calculus. In: POPL, pp. 250–259. ACM Press (1988)
28. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. *Science* **359**(6380), 1146–1151 (2018)
29. Wu, L., Liu, H.: Tracing fake-news footprints: Characterizing social media messages by how they propagate. In: WSDM, pp. 637–645. ACM (2018)
30. Xu, Z., Nettekoven, A.J., Julius, A.A., Topcu, U.: Graph temporal logic inference for classification and identification. In: CDC, pp. 4761–4768. IEEE (2019)
31. Zhou, X., Zafarani, R.: A survey of fake news: fundamental theories, detection methods, and opportunities. *ACM Comput. Surv.* **53**(5), 109:1–109:40 (2021)



Tracking Narrative Dynamics Using Churn Management on Social Networks

Ahmed Al-Taweel¹, Nitin Agarwal^{1,2(✉)}, and Abiodun Quadri¹

¹ COSMOS Research Center, University of Arkansas - Little Rock, Arkansas, USA
{asaltaweel,adquadri}@ualr.edu

² International Computer Science Institute, University of California, Berkeley, USA
nxagarwal@ualr.edu

Abstract. Churn analysis studies user growth or shrinkage using behavioral features. This study uses churn analysis to examine pro-Russian and pro-Ukraine Telegram communities' dynamics during the Russia-Ukraine conflict. The study answers two questions - (1) can churn management strategies help track campaigns/social movements based on user engagement trends? (2) can churn analysis help evaluate the effect of the growth or shrinkage of users on narrative propagation? The pro-Russian and pro-Ukraine narratives were generated from the Telegram channels to predict churners. We observed a significant change in the churn rate of pro-Russian narratives, while there was a notable fluctuation in user engagement with the pro-Ukraine narratives. Consequently, pro-Ukrainian narratives propagated faster than pro-Russian narratives, as demonstrated by the epidemiological model assessing narrative spread. This suggests a dynamic landscape in the dissemination and reception of these competing narratives.

Keyword: churn analysis, social media, Telegram, user engagement, campaign, social movement, narrative analysis

1 Introduction

Social media platforms have become key spaces for opinion formation, especially during conflicts like the Russia-Ukraine war, where pro-Russian and pro-Ukrainian factions use Telegram to spread their narratives[1,2]. This study introduces a unique and novel approach using strategic churn management traditionally used in subscription services to predict customer attrition-to analyze user engagement and narrative effectiveness in these Telegram channels. By examining user behavior, transitions between narratives, and engagement ‘burstiness’ around significant events, we assess how these narratives retain or lose followers. Metrics like Daily Active Users (DAU) help quantify narrative stickiness, while high churn rates may indicate a decline in credibility or emotional appeal[3].

Our research uniquely focuses on the patterns of narrative distribution within and across Telegram channels, examining how stories start, evolve, and spread

as they move between groups [4]. This aligns with the “pinball” model of brand narratives, where interactions continuously shape the stories[5]. We explore “narrative defection,” or the frequent switching of users between opposing groups, offering insights into the persuasiveness of each narrative and the dynamics of opinion formation during conflicts. Our study addresses two key questions: RQ1 Can churn management strategies track social movements based on user engagement? RQ2: Can churn analysis help evaluate the effect of users’ growth or shrinkage on narrative propagation?

In conclusion, we introduce a novel approach to analysing narrative conflicts on social media by applying churn management concepts to Telegram data from the Russia-Ukraine war. This methodology sheds light on how public perceptions are shaped in modern conflicts and provides predictive tools for understanding narrative resilience and shifts. Our findings are valuable for policymakers, media analysts, and others seeking to grasp the dynamics of public opinion in a hyper-connected world. The paper is structured as follows: Sect. 2 reviews related literature, Sect. 3 details our methodology, Sect. 4 presents the mathematical framework for churn analysis, Sect. 5 discusses the experimental results, and the final section concludes with discussions and future directions.

2 Literature Review

Churn, a term used to describe the loss of customers, is a significant concern in various industries, including telecommunications[6], banking [7], Internet service providers [8], and online games [9]. It measures risk and uncertainty, with annual churn rates ranging from 0.2-0.3. Acquiring new customers is often more expensive than retaining existing ones [10], and new customers are often less profitable [11]. The 2009 Vodafone annual report highlighted the importance of addressing customer churn, as it could significantly affect profitability. Recent studies [12] have explored the correlation between a user’s likelihood to churn and the number of their neighbours who have already churned, proposing diffusion models to explain how churn spreads among users. Network effects are critical factors in churn within social networks, and churn analysis has been widely used to predict customer retention and loss.

Churn prediction using clustering identifies churn users and groups them based on their online activities [13]. This approach involves two main steps: prediction and clustering. Targeted retention solutions are implemented to reduce churn rates by analyzing churn users. Machine learning techniques have improved the accuracy of these predictions. In our study, we propose combining churn analysis with narrative tracking to monitor user engagement and message activity on the Telegram platform for the Russia-Ukraine conflict.

3 Methodology

In this study, we investigate the concept of user attrition on social media platforms, where users engage in one-on-one interactions. We provide an overview

of the churn model, beginning with a description of our dataset and our approach to defining churn analysis in the context of social media. Next, we delve into the emergence of churn users within our model and the extraction of user information. Finally, we offer a detailed explanation of the model.

Data Collection: The study analyzed influential Russian political channels from TGStat, focusing on those with over 10,000 subscribers. A native Russian-speaking coder categorized these channels as Pro-Kremlin, Anti-Kremlin, Neutral, or Others based on their content. A non-Russian-speaking coder validated these labels using Telegram's translation feature, and a political science expert reviewed a subset of 100 channels. Cohen's kappa score was used to measure inter-rater agreement, achieving a high score of 0.97. The dataset included 404 Pro-Kremlin and 115 Anti-Kremlin channels, covering posts from December 21, 2020, to April 30, 2023. The top 5 channels were selected as the most influential from both channels. The pro-Russian narrative, "Special Military Operation," described the invasion as a mission to "demilitarize" and "denazify" Ukraine, while the pro-Ukrainian narrative, "Heroic Defense," emphasized Ukraine's resistance to rally domestic and international support.

User Churn: Predicting churn on social media platforms is challenging due to the need for a subscription model and the skewed data distribution. Users can cease engagement without notice, making a data-driven approach essential. The first week or two are crucial for predicting churners, with the first seven active days serving as observation and the subsequent seven days as prediction. The research uses an equation to compute user churn rate over time. In this research, we use the following equation for computing user churn rate over time Δt :

$$CR_{\Delta t}(t_i) = \frac{C_{\Delta t}}{U_{t_1}} \times 100\% \quad (1)$$

where CR is the user churn rate to t_1 , C is the number of churners who stop activity during Δt , U is the total number of users at the initial time t_1 and $\Delta t = \{t_i : t_i \text{ is in day, week, month, year}\}$. The daily percentages of non-active users for 60 d are presented in Fig. 1 utilizing Eq. 1. On a daily basis, the percentage of non-active users ranges from 95 percent to 99.99 percent most of the time. However, the rate of non-active users decreases from 75 percent to 80.3 percent between 2/26/2022 and 3/8/2022. The fraction of users who had no posts completed is around ten percentage points higher than those who had no events recorded for the first two weeks of the prediction period. However, this difference slightly decreases in the third week. This fluctuation in churn rates highlights potential periods of reduced user engagement, which could be critical for targeted intervention strategies.

Remark 1. To ensure sufficient data, we included users who engaged for at least one day during their first seven days. Users not engaging in the prediction period were labeled as churning; those who did were labeled as non-churners (active).

Epidemiological Model: We employed the SEIZ models to analyze the narrative spread on Telegram about the Russia-Ukraine Dispute, and we consider

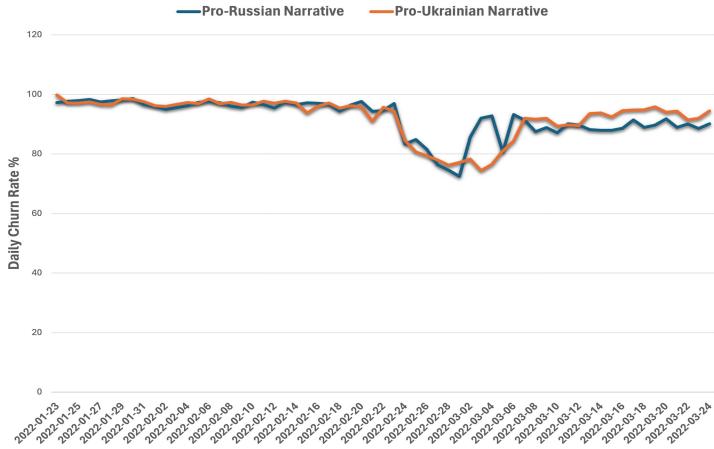


Fig. 1. Daily percentages of inactive users (churners) monitored using event and engagement approaches over a 60 d period.

fractional derivatives and integrals, followed by model comparisons. To better depict online narrative dynamics, we use the SEIZ model, which includes Skeptic and Exposed components and is shown to be better at modeling spread of narratives compared to other epidemiological models [15]. The SEIZ model comprises four compartments: Susceptible, Exposed, Infected, and Skeptic. When a Susceptible channel (S) encounters a narrative, it can quickly become an Infected channel (I) with probability p . Alternatively, it may analyze the narrative and move to Exposed (E) status with probability $(1-p)$.

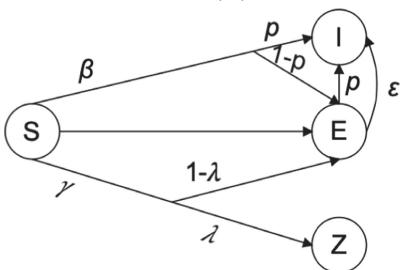


Fig. 2. SEIZ compartment model.

As depicted in Fig. 2, the SEIZ model rules can be summarized as follows:

$$\left\{ \begin{array}{l} \frac{dS}{dt} = -\frac{\beta SI}{N} - \frac{\gamma SZ}{N}, \\ \frac{dE}{dt} = \frac{(1-p)\beta SI}{N} + \frac{(1-\lambda)\gamma SZ}{N} - \frac{\eta EI}{N} - \epsilon E, \\ \frac{dI}{dt} = \frac{p\beta SI}{N} + \frac{\eta EI}{N} + \epsilon E, \\ \frac{dZ}{dt} = \frac{\lambda\gamma SZ}{N}. \end{array} \right. \quad (2)$$

To apply SEIZ models from equations (2), key parameters like contact rates (β , γ , η) must be defined, along with initial values for $S(t_0)$, $E(t_0)$, $I(t_0)$, and $Z(t_0)$. The implementation was done in Python using `scipy.optimize` for least squares fitting and `odeint` for solving ODEs. We used Nelder-Mead optimization within `scipy.optimize.minimize` to ensure parameter convergence by minimizing the difference between the Infected compartment (I) and corresponding Telegram data, $|I(t) - posts(t)|$. We tracked weekly cumulative post counts to set optimization boundaries.

4 Churn Model Proposed for Narrative Analysis

In this section, we present a mathematical model to analyze the narrative spread on Telegram during the Russia-Ukraine conflict, focusing on various components that contribute to understanding user retention and growth dynamics. We introduce the definitions and lemmas necessary for proposing the user churn technique that helps explain the suggested models.

Churn Definitions for Social Media Platform: In this section, we will first explore concepts related to the temporal aspects of user data. Let's consider x_{nt} as a D-dimensional vector representing the activity $n = 1, \dots, N$ at discrete time steps $t = t_1, \dots, t_{\beta+\gamma}$, where time intervals are evenly spaced. Now, we will divide these time intervals into several distinct and non-overlapping periods: an observation period spanning from $t = t_1, \dots, t_\beta$, which will be utilized for observing user behavior, and a prediction period from $t = t_{\beta+1}, \dots, t_{\beta+\gamma}$, designed for evaluating whether a user has churned or not. Here, β and γ denote the respective sizes of these periods in a week. Additionally, we introduce a third period called the activity period. This period, ranging from $t = t_{\beta-\eta}, \dots, t_\beta$, overlaps with the last η time steps of the observation period. The activity period is a crucial step in our analysis. This period overlaps with the last η time steps of the observation period and plays a significant role in our efforts to include users who churned a substantial amount of time ago in our predictive modeling, as illustrated in Fig. 3.

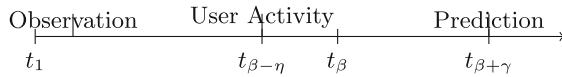


Fig. 3. Time period window.

Tracking user engagement is crucial to assessing an audience's interaction on social media platforms or in any movement. In other words, measuring engagement dynamicity across various periods (daily, weekly, and monthly) is essential to gain valuable insights. Different user behaviors in activity or engagement can lead to different churn outcomes.

Definition 1. *The Weekly Active User (WAU) metric monitors the count of users who engage with the initiative during a given t [13].*

Definition 2. *The Weekly Activity Churn (WAC) measures recurring user engagement that changes from active to inactive due to stop in engagement, such as a desire for variety and freshness, and measures the analysis of post performance within a specified period.*

$$WAC_{t_i, t_j}(x, y) = \begin{cases} y_n x_i, & \text{if } x_j = 0 \text{ and } x_i > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

Definition 3. The **Shrink metric**, denoted as S_{t_i, t_j} , is a parameter for analyzing user engagement and identifying instances of user attrition during a specific time frame, such as a week [19].

Note 1. If $x_i, x_j > 0$ and $x_i > x_j$ for $j > i$, the Expansion (E_{t_i, t_j}) measure yields $y_n(x_i - x_j)$. This metric offers insights into how users' engagement levels evolve, aiding in understanding the growth dynamics of online communities.

Definition 4. A Gross Activity (GA) is the proportion of ongoing user engagement in activities that an initiative has lost due to churners cutting their involvement in a given period, defined as follows:

$$GA_{t_i} = BOP_{t_i} - \sum_{i=1}^n \sum_{j=i+1}^n (WAC_{t_i, t_j} + S_{t_i, t_j}), \quad (4)$$

where the Beginning of Period (BOP_{t_i}) refers to the point in time that marks the start of a movement.

Definition 5. A Net Activity Churn (NAC): Instead of only considering the percentage of recurring activity an expedition lost from churners t_i , this metric considers attrition (expansion) and growth activities, defined as follows.

$$NAC_{t_i} = GA_{t_i} + \sum_{i=1}^n \sum_{j=i+1}^n E_{t_i, t_j}. \quad (5)$$

Definition 6. The End of Period (EOP) activity metric represents the total activity recovery at the end of the analysis period t_i , accounting for all changes in user engagement defined below.

$$EOP_{t_i} = NAC_{t_i} + \sum_{i=1}^n \sum_{j=i+1}^n NWAC_{t_i, t_j}, \quad (6)$$

where EOP_{t_i} activity represents the collective activity observed over the analysis duration. Figure 4 presents a mathematical model delineating user churn within a social media platform, utilizing data sourced from Telegram. The findings of this investigation propose a correlation between users' motivations for utilizing social media and their perceptions of promotional causes.

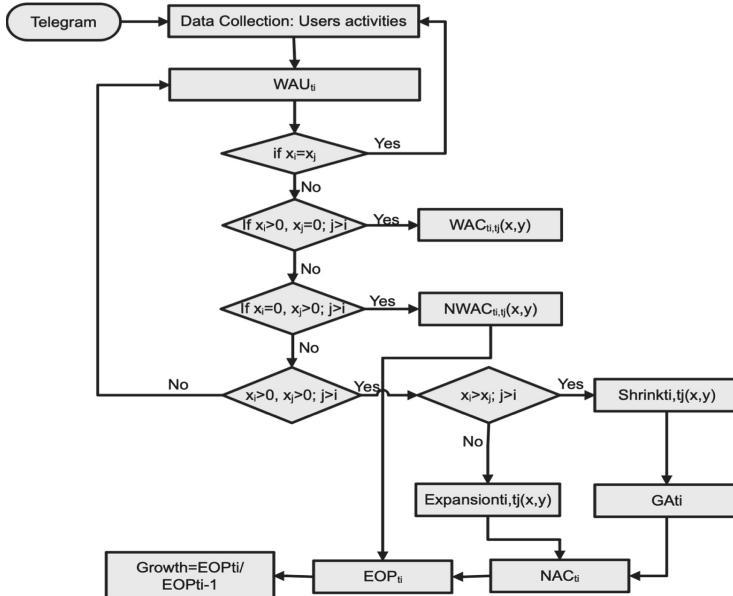


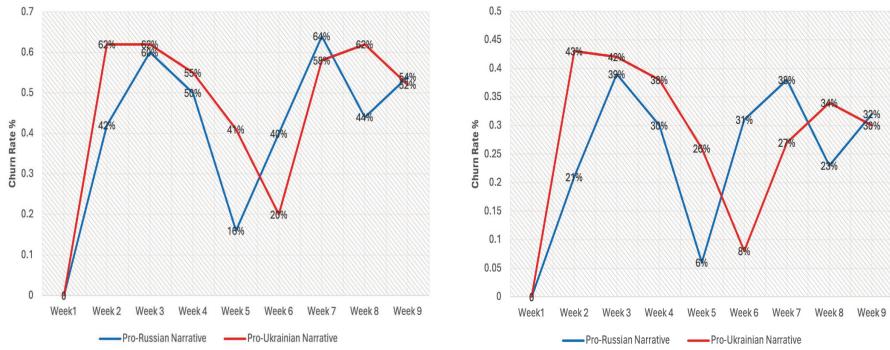
Fig. 4. Mathematical Model for Analyzing Churn via Narrative Analysis on Telegram.

5 Results

The study examines the impact of narratives on the expansion of pro-Russian and pro-Ukrainian Telegram channels during the ongoing Russia-Ukraine conflict. The analysis shows a significant increase in churn rates, with a 60% rise for pro-Russian channels and a 62% rise for pro-Ukrainian channels in the first two weeks, as shown in Fig. 5a. This surge coincided with the intensification of Russian military activities along the Ukrainian border, likely driving a surge in user engagement. However, the high level of activity did not last, as seen in the increased churn rates that followed. The third week was critical, with pro-Ukraine and pro-Russian narratives showing nearly identical churn rates, linked to NATO's heightened alert status and the strengthening of military forces in Eastern Europe. The analysis indicates that churn rates mirror the broader socio-political environment, with significant events in the Russia-Ukraine conflict as key user engagement points. The narratives surrounding "Special Military Operation" demonstrated how it caused users to churn during Week 5. However, the pro-Russian channels' higher churn rate indicates a less persuasive and contentious story. There is a correlation between significant events in the Russia-Ukraine war and variations in churn rates, as shown in Fig. 5a. For example, the pro-Russian platforms may have covered a significant event during Week 5, the week of the invasion, negatively, which could account for the increase in the churn rate. On the other hand, despite ongoing churn, the Pro-Ukrainian

channels were able to hold onto a marginally more consistent audience, possibly because of a narrative that resonated better.

The analysis of the Russia-Ukraine war on social media platforms reveals a competitive narrative landscape, with public interest oscillating between the two narratives, as shown in Fig. 5b. The Pro-Russian narrative shows an initial rise in churned posts/messages from Week 1 to Week 2, indicating a surge in disengagement or waning interest following the conflict's onset. Subsequent fluctuations suggest varying engagement levels, likely in response to the evolving situation and shifts in narrative strategies. For instance, a peak in the churn rate might align with a significant event where there was an heightened activity of the Russian troops in Week 1 that led to narrative change that either attracted or repelled subscribers. In contrast, the Pro-Ukrainian narrative starts with a lower churn rate than the Pro-Russian one, but surpasses it by Week 2 where we have growth rate increase of 72% (Table 2), indicating stronger initial engagement or interest in their messaging.



(a) Tracked churned users with narrative analysis per week. (b) Tracked churned posts with narrative analysis per week.

Fig. 5. Changes in the churned posts and users per week

The analysis measures users growth and shrinkage with narrative propagation in influencing public engagement on social media platforms. Effective storytelling can significantly affect whether an audience remains engaged with or starts to disengage from any channels. Understanding the correlation between churn rates and narrative strategies offers valuable insights into the dynamics of information dissemination and public sentiment during conflicts.

Figure 6 shows that there is both shrinking and expanding of active user engagement over time for the pro-Russian and pro-Ukrainian narratives. Weeks 2 and 3 show more active users in the pro-Ukrainian narratives, as displayed in the higher peaks of the blue and orange lines. Week 4 and week 5 present a momentous shift in active users; sharp peaks of green and light blue lines

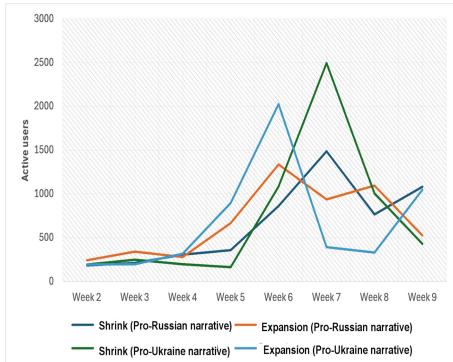


Fig. 6. Shrinking and Expanding Active Users for Pro-Russian and Pro-Ukrainian Narratives Over Time.



Fig. 7. Propagation of Pro-Russian vs. Pro-Ukrainian Narratives Over Time.

Table 1. Fitting Error Analysis of SEIZ Models in Telegram Narrative.

Metrics	SEIZ epidemiological Model			
	L^2 Error	MAE	Correlation	\mathfrak{R}_0
Pro-Russian				
Growth	0.0698	167.4	0.9955	6.49
Shrink	0.0887	187.4	0.9942	2.54
Pro-Ukrainian				
Growth	0.0900	228.4	0.9938	8.30
Shrink	0.0928	214.3	0.9960	0.88

Table 2. Analysis of Retention and Growth Over Time Periods.

Metrics	Time Periods								
	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	
Pro-Russian									
Gross Retention (GA_{t_i}/BOP_{t_i})	62%	44%	50%	79%	45%	36%	57%	30%	
Net Retention (NAC_{t_i}/BOP_{t_i})	84%	75%	78%	212%	78%	46%	91%	38%	
% Growth ($EOP_{t_i}/EOP_{t_i} - 1$)	56%	8%	16%	312%	43%	↓47%	20%	↓51%	
Pro-Ukrainian									
Gross Retention (GA_{t_i}/BOP_{t_i})	40%	38%	46%	67%	74%	31%	33%	59%	
Net Retention (NAC_{t_i}/BOP_{t_i})	58%	55%	72%	310%	128%	33%	38%	90%	
% Growth ($EOP_{t_i}/EOP_{t_i} - 1$)	23%	↓5%	35%	475%	74%	↓64%	↓51%	23%	

display this. The pro-Ukrainian narrative surpasses the pro-Russian narrative concerning active user engagement.

Table 3. Weekly Recurring User Metrics.

Time	BoP	WAC	Gross	NWAC	EoP	% Growth
Pro-Russian						
t_2	1082	459	623	1234	1,857	72%
t_3	1857	1,123	734	737	1,471	↓21%
t_4	1471	736	735	747	1,482	1%
t_5	1482	237	1,245	3080	4,325	192%
t_6	4325	1,730	2,595	6804	9,399	117%
t_7	9399	6,022	3,377	1364	4,741	↓50%
t_8	4741	2,070	2,671	2403	5,074	7%
t_9	5,074	52,759	2,315	1,664	3,979	↓22%
Pro-Ukrainian						
t_2	1461	910	551	1276	1,827	25%
t_3	1827	1,131	696	1028	1,724	↓6%
t_4	1724	948	776	1424	2,200	28%
t_5	2200	904	1,296	4472	5,768	162%
t_6	5768	1,167	4,601	5142	9,743	69%
t_7	9743	5,627	4,116	1028	5,144	↓47%
t_8	5144	3,210	1,934	1162	3,096	↓40%
t_9	3096	1,615	1,481	1318	2,799	10%

The results in Fig. 7 show that the pro-Ukrainian narrative propagated slightly more than the pro-Russian narrative at the initial stage (t_1 and t_2) (Table 3), possibly indicating more resonance with the international community's concerns about the threat of Russian invasion. From Week 3 to Week 5, with Russia launching its full-scale invasion of Ukraine on February 24th, the pro-Ukrainian narrative maintained a higher propagation level due to the widespread condemnation of Russia's military actions and the support behind Ukraine[14]. After the initial invasion period, both pro-Ukraine and pro-Russian narratives had a sharp decline in propagation.

Remark 2. We further employ the SEIZ epidemiological model [15] to study the narrative spread with respect to the churn model results. This helps us examine the effect of community churn (growth or shrinkage) on narrative propagation (i.e., RQ2). Table 1 shows the convergence rate in L^2 -norm, mean absolute error (MAE), and reproduction number \mathfrak{R}_0 for narrative propagation with respect to community churn (growth and shrinkage) at the final time. From Table 1, we can see that the pro-Ukrainian narrative spread rate $\mathfrak{R}_0 = 8.30$ is slightly more than the pro-Russian narrative rate $\mathfrak{R}_0 = 6.49$. Together with the churn analysis results from Figs. 6 and 7, we can conclude that the pro-Ukrainian community observed higher growth as compared to pro-Russian community and pro-Ukraine narratives spread faster than the pro-Russian narratives.

6 Conclusion and Future Research

This research presents a mathematical model designed for gathering and analyzing user engagement metrics in Pro-Russian and Pro-Ukrainian narrative channels on Telegram. We observed a significant shift in the churn rate for Pro-Russian narratives, along with changes in user engagement levels, displaying the crucial role narratives play in shaping public sentiment and interest. The study also explores the fluctuating patterns of users' interactions in both narrative domains, examining how strategic narrative construction impacts audience engagement during the conflict. We emphasize the significance of this data for predicting user attrition, especially for the power of the initial day's interactions. Our study has yielded valuable insights into the complex dynamics of the causes of users' churn, shedding light on critical aspects that can inform strategies for various users' behaviors. In this research, we have explored when specific online users will likely discontinue their engagement within a specified time frame, underscoring the importance of timely intervention to prevent churn. Furthermore, our investigation delves into the potential for users who have previously churned to return and re-engage, offering insights into possibilities for users' reactivation behavior. Churn analysis can measure the growth or decline of subscribers in narrative propagation by tracking the churn rate, which indicates the percentage of users who disengaged from a channel. A higher churn rate reflects a decrease in users and lower engagement with the propagated narrative, while a lower churn rate suggests an increase in users and stronger narrative propagation. By analyzing user churn rates in relation to narrative strategies and significant events like the Russia-Ukraine conflict, valuable insights can be gained into the dynamics of information dissemination and the factors influencing the spread of competing narratives during conflicts or other socio-political events. Additionally, this research has scrutinized whether the level of user engagement within a specific time frame during an online movement can serve as a predictor of its success or failure, thereby offering guidance for optimizing or enhancing the promotion strategies.

One of the limitations of the churn rate is its failure to account for the specific characteristics of churners. User decay is more noticeable among newly acquired users. In analyses, the churn rate lacks the ability to distinguish between different types of social media platforms, such as startups, those in the growth phase, or mature platforms.

In future work, we will explore the interplay of motivational factors on social media platforms and aim to create distinct behavioral profiles for different segments. We will also enhance our ensemble approach using machine learning and various evaluation metrics for user churn analysis.

Acknowledgments. This research is funded in part by the U.S. National Science Foundation (OIA-1946391, OIA-1920920), U.S. Office of the Under Secretary of Defense for Research and Engineering (FA9550-22-1-0332), U.S. Army Research Office (W911NF-23-1-0011, W911NF-24-1-0078), U.S. Office of Naval Research (N00014-21-1-2121, N00014-21-1-2765, N00014-22-1-2318), U.S. Air Force Research Laboratory, U.S.

Defense Advanced Research Projects Agency, Arkansas Research Alliance, the Jerry L. Maulden/Entergy Endowment at the University of Arkansas at Little Rock, and the Australian Department of Defense Strategic Policy Grants Program (SPGP) (award number: 2020-106-094). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations. The researchers gratefully acknowledge the support.

References

1. Gurung, M.I., Agarwal, N., Al-Taweel, A.: 'The 16th International Conference on Advances in Social Networks Analysis and Mining', in Are Narratives Contagious? Modeling Narrative Diffusion Using Epidemiological Theories. Calabria (2024)
2. Gurung, M.I., Bhuiyan, M.M.I., Al-Taweel, A., Agarwal, N.: Decoding YouTube's recommendation system: a comparative study of metadata and GPT-4 extracted narratives. In: Companion Proceedings of the ACM on Web Conference, pp. 1468–472 (2024). <https://doi.org/10.1145/3589335.3651913>
3. Edney, S., et al.: User engagement and attrition in an app-based physical activity intervention: secondary analysis of a randomized controlled trial. *J. Med. Internet Res.* **21**(11), e14645 (2019)
4. Geiger, A.W.: Key findings about the online news landscape in America. *PEW Res. Center* **11**, 13 (2019)
5. Singh, S., Sonnenburg, S.: Brand performances in social media. *J. Interact. Mark.* **26**(4), 189–197 (2012)
6. Dasgupta, K., Singh, R., Viswanathan, B., Chakraborty, D., Mukherjea, S., Nana-vati, A.A., Joshi, A.: Social ties and their relevance to churn in mobile telecom networks. In: Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology, pp. 668–677 (2008)
7. Xie, Y., Li, X., Ngai, E., Ying, W.: Customer churn prediction using improved balanced random forests. *Expert Syst. Appl.* **36**(3), 5445–5449 (2009)
8. Huang, B.Q., Kechadi, M.T., Buckley, B.: Customer churn prediction for broadband internet services. In: International Conference on Data Warehousing and Knowledge Discovery, pp. 229–243 (2009)
9. Kawale, J., Pal, A., Srivastava, J.: Churn prediction in MMORPGs: a social influence based approach. In: 2009 International Conference on Computational Science and Engineering, vol. 4, pp. 423–428 (2009)
10. Mozer, M.C., Wolniewicz, R., Grimes, D.B., Johnson, E., Kaushansky, H.: Predicting subscriber dissatisfaction and improving retention in the wireless telecommunications industry. *IEEE Trans. Neural Netw.* **11**(3), 690–696 (2000)
11. Buckinx, W., Van den Poel, D.: Customer base analysis: partial defection of behaviourally loyal clients in a non-contractual FMCG retail setting. *Eur. J. Oper. Res.* **164**(1), 252–268 (2005)
12. Karnstedt, M., Hennessy, T., Chan, J., Basuchowdhuri, P., Hayes, C., Strufe, T.: Churn in social networks. In: Handbook of Social Network Technologies and Applications, pp. 185–220 (2010)
13. Ngomang, B., Viennet, E., Tchuente, M.: Churn prediction in a real online social network using local community analysis. In: IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 282–288 (2012)
14. Bigg, M.M.: Russia invades Ukraine: a timeline of events. *The New York Times* (2023)

15. Maleki, M., Mead, E., Arani, M., Agarwal, N.: Measuring the interference effect of bots in disseminating opposing viewpoints related to Covid-19 on twitter using epidemiological modeling (2023)



Weakly Supervised Contrastive Representation Learning to Encode Narrative Viewpoint of COVID-19 Tweets

Kin Wai Ng^{1,2}, Nathan Wendt², Jasmine Eshun², and Emily Saldanha^{2(✉)}

¹ University of South Florida, Tampa, USA
kinwaing@usf.edu

² Pacific Northwest National Laboratory, Richland, USA
emily.saldanha@pnnl.gov

Abstract. The ability to detect and characterize online information campaigns will rely on the ability to infer shared underlying narrative viewpoints of online content. Existing methods typically can cluster documents by topic but struggle to distinguish between different points of view. The task is challenging due to the inherent characteristics of social media texts, which are noisy, short, and often provide very little context. Yet, due to the widespread prevalence of harmful misinformation in online media, the development of viewpoint detection approaches is crucial to enable the identification of information campaigns, the characterization of their sources, and their evolution. Our work proposes a weakly supervised contrastive representation learning approach to infuse latent text representations with viewpoint information by leveraging proxy signals observed through social interaction networks. We test our solution on a Twitter dataset related to COVID-19 discussions. We demonstrate the ability of our approach to separate COVID-19 tweets by their narrative viewpoint compared to baseline pre-trained embeddings.

Keywords: social media · weak supervision · contrastive learning · semantic embeddings

1 Introduction

Viewpoint detection is a crucial step for characterizing narratives and understanding their diffusion and evolution in the online space. The detection of shared narrative viewpoint in social media text has many inherent challenges. Online posts are typically short, noisy and written using unique styles and lexicons [3]. They usually offer limited context or only a small portion of the viewpoint they promote. Existing methods have shown progress in identifying collections of documents sharing similar topics via learned semantic embeddings [2, 7]. However, such approaches often struggle to differentiate between different points of view

related to the same topic, which is necessary for analytical studies aimed at understanding public opinion, especially on controversial issues with prevalent misinformation such as the COVID-19 pandemic. Another challenge is related to the lack of availability of labeled data for training, which can be attributed to the high-costs of the data annotation process and the large-scale of social media data.

These challenges motivate the use of weak supervision approaches for viewpoint detection. Particularly, we aim to develop weakly supervised contrastive methods which can leverage self-supervised signals present in social media data to learn a latent space representation of tweet content which is organized by shared narrative viewpoint. The use of self-supervised signals also has the benefit of avoiding reliance on manually annotated data which would not allow easy generalization to new topic areas or emerging events within the domain of interest. Motivated by the observed homophily of online social networks [14], we leverage social proximity as the weak proxy signal for our contrastive methods and train the model to infer the social network proximity of the authors of a given pair of tweets based only on the text of those tweets.

In this paper, we demonstrate the methodology on Twitter discussions related to COVID-19. Through multiple experiments, we show that fine-tuned embeddings leveraging social proximity signals present in retweet networks demonstrate the capability to incorporate rich viewpoint information. We find that domain-specific embeddings are crucial to improve the encoding of viewpoint information compared to out-of-domain pre-trained embeddings. However, while viewpoint encoding tends to improve when trained with this approach, it comes with a trade-off in the topic encoding performance. We propose an approach to partially mitigate this issue by incorporating topic-aware representations which maintain the original topic information and augment the existing representations with learnable viewpoint-encoding dimensions. Our results confirm that improvement on the narrative viewpoint task correlates with performance on the proxy task of weak social signals. This observation highlights the usefulness of weakly supervised approaches for this task and calls for future research on investigating the incorporation of additional signals for further improving the embeddings. The code to replicate our methods can be found here: <https://github.com/pnnl/NARREMB>.

2 Related Work

The target task of narrative viewpoint detection is related but not identical to several other tasks designed to group and categorize text such as topic detection or stance detection. Topic detection involves the detection of groups of documents related to the same subject area. Methods for topic detection typically do not differentiate between different points of view related to the same topic. For example, discussion in support of mask wearing and discussion opposed to mask wearing will likely be grouped into the same topic unless the vocabulary used by the two groups becomes highly disjoint. Meanwhile, the task of stance detection

typically focuses on the detection of different opinion categories towards a given specific entity or viewpoint. Our task differs from stance detection, as our objective is not to categorize tweets into predefined stance labels. Instead, we aim to implicitly encode viewpoint information into learned tweet embeddings such that tweets which share both a topic and a viewpoint are clustered together. Embeddings with improved encoding of viewpoint can be used for many downstream tasks relative to those which encode only topics. Here we highlight prior work in related areas and modeling techniques.

Topic Modeling. has been widely applied as an unsupervised method to discover the underlying organization of a corpus into topical groups [2, 4]. For example, it has been used to compare topics discovered from news and from Twitter to perform narrative comparison analysis [22]. *Stance detection* focuses on the extraction of an author’s stance towards a given target from a fixed set of options (e.g. Favor or Against) [16]. While these methods typically require pre-defined target and stance categories, recent work has tackled the unsupervised stance detection task [9]. *Weak supervision* is a model training framework which leverages large datasets of automatically generated noisy labels in lieu of high quality annotations which are time and effort intensive to generate [23]. Weakly supervised approaches have been previously applied within NLP problems ranging from text and document classification [19] to information extraction such as named entity recognition [17]. In this work, we leverage the social proximity of users observed through interaction networks as our weak signal to infer the similarity of tweets and their viewpoints towards a particular topic. *Contrastive representation learning* is often applied to different views of the same data instance. For example, in computer vision, the goal of the contrastive learning framework is to map similar images (e.g., original and augmented) close together while pushing dissimilar ones further away in the embedding space. The adoption of this framework has led to state-of-the-art performance on unsupervised image classification tasks [6]. The same idea has also been applied to natural language text, especially in tasks involving text retrieval [25], text generation [1], and sentence embedding representations [11], often resulting in significant performance improvements over multiple baselines.

Previous studies have applied topic modeling and narrative characterization to analyze online discussions related to COVID-19 [5, 10, 15, 26, 27]. However, these studies often rely on pre-trained embeddings to generate candidate clusters of text, which can be less effective due to the context-limited nature of online posts. Our proposed approach aims to enhance the pre-trained embeddings of tweets by incorporating viewpoint information. These resulting embeddings can help to identify more meaningful and fine-grained tweet clusters, and thus improve the characterization of narratives.

3 Data

We leverage *Avax* [21], a dataset on COVID-19 Twitter discussions, especially about vaccine adoption and hesitancy. It is publicly available

(<https://github.com/gmuric/Avax-tweets-dataset>) and contains 1.2M records posted by 568K users from Oct. 18th, 2020 to April 21st, 2021.

Table 1. Identified set of controversial topics and viewpoints for evaluation. The number of tweets for each topic is in parentheses. VP = Viewpoint, S = Support, O = Oppose, B = Believe, R = Refute.

Topic	Category	Discussion Description	VP 1	VP 2
Face masks	Policy	The use of face masks	S (36)	O (30)
Re-open schools	Policy	Reopen schools for in-person learning	S (27)	O (30)
Vaccine passports	Policy	Adoption of vaccine passports	S (31)	O (27)
Depopulation agenda	Conspiracy	Vaccines as part of a depopulation plan	B (29)	R (33)
Miscarriage agenda	Conspiracy	Miscarriages following vaccinations	B (30)	R (30)
Vaccines	General	General opinions of the vaccines	S (33)	O (29)

We preprocessed the tweets as follows: (1) removed duplicate entries (2) ignored retweets and only considered actions with textual content written by the user (i.e., tweets, replies, and quotes), (3) removed mentions, URLs, and special characters from tweets to focus on the natural language content, (4) removed non-English tweets, and (5) performed a fuzzy text deduplication technique to remove repeated tweets with almost the exact same content. We also removed tweets containing less than 20 characters as they constitute less than 1% of the total tweets in the dataset and often lack textual signals necessary for identifying viewpoints. We employ Top2Vec [2] to assign a broad topic to each tweet in the dataset. This results in 67 topics with sizes ranging from 861 to 26,585 tweets. After preprocessing, we end up with 362,146 unique tweets from 191,823 users. We further explore these broad Top2Vec topics and grouped them into ten higher-level categories of selected topics (as shown in Table 1).

Given the absence of ground truth data for viewpoint detection, we performed manual annotation of binary user viewpoints towards selected topics to facilitate the evaluation of various embedding approaches in their ability to capture both the semantic content and viewpoint of tweets. A set of selected tweets from each of the ten selected topics were manually labeled by the authors according to viewpoint of each tweet towards its topic. Table 1 shows the statistics for each selected topic. We use these 365 annotated tweets as a test set to evaluate the performance of trained models on encoding shared viewpoints of users.

Social Network Data: By leveraging social network information as weak signals, we can improve the representation of tweets, as the implicit connections between users can offer valuable insights into user perspectives and viewpoints. Hence, we construct the social network by leveraging user retweet interactions. We focus on the retweet network because these interactions are most likely to indicate endorsement of a shared viewpoint between users as opposed to other actions such as replies or mentions, where disagreement may be expressed. We

expect that users with close proximity in the network would likely share similar viewpoints while users who are distant in the retweet network are more likely to hold differing views. In this work, we build three distinct retweet networks, each designed to measure the strength of social connections in different ways and study the impact of noise or outliers. The different types of retweet networks constructed are as follows: (1) *Original* network, which includes all retweet interactions without filtering (382K nodes and 646K edges), (2) *Active* network, which includes users who have made at least two retweets over the entire observation period (118K nodes and 378K edges), and (3) *Strong* network (19K nodes and 27K edges), which includes only edges with an edge weight of at least 2 (i.e., each user must have retweeted at least twice another user). The Active network is a subset of the Original network, and the Strong network is a subset of the Active network.

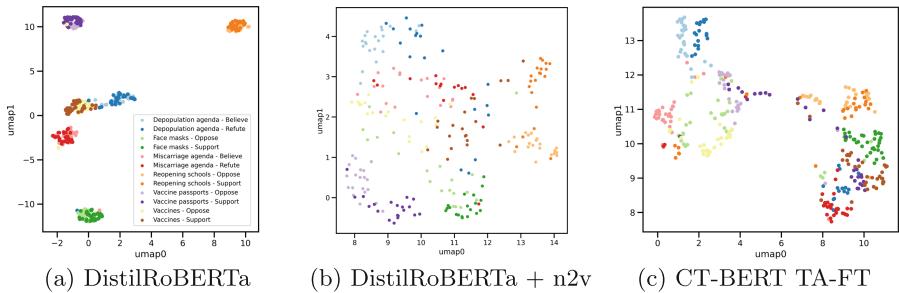


Fig. 1. UMAP latent space visualization of the baseline embedding (a); a concatenated embedding combining DistilRoBERTa with node2vec (b); and the best performing topic-aware fine-tuned model trained with the strong retweet network (c) for a selected set of topics with manually annotated binary viewpoints.

4 Methodology

To motivate the need for our proposed approach, we first demonstrate that baseline sentence embedding methods struggle to encode viewpoints. In Fig. 1a, we show a 2D UMAP projection [18] of a standard pretrained sentence embedding model from SentenceTransformer (<https://huggingface.co/sentence-transformers/all-distilroberta-v1>) [24] on our set of tweets with annotated binary viewpoints using dark versus light colors to represent different viewpoints on the same topic. We find that the tweets are strongly clustered by topic but achieve no separation by viewpoint. We aim to improve the embedding space with viewpoint information such that tweets are clustered by both topic and viewpoint.

4.1 Social Signals

We use social network information to first demonstrate the utility of social signals for viewpoint encoding by generating node2vec [13] embeddings for each

user in the *Avax* Original network. We concatenate the baseline DistilRoBERTa embeddings with these node2vec embeddings. The 2D UMAP projections of this concatenated embedding space, shown in Fig. 1b, show that this approach is able to partially separate the annotated test set by both topic and user viewpoint, in contrast with the sentence embeddings alone, showing the utility of social information. However, this approach relies on the availability of the complete social network information. In *Avax*, only 53% of tweets were authored by users who have interacted with other users via retweet interactions. Hence, we cannot utilize the node2vec representations to infer the stance or viewpoint of the users who authored the remaining 47% of the tweets.

Our weak supervision approach is designed to transfer the weak signals from tweets where social network information is available to those where they are lacking. For example, we can identify the textual signals that make a pair of tweets appear as if their authors would be closely related or distant in the social network. This allows us to cluster all tweets using their text alone with a focus on the textual features that are indicative of the likely social relationships of the user. To this end, we leverage the social proximity between two users, which is represented by their shortest path distance in the retweet network.

4.2 Contrastive Model

We develop a contrastive model to infuse the text embeddings with social network information. The goal of our contrastive fine-tuning task is to encode social proximity likelihood by making the embeddings for pairs of tweets similar if their authors are close (positive samples) and dissimilar if their authors are distant (negative samples) in the corresponding retweet networks. We leverage a contrastive loss function based on cosine similarity to measure proximity in the latent space.

$$\text{loss} = \begin{cases} 1 - \cos(x_1, x_2) & \text{if } y = 1 \\ \max(0, \cos(x_1, x_2) - m) & \text{if } y = -1 \end{cases}$$

where x_1 and x_2 are vectors, y is the label (1 for positive samples and -1 for negative samples), \cos is the cosine similarity function, and m is a margin value. Through experimentation on the full training data, we tested margin values between 0 and 0.5 selecting m equal to 0. This means that the cosine embedding loss will penalize dissimilar pairs that have positive cosine similarity.

We tested two different pre-trained embedding models implemented in huggingface: DistilRoBERTa which uses all-distilroberta-v1 model [24], and CT-BERT which uses the covid-twitter-bert-v2 model [20]. We expect that DistilRoBERTa will better preserve the topical similarity of tweets, as the model was specifically trained on semantic similarity-related tasks, but potentially struggle in separating user viewpoints. On the other hand, we expect that the embeddings produced by CT-BERT would better match the data domain. We emphasize that there is no risk of train-test data contamination for CT-BERT as its data collection period is different from that of the *Avax* dataset. Specifically, the CT-BERT

model was trained on tweets from January 12th to July 5th, 2020, while the *Avax* data starts on October 18th, 2020.

While the pre-trained embedding representations successfully encode topic similarity of tweets, the embeddings resulting from fine-tuning on the social proximity task might not be able to preserve this information. To address this issue, we experiment with a simple approach consisting of concatenating the initial pre-trained embedding representations (topic vector) with an extra n -dimensional vector. This new n -dimensional vector is learned by fine-tuning the embeddings of the pre-trained model and then passing them through a linear layer which reduces the vector to a size of n . The concatenation of the frozen pre-trained embedding and the learned extra dimensions is then used in the contrastive loss function. This approach aims to preserve topic similarity of tweets in the latent space by using the frozen embedding while also encoding additional information related to shared-viewpoints through the additional dimensions.

4.3 Evaluation Metrics

Correlation. We compute the Pearson, Spearman, and Kendall Tau correlation scores between the content similarity of tweet pairs, given by the cosine similarity of their text embeddings, and the social proximity of the authors of the tweets, given by their shortest path length in the retweet networks. The correlation metric does not measure the model’s ability to encode viewpoints, but indicates the performance on the proxy task corresponding to weak social signals.

ViewpointNN. To probe the ability of the embeddings to encode viewpoint towards a particular topic, we used the idea of k nearest neighbors (k -NN). Specifically, for each tweet in the manually annotated evaluation set, we find its k nearest neighbors in the embedding space of all tweets from the same topic (i.e., we do not consider tweets belonging to a different topic). We then predict the viewpoint class (e.g. Support or Oppose) by majority voting. The ViewpointNN metric is the proportion of correctly classified tweets.

TopicNN. To evaluate the ability of the embeddings to capture a shared topic, we also leverage k -NN using the manually annotated data. In this case, for each tweet, we find the k nearest neighbors in the embedding space of all other tweets in the annotated set (i.e., we consider all topics), and predict the topic class by majority voting. The TopicNN metric is the proportion of correctly classified tweets among all tweets in the annotated set.

4.4 Experimental Setup

To create the train, validation, and test data splits for the contrastive models, we sample 70% of users for training and 15% each for validation and testing using a stratified approach to maintain the distribution of topics across splits. For each split, we sample user pairs and compute their shortest path length in the corresponding retweet network. We transform the shortest path length

values into a two-class representation - user pairs in close proximity and user pairs in distant proximity. We rely on a two-stage sampling approach to generate candidate pairs. First, we use a snowball sampling strategy to sample close connections by exploring the 1-hop and 2-hop neighborhoods of a random set of users. Second, we incrementally add more examples by randomly selecting two users, computing their shortest path length, and retaining them if their distance is larger than 2.

Table 2. Model performance over the three types of retweet networks. We report performance on ViewpointNN and TopicNN metrics with $k=5$, and the Pearson correlation metric. The Baseline performance for ViewpointNN and TopicNN metrics is the same across different retweet networks as the baselines do not leverage the social network information. The correlation metric is computed on the corresponding test split for each network type. FT refers to fine-tuned contrastive models, and TA-FT refers to topic-aware contrastive models.

Network Type	Metric	DistilRoBERTa			CT-BERT		
		Baseline	FT	TA-FT	Baseline	FT	TA-FT
Original	ViewpointNN	0.65	0.73	0.72	0.82	0.92	0.92
	TopicNN	0.98	0.79	0.90	0.88	0.53	0.73
	Correlation	-0.24	-0.32	-0.37	-0.34	-0.44	-0.46
Active	ViewpointNN	0.65	0.69	0.65	0.82	0.92	0.89
	TopicNN	0.98	0.68	0.98	0.88	0.60	0.78
	Correlation	-0.26	-0.24	-0.26	-0.31	-0.39	-0.44
Strong	ViewpointNN	0.65	0.69	0.69	0.82	0.92	0.92
	TopicNN	0.98	0.59	0.86	0.88	0.60	0.81
	Correlation	-0.28	-0.30	-0.37	-0.33	-0.35	-0.43

Furthermore, we augment the number of close pairs in each split by adding pairs of tweets authored by the same user, which are likely to reflect similar viewpoints. We select pairs of tweets from each pair of users considering only tweets from the same topic to encourage the model to learn differing viewpoints towards the same topic. We use a shortest path length cutoff value of 4, which means that path lengths in the range of 1–3 are considered close while values of 4 or greater are distant.

5 Experimental Results

In this section, we report the performance of our fine-tuned embeddings in the task of encoding viewpoint and topic information. We compare our model against the two baseline sentence embeddings to demonstrate its effectiveness and limitations.

Table 2 shows the full set of metric results for all three social networks (original, active, and strong), each starting embedding (DistilRoBERTa and CT-BERT), and each weak supervision method (fine-tuned and topic-aware fine-tuned). First, we evaluate our models on the proxy task of encoding the social proximity of users within the tweet embeddings. The contrastive training leads to consistent improvements relative to the starting embeddings in all correlation scores, which measure the similarity between text embeddings and the shortest path distance of their respective authors. Our fine-tuned embeddings (FT) have an average improvement of 0.05 in correlation and the topic-aware embeddings (TA-FT) have an average improvement of 0.10 in correlation. This suggests that our fine-tuned embeddings capture more informative signals for discerning social proximity based solely on the content of the tweets.

To assess the ability of our trained models in encoding text representations with viewpoint information, we evaluate the fine-tuned embeddings and the two baseline embeddings on the annotated set of tweets described in Table 1. We observe an improvement in ViewpointNN performance with both fully fine-tuned models (FT) and topic-aware models (TA-FT) compared to their corresponding baselines. Specifically, when considering models trained on the original retweet network, the FT models have a performance improvement of 12.3% with DistilRoBERTa embeddings and 12.2% with CT-BERT embeddings. Similarly, the TA-FT models also present a performance improvement of 10.7% with DistilRoBERTa embeddings and 12.2% with CT-BERT embeddings.

The observed improvement in viewpoint performance is associated with a degradation in topic performance as measured by TopicNN. Specifically, the FT models show a decrease in performance of 19.4% compared to the DistilRoBERTa baseline and 39.8% compared to the CT-BERT baseline. The best performance on the TopicNN metric is achieved by the DistilRoBERTa baseline, which is expected given that its embeddings are trained on sentence-similarity tasks, allowing them to encode more robust semantic similarities. However, these baseline embeddings have the worst ViewpointNN performance. The observed trade-off between viewpoint and topic performance is driven by the nature of our proxy task. For example, tweets from distinct topics but likely similar viewpoints may be projected into similar regions in the embedding space, which inevitably leads to a reduction in topic performance. TA-FT models can mitigate this problem as they achieve a considerably smaller degradation in TopicNN in comparison to the FT models. Specifically, the TA-FT models show a decrease in TopicNN performance of only 8.1% compared to the DistilRoBERTa baseline and 17.0% compared to the CT-BERT baseline while achieving an improvement on ViewpointNN performance.

The domain-specific pre-training of CT-BERT improves the encoding of viewpoint embeddings by 26%. This highlights the importance of leveraging domain-specific embeddings over out-of-domain embeddings, as the former contains more relevant contextual information and vocabulary tailored to the specific domain of interest. In comparing the performance across the three different networks, we found that the model trained with the retweet network of strong relationships

demonstrated the best performance with an improvement of 12.2% in ViewpointNN and a reduction of 8.0% in TopicNN compare to the CT-BERT baseline. This can be attributed to its consideration of less noisy interactions, thus avoiding spurious interactions by chance.

Figure 1c shows the latent space embedding of the best performing model, *CT-BERT TA-FT Strong*, and demonstrates that tweets sharing similar viewpoints tend to group together while those with opposite viewpoints are pushed to distant regions of the latent space showing a pattern that likely reflects the polarization of the social network.

6 Conclusions and Future Work

In this work, we have introduced a weakly supervised contrastive approach to encode narrative viewpoint of tweets. Our approach leverages social proximity signals observed through user interaction networks to identify the textual features that make a pair of tweets appear as if their authors would be closely related in the social network. Using a Twitter dataset related to COVID-19 discussions, we show that our models are better than baseline sentence embedding methods in distinguishing tweets with shared and opposing viewpoints towards particular topics, and in effectively grouping them into similar regions of the latent space.

One limitation of the current approach to the evaluation of our method is that it is based on only binary viewpoints on each topic. A more complex set of narrative annotations would be needed to infer whether the methodology can infuse the embedding representations with more nuanced and fine-grained viewpoint information. Further work is also needed to leverage the learned representations for the automated identification and characterization of online narratives through clustering of the latent space or other approaches.

Social network proximity provides one source of signal on user viewpoints through the observed assortativity and polarization of such interactions [8, 12]. However, the methods described here could be augmented by incorporating additional weak signals into the model supervision. Future work could investigate the incorporation of additional signals, for example, contrastive models which can infer whether two tweets use the same hashtag or share the same URL based only on the natural language text of the tweet. The incorporation of multiple such weak signals would likely provide additional information which could enhance the topic representation of embeddings and further mitigate the trade-off between viewpoint and topic performance, leading to more robust representations.

Acknowledgements. The research described in this paper was conducted under the Laboratory Directed Research and Development Program at Pacific Northwest National Laboratory, a multi-program national laboratory operated by Battelle for the U.S. Department of Energy.

References

1. An, C., Feng, J., Lv, K., Kong, L., Qiu, X., Huang, X.: CoNT: contrastive neural text generation. In: Oh, A.H., Agarwal, A., Belgrave, D., Cho, K. (eds.) Advances in Neural Information Processing Systems (2022). <https://openreview.net/forum?id=mjVZw5ADSbX>
2. Angelov, D.: Top2Vec: distributed representations of topics. arXiv preprint [arXiv:2008.09470](https://arxiv.org/abs/2008.09470) (2020)
3. Benamara, F., Inkpen, D., Taboada, M.: Introduction to the special issue on language in social media: exploiting discourse and other contextual information. Comput. Linguist. **44**(4), 663–681 (2018). https://doi.org/10.1162/coli_a_00333. <https://aclanthology.org/J18-4006>
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
5. Boon-Itt, S., Skunkan, Y., et al.: Public perception of the Covid-19 pandemic on Twitter: sentiment analysis and topic modeling study. JMIR Public Health Surveill. **6**(4), e21,978 (2020)
6. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Adv. Neural. Inf. Process. Syst. **33**, 9912–9924 (2020)
7. Chaudhary, Y., Gupta, P., Saxena, K., Kulkarni, V., Runkler, T., Schütze, H.: TopicBERT for energy efficient document classification. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 1682–1690. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.findings-emnlp.152>. <https://aclanthology.org/2020.findings-emnlp.152>
8. Conover, M., Ratkiewicz, J., Francisco, M., Gonçalves, B., Menczer, F., Flammini, A.: Political polarization on twitter. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 5, pp. 89–96 (2011)
9. Darwish, K., Stefanov, P., Aupetit, M., Nakov, P.: Unsupervised user stance detection on twitter. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 14, pp. 141–152 (2020)
10. Doogan, C., Buntine, W., Linger, H., Brunt, S., et al.: Public perceptions and attitudes toward Covid-19 nonpharmaceutical interventions across six countries: a topic modeling analysis of twitter data. J. Med. Internet Res. **22**(9), e21,419 (2020)
11. Gao, T., Yao, X., Chen, D.: SimCSE: simple contrastive learning of sentence embeddings. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 6894–6910. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (2021). <https://doi.org/10.18653/v1/2021.emnlp-main.552>. <https://aclanthology.org/2021.emnlp-main.552>
12. Garimella, V.R.K., Weber, I.: A long-term analysis of polarization on twitter. In: Eleventh International AAAI Conference on Web and Social Media (2017)
13. Grover, A., Leskovec, J.: Node2Vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
14. Halberstam, Y., Knight, B.: Homophily, group size, and the diffusion of political information in social networks: evidence from twitter. J. Public Econ. **143**, 73–88 (2016)
15. Jing, E., Ahn, Y.Y.: Characterizing partisan political narrative frameworks about Covid-19 on twitter. EPJ Data Sci. **10**(1), 53 (2021)

16. Küçük, D., Can, F.: Stance detection: a survey. *ACM Comput. Surv.* (CSUR) **53**(1), 1–37 (2020)
17. Lison, P., Barnes, J., Hubin, A., Touileb, S.: Named entity recognition without labelled data: a weak supervision approach. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1518–1533. Association for Computational Linguistics, Online (2020). <https://doi.org/10.18653/v1/2020.acl-main.139>. <https://aclanthology.org/2020.acl-main.139>
18. McInnes, L., Healy, J., Saul, N., Großberger, L.: UMAP: uniform manifold approximation and projection. *J. Open Source Softw.* **3**(29), 861 (2018). <https://doi.org/10.21105/joss.00861>. <https://doi.org/10.21105/joss.00861>
19. Mekala, D., Shang, J.: Contextualized weak supervision for text classification. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 323–333 (2020)
20. Müller, M., Salathé, M., Kummervold, P.E.: Covid-twitter-BERT: a natural language processing model to analyse Covid-19 content on twitter. arXiv preprint [arXiv:2005.07503](https://arxiv.org/abs/2005.07503) (2020)
21. Muric, G., Wu, Y., Ferrara, E.: COVID-19 vaccine hesitancy on social media: building a public twitter data set of antivaccine content, vaccine misinformation, and conspiracies. *JMIR Public Health Surveill* **7**(11), e30,642 (2021). <https://doi.org/10.2196/30642>. <https://publichealth.jmir.org/2021/11/e30642>
22. Nerges, A., Lee, J.S.: Narratives of the refugee crisis: a comparative study of mainstream-media and Twitter. *Media Commun.* **7**(2 Refugee Crises Disclosed), 275–288 (2019)
23. Ratner, A., Bach, S.H., Ehrenberg, H., Fries, J., Wu, S., Ré, C.: Snorkel: rapid training data creation with weak supervision. In: Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases, vol. 11, p. 269. NIH Public Access (2017)
24. Reimers, N., Gurevych, I.: Sentence-Bert: sentence embeddings using Siamese Bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (2019). <https://arxiv.org/abs/1908.10084>
25. Xiong, L., et al.: Approximate nearest neighbor negative contrastive learning for dense text retrieval. In: International Conference on Learning Representations (2021). <https://openreview.net/forum?id=zeFrfgyZln>
26. Xue, J., Chen, J., Chen, C., Zheng, C., Li, S., Zhu, T.: Public discourse and sentiment during the Covid 19 pandemic: using latent dirichlet allocation for topic modeling on twitter. *PLoS one* **15**(9), e0239,441 (2020)
27. Yin, H., Song, X., Yang, S., Li, J.: Sentiment analysis and topic modeling for Covid-19 vaccine discussions. *World Wide Web* **25**(3), 1067–1083 (2022)



Enhancing Multilingual Fake News Detection Through LLM-Based Data Augmentation

Razieh Chalehchaleh^(✉), Reza Farahbakhsh, and Noel Crespi

Télécom SudParis, Institut Polytechnique de Paris, Palaiseau, France
{razieh.chalehchaleh,reza.farahbakhsh,noel.crespi}@telecom-sudparis.eu

Abstract. The rapid growth of online news consumption has intensified the spread of misinformation, underscoring the critical need for effective fake news detection methods. Despite significant advancements in this area, the scarcity and inadequacy of high-quality labeled datasets necessary for training effective detection models remains a major challenge. In this paper, we introduce a novel approach to address this issue by leveraging large language models (LLMs) for data augmentation. Specifically, we employ Llama 3 to generate multiple synthetic news samples per original article, enriching existing fake news datasets to enhance fake news detection. We explore various augmentation strategies like different augmentation rates, random or similarity-based subsampling, and selectively augmenting data from specific classes to optimize the augmented datasets to train better classifiers. We evaluate the efficacy of our approach using BERT-based classifiers on two multilingual datasets. Our findings reveal notable improvements particularly when augmenting only the fake class with rate 1.

Keywords: Multilingual Fake News Detection · Large Language Models (LLMs) · Data Augmentation

1 Introduction

In recent years, rapid technological advancements and their widespread adoption have reshaped the landscape of news consumption. This transformation has triggered a notable surge in the usage of online platforms for accessing news content. A 2023 survey¹ found that 86% of U.S. adults frequently or occasionally access news via digital devices with 50% at least sometimes obtaining news from social media. Another 2023 study² highlights that 40% of Americans who use social media to access news are concerned about inaccuracies-encompassing issues such as unverified facts, misinformation, the spread of fake news, and reliance on unreliable sources. The COVID-19 pandemic accelerated the spread of fake news,

¹ <https://www.pewresearch.org/journalism/fact-sheet/news-platform-fact-sheet>.

² <https://www.pewresearch.org/short-reads/2024/02/07/many-americans-find-value-in-getting-news-on-social-media-but-concerns-about-inaccuracy-have-risen>.

emphasizing the need to combat false information. A poll involving 806 U.S. physicians³ reveals that misinformation is perceived as a significant issue with 44% of the respondents indicating that more than half of the COVID-19 information they encountered from patients was medically inaccurate. Moreover, 72% reported that misinformation aggravated difficulties in treating patients.

While there is no universally accepted definition of “fake news” in the literature, Allcott and Gentzkow[3] offer a widely recognized definition, describing it as “news articles that are intentionally and verifiably false and could mislead readers.” Extensive research has been devoted to addressing the critical task of fake news detection. A thorough review of research and methods for detecting and preventing the spread of fake news on online social networks is presented in [2]. One of the most critical aspects of training supervised models and enhancing their efficiency is the training data. The lack of labeled data, particularly in under-resourced languages, poses a significant challenge. Although several datasets are available for fake news detection, they often face limitations such as size, modality, granularity, and becoming outdated over time [13]. To tackle the issue of limited data availability, researchers have utilized diverse techniques to augment existing fake news datasets. These methods aim to enhance both the volume and diversity of training data, thereby improving model performance without the need for further data collection efforts. However, the outcomes of these augmentation approaches in fake news detection have been varied. While some studies have shown improvements in overall outcomes [14, 17], others have reported insignificant, negative, or mixed effects on the results [1, 4, 5, 15, 16].

Large Language Models (LLMs) have revolutionized the field of natural language processing (NLP) [20, 27]. These models demonstrate robust abilities to comprehend natural language and tackle complex tasks, particularly through text generation [26]. Leveraging LLMs for data augmentation presents a valuable use case as they can sometimes produce high-quality synthetic data that exceeds human-curated ones. This approach helps address the scarcity of human-annotated data and permits the expansion of training datasets without proportionally increasing computational costs. Additionally, the use of synthetic data can substantially reduce data collection expenses and energy consumption [11].

In this study, we aim to address the challenges posed by the scarcity and inadequacy of labeled fake news datasets by augmenting the existing fake news datasets with synthetic news generated by LLMs. We utilize Llama 3⁴, an open-source LLM developed by META GenAI, for fake news data augmentation. We will instruct the model to generate multiple news samples per original article, maintaining the same labels as the original pieces. To optimize our dataset augmentation for fake news detection, we will explore various strategies. After generating the augmented datasets, we will train BERT (Bidirectional Encoder Representations from Transformers) [10] classifiers for the news classification. Our strategies will include experimenting with different augmentation rates—the

³ <https://debeaumont.org/news/2023/physician-poll-medical-misinformation-is-harming-patients>.

⁴ <https://llama.meta.com/llama3>.

number of additional news samples generated per original sample, using random or similarity-based subsampling, and selectively augmenting data from specific classes (both fake and real, only fake, or only real). A schematic overview of our approach is provided in Fig. 1.

Our results on the two multilingual real-world datasets TALLIP [9] and MM-COVID [18] underscored the effectiveness of data augmentation in enhancing model performance. Notably, when augmenting only the fake class with one synthetic sample per original article (rate 1 augmentation), the approach consistently outperformed others. For instance, in the TALLIP dataset, augmenting only the fake class led to a notable improvement in F1 scores, with English increasing by 7.7% and Hindi by 4.4%. In the MM-COVID dataset, augmenting only the fake class improved F1 scores by 1.5% for Spanish and 1.1% for both Hindi and English. While similarity-based sampling generally yielded higher F1 scores, the difference was less pronounced when augmenting only real or fake news, suggesting consistent quality across the generated samples.

The rest of the paper is organized as follows: Section 2 reviews related work. Section 3 details our augmentation techniques and classification method. Section 4 covers the datasets, evaluation setting, and results. Section 5 addresses limitations and future research directions. Section 6 summarizes our approach and findings.

2 Related Work

Extensive research has been dedicated to the crucial task of fake news detection. A comprehensive review of fake news research and the examination of current methods for detecting and preventing the spread of fake news on online social networks is presented in [2]. A critical factor in training supervised models and enhancing their efficiency is access to high-quality datasets. Although numerous datasets are available for fake news detection [12], they often suffer from limitations such as size, modality, granularity, and becoming outdated over time [13]. These challenges are particularly pronounced for under-resourced languages.

In the literature, numerous researchers have attempted to augment fake news datasets using various techniques, addressing the challenges of not having a sufficient amount of data. Data augmentation refers to utilizing innovative techniques to enhance model performance by expanding the volume and diversity of training data, without the need for additional data collection. Bayer et al. conducted a comprehensive survey of various augmentation methods in [6]. The results of these augmentation approaches in the field of fake news detection have shown inconsistency. While some studies have reported improvements in overall outcomes, many others have found insignificant, negative, or mixed effects on the results. Amjad et al. [4] employed an augmentation approach to train a fake news classifier in Urdu, using both manually annotated and machine-translated English datasets, but found no significant improvement in performance.

Pre-trained language models (PLMs) based on transformer architectures [25] have revolutionized the field of natural language processing (NLP). Trained on large-scale corpora, these models exhibit robust capabilities across a spectrum of tasks. Models like BERT (Bidirectional Encoder Representations from Transformers) [10] and RoBERTa (Robustly optimized BERT approach) [19], developed using transformer architecture and self-attention mechanisms, have shown exceptional abilities in capturing linguistic patterns and semantic relationships. The introduction of these comprehensive context-aware representations has led to widespread adoption by researchers across various tasks and applications. Hua et al. [14] introduced a multimodal approach that used a BERT-based back-translation method to augment the dataset size. Their results demonstrate promising improvements. Keya et al. [17] introduced AugFake-BERT, which leverages BERT to generate synthetic texts, thereby expanding the dataset with augmented fake data. Their study revealed an improvement in classification outcomes when applying text augmentation to the minority class. Kapusta et al. [16] used various text data augmentation techniques, including synonym replacement, back translation, and reduction of function words, to optimize a text corpus for fake news classification, revealing significant differences in classifier outcomes between augmented and original corpora. Ashraf et al. [5] applied random word insertion or replacement using Word2Vec similarity search for news claim classification but found no improvement. Junior et al. [15] augmented an English dataset by replacing nouns with synonyms based on part of speech tagging and similarity thresholds, resulting in a decrease in validation and test accuracy when translated to Portuguese for BERT classification.

Large language models (LLMs) have gained significant attention for their impressive performance across various tasks ranging from general NLP to domain-specific applications, especially following the release of OpenAI's ChatGPT⁵. These models, known for their proficiency in language understanding and generation, are commonly recognized as transformer-based language models with hundreds of billions of parameters, trained on extensive textual corpora [22]. Minaee et al. provided an extensive review of LLMs in their work [20]. Ding et al. [11] conducted a comprehensive study on the potential of using LLMs for data augmentation in NLP. Ahlback et al. [1] investigated various data augmentation methods, including pre-trained LLMs like Vicuna [8] and ChatGPT, along with a word substitution technique. They found modest improvements in fake news classification performance but no statistical significance. In our earlier research [7], we explored various multilingual models and training scenarios for detecting multilingual fake news. Building on this foundation, the current research aims to address the challenge of insufficiently sized annotated fake news datasets by augmenting them with synthetic news generated by LLMs. We introduce a novel approach using Llama 3 for data augmentation, with the goal of training better-performing detection models.

⁵ <https://openai.com/index/chatgpt/>.

3 Methodology

To address the challenge of a limited supply of labeled fake news datasets for effective training of detection models, our research endeavors to augment multilingual datasets with additional synthetic news generated by large language models (LLMs). A schematic overview of our approach is provided in Fig. 1. First, all news articles will be translated into English to ensure a common environment. We will use a specific prompt on Llama 3 to generate N new samples per news item, retaining the original labels. To determine the optimal dataset augmentation settings various augmentation strategies, including different rates (using random or similarity-based subsampling) and selective class augmentation, will be tested. After augmentation, we will train BERT-based classifiers. Further details are elaborated in the subsequent subsections.

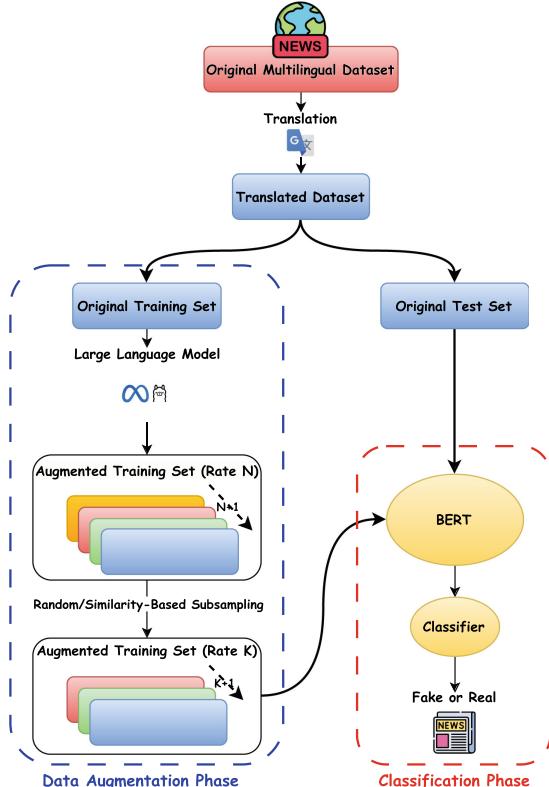


Fig. 1. A schematic overview of the proposed approach for enhancing fake news detection using LLM-based data augmentation

3.1 Data Augmentation with Llama 3

Large Language Models (LLMs) have demonstrated significant potential, excelling in complex reasoning tasks that require expert knowledge across various fields. Llama 3, developed by META GenAI, is a family of open-source pre-trained and fine-tuned LLMs ranging from 8 billion to 70 billion parameters, outperforming many existing models. For our task, we use the 8B instruction-tuned variant optimized for dialogue and chat use cases. Llama 3 builds on the strengths of its predecessors, Llama 1 [23] and Llama 2 [24]. To augment datasets with Llama 3, we use a carefully crafted prompt to generate paraphrased versions of each news item, retaining the original meaning and label. We employed prompt engineering and tested various structures to finalize the prompt used for generation. An example of the news generation is illustrated in Fig. 2. Further details on the selected prompt are provided in Sect. 4.2. We will explore different augmentation strategies to find the best settings for our fake news detection task, including varying augmentation rates (using random or similarity-based subsampling) and selectively augmenting specific classes (only fake, only real, or both).

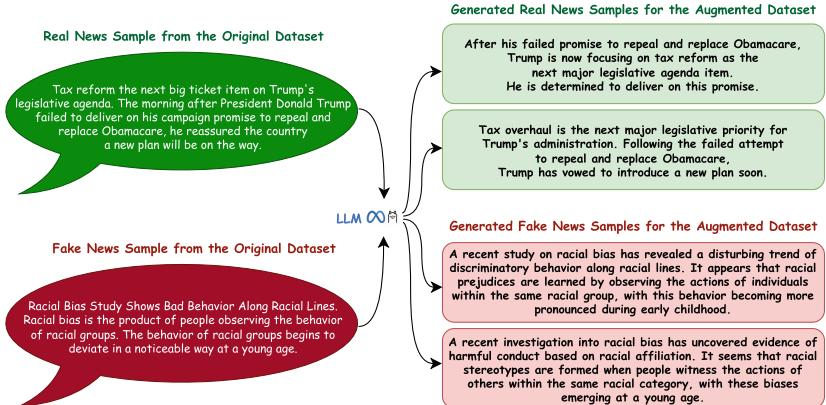


Fig. 2. An example of LLM-based synthetic news generation for dataset augmentation

Different Augmentation Rates. The augmentation rate refers to the number of generated news samples per original news item. We aim to identify an optimal rate for the fake news dataset to achieve the best outcome without risking overfitting or introducing excessive noise. For this purpose, first, we generate N samples for each news item and then select K ($K \leq N$) subsamples, either randomly or based on similarity to form the K rated augmented dataset. The similarity-based approach uses Sentence-BERT (SBERT) [21] to obtain embeddings for generated samples and calculates their cosine similarity to the original news item's embedding to identify the most similar samples.

Different Classes to be Augmented. Another crucial factor that could impact the results is the choice of classes for augmentation. Specifically, given that real-world datasets are often imbalanced, prioritizing the augmentation of certain classes, such as fake news, may be advantageous. We will evaluate the effects of three different augmentation strategies: augmenting both classes, augmenting only the fake class, and augmenting only the real class, to understand how these class-specific approaches impact detection accuracy.

3.2 BERT-Based Classification

After acquiring the augmented datasets, we utilize BERT (Bidirectional Encoder Representations from Transformers) [10] to generate text embeddings. BERT, pre-trained on large corpora with masked language modeling and next-sentence prediction, produces contextual embeddings through multiple transformer layers. A linear classification layer then maps these embeddings to binary labels (fake or real). The model is trained to minimize classification loss, enhancing its ability to accurately classify news as fake or real.

4 Experiments

In this section, we detail the datasets, experimental settings, and the results of our study. We first introduce the two multilingual datasets and experimental setup, then explore the key findings and the impact of different strategies.

4.1 Dataset

In our study, we leverage two multilingual datasets: TALLIP⁶ [9] and MM-COVID⁷ [18]. The TALLIP dataset encompasses five languages: English (en), Hindi (hi), Indonesian (id), Swahili (sw), and Vietnamese (vi). It includes articles from various domains, including business, education, technology, entertainment, politics, sports, and celebrity news. This dataset maintains a balance between the fake and real classes, with 490 fake and 490 real news articles for each language, yielding a total of 2,450 articles for each class. MM-COVID comprises multilingual COVID-19-related news; we focus on 6 languages, English (en), Spanish (es), French (fr), Hindi (hi), Italian (it), and Portuguese (pt). The statistics of the MM-COVID dataset used in our experiments can be seen in Table 1. English and Hindi are the two languages shared by both datasets.

4.2 Experimental Setting

We conducted our experiments using the computing resources available through Google Colaboratory. Our models are developed using PyTorch and the Hugging

⁶ <https://github.com/Arko98/TALLIP-FakeNews-Dataset>.

⁷ <https://github.com/bigheiniu/MM-COVID>.

Table 1. MM-COVID dataset statistics

Dataset	Language	Label	
		Fake	Real
MM-COVID	English (en)	2046	5081
	Spanish (es)	596	2271
	French (fr)	164	691
	Hindi (hi)	355	1205
	Italian (it)	107	938
	Portuguese (pt)	313	638
	Total	3581	10824

Face Transformers libraries. For translation, we employed Google Translate. For Llama 3, we utilize a directly quantized 4bit 8B instruction tuned model⁸ optimized for dialogue use cases. We configure the `max_new_tokens` parameter to 2000, with the temperature set to 0.6. For classification tasks, we employ the base uncased version of the BERT model. For text embedding generation before calculating the cosine similarity, we employ a SentenceTransformers [21] model⁹ that maps sentences and paragraphs to a dense 768-dimensional vector space. We conducted training for our models over 10 epochs. The learning rate was configured to 2×10^{-5} , and the batch size was set to 28. During testing, 40% of each dataset was randomly selected, while the remaining 60% was allocated for training purposes. To augment the datasets using Llama 3, we employed a specific prompt to generate five paraphrased versions of news texts while preserving their original meanings and labels. The prompt is provided in the following box. In practice, each news text was substituted for `statement` in the template. An example of the paraphrased news generated using this prompt is shown in Fig. 2.

Paraphrase the following text delimited by triple backquotes in 5 different ways. The generated texts must retain the exact same meaning as the original text. Enclose each paraphrased text within the following keyword pair: [BEGINPARA] ... [ENDPARA]. Only return the texts and no extra explanation.

```
```{statement}```
```

NUMBERED GENERATED TEXTS:

### 4.3 Results

In this section, we will analyze the results of our experiments. The F1 results based on different augmentation classes and rates (with random subsampling) for the TALLIP and MM-COVID datasets using BERT-based classifiers are presented in Table 2. The top two F1 scores for each language are bolded in with

<sup>8</sup> <https://huggingface.co/unslloth/llama-3-8b-Instruct-bnb-4bit>.

<sup>9</sup> <https://huggingface.co/sentence-transformers/all-mnlp-base-v2>.

the largest score in each column underlined. We will delve into the details of the impacts of different augmentation strategies in the following subsections.

**Table 2.** F1 scores for BERT-based classifiers on augmented datasets with varying augmentation rates (random sampling) and classes. (Aug.: Augmentation)

Aug.Mode	Aug.Rate	TALLIP						MM-COVID						
		vi	sw	id	hi	en	All	en	es	fr	hi	it	pt	All
<b>None</b>	<b>0</b>	0.693	0.686	0.675	0.662	0.681	0.799	0.940	0.949	<b>0.961</b>	0.936	0.889	0.990	0.947
<b>OnlyFake</b>	<b>1</b>	<b>0.725</b>	<b>0.704</b>	<b>0.713</b>	0.668	0.725	<b>0.801</b>	<b>0.951</b>	<b>0.964</b>	0.909	0.932	<b>0.892</b>	<b>0.993</b>	<b>0.953</b>
	<b>3</b>	<b>0.703</b>	0.660	<b>0.717</b>	<b>0.706</b>	<b>0.758</b>	<b>0.801</b>	<b>0.950</b>	0.958	0.940	<b>0.947</b>	<b>0.892</b>	<b>0.997</b>	<b>0.954</b>
	<b>5</b>	0.679	0.672	0.691	<b>0.690</b>	<b>0.739</b>	0.799	0.946	<b>0.962</b>	0.932	0.943	0.865	0.987	0.951
<b>OnlyReal</b>	<b>1</b>	0.634	0.665	0.665	0.647	0.709	0.774	0.940	0.919	0.912	0.938	0.812	0.944	0.929
	<b>3</b>	0.646	<b>0.690</b>	0.679	0.659	0.724	0.772	0.930	0.909	0.938	0.925	0.812	0.937	0.919
	<b>5</b>	0.692	0.682	0.676	0.623	0.698	0.773	0.926	0.884	0.921	0.924	0.794	0.948	0.920
<b>Fake + Real</b>	<b>1</b>	0.692	0.668	0.662	0.641	0.688	0.780	0.942	0.955	0.953	0.931	0.829	0.951	0.941
	<b>3</b>	0.665	0.649	0.649	0.614	0.673	0.746	0.942	0.949	0.953	<b>0.951</b>	0.829	0.955	0.938
	<b>5</b>	0.644	0.641	0.651	0.618	0.665	0.752	0.931	0.947	0.946	0.943	0.861	0.955	0.936

**The Impact of Class-Specific Augmentation.** To explore class-specific augmentation effects, we augmented the datasets using three different modes, augmenting only fake, only real, or both fake and real. In the TALLIP dataset, augmenting with only fake news consistently yields the highest F1 scores with English, Hindi, Indonesian, Vietnamese, and Swahili scores improving by 7.7%, 4.4%, 4.2%, 3.2%, and 1.8% respectively. Augmenting with only real news produced mixed results, with some improvements observed but less consistent. Augmenting with both real and fake news generally led to a reduction in performance compared to the baseline (no augmentation). Similarly, in the MM-COVID dataset, “only fake” class augmentation exhibits the most significant improvements. The “only real” mode results in decreased F1 scores across all languages, indicating that this approach is not effective on this dataset. The “Fake + Real” mode shows mixed results.

**The Effects of Augmentation Rates.** As outlined earlier, we aim to determine the optimal augmentation rate by training the classifiers with datasets having varying numbers of augmented samples. We investigate rates 1, 3, and 5. In both datasets lower augmentation rates generally improve model performance, but as the rate increases, performance tends to decline, suggesting potential overfitting and noise introduction. Generally, rate 1 yields the highest scores, rate 3 shows mixed effects, and rate 5 leads to decreased performance. For instance, in TALLIP, the Vietnamese F1 scores dropped from 0.725 at rate 1 to 0.703 at rate 3 and further decreased to 0.679 at rate 5.

**Random vs. Similarity-Based Selection.** The impact of two subsampling methods on rate 1 augmentation is exhibited in Table 3. The results indicate that similarity-based selection generally leads to higher F1 scores when augmenting both real and fake news samples. However, the difference is less pronounced when only real or fake news is augmented. This limited performance gain may be attributed to the relatively consistent text generated by the Llama 3 model under our specified settings.

**Table 3.** F1 scores of BERT-based classifiers on rate 1 augmented datasets: selection methods comparison. (Aug.: Augmentation, Sel.: Selection, R: Random, S: Similarity)

Aug.Mode	Sel.Mode	TALLIP						MM-COVID						
		vi	sw	id	hi	en	All	en	es	fr	hi	it	pt	All
<b>None</b>	-	0.693	0.686	0.675	0.662	0.681	0.799	0.940	0.949	0.961	0.936	0.889	0.990	0.947
<b>OnlyFake</b>	<b>R</b>	0.725	0.704	0.713	0.668	0.725	0.801	0.951	0.964	0.909	0.932	0.892	0.993	0.953
	<b>S</b>	0.719	0.717	0.715	0.695	0.721	0.789	0.945	0.971	0.925	0.935	0.883	0.993	0.956
<b>OnlyReal</b>	<b>R</b>	0.634	0.665	0.665	0.647	0.709	0.774	0.940	0.919	0.912	0.938	0.812	0.944	0.929
	<b>S</b>	0.642	0.672	0.667	0.661	0.700	0.773	0.927	0.912	0.912	0.945	0.806	0.944	0.928
<b>Fake+Real</b>	<b>R</b>	0.692	0.668	0.662	0.641	0.688	0.780	0.942	0.955	0.953	0.931	0.829	0.951	0.941
	<b>S</b>	0.706	0.670	0.686	0.653	0.680	0.782	0.952	0.956	0.953	0.949	0.861	0.962	0.940

## 5 Limitations and Future Work

One notable issue with large language models (LLMs) is their proneness to produce hallucinations- content that is nonsensical or unfaithful to the source. This can result in irrelevant or unexpected answers when augmenting data, lowering the quality of our augmented datasets and introducing noise. Additionally, similar prompts may yield vastly different outputs, and generated news may undergo changes that alter their intended labels. Grammatical or punctuation errors in the original news may also mislead the model during generation. In our work, we sought to address these issues through iterative testing and experimentation with various prompts, along with randomly inspecting the generated outputs to ensure optimal results. However, this process is time-consuming, and there remains significant room for further improvement. Another limitation is the substantial computational resources required by LLMs. These resources may not always be available, necessitating adjustments to parameters. For instance, shortening the generated output’s sequence length can improve generation speed but may also result in cut-off or low-quality augmented data, particularly for longer news articles.

Future work could involve experimenting with different LLMs and settings to improve performance and reduce hallucinations, leading to higher-quality augmented datasets. Employing more advanced prompting techniques can help minimize irrelevance and ensure that the generated content aligns with the intended

labels, improving the training data quality. This enhancement will lead to more robust detection models that effectively combat fake news in a timely manner. Additionally, leveraging multilingual models offers noteworthy benefits, as they can capture the subtle nuances of different languages, unlike translation-based approaches that may lose valuable language-specific features.

## 6 Conclusion

In this study, we addressed the challenge of limited labeled datasets for fake news detection by augmenting the datasets using the capabilities of large language models (LLMs), specifically the open-source Llama 3 model. Our approach involved experimenting with various Llama 3-based augmentation strategies to identify the most effective settings for enhancing the dataset while minimizing noise and the risk of overfitting. We explored various augmentation rates (1, 3, and 5), the impact of random vs. similarity-based subsampling, and the effects of augmenting specific classes (only fake, only real, or both).

Our results demonstrated the effectiveness of data augmentation in training a more robust model and improving performance compared to the baseline, which did not involve data augmentation. However, it is important to note that excessive augmentation in most cases led to negative effects, resulting in reduced performance or inconsistent outcomes. Our findings highlighted that the “only fake” augmentation mode consistently outperforms the others, and lower augmentation rates generally provide the best balance between introducing useful variability and avoiding overfitting and noise. Notably, augmenting only the fake class in the TALLIP dataset significantly boosted F1 scores, with increases of 7.7% for English and 4.4% for Hindi. In the MM-COVID dataset, this approach resulted in a 1.1% improvement for both languages. Conversely, higher augmentation rates tend to degrade performance, suggesting a threshold beyond which augmentation becomes detrimental to the model’s performance. Additionally, our comparison between random and similarity-based sampling showed that similarity-based selection often yields higher F1 scores, though the difference is less notable when augmenting only one class. Overall, employing LLMs for augmenting fake news datasets demonstrated significant potential, highlighting the importance of leveraging state-of-the-art methods to address the challenges of limited labeled data and enhance the performance of fake news detection models.

## References

1. Ahlbäck, E., Dougly, M.: Can large language models enhance fake news detection?: improving fake news detection with data augmentation (2023)
2. Aïmeur, E., Amri, S., Brassard, G.: Fake news, disinformation and misinformation in social media: a review. *Soc. Netw. Anal. Min.* **13**(1), 30 (2023)
3. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. *J. Econ. Perspect.* **31**(2), 211–236 (2017)

4. Amjad, M., Sidorov, G., Zhila, A.: Data augmentation using machine translation for fake news detection in the Urdu language. In: Proceedings of the Twelfth Language Resources and Evaluation Conference, pp. 2537–2542 (2020)
5. Ashraf, N., Butt, S., Sidorov, G., Gelbukh, A.F.: CIC at CheckThat! 2021: fake news detection using machine learning and data augmentation. In: CLEF (Working Notes), pp. 446–454 (2021)
6. Bayer, M., Kaufhold, M.A., Reuter, C.: A survey on data augmentation for text classification. ACM Comput. Surv. **55**(7), 1–39 (2022)
7. Chalehchaleh, R., Farahbakhsh, R., Crespi, N.: Multilingual fake news detection: a study on various models and training scenarios. In: Intelligent Systems Conference, pp. 73–89. Springer (2024). [https://doi.org/10.1007/978-3-031-66428-1\\_5](https://doi.org/10.1007/978-3-031-66428-1_5)
8. 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/>
9. De, A., Bandyopadhyay, D., Gain, B., Ekbal, A.: A transformer-based approach to multilingual fake news detection in low-resource languages. Trans. Asian Low-Resource Lang. Inf. Process. **21**(1), 1–20 (2021)
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)
11. Ding, B., et al.: Data augmentation using LLMS: data perspectives, learning paradigms and challenges. arXiv preprint [arXiv:2403.02990](https://arxiv.org/abs/2403.02990) (2024)
12. D’Ulizia, A., Caschera, M.C., Ferri, F., Grifoni, P.: Fake news detection: a survey of evaluation datasets. PeerJ Comput. Sci. **7**, e518 (2021)
13. Hamed, S.K., Ab Aziz, M.J., Yaakub, M.R.: A review of fake news detection approaches: a critical analysis of relevant studies and highlighting key challenges associated with the dataset, feature representation, and data fusion. Heliyon (2023)
14. Hua, J., Cui, X., Li, X., Tang, K., Zhu, P.: Multimodal fake news detection through data augmentation-based contrastive learning. Appl. Soft Comput. **136**, 110,125 (2023)
15. Júnior, W.O., da Cruz, M.S., Wyzykowski, A.B.V., de Jesus, A.B.: The use of data augmentation as a technique for improving neural network accuracy in detecting fake news about Covid-19. arXiv preprint [arXiv:2205.00452](https://arxiv.org/abs/2205.00452) (2022)
16. Kapusta, J., Držík, D., Šteflovič, K., Nagy, K.S.: Text data augmentation techniques for word embeddings in fake news classification. IEEE Access **12**, 31538–31550 (2024)
17. Keya, A.J., Wadud, M.A.H., Mridha, M., Alatiyyah, M., Hamid, M.A.: AugFake-BERT: handling imbalance through augmentation of fake news using BERT to enhance the performance of fake news classification. Appl. Sci. **12**(17), 8398 (2022)
18. Li, Y., Jiang, B., Shu, K., Liu, H.: MM-Covid: a multilingual and multimodal data repository for combating Covid-19 disinformation. arXiv preprint [arXiv:2011.04088](https://arxiv.org/abs/2011.04088) (2020)
19. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692) (2019)
20. Minaee, S., et al.: Large language models: a survey. arXiv preprint [arXiv:2402.06196](https://arxiv.org/abs/2402.06196) (2024)
21. Reimers, N., Gurevych, I.: Sentence-BERT: sentence embeddings using siamese bert-networks. arXiv preprint [arXiv:1908.10084](https://arxiv.org/abs/1908.10084) (2019)
22. Shanahan, M.: Talking about large language models. Commun. ACM **67**(2), 68–79 (2024)
23. Touvron, H., et al.: Llama: open and efficient foundation language models. arXiv preprint [arXiv:2302.13971](https://arxiv.org/abs/2302.13971) (2023)

24. Touvron, H., et al.: Llama 2: open foundation and fine-tuned chat models. arXiv preprint [arXiv:2307.09288](https://arxiv.org/abs/2307.09288) (2023)
25. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
26. Yang, J., et al.: Harnessing the power of LLMS in practice: a survey on ChatGPT and beyond. ACM Trans. Knowl. Discov. Data **18**(6), 1–32 (2024)
27. Zhao, W.X., et al et al.: A survey of large language models. arXiv preprint [arXiv:2303.18223](https://arxiv.org/abs/2303.18223) (2023)



# Anomalous Channel Detection for YouTube Through Label Propagation

Ridwan Amure<sup>1</sup> and Nitin Agarwal<sup>1,2(✉)</sup>

<sup>1</sup> COSMOS Research Center, UA - Little Rock, Arkansas, USA  
[{raamure,nxagarwal}@ualr.edu](mailto:{raamure,nxagarwal}@ualr.edu)

<sup>2</sup> International Computer Science Institute, University of California, Berkeley, USA

**Abstract.** The prevalence of anomalous behavior on social media platforms like YouTube disrupts user experience and undermines trust. Identifying these patterns can be challenging, particularly when analyzing commenter activity on YouTube channels. We propose a semi-supervised model called Dual Similarity-based Propagation Scoring (DSPS) to identify channels with anomalous behavior on YouTube by leveraging association between channels across a shared co-commenter network. The co-commenter network represents interactions between users who comment together, providing a magnifying lens into the connectivity between users who interact with videos across channels. We evaluated our approach and model on data from 97 channels with over 640,000 videos, 12 million commenters, and 123 million comments. Additionally, we created synthetic data representing real-world situations for a rigorous evaluation. The model achieved an F1 score of 0.92 on the synthetic graph. The findings show that analyzing the affinities between suspended and active channels can reveal previously unidentified features useful for characterizing anomalous behavior. This research contributes to the broader effort to ensure genuine and reliable user interaction on digital platforms.

**Keywords:** YouTube anomaly detection · Anomalous behavior · Label Propagation · Co-commenter Network

## 1 Introduction

YouTube, a popular video-sharing platform, has become a hub for abusive behavior that threatens user interaction authenticity. This can range from artificially manipulating engagement metrics to engaging in fraudulent methods, distorting a channel's perceived popularity and legitimacy. Anomalous behaviors on YouTube are marked by deviations from expected patterns and often violate YouTube's rules and terms of service. These actions usually manifest through sudden or unusual patterns in commenter interaction. The platform's open nature and wide audience have led to concerns about the authenticity of user interactions and the overall trustworthiness of the platform.

This research aims to identify anomalous patterns in YouTube channels by examining the relationships between suspended and active channels through a

shared co-commenter network. The co-commenter network is defined as a network of commenters who commented together on at least five videos in a channel [13]. The term “anomalous” refers to commenters manipulating engagement metrics to present a misleading image of popularity and activity. This behavior has broader implications for the platform’s ecosystem, influencing content creators and consumers. Our study examined a set of 97 YouTube channels reporting on events related to the Indo-Pacific region, which are often linked to dishonest behavior due to controversial subjects and content. The data comprises over 640,000 videos, 12 million commenters, and 123 million comments. Additionally, for a systematic evaluation, we created synthetic data based on the characteristics of the real-world data. The model achieved an F1 score of 0.92 on the synthetic graph.

The findings show that analyzing the affinities between suspended and active channels can reveal previously unidentified features useful for characterizing anomalous behavior. This research contributes to maintaining the integrity of online platforms, ensuring genuine user engagement and authentic interactions. Next, we will go ahead and present the related work.

## 2 Related Work

Identifying anomalous behavior on digital platforms, especially YouTube, has attracted substantial interest lately because of the platform’s enormous impact and scope. Previous studies in this field have mostly concentrated on detecting patterns of deceptive engagement. An example of such scientific work is Kirdemir[9], where the authors studied anomalous behaviors in YouTube channels using their engagement metrics. Hussain [6] and Benevenuto [2] both highlighted the presence of disinformation, propaganda, and self-promotion on YouTube.

Kaushal [7] focused on the presence of child-unsafe content and its promoters. The authors used a Convolutional Neural Network model on video frames to detect unsafe content for children with an accuracy of 85.7%. Using a network of unsafe content promoters and other engagement metrics, Kaushal [7] found that unsafe content is usually very close to safe content. Similarly Akinnuni et al. [1] showed that networks can be used to identify unscrupulous actors on social media. This emphasizes the use of a network to analyze anomalous behavior on the YouTube platform. Similarly, Papadamou et al. [12] discovered that inappropriate content aimed at young audiences can bypass YouTube’s filtering systems by closely imitating legitimate content.

Galeano [3] emphasized the significant role commenters play in enhancing channel influence and shaping public perceptions, particularly in the dissemination of disinformation. Similarly, Khan[8] identified various motives behind user engagement, highlighting that social interaction is a strong predictor of commenting behavior.

The concept of using co-commenter networks to detect anomalous behavior is relatively novel. Co-commenter networks represent the interactions between

users commenting on the same videos, providing a richer context for understanding engagement patterns. Our study expands on these pioneering investigations by examining the relationship between suspended and active YouTube channels. Our analysis is based on the shared co-commenter network, enabling us to find hidden connections and abnormalities. The major contributions introduced in this paper are listed below.

- We propose a semi-supervised Dual Similarity-Based Propagation Scoring (DSPS) model for identifying YouTube channels with anomalous behavior using a co-commenter network.
- We develop a semi-supervised model for propagating labels from anomalous channels to associated channels and create a synthetic dataset for the quantitative evaluation of the model.

### 3 Problem Statement

We formulate the problem of finding anomalous channels on YouTube in this section. A YouTube channel can be said to exhibit anomalous behavior based on commenter activities and user engagement [9]. Therefore, we propose the following theory for calculating anomalous scores for active YouTube channels (an active YouTube channel is a channel that has not been suspended or deactivated by YouTube). Given a graph  $G = (V, E)$  where  $V$  consist of channel nodes ( $C_a$ ) and commenter nodes ( $C_o$ ), then the edges  $E$  are defined as follows:

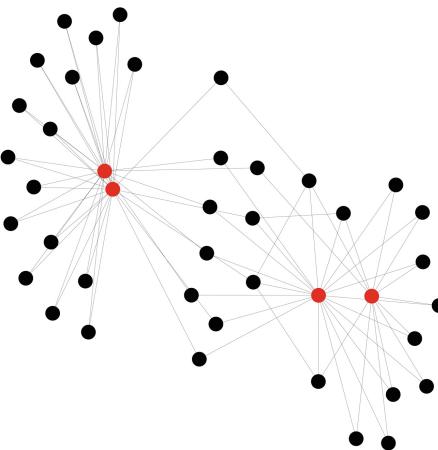
- $E_{ij}$  exists  $\forall i, j \in V$  if  $i$  and  $j$  are both commenter nodes and  $i$  and  $j$  have commented together on five or more videos in the same channel.
- $E_{ij}$  exists  $\forall i, j \in V$  if  $i$  is a commenter node,  $j$  is a channel node and  $i$  has commented on a video in channel  $j$ .

Given  $L \subset C_a$  consisting of suspended channels on YouTube, it is possible to assign an anomalous score to active channels  $L^i = C_a - L$  in the network by traversing through the shared co-commenter network.

We provide evidence for our theory by considering if a random node  $v \in L$  can be reached by starting from another random node  $u \in L$ . From the co-commenter network of suspended channels from YouTube, we found that the suspended channels shared similar commenters, and each suspended YouTube channel could be accessed by following a path in the commenter network. We show an example graph in Fig. 1 containing suspended channels (red nodes) and their shared commenters (black nodes). Figure 1 shows a suspended channel can be reached from another random suspended channel in a single hop through the co-commenter network.

### 4 Method

This section describes the data collection and approach used in calculating the anomalous score of active YouTube channels. This section is subdivided as follows:



**Fig. 1.** Graph of suspended channel nodes (red) and common commenters node (black) showing few hops between suspended channel nodes.

- Section 4.1 describes the dataset used in the research.
- Section 4.2 describes the method used in calculating the anomalous scores for the YouTube channels.

#### 4.1 Data

In this study, we used a data extraction tool [10] to gather information from YouTube. This tool relies on YouTube’s system for accessing data (API) [4]. We collected daily uploads, the number of comments on those videos, and details about those comments (e.g., text contents, commenters, video details, and timestamps). We gathered data from 97 channels—YouTube had suspended seven for anomalous behaviors—including information on over 640,000 videos, more than 12 million commenters, and over 123 million comments. The data was collected using various keywords that were identified by studying coverage in the Indo-Pacific region with further reviews to improve the inclusiveness of the keywords. “Komunis Cina China | pengaruh Indonesia”, “Muhammadiyah Cina | China | Tiongkok | Tionghoa”, “Kejam Uighur | Uyghur”, “Muslim Brother | Indonesia Uighur | Uyghur”, etc., are examples of keywords used in the data collection. The commenter’s nodes were connected if they had more than five comments among videos in the same channel. The threshold (five comments) is enough to establish the edge between the commenters [13], and each commenter’s node was connected to at least one channel node. In the subsequent sections, the graph created from the extracted data is referred to as the actual graph, or **AG**.

To augment the lack of ground truth in this study, we created a synthetic graph dataset to imitate a social media network. Experimentation with SG data allows us to study the characteristics of the developed model. Furthermore, due

to a lack of a baseline approach (at the time of writing the paper) for a comparative evaluation of our developed model, SG data allows us to identify the limitations of our approach and refine it further. The synthetic graph, or **SG**, was created as a pseudo-social network. To achieve this, the commenter's node and edges were generated using the Watts-Strogatz model [14] with the probability of rewiring each edge as 0.4, and each node was joined with its 10 nearest neighbours in a ring topology. The graph consisted of 100,000 commenters and 1,000 channels. The characteristics of the two graphs are shown in Table 1. We identified 200 channels within the network that displayed structural characteristics similar to those of suspended nodes in graph AG, and consequently labeled them as suspended nodes in our synthetic graph as shown in Algorithm 1. By using this model, we created a graph with a small-world property that is close to a real-life representation. The labeled datasets were then partitioned into training and testing sets, comprising 70% and 30% of the data, respectively.

---

**Algorithm 1.** Labeling algorithm for channels node in graph SG

---

```

1: Input:
2: channels_graph list of graphs for all channels
3: check_graph function that checks if the graph meets the graph statistic requirements
4:
5: Label channels nodes:
6: for each graph in channels_graph do
7: if check_graph(graph) then
8: label \leftarrow 1
9: end if
10: end for

```

---

**Table 1.** Distributions of nodes in the graphs used in this research

Graph	No. of commenter	No. of channels	No. of labeled channels
AG	31179	97	7
SG	100000	1000	200

## 4.2 Calculating the Anomalous Score

Our method of calculating anomalous scores for YouTube channels was based on affinity with suspended channels. We observed that affinity can be measured by estimating the distance or similarities between an active channel and any other suspended channels. Given a graph  $G$  with commenter's nodes ( $C_o$ ) and

channel nodes ( $C_a$ ) described in Sect. 3. The approach used in calculating the anomalous score is described in the subsequent sections. First, we extract the node-level features using the Node2Vec algorithm [5], which involves the major steps described below.

Given a graph  $G = (V, E)$ , where  $V$  is the set of nodes and  $E$  is the set of edges. The neighborhood of  $v \in V$  represented as  $N_s \subset V$  can be a breadth-first (BF) neighborhood comprised of nodes that are one step away or a depth-first (DF) neighborhood comprised of nodes that are further away. The first step in the Node2Vec algorithm involves performing a random walk. At each step, the random walker reaches a neighboring node based on a biased transition probability that is necessary for optimizing the BF and DF search. After generating the random walk, a skip-gram model was trained to learn the embedding for each node. The skip-gram objective was to maximize the average log probability of predicting the neighborhood  $N_s(v)$  for a node  $v$  conditioned on its embedding given by  $f$ . This can be represented as

$$\max_f \sum_{v \in V} \log P(N_s(v) | f(v)) \quad (1)$$

The embedding vector representation  $f$  for each node can be obtained from the trained skip-gram model.

We defined the DSPS model for estimating the anomalous score based on the distance factors in Eq. 2,

$$S(n) = \frac{k}{dx} \quad (2)$$

where  $k$  is a scaling factor and the  $dx$  represents the distance factors. We define the distance factor as a function of cosine distance ( $s_i$ ) between the active and the suspended channel nodes and the path described by the commenter's node ( $s_2$ ), hence  $dx = \mathcal{F}(s_1, s_2)$ . The first distance factor ( $s_1$ ) measures the similarity between the active and suspended channel nodes in terms of the distance between their vectors, while the second distance factor ( $s_2$ ) is a measure of the spatial distance between the channels. In both cases, lower values mean the two channels under comparison are similar. So, given the feature vector  $f_v \in \mathbb{R}^{64*1}$  for each node, we define the first distance factor ( $s_1$ ) as the cosine distance [11] between an active channel node  $v_i$  and a suspended channel  $v_j$  as

$$s_1(v_i, v_j) = 1 - \frac{f_{v_i} \cdot f_{v_j}}{\|f_{v_i}\| \cdot \|f_{v_j}\|} \quad (3)$$

Given  $P \leftarrow$  all paths between nodes  $v_i$  and  $v_2$  Then we define a second distance factor ( $s_2$ ) as follows

$$s_2(v_i, v_j) = \frac{\sum_p^P |p|}{|P|} \quad (4)$$

In Eq. 4, the numerator can be regarded as the number of commenter nodes common to channels  $v_i$  and  $v_j$ , and the denominator represents the number of

unique paths between  $v_i$  and  $v_j$ . We define the scaling factor  $k$  as the number of suspended channel nodes that can be reached from an active channel node  $i$ . Given suspended node channels  $L$ , the anomalous score  $S(i)$  for an active channel  $i$  can be rewritten as

$$S(i) = \frac{k}{\sum_{i \notin L, j \in L} s_1(i, j) \cdot s_2(i, j)} \quad (5)$$

We then normalized the score to a 0-1 range and converted it into classes using a 0.5 threshold. Channels with a score of 0.5 or higher were classified as class 1, while those with a score below 0.5 were classified as class 0.

## 5 Results

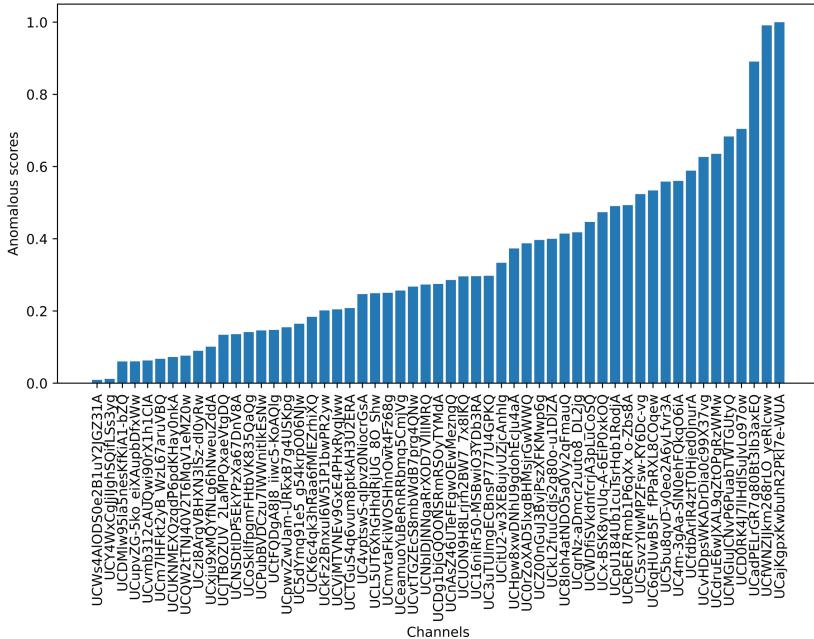
In this section, we present the results of applying the methods described in Sect. 4 on the different datasets. The rest of this section is subdivided as follows:

- Section 5.1 details the results from the application of the algorithm on graph AG.
- Section 5.2 details the performance of the algorithm when applied to the synthetic graph SG.

### 5.1 Anomalous Detection Algorithm Results on Graph AG

In Fig. 2, most channels have anomalous scores less than or equal to 0.5. These channels can be classified as channels with less anomalous behavior. Meanwhile, the channels with anomalous scores greater than 0.5 were classified as anomalous. Afterwards, examples of channels in this category were examined closely. Starting with the highly suspicious category, Fig. 3(a) shows a channel with a high anomalous score. The graph shows a close affinity to four suspended channels used in this study. There is an average of one commenter node (black) between the active channel (green) and the suspended channel (red) nodes, which indicates a close proximity or relationship with the channels that exhibit anomalous behavior. Also, multiple paths connect this channel with the suspended nodes.

In contrast to Fig. 3(b), which exhibits a low anomalous score, Fig. 3(a) presents an average of five commenter nodes separating active (green) and suspended (red) channel nodes. To further elucidate this disparity, Fig. 3(c) introduces an active channel node with a comparable hop distance to suspended channels as the node in Fig. 3(a). Despite this similarity in hop distance, the anomalous score for the channel in Fig. 3(c) is lower (0.47) than that of Fig. 3(a). This discrepancy in scores was attributed to the combined influence of both distance factors. The two distance factors, similarity and average path length, measure the channel relationship and the shared interest between the users who engage with their videos. The channels UCfWNZIJkm268rLO-yeRlcww, UCad-PELrGR7q80Bt3Ib3axEQ, and UCajKgpxKwbuH2PkI7e-WUA were the top



**Fig. 2.** Channels and their UCfWNZIJkm268rLO\_yeRlcww, UCajKgpXKwbuhR2PkI7eWUA

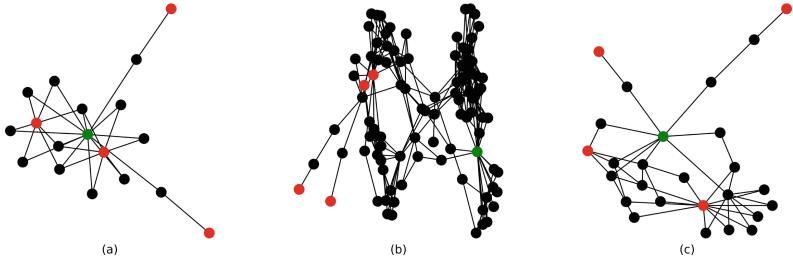
Anomalous score. The top channels are UCadPELrGR7q80Bt3Ib3axEQ, and

channels with the highest anomalous score. Our analysis identified YouTube channels promoting militaristic content, unverified economic reports, clickbait, and misleading information as the most prominent source of anomalies. Notably, the channel “US Military Power” (UCajKgpXKwbuhR2PkI7eWUA) appears on a previously established list of misleading and clickbait channels<sup>1</sup>. This list, containing YouTube channels exhibiting anomalous behavior, serves as a form of pseudo-ground truth, further validating the methodology and outcomes presented in this study.

## 5.2 Anomalous Detection Algorithm Results on Graph SG

The model showed impressive results on the synthetic graph dataset. It had an F1 score of 0.92 on the SG data. The model performance can be attributed to its ability to measure the closeness between the nodes from two directions. The combined measure used in the model makes it more robust and hence has strong performance results. Additionally, the model’s efficiency in handling diverse network structures further validates its effectiveness in real-world applications.

<sup>1</sup> <https://gist.github.com/RubenKelevra/06865c3cc2b4c1dd96faadde2aa6ab43>.



**Fig. 3.** Graph of channels and their shared co-commenter network. The active, suspended, and comment channels are represented with green, red, and black nodes, respectively. (a) The graph shows an active channel with id = UCajKgpxKwbuhR2PkI7e-WUA with a high anomalous score. (b) The graph shows an active channel with id = UCVjMTVNEv9GxE4PHxRyqJww with a low anomalous score. (c) The graph shows an active channel with id = UCx-BSK8ynUq-A-6EiP0xkOQ with a relatively low anomalous score.

## 6 Conclusion and Future Works

This research aims to identify anomalous patterns in YouTube channels by examining the relationships between suspended and active channels through a shared co-commenter network. The co-commenter network is defined as a network of commenters who commented together on at least five videos in a channel [13]. The idea of using co-commenter networks to discover the anomalous behavior of entities in the YouTube platform is a relatively new strategy for the domain of online engagement analysis. In this study, we further develop the research domain, using the connections between the co-commenters and channels to identify channels with anomalous behavior. We proposed a method to calculate the anomalous scores and identified the top three anomalous channels by label propagation through a shared co-commenters network. The term “anomalous” refers to commenters manipulating engagement metrics to present a misleading image of popularity and activity. This behavior has broader implications for the platform’s ecosystem, influencing content creators and consumers. Our study examined a set of 97 YouTube channels reporting on events related to the Indo-Pacific region, which are often linked to dishonest behavior due to controversial subjects and content. The data comprises over 640,000 videos, 12 million commenters, and 123 million comments. Additionally, for a systematic evaluation, we created synthetic data based on the characteristics of the real-world data. The model achieved an F1 score of 0.92 on the synthetic graph.

This shows the reliability and robustness of employing co-commenter networks for assessing anomalous behavior. By analyzing the affinity between suspended and active channels, we demonstrate that it is possible to identify channels exhibiting anomalous activity effectively. Given the absence of ground truth data and comparative studies at the time of this research, synthetic data was employed to evaluate the strengths and weaknesses of our model. The model proposed in this study can be used to identify previously undetected YouTube

channels with anomalous behavior. We will extend this study to other platforms in the future.

This study transcends its immediate findings and holds significant value for various stakeholders within the online information landscape. Users are empowered to make informed decisions about content consumption by recognizing anomalous channels that exhibit suspicious engagement patterns and disseminate unverified information, promoting misinformation or harmful narratives. This fosters a more critical online experience.

Within the research community, this study contributes to exploring anomalous behavior on online platforms. The proposed framework for identifying anomalous channels paves the way for further investigation into the motivations behind such activities and developing more sophisticated detection algorithms, ultimately benefiting all stakeholders. In future works, we will extend the range of features considered in detection beyond the co-commenter network and use more data for development and validation. Likewise, we will investigate and compare the performance in different time frames to validate the methods more definitively and better understand their long-term detection capabilities.

**Acknowledgments.** This research is funded in part by the U.S. National Science Foundation (OIA-1946391, OIA-1920920), U.S. Office of the Under Secretary of Defense for Research and Engineering (FA9550-22-1-0332), U.S. Army Research Office (W911NF-23-1-0011, W911NF-24-1-0078), U.S. Office of Naval Research (N00014-21-1-2121, N00014-21-1-2765, N00014-22-1 -2318), U.S. Air Force Research Laboratory, U.S. Defense Advanced Research Projects Agency (W31P4Q-17-C-0059), Arkansas Research Alliance, the Jerry L. Maulden/ Entergy Endowment at the University of Arkansas at Little Rock, and the Australian Department of Defense Strategic Policy Grants Program (SPGP) (award number: 2020-106-094). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations. The researchers gratefully acknowledge the support.

## References

1. Akinnubi, A., Alassad, M., Agarwal, N., Amure, R.: Identifying contextualized focal structures in multisource social networks by leveraging knowledge graphs. In: International Conference on Complex Networks and Their Applications, pp. 15–27. Springer (2023). [https://doi.org/10.1007/978-3-031-53472-0\\_2](https://doi.org/10.1007/978-3-031-53472-0_2)
2. Benevenuto, F., Duarte, F., Rodrigues, T., Almeida, V.A., Almeida, J.M., Ross, K.W.: Understanding video interactions in Youtube. In: Proceedings of the 16th ACM international conference on Multimedia, pp. 761–764. ACM, Vancouver British Columbia Canada (2008). <https://doi.org/10.1145/1459359.1459480> <https://dl.acm.org/doi/10.1145/1459359.1459480>
3. Galeano, K., Galeano, R., Agarwal, N.: An evolving (dis) information environment—How an engaging audience can spread narratives and shape perception: a trident juncture 2018 case study. Disinformation, Misinformation, and Fake News in Social Media: Emerging Research Challenges and Opportunities, pp. 253–265 (2020)

4. Google developers: API reference — Youtube data API (2021). <https://developers.google.com/youtube/v3/docs>. Visited on January 21, 2022
5. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
6. Hussain, M.N., Tokdemir, S., Agarwal, N., Al-Khateeb, S.: Analyzing disinformation and crowd manipulation tactics on YouTube. In: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 1092–1095. IEEE (2018). <https://ieeexplore.ieee.org/abstract/document/8508766/>
7. Kaushal, R., Saha, S., Bajaj, P., Kumaraguru, P.: KidsTube: Detection, characterization and analysis of child unsafe content & promoters on YouTube. In: 2016 14th Annual Conference on Privacy, Security and Trust (PST), pp. 157–164. IEEE (2016). <https://ieeexplore.ieee.org/abstract/document/7906950/>
8. Khan, M.L.: Social media engagement: what motivates user participation and consumption on Youtube? Comput. Hum. Behav. **66**, 236–247 (2017)
9. Kirdemir, B., Adeliyi, O., Agarwal, N.: Towards characterizing coordinated inauthentic behaviors on Youtube. In: ROMCIR@ ECIR, pp. 100–116 (2022). <https://ceur-ws.org/Vol-3138/paper6.jot.pdf>
10. Kready, J., Shimray, S.A., Hussain, M.N., Agarwal, N.: Youtube data collection using parallel processing. In: 2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 1119–1122. IEEE (2020)
11. Lahitani, A.R., Permanasari, A.E., Setiawan, N.A.: Cosine similarity to determine similarity measure: Study case in online essay assessment. In: 2016 4th International conference on cyber and IT service management, pp. 1–6. IEEE (2016). <https://ieeexplore.ieee.org/abstract/document/7577578/>
12. Papadamou, K., et al.: Disturbed Youtube for kids: characterizing and detecting inappropriate videos targeting young children. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 14, pp. 522–533 (2020)
13. Shajari, S., Amure, R., Agarwal, N.: Analyzing anomalous engagement and commenter behavior on youtube. In: AMCIS 2024 Proceedings (2024). <https://aisel.aisnet.org/amcis2024/socialcomp/socialcomput/6>
14. Watts, D.J., Strogatz, S.H.: Collective dynamics of ‘small-world’networks. Nature **393**(6684), 440–442 (1998)



# Decoding Political Polarization in Social Media Interactions

Giulio Pecile<sup>(✉)</sup>, Niccolò Di Marco, Matteo Cinelli, and Walter Quattrociocchi

Department of Computer Science, Sapienza University of Rome, Rome, Italy  
giulio.pecile@uniroma1.it

**Abstract.** Social media platforms significantly influence ideological divisions by enabling users to select information that aligns with their beliefs and avoid opposing viewpoints. Analyzing approximately 47 million Facebook posts, this study investigates the interactions of around 170 million users with news pages, revealing distinct patterns based on political orientations. While users generally prefer content that reflects their political biases, the extent of engagement varies even among individuals with similar ideological leanings. Specifically, political biases heavily influence commenting behaviors, particularly among users leaning towards the center-left and the right. Conversely, the ‘likes’ from center-left and centrist users are more indicative of their political affiliations. This research illuminates the complex relationship between social media behavior and political polarization, offering new insights into the manifestation of ideological divisions online.

**Keywords:** Social Media · Polarization · Selective Exposure · News Consumption

## 1 Introduction

The World Wide Web has vastly increased information accessibility, transforming how information is distributed and consumed globally. This evolution has revolutionized communication methods, erasing geographical and temporal constraints. Over recent decades, expanding social networks and shifting public discourse to digital platforms have led scholars to explore how users interact with information and form behavioral clusters, often challenging traditional expectations. It is well-documented that online users tend to engage with information that confirms their pre-existing beliefs, typically ignoring conflicting viewpoints [1, 2]. Such behaviors create echo chambers, where like-minded individuals reinforce each other’s views [3–5]. Such configurations, alongside growing polarization, manifest with varying intensity across different social media platforms [6], suggesting that the users’ base as well as platform designs and algorithms, which aim to maximize engagement, play a significant role in shaping these social dynamics.

Existing literature [7] suggests that although individuals generally prefer political news that aligns with their pre-existing beliefs, they do not entirely ignore opposing viewpoints. Conversely, Guess [8] finds that most users gravitate towards centrist news outlets despite a small, highly engaged group favoring partisan sources. These patterns indicate a notable convergence in the information available to the broader public. However, the ability of individuals to completely shield themselves from conflicting information is limited due to the minimal control they have over the content presented to them and the dynamics of information spread on social media platforms. Notably, ‘weak ties’-connections between loosely affiliated individuals like acquaintances or distant relatives-are crucial in spreading new information across social networks [9].

Notably, most content users interact with on social media is not actively sought but is delivered through a network algorithm tailored to maximize user engagement. This characteristic is pivotal as it may foster unrecognized behavioral patterns, complicating efforts to measure such effects. Due to concerns over potential manipulation by malicious actors, social networks often refrain from disclosing the specifics of these algorithms. Sunstein, in his book *Republic: Divided Democracy in the Age of Social Media* [10], discusses the *Age of the Algorithm*, where users lack control over their news consumption, inadvertently contributing to the formation of echo chambers.

Not all users exhibit polarization similarly. However, their behaviors show significant nuances. Zaller observes that politically engaged citizens are particularly receptive to messages that align with their beliefs [11]. Similarly, Taber and Lodge find that highly partisan users are more likely to embrace supporting arguments while dismissing contrary ones without question [12]. This suggests that increasing polarization complicates efforts to mitigate it, and using counterfactuals might even be counterproductive [2]. The debate over which political group is more prone to selective exposure and biased information processing remains unresolved. Some studies indicate that conservatives are more likely to engage in such behaviors [13–15], while others present contradictory findings [16–18]. Furthermore, there is no consensus on cross-party discussions; Barberá suggests that liberals are more involved in cross-party interactions [19], whereas Wu argues that conservatives are more likely to engage in such discussions [20].

In this study, we explore the phenomenon of selective exposure by analyzing how about 170 million Facebook users interact with approximately 47 million posts from news agencies with varying political leanings. We specifically measure the intensity of selective exposure and assess whether users prefer specific ideologies or news agencies. The analysis is structured around several distinct scenarios of user selectivity:

- users who are not selective at all;
- users who are selective both in terms of pages viewed and the political leaning of the pages;
- users who are selective only in terms of political leaning;
- users who are selective in terms of pages viewed.

By categorizing user behavior into these scenarios, we aim to uncover the extent and nature of selective exposure, identifying whether it is more pronounced towards particular ideologies or news providers.

We find that all users exhibit strong selectivity in terms of political leaning, a phenomenon explained mainly by a marked preference for specific pages [21, 22]. However, this preference does not fully account for the leaning-based selective exposure observed in specific user groups. These results are crucial for understanding the primary drivers of selective exposure and the resultant polarization within online communities. Our framework enables an examination of the role political affiliation plays in the selectivity exhibited by users. The structure of this paper is organized as follows: Initially, we discuss related works that explore the concepts of polarization, the echo chamber hypothesis, and their interactions with social media. We then describe the theoretical methods used in our analysis, including entropy as a proxy for measuring selective exposure and two randomization strategies to evaluate the robustness of our findings. Following this, we detail the patterns of user activity concentration and demonstrate how this concentration aligns predominantly with ideological page biases. The paper concludes by presenting evidence of leaning-driven selectivity, predominantly among users who follow pages with right and center-left biases.

## 2 Materials and Methods

### 2.1 Entropy and Selective Exposure

To fully measure the phenomenon of selective exposure defined as *a tendency for people both consciously and unconsciously to seek out material that supports their existing attitudes and opinions and to actively avoid material that challenges their views* [23], one would need the full digital trace of a user and the reasons that motivated the user to interact with one page instead of another. In its absence, we use the entropy of the interactions as a proxy. We recall that Shannon Entropy is commonly used in information theory to measure the concentration of a distribution. In our framework, users with low entropy are the most selective, while those interacting with different bias levels uniformly have high entropy.

In particular, here we also employ some of its properties that arise when considering sub-partitions.

Consider a set universe  $U$  and a partition  $\sigma = \{s^1, \dots, s^n\}$  of it. We denote  $|s^i| \equiv c_i$ . Suppose that  $\rho$  is a partition of  $\sigma$ , i.e. if for each  $r \in \rho, r \subseteq s_i \in \sigma$  for exactly one  $i$ . Then, each set of  $\rho$  can be written as:

$$\rho = \{s_1^1, \dots, s_{c_1}^1, \dots, s_1^i, \dots, s_{c_i}^i, \dots, s_1^n, \dots, s_{c_n}^n\},$$

i.e.  $s_j^i$  its the  $j$ -th subset of  $s^i$ .

Now, consider a random variable  $X_\rho$  having image in  $\rho$ . Obviously,  $X_\rho$  can be naturally extended to  $X_\sigma$  having image in  $\sigma$ . We define  $p(X_\rho \in s_j^i) \equiv p_j^i$ . It follows,  $p(X_\rho \in s^i) = p(X_\sigma \in s^i) = \sum_{j=1}^{c_i} p_j^i \equiv p^i$ . We have that:

$$\begin{aligned}
H(X_\rho) &= - \sum_i \sum_j p_j^i \log p_j^i \\
&= - \sum_i \sum_j \left( p_j^i \log(p^i) + p_j^i \log \frac{p_j^i}{p^i} \right) \\
&= - \sum_i p^i \log(p^i) - \sum_i p^i \sum_j \bar{p}_j^i \log \bar{p}_j^i,
\end{aligned}$$

where  $\bar{p}_j^i = \frac{p_j^i}{p^i}$ . In a more compact form, we have obtained:

$$H(X_\rho) = H(X_\sigma) + \sum_i p^i H(X_\rho | X_\sigma \in s^i). \quad (1)$$

In this work, we consider interactions over pages having a certain political leaning. Therefore, given the general set of spaces,  $\sigma$  will be the partition induced by their political leaning, while  $\rho$  will be simply the partition in which each page is considered alone.

Having reached Eq. (1), it is immediate to find the theoretical minimum and maximum entropy of the finer partition  $\rho$  (i.e. the pages) given the interaction over partition  $\sigma$  (i.e. the leanings).

This follows from the observation that the  $H(X_\sigma)$  and  $p^i$  terms are fixed, and the terms  $H(X_\rho | X_\sigma \in s^i)$  are all independent of one another and thus can be minimized (maximized) independently. The minimum is clearly found when  $H(X_\rho | X_\sigma \in s^i) = 0 \forall i$  i.e. when the activity of the user is concentrated in at most one page for every  $s^i$  (i.e. leaning).

On the other hand, we recall that the maximum entropy is reached by a uniform distribution. Since we consider interactions, which are discrete and often not large, it is not always possible to obtain an exact uniform distribution. Therefore, we compute the maximum possible value of entropy with the below criteria.

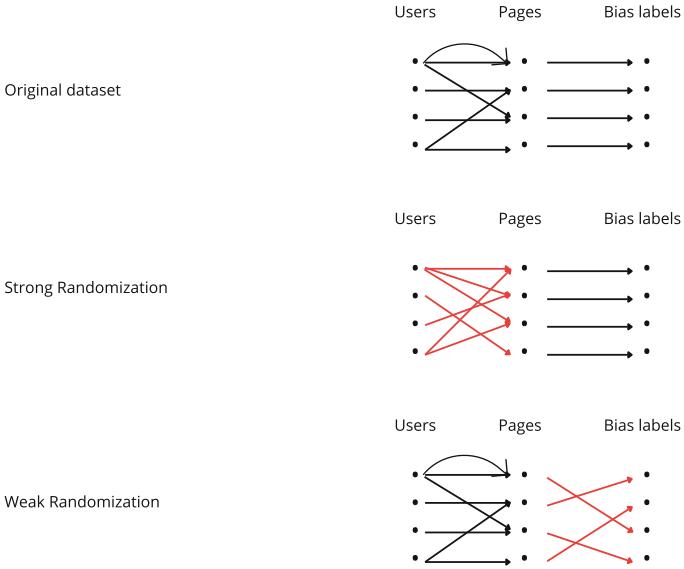
Consider a user  $u$  with  $n$  interactions in  $c_i$  pages from  $s^i$ . The maximum entropy is found when  $r$  pages receive  $q+1$  interactions and  $c_i - r$  pages receive  $q$  interactions, where  $q$  and  $r$  are the quotient and remainder of the integer division of  $\frac{n}{c_i}$ . Note that, if  $n \leq c_i$ , the maximum entropy is simply  $\log n$ .

Finally, we note that in every  $s^i$  the minimum and maximum entropies are the same only when there is only one interaction, in which case  $H(X_\rho | X_\sigma \in s^i) = 0$ .

## 2.2 Strong and Weak Randomization

In our analysis, we compare the actual interaction patterns to those resulting from two randomization processes: a stronger one, where each interaction between users and pages is randomized, and a weaker one, where the Bias labels of the pages are randomized. In both cases, the number of interactions made by each user and received by each page remains the same, and the number of pages for each bias label remains unchanged. Furthermore, we notice that in the strong randomization process, the users distribute their activity uniformly across

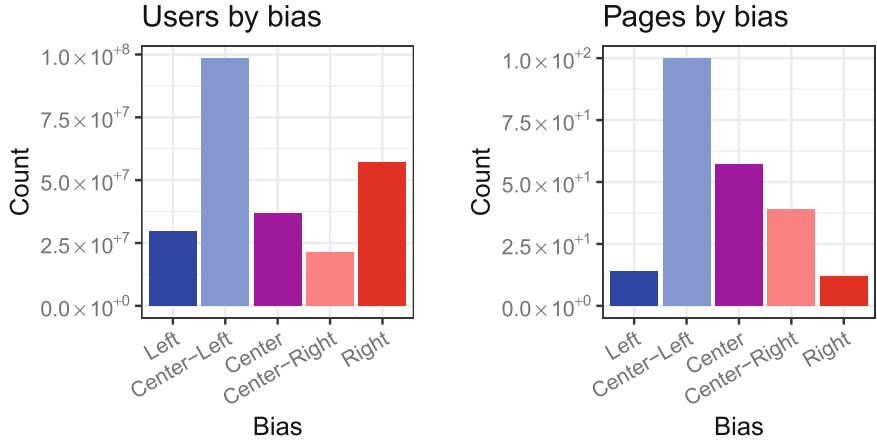
pages as the original multiple interactions between the same user and the same page are now spread between multiple pages. This uniformity of interactions implies that the result of a strong randomization process would be unaffected by a further weak randomization. Thus, the strong randomization process implies the weaker one. In Fig. 1, the details of the randomization processes are visually explained. Note that, in the bipartite representation, each edge indicate an interaction between a user and a page and multiple edges are allowed.



**Fig. 1.** In the tripartite representation of the interactions, the strong randomization process affects the user-pages interactions, while the weak one affects the page-bias affiliations.

### 3 Results

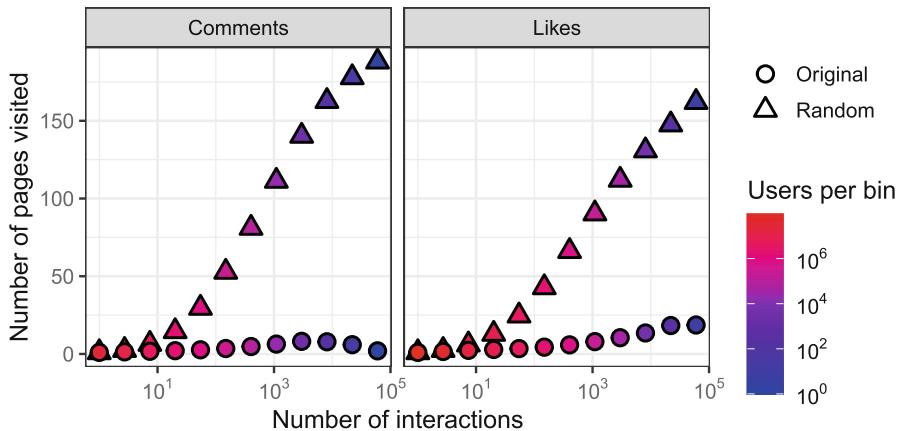
In our analysis, we use the same dataset as two previous papers [6, 21], which comprise roughly 266 million comments and 1.5 billion likes from about 170 million users on 47 million Facebook posts by 222 pages of news agencies. Those news agencies have a Bias score, provided by Media Bias/Fact Check [24], with five possible leanings ranging from left to right. We infer the political bias of users using the mode of the leanings of their interactions. The distribution of pages and users by political leaning is shown in Fig. 2, where we see that while most pages and users are identified as center-left leaning, there are in proportion many more right-wing users than pages.



**Fig. 2.** Number of Facebook pages and users grouped by political affiliation.

### 3.1 Measuring Selective Exposure

We first explore patterns of activity concentration by grouping users by the number of their likes and comments. We create 12 logarithmic bins and, for each bin, we compute the average number of pages they interact with. We then replicate this analysis on the strongly randomized dataset (see Materials and Methods for further details). Figure 3 compares the two scenarios.

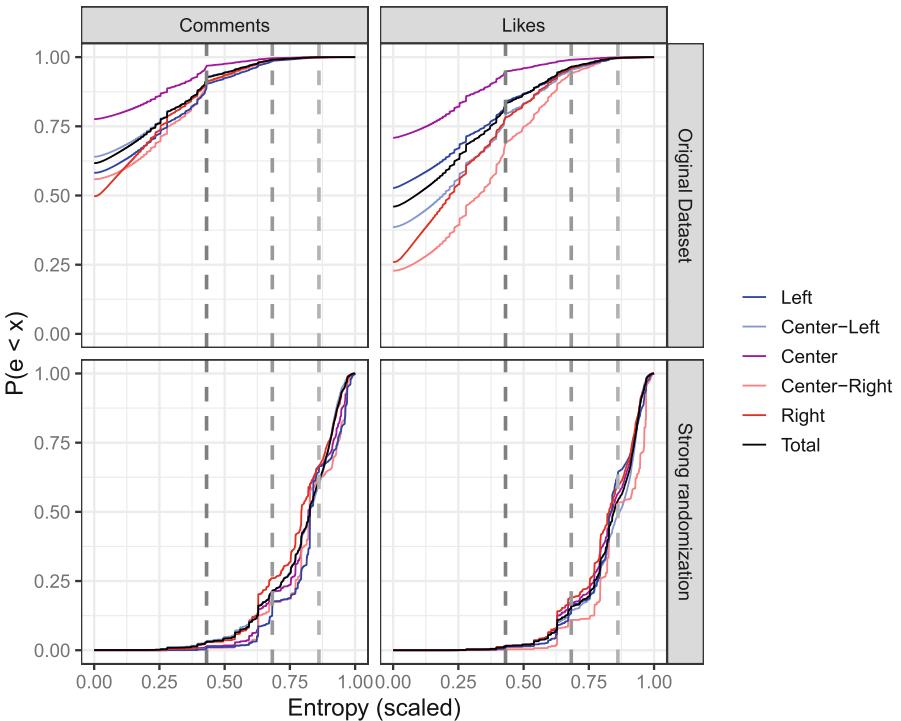


**Fig. 3.** Number of pages among which users divide their attention.

The results show that the real behavior of users is much more concentrated than the randomized case. Note that, in this latter case, the probability that a user interacts with a page is solely based on the page's popularity i.e. the number

of users interacting with it. While this on itself does not imply that there is a specific tendency for users to prefer content that is congruent with their specific view, it already suggests that users' activity is strongly dependent on factors that transcend the simple popularity of the page, as in the randomization case.

To understand if political leaning could be one of these factors, we group pages by their bias level and, for each user, we evaluate the Shannon entropy of their distribution of interactions left in each political class. We use this metric because it clearly indicates how concentrated or spread out each user's activity is. We scale the entropy values to a  $[0, 1]$  interval by dividing by  $\log 5$ , the theoretical maximum entropy value for 5 classes. To ensure that the observed selectivity is not due to low activity, we consider only users who have made more than 5 comments (likes) in this and all further analyses. The empirical cumulative distribution function (eCDF) of the values of Shannon's entropy for all groups of users is presented in Fig. 4. Additionally, we highlight the entropy levels corresponding to a user interacting evenly with two, three, and four different leanings using grey lines.



**Fig. 4.** Distributions of bias entropy of users compared with the strongly randomized scenario.

Clearly, the leaning classification of users is most reliable for those with entropy at the left of the first line, since they comment very heterogeneously.

All groups of users display a strong level of selectivity, with center-leaning users being the most selective. We observe that, in all groups, at least 50% of the users concentrated their commenting activity on pages of the same political leaning. This trend is also present, albeit less prominent, with likes, with center-right-leaning users being less selective than other groups of users. We compare this situation with the strong randomization process, that describes users interacting with pages based solely on popularity.

Interestingly, we observe a substantial difference with the randomized scenario, suggesting again that even the least selective users choose content according to a criterion compatible with political leaning.

### 3.2 Page and Bias Selectivity

In the previous section we have shown that users display a strong level of selectivity compatible with an ideological classification of the pages. However, this can only partially capture users' preferences in their activity. If users base their interactions solely on the political leaning of a page, they will treat all pages with the same leaning similarly. On the other hand, if users are selective about specific pages, their interactions within each bias level will reflect that page-driven selectivity. We observe that since pages can be thought of as a sub-partition of the bias levels, the values of the two entropies (the one calculated by considering the interaction on pages and bias levels) are actually dependent, as explained in Materials and Methods. In particular, we find that the entropy calculated on pages is equal to the entropy calculated on the bias levels summed with a weighted average of the entropies of the pages inside each bias level. Since each term of this average is independent, they can be trivially minimized and maximized. As explained in Materials and Methods, for each user, we find the theoretical interval  $I(u) = [m(u), M(u)]$  in which the actual page entropy  $H_p(u)$  can be found and scale the result using

$$x(u) = \frac{H_p - m}{M - m}.$$

Doing that, users who make at most one interaction per bias level are removed as the theoretical minimum and maximum coincide. This phenomenon is desirable, as it is impossible to decide if further selectivity is motivated by a preference for specific pages. We also recall that the analysis is performed only on users who have made at least five comments (likes) to ensure that the selectivity measured cannot be attributed to low activity. Table 1 reports the summary characteristics of the  $x(u)$  values distribution. Interestingly, we can observe that users tend to be very selective regarding pages, often concentrating their activity on one page per bias level.

### 3.3 Bias and Page Selectivity

In the previous sections, we have observed that, although users exhibit a selective behavior that is compatible with political segregation, it is not the only explaining driver of user activity. As users interact with -relatively- few pages, many

**Table 1.** Quartiles of the  $x$  statistics describing inter-bias selectivity.

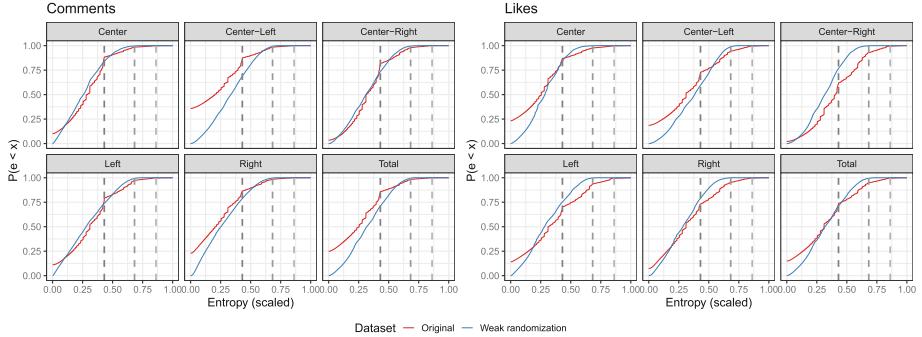
	Min	1st Quartile	Median	3rd Quantile	Max
Likes	0	0	0	0.2108	1
Comments	0	0	0	0.09045	1

alternative divisions could be compatible with the observed phenomenon, and thus, it is necessary to assess the divergence between the real selective exposure expressed by the users and a benchmark value found by grouping pages randomly. To accomplish this, we perform the weak randomization process described in Materials and Methods to account for measurements of bias selectivity that may arise from the simple page selectivity or other sorts of selective behaviors (for instance, locality-driven selectivity). We notice that the users interacting with only one page will naturally interact with only one bias class, regardless of the randomization. For those users, deciding if a preference in political leaning, locality, or any arbitrary classification of pages drives their choice of activity is impossible. For this reason, we set those users aside and only compute the distributions of users who interact with at least two pages and have a minimum of five interactions. Upon performing the weak randomization process, we note that since the user-page relations are unchanged, the page entropy of each user remains unaffected by this particular randomization process. This allows us to measure the bias selectivity while accounting for the strong patterns of page selectivity measured previously.

We perform a Monte Carlo experiment where each simulation is obtained using the weak randomization process. To ensure the manageability of the procedure, we used a random sample of the dataset (slightly over 2% of the users). By averaging over multiple groupings of pages, we obtain a benchmark value for the bias entropy of the users, which we can then compare to the actual bias entropy. If the political leaning determines a further selectivity, the distribution of the original dataset will have higher entropy values, and, especially at the left of the first line, the entropy eCDF will have higher values. If, on the contrary, the two distributions are similar or the weak randomization process produces higher entropy values, it means that the political leaning of the pages does not drive the selectivity of the users.

Figure 5 compares the real and randomized eCDF distributions and Table 2 summarizes the Kullback-Leibler divergence between the distributions.

We observe that users primarily commenting on center and center-right-leaning pages align closely with the randomized distribution, suggesting that the political leaning of these pages aligns with their usual consumption patterns. In contrast, users interacting with center-left-leaning pages show the most significant divergence from the randomized model, indicating that political leanings heavily influence their news consumption habits. This is evident when comparing these findings with those illustrated in Fig. 4, where users engaging with only one page are included. Notably, the exclusion of such users significantly reduces



**Fig. 5.** Distribution of users' bias entropy compared with the benchmark values obtained via weak randomization.

the number of those with zero entropy. Center-leaning users, who are the most selective, are most impacted by this exclusion. This suggests that less politically engaged users, who typically follow only one page, tend to prefer center-leaning pages. Conversely, users with more explicit political preferences display greater selectivity in their interactions. For likes, the patterns slightly differ. Center-leaning users clearly exhibit bias-driven engagement, a trend that is less pronounced among right-leaning users and even less so among center-right-leaning ones.

In particular, center-right-leaning users are less selective than their randomized counterparts. Left- and center-left-leaning users display prominent levels of divergence.

**Table 2.** The Kullback-Leibler divergence between the distribution of bias-entropies and the benchmark obtained via the weak randomization.

	Left	Center-Left	Center	Center-Right	Right	Total
Likes	1.029	1.111	0.949	0.879	0.645	1.002
Comments	0.520	1.317	0.430	0.480	0.658	1.014

## 4 Conclusion

In this paper, we conduct a comprehensive analysis of the phenomenon of selective exposure on social media platforms. Initially, we observe that user activity predominantly concentrates on a limited number of pages. As user engagement increases, these pages quickly become saturated, suggesting that user interactions are focused despite the availability of vast content. Employing Shannon Entropy to explore the homogeneity of user behavior, we identify a strong preference for content that aligns with users' pre-existing political views. This finding

supports the hypothesis that social media serves as an echo chamber, amplifying and reinforcing similar viewpoints. Further analysis reveals that user engagement is not uniform across pages with similar political leanings. Users prefer specific news sources within political categories, indicating a more nuanced approach to selective exposure that includes specific sources resonating deeply with individual users. This observation prompted developing and applying a novel methodology designed to dissect and analyze the reasons behind such selective behaviors. Our findings indicate that political congruence is a more significant driver of user behavior than random selection, with effects particularly pronounced among users leaning towards the center-left. Our work establishes a robust framework for analyzing and comparing the mechanisms of selective exposure across various user groups on social media. This framework enhances our understanding of why users gravitate towards certain content and improves our ability to predict page-level selectivity based on political bias. Additionally, it helps identify which user groups are most susceptible to selective exposure, shedding light on how echo chambers form and persist. However, this study has limitations. It does not account for the tone or nature of interactions—whether users support, criticize, or comment on content in a neutral or hostile manner. Despite this, we are confident in the robustness of our findings. We observe that negative or hostile interactions, though present, tend to be a small minority of all interactions and thus do not significantly alter the overall patterns of selective exposure.

## References

1. Bakshy, E., Messing, S., Adamic, L.A.: Exposure to ideologically diverse news and opinion on Facebook. *Science* **348**(6239), 1130–1132 (2015). <https://doi.org/10.1126/science.aaa1160> <https://www.science.org/doi/abs/10.1126/science.aaa1160>
2. Zollo, F., et al.: Debunking in a world of tribes. *PLOS ONE* **12**(7), 1–27 (2017). <https://doi.org/10.1371/journal.pone.0181821>
3. Zollo, F., et al.: Emotional dynamics in the age of misinformation. *PLOS ONE* **10**(9), 1–22 (2015). <https://doi.org/10.1371/journal.pone.0138740>
4. Garimella, K., Morales, G.D.F., Gionis, A., Mathioudakis, M.: Quantifying controversy on social media. *Trans. Soc. Comput.* **1**(1), 1 (2018). <https://doi.org/10.1145/3140565>
5. Del Vicario, M., et al.: The spreading of misinformation online. *Proc. Natl. Acad. Sci.* **113**, 01 (2016)
6. Cinelli, M., De Francisci Morales, G., Galeazzi, A., Quattrociocchi, W., Starnini, M.: The echo chamber effect on social media. *Proc. Natl. Acad. Sci.* **118**(9), e2023301118 (2021). <https://doi.org/10.1073/pnas.2023301118> <https://www.pnas.org/doi/abs/10.1073/pnas.2023301118>
7. Garrett, R.K.: Echo chambers online?: Politically motivated selective exposure among internet news users. *J. Comput.-Mediat. Commun.* **14**(2), 265–285 (2009)
8. Guess, A.: Media choice and moderation: evidence from online tracking data. Unpublished Manuscript (2016)
9. Bakshy, E., Rosenn, I., Marlow, C., Adamic, L.: The role of social networks in information diffusion. In: *WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web* (2012)

10. Sunstein, C.R.: Republic: Divided Democracy in the Age of Social Media, NED - New Edition. Princeton University Press (2018). <http://www.jstor.org/stable/j.ctv8xnhdt>
11. Zaller, J.R.: The Nature and Origins of Mass Opinion. Cambridge Studies in Public Opinion and Political Psychology, Cambridge University Press (1992)
12. Taber, C., Lodge, M.: Motivated skepticism in the evaluation of political beliefs. *Am. J. Polit. Sci. Am. J. Polit. Sci.* **50**(3), 755–769 (2006)
13. Lau, R.R., Redlawsk, D.P.: How Voters Decide: Information Processing in Election Campaigns. Cambridge Studies in Public Opinion and Political Psychology, Cambridge University Press (2006)
14. Nyhan, B., Reifler, J.: When corrections fail: the persistence of political misperceptions. *Polit. Behav.* **32**(2), 303–330 (2010)
15. Nam, H.H., Jost, J.T., Van Bavel, J.J.: Not for all the tea in China! Political ideology and the avoidance of dissonance-arousing situations. *PLOS ONE* **8**(4), 1–8 (2013). <https://doi.org/10.1371/journal.pone.0059837>
16. Munro, G.D., Ditto, P.H., Lockhart, L.K., Fagerlin, A., Gready, M., Peterson, E.: Biased assimilation of sociopolitical arguments: evaluating the 1996 us presidential debate. *Basic Appl. Soc. Psychol.* **24**(1), 15–26 (2002)
17. Iyengar, S., Sood, G., Lelkes, Y.: Affect, not ideology: a social identity perspective on polarization. *Public Opin. Q.* **76**(3), 405–431 (2012)
18. Nisbett, E., Cooper, K., Garrett, R.K.: The partisan brain. *An. Am. Acad. Polit. Soc. Sci.* **658**, 36–66 (2015)
19. Barberá, P., Jost, J., Nagler, J., Tucker, J., Bonneau, R.: Tweeting from left to right: is online political communication more than an echo chamber? *Psychol. Sci.* **26**, 08 (2015)
20. Wu, S., Resnick, P.: Cross-partisan discussions on YouTube: conservatives talk to liberals but liberals don't talk to conservatives. In: Proceedings of the International AAAI Conference on Web and Social Media, pp. 808–819 (2021)
21. Cinelli, M., Brugnoli, E., Schmidt, A.L., Zollo, F., Quattrociocchi, W., Scala, A.: Selective exposure shapes the Facebook news diet. *PLOS ONE* **15**(3), 1–17 (2020). <https://doi.org/10.1371/journal.pone.0229129>
22. Schmidt, A., et al.: Anatomy of news consumption on Facebook. *Proc. Natl. Acad. Sci.* **114**, 03 (2017)
23. Selective exposure. <https://www.oxfordreference.com/view/10.1093/oi/authority.20110803100452931>
24. Media bias/fact check. <https://mediabiasfactcheck.com>



# Modelling the Influence of Deinfluencers

Jiarui Hu and Tzu-Yi Chen<sup>(✉)</sup>

Computer Science Department, Pomona College, Claremont, CA 91711, USA  
[{jhaa2021,tzuyi.chen}@pomona.edu](mailto:{jhaa2021,tzuyi.chen}@pomona.edu)

**Abstract.** While influencers on social media are advocating for products, deinfluencers are encouraging users to refrain from buying those same products. We introduce an influence maximization model that captures unique aspects of the interaction between influencers and deinfluencers. We then use the model on both synthetic and real-world networks to explore different strategies for choosing an initial deinfluencers seed set affects the impact of both influencers and deinfluencers.

**Keywords:** deinfluencers · influence maximization · competitive influence maximization

## 1 Introduction

As of May 2023, the #deinfluencing hashtag on TikTok had close to 584 million total views, 582 million of which were recorded in the preceding 12 months [13]. Motivated by reasons including disillusionment with past purchases [24] and a general rejection of consumption culture [1], deinfluencers try to convince others not to buy products. Their impact can be significant since negative online reviews have a disproportionate impact on purchase decisions [7, 17, 23, 26].

In this work we focus on modelling the interactions of influencers, deinfluencers, and potential consumers. After briefly discussing relevant previous work in the field of influence maximization, we describe our model and examine two specific questions:

- How important is the strategy, whether random or greedy or metric-based, used to choose the initial deinfluencers? For a greedy strategy, does it matter whether the goal is to minimize influence or to maximize deinfluence?
- What is the impact of restricting the viable set of initial deinfluencers to either only already influenced nodes or only still susceptible nodes?

We explore these questions on both random and real-world networks, thus giving insight into how underlying graph topology affects the answers.

## 2 Background

The influence maximization problem was formalized as a discrete optimization problem in [14]: given a graph  $G = (V, E)$ , what is a seed set  $V' \subseteq V$  that maximizes the expected influence spread? The authors of [14] explore this question

under two models of influence spread: Independent Cascade and Linear Threshold. They show the influence maximization problem is NP-hard for both models and that a greedy hill-climbing strategy achieves slightly better than 63% of the optimal solution. Subsequent work has explored improved heuristics for choosing the seed set, more thorough experimentation, faster implementations, and more nuanced models designed to reflect a range of real world phenomena; surveys of this work include [19, 20]. Concepts that are particularly relevant to our work include competitive influence maximization and influence blocking.

Competitive influence maximization models situations where two or more groups compete to maximize their influence. This happens, for example, in political campaigns and when brands compete in the same market. Work in this area includes [3–5, 9, 21].

As opposed to competitive influence maximization, influence blocking studies the situation where one group’s goal is to minimize the spread of influence by the other group rather than to maximize its own influence. This work is commonly motivated by the problem of curbing the spread of misinformation or disinformation. Work on influence blocking includes [5, 11, 12, 22].

Previous work in both competitive influence maximization and influence blocking typically assumes that a node which has been influenced by one group cannot subsequently be influenced by another. By way of contrast we assume an influenced node can become deinflused if, say, someone becomes disenchanted with a previous purchase. However, because people are reluctant to be seen as flip-flopping, we treat influenced and deinflused nodes non-symmetrically: while a consumer can become disenchanted, the consumer cannot oscillate between being thrilled and disenchanted by the same product. Additionally in our experiments we explore different viable deinfluser seed sets to capture the distinction between deinflusers who are consumers that subsequently change their minds and those who reject all consumption. Finally, instead of focusing on developing better heuristics, we look first to understand the dynamics of influencer-deinfluser interactions.

### 3 Model

Given a directed graph  $G = (V, E)$ , each vertex  $v \in V$  is in one of three states: Unactivated-Susceptible (S), Activated-Influenced (I), or Activated-Deinflused (D). Activated nodes can affect the states of their neighbors. A susceptible node can become either influenced or deinflused, an influenced node can only become deinflused, and deinflused nodes cannot change state. The edges model lines of influence and an edge  $(i, j)$  is assigned three weights to capture the likelihood of node  $j$  accepting node  $i$ ’s influence as a function of the state of nodes  $i$  and  $j$ . Summarizing, the state of the network is captured by the triple  $(G, \mathbf{S}, \mathbf{P})$ , where  $G = (V, E)$ . The function  $\mathbf{S} : V \rightarrow \{S, I, D\}$  assigns each node  $v \in V$  one of three possible states. The function  $\mathbf{P} : E \rightarrow [0, 1]^3$  assigns to each directed edge  $(v_i, v_j) \in E$  a tuple of probabilities  $(p_{ij}^{IS}, p_{ij}^{DI}, p_{ij}^{DS})$ .

We then model spread through a series of discrete time steps using a modified version of the Independent Cascade model. The probability that a susceptible

node  $j$  becomes influenced or deinfluenced at time step  $t$  is given, respectively, by:

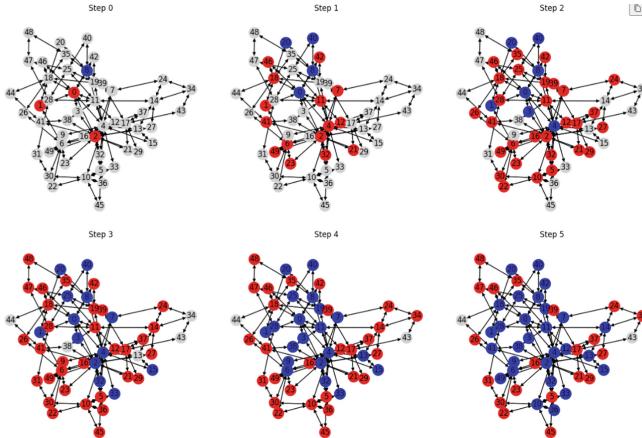
$$P(S \rightarrow I|N_I(j), N_D(j)) = 1 - \prod_{i \in N_I(j)} (1 - p_{ij}^{IS})$$

$$P(S \rightarrow D|N_I(j), N_D(j)) = 1 - \prod_{i \in N_D(j)} (1 - p_{ij}^{DS})$$

where  $N_I(j)$  is the set of neighbors of node  $j$  that are in the I state at the start of time step  $t$  and  $N_D(j)$  is the set of neighbors in the D state. An influenced node  $j$  can only transition to the deinfluenced D state and it does so with probability:

$$P(I \rightarrow D|N_D(j)) = 1 - \prod_{i \in N_D(j)} (1 - p_{ij}^{DI})$$

whereas the original Independent Cascade model gives an influenced node only one chance to affect its neighbors, we allow nodes multiple opportunities. This models the observation that users may see a message more than once if, for example, the social media platform chooses to display it again. Additionally if a node in an I state and another in a D state are both trying to influence a susceptible node, we use a parameter  $p_{conf}$  to determine which state the susceptible node enters. Note that this parameter allows the modelling of either influencers or deinflueners having greater authority. This contrasts with prior work in which one side is always favored. Figure 1 illustrates our model on a small network.

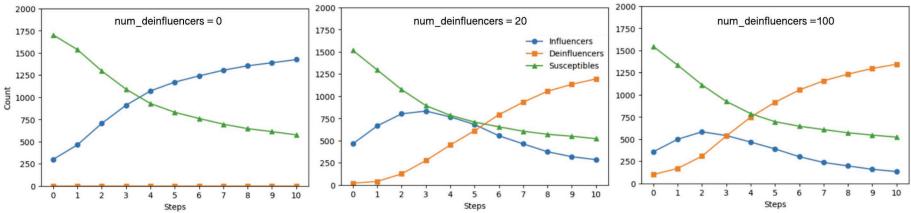


**Fig. 1.** Influencer-deinfluenecer simulation. Susceptible nodes (S) are uncolored; activated-influenced (I) notes are red; activated-deinfluenced (D) nodes are blue.

Because a product may need to reach a certain level of popularity before users advocate against it, we introduce a parameter  $t_{pre}$  which is the number of

steps of influence spread by the initial  $k$  influencers before  $m$  seed deinfluencers are introduced. While one could mimic this by first doing the initial spread and then having all influenced nodes form the seed set of influencers, this would not allow us to differentiate between selecting the seed deinfluencers from the initial  $k$  influencers as opposed to those that were influenced by them. Once the seed deinfluencers are chosen the simulation is run for another  $t_{gen}$  steps, where  $t_{gen}$  captures the fact that attention spans on social media are limited.

Putting it all together, Fig. 2 illustrates the behavior of our model on a Barabási-Albert graph with 2000 nodes and 4000 directed edges for different values of  $m$ . In all cases the  $m$  deinfluer seed nodes are chosen at random. When  $m = 0$  this is a modified independent cascade model so, as expected, the number of influenced nodes steadily increases. When  $m > 0$ , the number of influencers initially rises before peaking and falling while the number of deinfluencers increases; increasing the value of  $m$  has the expected effect.



**Fig. 2.** The number of susceptible (green), influenced (blue), and deinfluer (orange) nodes as a function of the number of time steps on a Barabási-Albert graph for three values of  $m$ , the number of seed deinfluers. The value of  $m$  from left to right is 0, 20, and 100.

## 4 Experimental Framework

As described above, our model incorporates several parameters and options. In this section we summarize our experiments while highlighting those that are discussed in greater detail in the following Results section.

### 4.1 Parameters

**Network.** The first parameter is the network itself. We tested three types of directed synthetic networks: Erdős-Rényi, which are random homogeneous graphs; Barabási-Albert, which model social networks with preferential attachment properties; and Watts-Strogatz, which generates graphs with small world properties. We considered synthetic networks ranging in size from 300 to 3000 nodes and from 300 to 9000 edges. We also ran experiments on the directed real-world networks summarized in Table 1, all of which are in the SNAP dataset [18].

**Table 1.** Real world networks

network	nodes	edges	diam	description
email-Eu-core	1005	25571	7	email network
ego-Facebook	4039	88234	8	social network
wiki-Vote	7115	103689	7	wikipedia network
ca-GrQc	5242	14496	17	collaboration network

**Edge Weight Probabilities.** Each edge  $(i, j)$  needs to be assigned three weights  $(p_{ij}^{IS}, p_{ij}^{DI}, p_{ij}^{DS})$  capturing the likelihood of one endpoint being influenced (or deinfluenced) by the other. We experimented with assigning probabilities uniformly at random and with assigning fixed probabilities  $(p^{IS}, p^{DI}, p^{DS})$  to all edges. In the Results section we focus on edge weights assigned uniformly at random.

**Breaking Ties.** If a susceptible node can be influenced by either an I node or a D node, our model uses the parameter  $p_{conf}$  to determine the likelihood that the node becomes an I node. We experimented with different values, but the Results section focuses on the case  $p_{conf} = 0.5$ .

**Size of the Initial Influencer Seed Set.** Initially the simulation begins with  $k$  influencers. We experimented with values of  $k$  ranging from 5 to 500 and, for the purposes of illustrating general trends, use values ranging from 30 to 150 for the results shown in this paper.

**$t_{pre}$  and  $t_{gen}$ .** The parameter  $t_{pre}$  dictates how many steps of the simulation are run with only influencers. The parameter  $t_{gen}$  dictates how many steps of the simulation are run after the seed deinfluencers are introduced. Because the spread of information in social networks tends to either dissipate or reach a saturation point within a short timeframe [2, 10], we limited the values of  $t_{pre}$  and  $t_{gen}$  to small integers. In our results section the values are  $t_{pre} = t_{gen} = 3$ .

## 4.2 Choosing the Deinfluencer Seed Nodes

We varied both the strategy that was used for choosing the deinfluencer seed nodes and the set of valid nodes from which the deinfluencer seed nodes could be chosen.

**Strategies for Choosing the Deinfluencer Seed Nodes.** There are many possible ways to select the initial deinfluencers. The random method selects  $m$  nodes at random. Metric-based strategies rank nodes in order of some centrality measure and select the top  $m$  as the set of deinfluencers. Our experiments use measures including degree, closeness, betweenness, eigenvector, and PageRank based on [25]. Greedy heuristics choose the seed nodes iteratively based on their expected impact; in our experiments we used the heuristic described in [8].

**Viable Sets for the Deinfluencer Seed Nodes.** Finally we vary the set from which the deinfluer seed set can be chosen. We considered several options:

1. susceptible only (ExAllInf): deinfluer seed nodes are chosen from the set of nodes that are in the  $S$  state after the initial  $t_{pre}$  steps of the simulation.
2. influencer-seeds only (IniInf): deinfluer seed nodes are chosen from the set of influencer seed nodes.
3. influencers only (AllInf): deinfluer seed nodes are chosen from the set of nodes that are in the  $I$  state after the initial  $t_{pre}$  steps of the simulation.
4. influenced only (ExIniInf): deinfluer seed nodes are chosen from the set of nodes that are in the  $I$  state after the initial  $t_{pre}$  steps of the simulation, but cannot be one of the initial influencer seed nodes.

The first option mimics the situation in which deinfluers are motivated by a general commitment against overconsumption; the others are variations of the situation where deinfluers are users who previously purchased or advocated for a product before subsequently changing their minds.

### 4.3 Measurements

In each step we keep track of the number of nodes in each state. Because of the randomness in the model, we ran each simulation 5 times for each selection strategy and each possible number of deinfluer nodes, then plot the average number of nodes in each state.

### 4.4 Environment

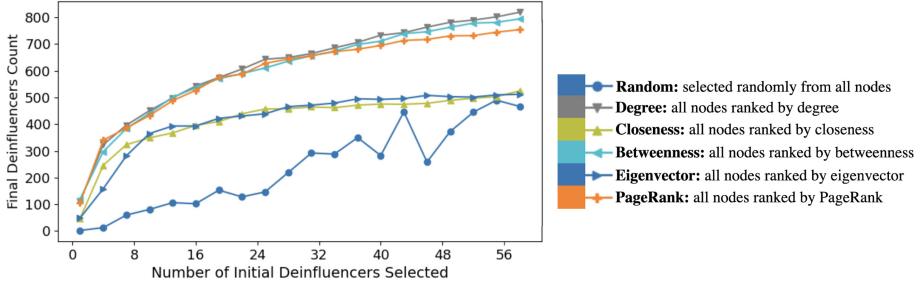
All experiments were conducted on a MacBook Pro with a M2 chip and 16 GB memory running macOS Darwin 23.3.0 and Python 3.11.8. All code was written in python; we used the `joblib` package to facilitate parallel processing.

## 5 Results

In this paper we focus on understanding how the model behaves with different choices for the deinfluer seed sets. We first look at the impact of seed selection strategy, and then of different viable sets, on different synthetic networks. We then look at the same questions on real-world networks.

### 5.1 Strategies for Choosing Deinfluer Seed Nodes

Figure 3 plots the number of deinflued nodes after  $t_{gen} = 3$  steps as a function of  $m$ , the initial number of deinfluer seed nodes, on a Barabási-Albert network with 2000 nodes and 4000 edges. While we experimented on other size

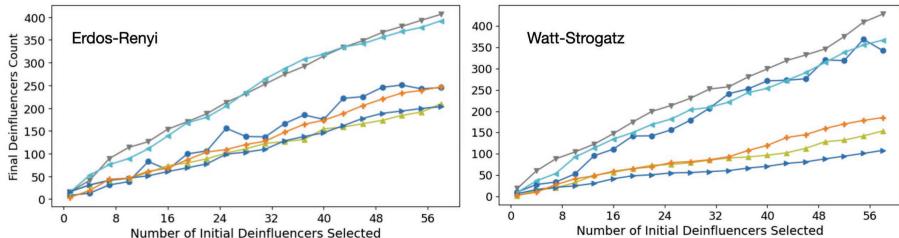


**Fig. 3.** The number of nodes in a D state after 3 iterations as a function of the size of the deinfluencer seed set for 6 different seed set selection strategies.

networks, the cascade properties were generally consistent across network size. Each line represents one strategy for choosing the  $m$  seed nodes without placing any restrictions on which of the nodes can be deinfluencer seed nodes. The strategies are: random selection (Random), the  $m$  nodes with the highest degree, closeness, or betweenness measure (Degree, Closeness, and Betweenness, respectively), the  $m$  nodes corresponding to the most influential nodes based on their connection to other highly connected nodes (Eigenvector), and the  $m$  nodes corresponding to both the quantity and “quality” of incoming connections (PageRank).

As expected, the curves all increase with the size of the deinfluencer seed set. Not surprisingly, the random strategy performs worst. We additionally observe that taking the nodes with the highest degree outperforms a more computationally expensive strategy such as PageRank.

In fact, as shown in Fig. 4, choosing the nodes with the highest degree for the deinfluencer seed set works well not only for Barabási-Albert networks, but also for Watts-Strogatz and Erdos-Renyi graphs as reflected by the fact that the grey line is consistently above the others. We further observe that while

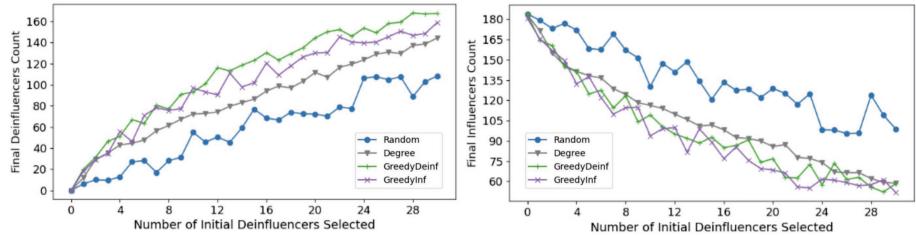


**Fig. 4.** The number of nodes in a D state after 3 iterations as a function of the size of the deinfluencer seed set for 6 different seed set selection strategies on Erdos-Renyi (left) and Watts-Strogatz (right) networks.

choosing deinfluencers at random does very poorly for Barabási-Albert networks as shown in Fig. 3, choosing deinfluencers at random does quite well for Watts-Strogatz networks. This suggests seed selection strategies that work well for social networks might not work as well on other networks and vice versa.

**Greedy Heuristics with Different Objective Functions.** Another strategy for selecting the deinfluencer seed nodes is to use a greedy heuristic.

Figure 5 again plots the number of deinfuenced nodes as a function of the number of deinfluer seed nodes, but this time choosing those seed nodes using two different greedy strategies. The graph on the left plots the number of deinfluencers and the graph on the right shows the number of influencers.

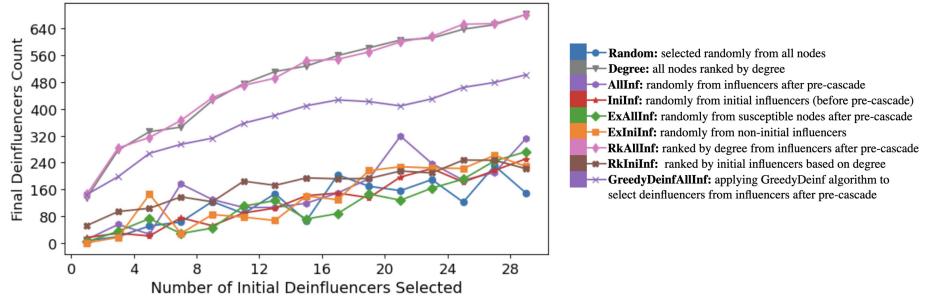


**Fig. 5.** Plotting the number of deinfuenced (left) and influenced (right) nodes as a function of the number of deinfluer in the initial seed set. GreedyDeinf (green) selects seed nodes likely to maximize the number of deinfuenced nodes whereas GreedyInf (purple) selects nodes expected to minimize the number of influenced nodes.

Not surprisingly, we found that the greedy heuristics are generally more effective than either metric-based or random methods (shown in Fig. 5 in grey and blue, respectively) at both promoting deinfuence and reducing influence. We additionally observed that optimizing for more deinfluencers (GreedyDeinf) outperforms a heuristic that tries to minimize the number of influencers (GreedyInf): at the end of the simulation both have similar numbers of influenced nodes but the former has notably more deinfuenced nodes.

## 5.2 Restricting the Viable Sets for Deinfluer Seed Nodes

Figure 6 plots the result of restricting the valid set of potential deinfluer seed nodes on the same size Barabasi-Albert network as used in Fig. 3; recall this models situations in which deinfluers are motivated by a commitment against overconsumption as opposed to a desire to campaign against a previously-purchased product. Ignoring the top three lines (labelled Degree, RkAllInf, and GreedyDeinfAllInf), we observe that the restriction does not have any obvious impact if the nodes are selected at random from the valid set. However, selecting from the valid set using a measure such as degree is naturally better.



**Fig. 6.** The number of nodes in a D state after 3 iterations as a function of the size of the deinfluencer seed set with different restrictions on valid seed nodes.

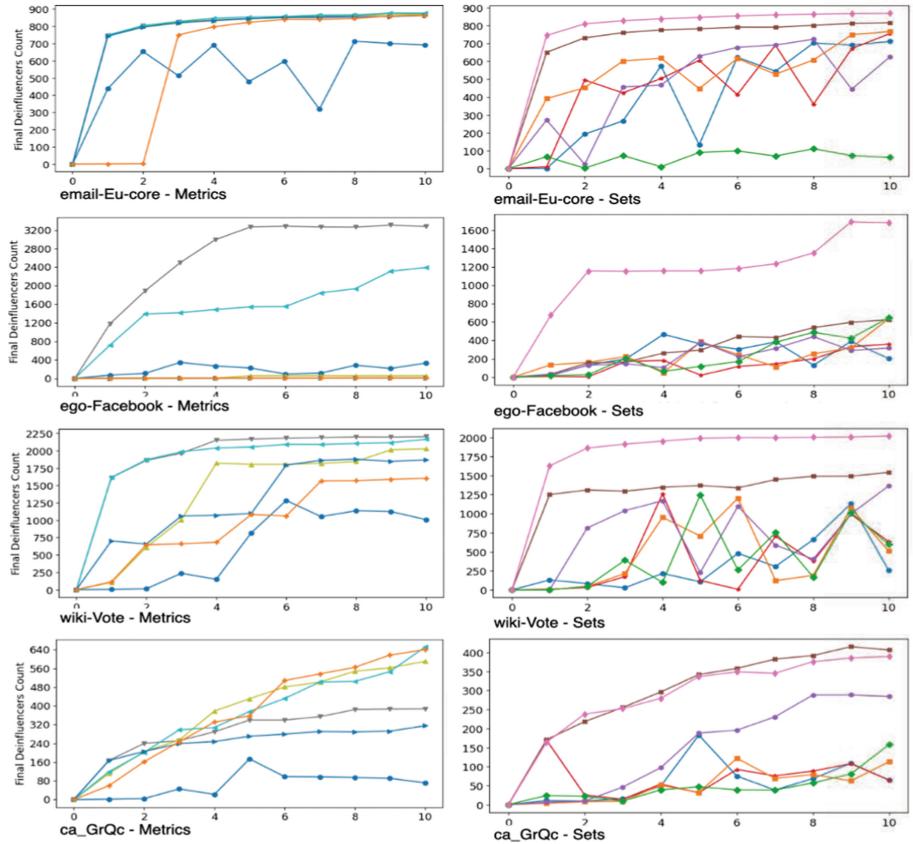
And while RdIniInf and RdAllInf performed similarly to the other strategies in terms of increasing the number of deinfluencers, both were more effective than other strategies at reducing the number of influencers.

Furthermore, restricting the valid set in different ways affects whether a heuristic is more effective at increasing the number of deinfluencers or at decreasing the number of influencers. We observe that RkAllInf performs nearly identical to Degree even though the former is selecting from only approximately 100 influenced nodes and the latter is selecting from the entire set of 3000 nodes. Additionally, using a greedy heuristic to select the seed nodes from this restricted set (GreedyDeinfAllInf) does very well and is also less computationally expensive than using a greedy heuristic on the entire set of nodes.

### 5.3 Results on Real-World Networks

Not surprisingly, seed selection strategies have different effects on real-world networks as compared to synthetic networks. Figure 7 shows the impact of different seed selection strategies and different restrictions on the valid set for four real-world networks.

We observe that while metric-based measures still generally perform well, choosing the nodes with highest degree is not as reliably good a strategy as with the synthetic networks tested. Additionally, a measure can be worse than random on one network but very effective on another, as shown in Fig. 7 when comparing the behavior of PageRank (the orange line) on ego-Facebook network vs ca-GrQc network. Finally the effect of restricting the valid set for the initial deinfluencer seed set depends not only on the selection strategy but also on the size of the deinfluencer seed set.



**Fig. 7.** The number of nodes in a D state after 3 iterations as a function of the size of the deinfuencer seed set (ranging from 0 to 10) with different selection strategies (left hand column) and different restrictions on the valid set (right hand column) on four real-world networks.

## 6 Conclusion

In this paper we describe an influence maximization model motivated by the impact of deinfluencers. We discuss how the model behaves experimentally when run on a range of synthetic and real-world networks. We focus on understanding the impact of different strategies for choosing the deinfuencer seed set and find that metric-based methods perform well on synthetic networks but less reliably so on real-world networks.

Future work includes extending our model by adding budget constraints as done in papers such as [6, 15, 16]. Preliminary results connect network characteristics to the impact of different pricing models on the efficacy of selecting high-degree nodes as seed nodes.

## References

1. Ansu-Mensah, P.: Green product awareness effect on green purchase intentions of university students': an emerging market's perspective. Future B. J. **7**(1), 48 (2021)
2. Bao, Y., Yi, C., Xue, Y., Dong, Y.: Precise modeling rumor propagation and control strategy on social networks, pp. 77–102. Springer, Cham (2015)
3. Bharathi, S., Kempe, D., Salek, M.: Competitive influence maximization in social networks. In: Internet and Network Economics: Third International Workshop, WINE 2007, pp. 306–311 (2007)
4. Borodin, A., Filmus, Y., Oren, J.: Threshold models for competitive influence in social networks. In: Saberi, A. (ed.) Internet and Network Economics, pp. 539–550 (2010)
5. Budak, C., Agrawal, D., El Abbadi, A.: Limiting the spread of misinformation in social networks. In: Proceedings of the 20th International Conference on World Wide Web, WWW '11, pp. 665–674 (2011)
6. Cellinese, F., D'Angelo, G., Monaco, G., Velaj, Y.: Generalized budgeted submodular set function maximization. Inf. Comput. **281**, 104741 (2021)
7. Charlett, D., Garland, R., Marr, N., et al.: How damaging is negative word of mouth. Mark. Bull. **6**(1), 42–50 (1995)
8. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09, pp. 199–208 (2009)
9. Goyal, S., Kearns, M.: Competitive contagion in networks. In: Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing, STOC '12, pp. 759–774 (2012)
10. Haerter, J.O., Jamtveit, B., Mathiesen, J.: Communication dynamics in finite capacity social networks. Phys. Rev. Lett. **109**, 168701 (2012)
11. He, X., Song, G., Chen, W., Jiang, Q.: Influence blocking maximization in social networks under the competitive linear threshold model. In: Proceedings of the Twelfth SIAM International Conference on Data Mining, pp. 463–474 (2012)
12. Hemmati, M., Smith, J.C., Thai, M.T.: A cutting-plane algorithm for solving a weighted influence interdiction problem. Comput. Optim. Appl. **57**(1), 71–104 (2014)
13. Karimi, F.: Forget the influencers. Here come the deinfluencers. CNN (2023). <https://www.cnn.com/2023/06/11/us/deinfluencing-tiktok-trend-explained-cec/index.html>
14. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, pp. 137–146 (2003)
15. Khuller, S., Moss, A., Naor, J.S.: The budgeted maximum coverage problem. Inf. Process. Lett. **70**(1), 39–45 (1999)
16. Krause, A., Guestrin, C.: A Note on the Budgeted Maximization of Submodular Functions. Carnegie Mellon University (2018). [https://kilthub.cmu.edu/articles/journal\\_contribution/A\\_note\\_on\\_the\\_budgeted\\_maximization\\_of\\_submodular\\_functions/6591131](https://kilthub.cmu.edu/articles/journal_contribution/A_note_on_the_budgeted_maximization_of_submodular_functions/6591131)
17. Lee, J., Park, D.H., Han, I.: The effect of negative online consumer reviews on product attitude: an information processing view. Electron. Commer. Res. Appl. **7**, 341–352 (2008)
18. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection (2014). <http://snap.stanford.edu/data>

19. Li, Y., Fan, J., Wang, Y., Tan, K.L.: Influence maximization on social graphs: a survey. *IEEE Trans. Knowl. Data Eng.* **30**(10), 1852–1872 (2018)
20. Li, Y., Gao, H., Gao, Y., Guo, J., Wu, W.: A survey on influence maximization: from an ML-based combinatorial optimization. *ACM Trans. Knowl. Discov. Data* **17**(9) (2023)
21. Lu, W., Chen, W., Lakshmanan, L.V.S.: From competition to complementarity: comparative influence diffusion and maximization. *Proc. VLDB Endow.* **9**(2), 60–71 (2015)
22. Nguyen, N.P., Yan, G., Thai, M.T., Eidenbenz, S.: Containment of misinformation spread in online social networks. In: Proceedings of the 4th Annual ACM Web Science Conference, WebSci '12, pp. 213–222 (2012)
23. Qahri-Saremi, H., Montazemi, A.R.: Negativity bias in the diagnosticity of online review content: the effects of consumers' prior experience and need for cognition. *Eur. J. Inf. Syst.* **32**(4), 717–734 (2023)
24. Quittner, E.: Is this the end of Instagram cookware? New York Times (2024). <https://www.nytimes.com/2024/05/24/dining/instagram-cookware.html>
25. Saxena, A., Iyengar, S.: Centrality measures in complex networks: a survey. arXiv preprint [arXiv:2011.07190](https://arxiv.org/abs/2011.07190) (2020)
26. Varga, M., Albuquerque, P.: The impact of negative reviews on online search and purchase decisions. *J. Mark. Res.* **51**(5), 803–820 (2023)

## **Multilayer/Multiplex**



# The Structure of Illegal Online Gambling Network

Ardian Maulana<sup>(✉)</sup> , Andhika Bernad, Fausto Keiluhu, and Hokky Situngkir

Department of Computational Sociology, Bandung Fe Institute, Bandung, Indonesia  
ardianmaulanaeff@gmail.com

**Abstract.** We investigate the operational structure of illegal online gambling (IOG) ecosystems by modeling the relationships between gambling sites as multiplex network. We examined the structural properties of the aggregate network as well as each layer within the multiplex network. In general, almost all layers exhibit assortative characteristics, indicating that high-degree nodes tend to connect with nodes of similar attributes. Interlayer and cluster analysis show that the relational layers of illegal online gambling sites exhibit a hierarchically nested structure. The characterization of illegal gambling site networks conducted in this study can provide insights for developing more targeted anti-gambling policies.

**Keywords:** multiplex network · illegal online gambling · assortative

## 1 Introduction

The operation of illegal online gambling (IOG) services has emerged as a priority issue currently being addressed in numerous countries. This concern is primarily related to the prevalence of fraud and cybercrime associated with IOG operations [1, 2], as well as the socio-economic losses experienced by the public. Several countries have escalated this issue to an emergency level and have implemented various policies to address it, including the intensification of content moderation through site blocking [3, 4]. However, comprehensive studies providing insights into IOG ecosystem have not been extensively conducted, particularly regarding the methods and strategies employed by operators to evade law enforcement. Such research is essential for formulating effective anti-gambling strategies and policies.

Previous studies on IOG technology have predominantly focused on cyber techniques used for promotion and strategies to evade detection and regulation. In [5, 6], the authors discussed how gambling site operators can utilize black hat SEO techniques to promote their websites to a broader internet audience. These operators can exploit high-ranking website reputations, manipulate dangling resources on DNS records, or covertly poison search engine results with illicit promotional texts (IPT). Authors in [7] reported on the characterization of mobile-based online gambling applications and the challenges faced in addressing covert distribution and supervising the operational infrastructure of these applications. In [8], the authors conducted an empirical and systematic study to measure the profit chain of illegal online gambling in China and investigated both the

upstream and downstream aspects of illegal gambling websites. They also reported on the strategies employed by site operators to control gambling sites on a massive scale, how they abuse network infrastructure, and how they gain profit.

This study aims to investigate the operational structure of IOG ecosystems. Our main objectives are to identify the relational patterns linking various IOG sites and to determine how these relations can be used to characterize the ecosystem. We first identify the key components necessary for operating IOG sites, starting with the web technology. Once these components are identified, we unveil the relationships among IOG sites, which indicate ownership by the same gambling groups. The relationships among these sites are then modeled as layers within a multiplex network, with each layer representing different key components of the IOG ecosystem. We apply network analysis to examine the structural properties of the aggregate network as well as each layer within the multiplex network. We then conduct interlayer analysis to identify similarities between layers and perform clustering analysis. The latter analyses elucidate the underlying structure among layers.

## 2 Data and Methods

We conducted web scraping on 35,229 active IOG sites with Fully Qualified Domain Names (FQDNs) that are confirmed to be operational or accessible. Through this process, we obtained initial data consisting of the source code from the main page of these sites, along with the Internet Protocol (IP) addresses associated with them. We enriched this data by gathering detailed information about the IP addresses, including network provider, associated country, and geolocation. We then performed scanning to profile web technologies and extract supplementary information from the page source code, which will be detailed in the following subsection. This procedure outputs a structured dataset that will be used in further analysis to characterize the IOG ecosystem.

### 2.1 Infrastructure and Sites Promotion

The IP address of a site serves as an identifier for the network infrastructure used to operate and access the site via the Internet. Out of the 35,229 sites collected, only 11,565 unique IP addresses were recorded, indicating that some IP addresses are shared among multiple sites with distinct Fully Qualified Domain Names (FQDNs). This observation may suggest two possible scenarios: 1) IOG sites are utilizing shared hosting services, where multiple websites share the same resources on a single physical server, or 2) an IOG site operates with more than one FQDN.

Based on our preliminary investigation [9], it is not difficult to find cases where an IOG site is assigned to multiple domains, considering the enforcement of illegal site blocking policies in numerous countries which are based on domain names blacklists. Operators of IOG sites anticipate the impact of these policies by assigning several domains for a single site as a contingency measure to provide alternative access for users when one of the domains is blacklisted [10]. In addition, the strategy of using multiple domains is also adopted to practice black hat search engine optimization (SEO) [6, 11]. Given that IOG sites cannot engage in legal SEO to reach potential users, they resort to

manipulating search engine results through techniques such as keyword stuffing [12], cloaking [13], and spider pooling [14]. This includes reusing dangling domain names from inactive old sites to retain backlink effects from those domains [5].

IOG site operators also utilize promotional methods through web pages on other sites, including both digital advertisements and external referral links. These sites may include other illegal sites (such as porn sites) or landing pages that redirect visitors to gambling websites [15]. We perform extraction of hyperlinks present on the analyzed IOG sites and check whether the Fully Qualified Domain Names (FQDNs) of these links are also included in the dataset. This procedure aims to elucidate the direct relationships between IOG sites and to understand how visitor navigation might occur between these sites.

## 2.2 Third-Party Services Identifiers

We applied string pattern extraction techniques to web pages of IOG sites to gather unique identifiers from third-party services utilized by these sites. Based on their functionality, we classified these identifiers into three distinct categories. These categories can be regarded as key components in the business processes related to the operation of IOG sites.

**Analytics Tags.** IOG site administrators need to monitor their site performance based on visitor traffic metrics and to acquire valuable data regarding user profiles and behavior. Furthermore, they need to gain insights into the primary sources of traffic directing visitors to the sites and evaluate the effectiveness of their promotional activities in reaching new potential users. Based on this data, site administrators can make more informed decisions on advertising strategies and site development initiatives. One key technology widely used in the advertising industry to support these needs is analytics tags [16]. Based on the extraction results from active IOG sites, we found that 23.8% of online gambling websites were detected using analytics tags embedded in their site interfaces. Most of these sites utilize Google Analytics and Facebook Pixel services (Table 1).

**Table 1.** Statistics of analytics tags

Analytics Tag	Illegal Online Gambling Sites		Identifiers	
	Count	Percentage	Count	Average usage (sites)
Facebook Pixel	4,565	13.0%	8,966	2.34
Google Analytics	5,518	15.7%	2,721	2.57
Google Tag Manager	1,265	3.6%	621	2.92

**Customer Services.** To facilitate question-and-answer activities with users, IOG sites also provide customer service (CS) support. This service allows users to submit inquiries and complaints to the site administrators. In some cases, customer service is involved

in the payment transaction process to provide information regarding recipient account details and to verify payments. This can be interpreted as a mechanism implemented by site administrators to prevent fraudulent payments and further to avoid account tracking by law enforcement agencies. In general, IOG websites make use of two primary methods for providing customer service support, i.e. through third-party customer service (CS) platforms and instant messaging applications. Third-party services can be directly integrated with the website's interface, allowing users to interact with site CS representatives without leaving the site. Typically, site CS representatives use digital avatar profiles to maintain anonymity. Nevertheless, identifiers for chat service accounts are still visible, allowing indication of account ownership to be identified. Our findings indicate that LiveChat is the most frequently used third-party service (26.2%) by IOG sites, and it is not uncommon for a single account to be utilized by multiple IOG sites. Instant messaging applications such as WhatsApp and Telegram are also frequently used. To direct users to a private chat with CS representatives, site administrators embed links on the IOG website, accompanied by information on the phone number or username of the CS account (Table 2).

**Table 2.** Statistics of customer services

Customer Service	Illegal Online Gambling Sites		Identifiers	
	Count	Percentage	Count	Average usage (sites)
LiveChat Account	9,237	26.2%	3,367	2.75
WhatsApp Number	9,081	25.8%	4,240	2.50
Telegram Account	4,536	12.9%	2,166	2.41

**Community.** IOG sites build online communities where users can join and exchange information and advice with other players. These communities serve not only as a channel for site administrators to provide customer support but also to disseminate up-to-date information about the IOG site. This includes promoting new sites, offering alternative access links, and distributing gambling applications. When a site domain is subject to blocking policies, the communication channels established through online communities remain intact, thereby facilitating the site to retain its existing customers. Moreover, the presence of online communities enables site operators to reach potential users more effectively while taking advantage of more difficult regulation and detection challenges [17]. IOG site operators leverage group features on social networking applications to build their communities, with Facebook groups being the most frequently used, followed by Telegram groups. To encourage user participation, site operators embed group links on their web pages (Table 3).

Based on the extraction of identifiers across the three components, we generally find that, on average, a unique identifier from a third-party service is used by more than two IOG sites. This suggests that several distinct IOG sites are operationally controlled by

**Table 3.** Statistics of online communities

Community	Illegal Online Gambling Sites		Identifiers	
	Count	Percentage	Count	Average usage (sites)
Facebook Group	2,734	7.8%	1,366	2.23
Telegram Group	1,555	4.4%	708	2.25
WhatsApp Group	261	0.7%	63	4.16

the same entity or business group, as evidenced by the relationships between sites based on the shared unique identifiers they used.

### 2.3 Multiplex Network of Illegal Online Gambling Sites

Based on preliminary analysis, we identify five key components in the ecosystem of IOG site operations: 1) infrastructure (IP addresses), 2) promotion and site navigation (hyperlinks), 3) analytics services used, 4) customer service (CS), and 5) online communities. In each component, we find that relational patterns between sites can be identified, both explicitly through site referrals (hyperlinks) and implicitly based on similarities in infrastructure and the use of third-party service identifiers. These relationships indicate that the connected gambling sites are owned by the same business entity.

**Relationship Between IOG Sites.** To ascertain implicit relationships between IOG sites, we first construct a bipartite graph  $G^\alpha = (L, \Gamma^\alpha, E^\alpha)$  where  $L \cap \Gamma^\alpha = \emptyset$ . Here,  $L = \{l_i\}$  represents the set of IOG sites,  $\Gamma^\alpha = \{\gamma_i^\alpha\}$  denotes the set of identifiers belonging to component  $\alpha$ , and  $E^\alpha$  is the set of edges connecting IOG sites to identifiers of type  $\alpha$ . We define  $C^\alpha = \{c_{ij}^\alpha\}$ , as the bi-adjacency matrix of dimensions  $|L| \times |\Gamma^\alpha|$ , with elements defined as follows:

$$c_{ij}^\alpha = \begin{cases} 1 & \text{if } (l_i, \gamma_j^\alpha) \in E^\alpha \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Given the bipartite graph  $G^\alpha$ , implicit relationships between IOG sites can be measured based on their co-occurrence, i.e. whether there are common neighbors for the two sites within the graph. Mathematically, this relationship can be represented by the adjacency matrix  $A^\alpha$  of dimensions  $|L| \times |L|$ , which constitutes a unipartite network of IOG sites  $L$ . The elements of this matrix are determined as follows:

$$a_{ij}^\alpha = \begin{cases} 1 & \text{if } \sum_k c_{ik}^\alpha c_{jk}^\alpha > 0 \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

We also represent the directly observed relationships between IOG sites based on hyperlinks as an adjacency matrix of an undirected graph. In this matrix,  $a_{ij} = a_{ji} = 1$  indicates the presence of hyperlink from site  $l_i$  to site  $l_j$  or vice versa. The value is 0 if no such relationship exists.

**Multiplex Network.** To investigate relational patterns between IOG sites and further characterize the IOG ecosystem, we utilize multiplex network model [18, 19]. A multiplex network is a type of multilayer network where there are no interconnections between nodes across different layers. In the multiplex network, each type of relationship between IOG sites, corresponding to different components of the ecosystem, is represented by a separate layer.

In our case, the multiplex network is composed of  $M = 5$  layers, each representing the network of relationships between sites for different components of the IOG ecosystem, namely IP, Hyperlink, Analytics, Customer Service (CS), and Community. By the construction of relationships between sites, we describe the multiplex network as a vector of adjacency matrices for  $M$  layers,

$$\mathbf{A} = \{A^1, \dots, A^M\} \quad (3)$$

where  $A^\alpha = \{a_{ij}^\alpha\}$  denotes the adjacency matrix associated with layer  $\alpha$ ,  $\alpha = 1, \dots, M$ . The degree of a node  $i$  in a given layer  $\alpha$  is defined as  $k_i^\alpha = \sum_{j \neq i} a_{ij}^\alpha$ . The mean degree of layer  $\alpha$ , denoted as  $z^\alpha$ , is computed as:

$$z^\alpha = \sum_{k^\alpha} k^\alpha p_{k^\alpha} \quad (4)$$

where  $p_{k^\alpha}$  denotes the probability of a randomly selected node has degree  $k^\alpha$ . Another important measure for characterizing the structural properties of a layer is degree assortativity coefficient  $r^\alpha$  [20], which can be calculated as follows:

$$r^\alpha = \frac{\sum_{ij} ij (e_{ij}^\alpha - q_i^\alpha q_j^\alpha)}{\sigma_q^2} \quad (5)$$

where  $e_{ij}^\alpha$  denotes the fraction of edges connecting nodes with degrees  $i - 1$  and  $j - 1$  within layer  $\alpha$ ,  $q_k^\alpha = \frac{(k^\alpha + 1)p_{k^\alpha} + 1}{z^\alpha}$ , and  $\sigma_q$  is the standard deviation of distribution of  $q^\alpha$ .

In addition to characterizing each layer separately within a multiplex network, it is also useful to analyze the aggregation of each network layer for comparative purposes. We define the aggregate network  $A = \{a_{ij}\}$ , which is an unweighted network with single layer or monoplex, and its elements are defined as follows:

$$a_{ij} = \begin{cases} 1 & \text{if } \exists \alpha : a_{ij}^\alpha = 1 \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

This network consists of each pair of nodes  $i$  and  $j$  that are connected by an edge in at least one of the layers of multiplex network. Previous studies have shown that this representation tends to be simplistic and insufficient for capturing the key characteristics of multilayer systems [19]. It has been recognized that it is important to consider the weighting of relationships based on the frequency with which an edge  $i - j$  appears across different layers. Therefore, we also analyze the aggregated overlapping network  $O = \{o_{ij}\}$ , where:

$$o_{ij} = \sum_\alpha a_{ij}^\alpha \quad (7)$$

with  $0 \leq o_{ij} \leq M$ . Based on this, we have the overlapping degree  $o_i = \sum_j o_{ij} = \sum_\alpha k_i^\alpha$  for a node  $i$  in aggregate network  $O$ .

**Jensen-Shannon distance.** We also conducted an analysis to compare the extent of the similarity between layers and to investigate how information of the relationship between IOG sites is structured in the multiplex network of the online gambling ecosystem. To answer this, we use the Jensen-Shannon distance metric which has been widely adopted to measure the similarity between two networks with same number of nodes, including two different layers in a multiplex network [18, 21].

Given adjacency matrix of a network  $A = \{a_{ij}\}$ , the density matrix  $\rho$  associated with  $A$  is defined as:

$$\rho = c(D - A) \quad (8)$$

where  $D$  is the diagonal matrix of the degrees of the nodes and  $c = 1/(\sum a_{ij})$ . One can then calculate the von Neumann entropy  $S$  of the density matrix  $\rho$  as follows:

$$S(\rho) = -\sum_i \lambda_i \log_2 \lambda_i \quad (9)$$

where  $\lambda_i$  denotes the  $i$ -th eigenvalue of matrix  $\rho$ . To measure the similarity between two layers,  $\alpha$  and  $\alpha'$ , we can compute the Jensen-Shannon distance  $d_{JS}$  as follows:

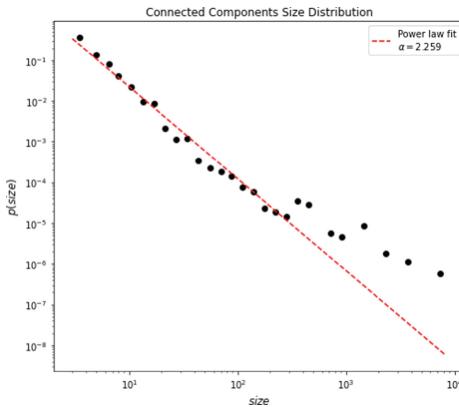
$$d_{JS}(\alpha, \alpha') = \sqrt{S\left(\frac{\rho_\alpha + \rho_{\alpha'}}{2}\right) - \frac{S(\rho_\alpha) + S(\rho_{\alpha'})}{2}} \quad (10)$$

where  $\rho_\alpha$  and  $\rho_{\alpha'}$  denote the density matrices corresponding to the adjacency matrices of the two layers,  $A^\alpha$  and  $A^{\alpha'}$ , respectively.

### 3 Results

#### 3.1 Structural Properties of IOG Site Multiplex Network

**Aggregate Network.** We begin our analysis by examining the aggregate network of IOG sites. This network comprises 35,229 nodes and 6,573,382 edges connecting the nodes. Within this network, there are 5,183 connected components, indicating that the network is rather fragmented into isolated clusters. Further analysis of the distribution of the sizes of these connected components reveals that the clusters vary significantly in size and even follow a power-law distribution  $p(x) \sim x^\alpha$  with exponent coefficient  $\alpha \approx 2.259$ . Notably, a small number of clusters contain a disproportionately large number of IOG sites compared to the majority of clusters. This finding suggests that most IOG groups manage a small number of sites, while only a few groups manage an exceptionally large number of sites (Fig. 1).



**Fig. 1.** Power law distribution of connected components size

Given the large number of connected components, we focus our analysis on the largest component by characterizing its network structure. This component comprises 8,295 nodes, representing over 20% of the IOG sites. As shown in Table 4, this component exhibits highly assortative properties, where the network as a whole tends to be less affected by disruption or removal of nodes compared to disassortative networks [22]. This also suggests a resilient nature of the IOG ecosystem, where targeting specific sites for disruption may be ineffective, as nearly every node holds comparable significance and can potentially substitute for others.

**Table 4.** Structural properties of giant component of aggregate network

Nodes	Edges	Density	Assortativity	Avg degree	Max degree	Min degree	Clustering coefficient	Avg. Path length
8,295	6,573,382	0.1910902	0.99993064	1,584.90	3,564	1	0.95299765	15.3749829

**Multiplex Network of Giant Component.** We continue our analysis by investigating the structural characteristics of each layer within the giant component. Table 5 presents the measurement results for each layer. In general, almost all layers exhibit assortative characteristics, indicating that high-degree nodes tend to connect with nodes of similar attributes. This is except for the Hyperlink layer, which is disassortative, where high-degree nodes are connected to nodes with lower degrees. Further investigation reveals that sites with higher degrees are promotional sites. This finding suggests a prevalence of promotional practices for IOG sites through landing pages, which direct users to numerous IOG site domains. Structurally, such sites function as hubs within the hyperlink-based network.

Another noteworthy finding is the relatively sparse nature of the network in almost all layers, except for the IP layer, which exhibits a significantly higher density. The IP layer also has the fewest connected components among the layers. These metrics suggest

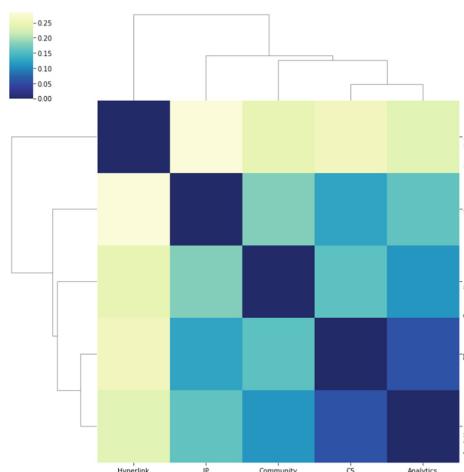
**Table 5.** Structural properties of multiplex layers of giant component

Layer	Nodes	Edges	Density	Connected components	Assortativity	Average degree
CS	8,295	8,187	0.0002380	6,126	0.99991742	1.97
Community	8,295	3,282	0.0000954	7,371	0.99983757	0.79
Analytics	8,295	52,657	0.0015308	6,084	0.99940439	12.70
IP	8,295	6,516,652	0.1894411	2,350	1.00000000	1,571.22
Hyperlink	8,295	346	0.0000101	7,975	-0.14420221	0.08

that, when viewed as an aggregate network, the IP layer acts as a connector between clusters that are otherwise isolated in the other layers. From operational infrastructure perspective, this finding also indicates that IOG sites owned by a same group are likely to share the same computing or network resources.

### 3.2 Hierarchically Nested Layers of IOG Sites Ecosystem

We applied similarity measurements for each possible pair of layers, resulting in Jensen-Shannon distance matrix between layers. The heatmap of this matrix, shown in Fig. 2, illustrates the varying degrees of similarity between layers. The Analytics and CS layers exhibit the highest similarity, indicating that these two layers provide relatively consistent information regarding the relationship patterns between IOG sites. This suggests that these layers, as key components of the ecosystem, play interrelated roles from the perspective of site administrators in monitoring and maintaining the performance of IOG sites based on player or customer activities.

**Fig. 2.** Jensen-Shannon distance matrix between layers

The results of the cluster analysis based on distance matrix indicate that the relational layers of IOG sites exhibit a hierarchically nested structure. The innermost layer consists of the Analytics and Customer Service (CS) layers, followed by the Community layer, with the IP and Hyperlink layers forming the outermost layers.

## 4 Discussion and Conclusion

Illegal online gambling (IOG) business groups develop many websites to reach a wide user base. Websites managed by the same group exhibit relatively similar appearances, which can be identified based on the key components in the IOG ecosystem. We model the network among sites into five layers of a multiplex network, where relationships between sites within these layers indicates that the sites belong to the same IOG group. Our findings reveal that the cluster size of these sites follows a power-law distribution, indicating that most IOG groups are small, with only a few groups being exceptionally large. This study then focuses on characterizing the structural network of IOG sites within the largest group.

In its management, IOG business owners consider three key aspects: manageability, reachability for new users, and operational security. The trade-offs among these three results in a network of relationships between sites structured into hierarchical, nested layers. At the Analytics and CS layers, efficiency in managing a large number of sites is a primary consideration. Consequently, at these layers, the network components are typically fully connected, with a single analytics tag deployed across multiple sites simultaneously, or a single CS channel managing many sites.

In the community layer, IOG business management focuses on reachability and security. To achieve this, they create numerous community groups to expand their reach to users and potential users. However, the number of sites sharing the same community is kept relatively small to minimize the impact when a community group is detected by security agencies, which could potentially disrupt traffic flow from the community group to the sites. Therefore, connectivity among sites at this layer is more fragmented compared to the previous two layers.

IOG eradication generally targets two things, namely blocking at the IP and domain levels. The assortative nature of the site networks means that these measures primarily affect only the outermost layer of the hierarchical relationship among sites. The characterization of IOG networks conducted in this study can provide insights for developing more targeted anti-gambling policies.

## References

1. Griffiths, M.D.: Crime and gambling: a brief overview of gambling fraud on the Internet. *Internet Journal of Criminology*. New University Press (2010)
2. McMullan, J.L., Rege, A.: Online crime and internet gambling. *Journal of Gambling Issues* **24**, 54 (2010)
3. Schmidt-Kessen, M.J., Hörnle, J., Littler, A.: Preventing Risks from Illegal Online Gambling Using Effective Legal Design on Landing Pages. *SSRN Electronic Journal* (2019)

4. French, M., Tardif, D., Kairouz, S., Savard, A.-C.: A Governmentality of Online Gambling: Quebec's Contested Internet Gambling Website Blocking Provisions. *Canadian Journal of Law and Society / Revue Canadienne Droit et Société* **36**(3), 483–504 (2021)
5. Frieß, J. et. al.: Cloudy with a Chance of Cyberattacks: Dangling Resources Abuse on Cloud Platforms. [arXiv:2403.19368v1](https://arxiv.org/abs/2403.19368v1) (2024)
6. Wu, S., et. al.: Reflected Search Poisoning for Illicit Promotion. [arXiv:2404.05320](https://arxiv.org/abs/2404.05320) (2024)
7. Gao, Y., et al.: Demystifying Illegal Mobile Gambling Apps. *Proceedings of the Web Conference 2021*(35), 1447–1458 (2021)
8. Yang, H., et. al.: Casino royale. In: *Proceedings of the 35th Annual Computer Security Applications Conference*, pp. 500–513 (2019)
9. Maulana, A., Lumbantobing, A., Keiluhu F.: Preliminary Analysis of Online Gambling Ecosystem. Internal Report WP-2–2024. Bandung Fe Institute (2024)
10. Oest, A., et. al.: Inside a phisher's mind: Understanding the anti-phishing ecosystem through phishing kit analysis. In: *2018 APWG Symposium on Electronic Crime Research (ECrime)* (2018)
11. John, J.P., et. al.: deSEO: Combating Search-Result Poisoning. *USENIX Security Symposium* (2011)
12. Liao, X., et. al.: Characterizing Long-tail SEO Spam on Cloud Web Hosting Services. In: *Proceedings of the 25th International Conference on World Wide Web* (2016)
13. Gruteser, M., Grunwald, D.: Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In: *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services* (2003)
14. Du, K., et. al.: The {Ever-Changing} labyrinth: A {Large-Scale} analysis of wildcard {DNS} powered blackhat {SEO}. In: *25th USENIX Security Symposium (USENIX Security 16)*, pp. 245–262. (2016)
15. Geng, G.G., et. al.: A Taxonomy of Hyperlink Hiding Techniques. (2013). [arXiv:1303.2438](https://arxiv.org/abs/1303.2438)
16. Mertens, G., et. al.: Google Tag Manager: Hidden Data Leaks and its Potential Violations under EU Data Protection Law. (2023). [arXiv:2312.08806v2](https://arxiv.org/abs/2312.08806v2)
17. Guo, Y., et. al.: Beyond App Markets: Demystifying Underground Mobile App Distribution Via Telegram. (2024). [arXiv:2408.03482](https://arxiv.org/abs/2408.03482)
18. De Domenico, M.: Multilayer Networks: Analysis and Visualization: Introduction to muxViz with R. Springer (2022)
19. Battiston, F., Nicosia, V., Latora, V.: Structural measures for multiplex networks. (2014). [arXiv:1308.3182v3](https://arxiv.org/abs/1308.3182v3)
20. Newman, M.J.: Mixing patterns in networks. (2003). [arXiv:cond-mat/0209450v2](https://arxiv.org/abs/cond-mat/0209450v2)
21. De Domenico, M., Nicosia, V., Arenas, A., Latora, V.: Structural reducibility of multilayer networks. *Nature Communications*, **6**(1) (2015)
22. Newman, M.J.: Assortative mixing in networks. (2002). [arXiv:cond-mat/0205405](https://arxiv.org/abs/cond-mat/0205405)



# Early Prediction of Power Outage Duration Through Hierarchical Spatiotemporal Multiplex Networks

Rafaa Aljurbua<sup>1,2(✉)</sup>, Jumanah Alshehri<sup>1,3</sup>, Shelly Gupta<sup>1</sup>,  
Abdulrahman Alharbi<sup>1,4</sup>, and Zoran Obradovic<sup>1</sup>

<sup>1</sup> Center for Data Analytics and Biomedical Informatics, Computer and Information Science Department, Temple University, Philadelphia, PA, USA

{rafaa.aljurbua,shehri.j,shelly.gupta,adalharbi,  
zoran.obradovic}@temple.edu

<sup>2</sup> Department of Computer Science, College of Computer, Qassim University, Buraydah, Saudi Arabia

<sup>3</sup> College of Business Administration, Imam Abdulrahman bin Faisal University, Dammam, Saudi Arabia

<sup>4</sup> Department of Computer Science, College of Engineering and Computer Science, Jazan University, Jazan, Saudi Arabia  
<https://dabi.temple.edu>

**Abstract.** Long power outages caused by weather can have a big impact on the economy, infrastructure, and quality of life in affected areas. It's hard to provide early and accurate warnings for these disruptions because severe weather often leads to missing weather recordings, making it difficult to make learning-based predictions. To address this challenge, we have developed HMN-RTS, a hierarchical multiplex network that classifies disruption severity by temporal learning from integrated weather recordings and social media posts. This new framework's multiplex network layers gather information about power outages, weather, lighting, land cover, transmission lines, and social media comments. Our study shows that this method effectively improves the accuracy of predicting the duration of weather-related outages. The HMN-RTS model improves 3 h ahead outage severity prediction, resulting in a 0.76 macro F1-score vs 0.51 for the best alternative for a five-class problem formulation. The HMN-RTS model provides useful predictions of outage duration 6 h ahead, enabling grid operators to implement outage mitigation strategies promptly. The results highlight the HMN-RTS's ability to offer early, reliable, and efficient risk assessment.

**Keywords:** power outage · multiplex networks · social media · spatiotemporal learning

## 1 Introduction

Weather conditions like freezing rain are often critical in predicting power outage risk and estimating the durations of power service disruptions, causing substan-

tial economic losses and affecting the quality of human life [13]. Previous research studied and analyzed the power outage problem from a statistical point of view, such as using quantile regression forests and Bayesian additive regression tree models [36]. In the new era of machine learning, research has moved toward predictive models, which use machine learning models to improve predictions of power outages and their causes [7, 22, 35]. For instance, a Random Forest is recently used to predict the risk of outage disruption [17]. Meanwhile, another disruption risk predictor is developed based on Bayesian deep learning in the KSTAR disruption database [16]. An adaptive ensemble learning approach is also applied for power outage prediction, while a hybrid mechanistic-machine learning outage risk prediction model is proposed to assess the effectiveness of various grid hardening actions [11]. Another study was focused on improving outage prediction risk models caused by a specific weather event such as rainfall [19]. Big data recorded by multiple phasor measurement units (PMUs) is used to improve the detection accuracy of outages in power systems [20]. Previous studies also utilized graph-based models to predict power outages [5, 28]. However, none of these studies utilized multiplex networks to enhance predictions; instead, they were restricted to using single-modality input.

Power outage predictions depend crucially on weather data, which often lacks many value during severe weather conditions. Previous studies explored methods to address this challenge, but the potential benefits of combining noisy weather data with information extracted from social media posts should be more adequately studied. In recent years, social media networks have become an integral part of society, allowing critical information to be communicated rapidly during severe weather events [27]. It has been found that weather conditions significantly affect tweeting behavior [15]. Social media data can provide information on weather-related impacts on infrastructure and human behavior and can also provide information back to observers [34]. In addition, a correlation was found between the number of weather-related tweets and the existing weather conditions [33]. Therefore, combining social media data with weather sensor data can provide valuable insights in predicting power outages caused by severe weather events.

While predicting power outages is crucial, it is essential to predict the duration of these outages to implement effective response and mitigation strategies. Several statistical methods were used to predict power outage duration [24]. Others applied machine learning models [6]. However, researchers have yet to fully explore the benefits of multiplex networks that utilize data from multiple modalities to predict power outage duration. In our previous study [1], multiplex graphs successfully predicted the occurrence of power outages three hours in advance at the county level. Still, our previous analysis did not include the duration of the outages. This limitation is addressed in our current study by developing HMN-RTS, an approach that forecasts the likelihood of power outages and their durations while expanding the prediction time intervals at the county level within the entire U.S. Pacific Northwest region. The HMN-RTS is a hierarchical spatiotemporal multiplex network model that leverages structured

data, such as weather and transmission line information, and unstructured data from social sensors collected across time and space. The HMN-RTS takes real-time input to update power outage duration prediction three hours in advance based on weather conditions and social media activity changes. This model provides earlier warnings of outage risks, potentially enhancing outage management efficiency and response efforts. The main contributions of this paper are as follows:

1. This study develops a principal framework for improving spatiotemporal classification by integrating multimodal data. It incorporates social media information to address the gaps in weather recording data.
2. A hierarchical county-level, spatiotemporal multiplex network multi-modal HMN-RTS approach is proposed to predict the duration of a power outage three hours ahead.
3. The HMN-RTS model updates the power outage duration prediction in real-time based on changes in weather and social media activity.
4. The HMN-RTS's predictive capabilities for estimating the duration of probable outages are evaluated across multiple time horizons, ranging from 3 to 24 h in advance.

## 2 Related Work

Three significant components summarize the related work for the proposed approach: 1) power outage duration prediction, 2) handling missing and incomplete weather data, and 3) hierarchical spatiotemporal multiplex network and multimodal data development for power outage duration prediction.

### 2.1 Weather Data

The Automated Surface Observing Systems (ASOS) ceilometer measures clouds at or below 12,000 ft (3.6 km), compromising ASOS data accuracy. Thus, missing data would result from incomplete atmospheric coverage [37]. Additionally, missing data is a common issue when dealing with sensor malfunctions and cloud contamination [10]. Numerous studies have examined various strategies for managing missing data. Mean imputation is the most commonly used method [32]. Common alternatives are disregarding records with missing values and replacing missing data with multiple imputations [30] or regression-based estimation [31]. However, missing data should be handled carefully for accurate analysis since ignoring missing instances can pose significant risks in the context of analysis.

### 2.2 Power Outage Duration Prediction

For utility companies to plan power restoration more effectively, predicting the duration of power outages early and accurately is essential. Statistical methods to predict power outage durations include accelerated failure time regression,

Cox proportional hazards regression, Bayesian additive regression trees, regression trees, and multivariate adaptive regression splines [24]. The researchers also combined statistical and geographic information systems to analyze the performance of a winter storm-affected urban distribution system by using a GIS to plot the data of the repair crews to study the duration of each outage [29]. In another study [18], Accelerated Failure Time (AFT) and Cox Proportional Hazard (CPH) were applied to estimate the duration of storm-caused power outages. The daily Night Time Lights (NTL) imagery data is also used to assess Puerto Rico's outage duration [2]. Previous studies show that socioeconomic factors influence outage duration [12, 23]. In recent years, researchers have moved toward machine learning models hoping to predict the duration of power outages. For instance, the duration of hurricane-related power outages is predicted by applying a Random Forests-based forecast mode using input variables such as wind duration and wind speed [25]. The duration during typhoon disasters is predicted by integrating Extra Tree (ET) [9], Extreme Gradient Boosting (XGBoost), Light Gradient Boosting Machine (LightGBM), Random Forest (RF), Gradient Boosting Regression (GBR), and Decision Tree (DT).

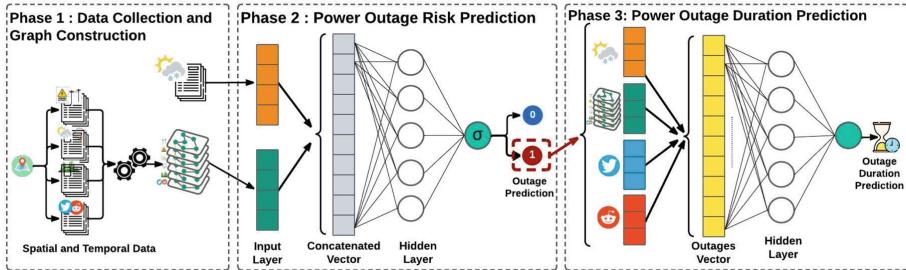
### 2.3 Multi-modal Learning in Ahierarchical Spatiotemporal Multiplex Network for Predicting Power Outage Duration

Weather data is often crucial for predicting power outages [14]. However, a high incidence of missing values in weather recordings during severe weather complicates learning. Our research combines weather data with social sensor data to address this challenge. We investigate the influence of social networks (multiplex networks) and social sensor data on predicting the duration of power outages. This study is one of the first to address power outage duration prediction three hours in advance by incorporating social sensors within a multiplex network representation. Precisely, we assess the advantages of learning from a spatiotemporal multiplex network that encompasses data from six sources: Bonneville Power Administration power outages, weather, lightning, Bonneville Power Administration transmission lines, land cover, and social sensor data from two prominent platforms: Reddit and Twitter.

## 3 Methodology

This study aims to assess whether the proposed HMN-RTS hierarchical model, which integrates multiplex networks and multi-modal data, enhances the prediction of power outage duration three hours ahead. The HMN-RTS model operates in two phases: first, predicting the risk for the occurrence of power outages, and second, estimating the severity of the predicted outage by predicting the duration of the expected outage, as shown in Fig. 1. The research is structured in three phases (1) data collection and spatiotemporal graph construction, (2) the development of the spatiotemporal multiplex network that estimates outage risk, and (3) the development of the model that predicts duration for expected outages.

Each component of this hierarchical framework is detailed in the subsequent subsections.



**Fig. 1.** The architecture of the HMN-RTS hierarchical spatiotemporal multiplex network for multi-modal prediction of power outage duration.

### 3.1 Data Collection

Previous research has highlighted the benefits of using data from multiple sources to enhance the prediction of power outage severity [1]. As the first step, we identify and collect critical factors to predict outage duration. These factors include power outage data, weather conditions, lightning, land cover, transmission lines, and social sensor data from two prominent platforms: Twitter (now referred to as X) and Reddit. This subsection provides an overview of the data collection process.

1. **Power outages:** We focus on the geographical area of the U.S. Pacific Northwest. We collected transmission services power outage events data covering a territory of more than 15,000 circuit miles over two years, from January 1, 2021, to December 31, 2022. This data is provided by Bonneville Power Administration (BPA), an American federal agency that operates in the U.S. Pacific Northwest. The BPA publicly reports all power outage events, regardless of cause. However, not every outage is relevant to this study; therefore, we only collect weather-related power outages, such as ice and lightning. We identify all weather-related power outages and map them to county and state using our dictionary that links every county to its Federal Information Processing Standards (FIPS) code. This results in 2,411 weather-related outages with a mean duration of 310 minutes<sup>1</sup>.
2. **Weather:** We collect historical weather data from Automated Surface Observing Systems (ASOS) stations. These stations are located at airports and include sensors to measure wind, ambient temperature, pressure, obstructions to vision, and sky conditions. We map each station to its county and state using latitude and longitude. As a result, we gather around 39 million weather observations in the five states<sup>2</sup>.

<sup>1</sup> <https://www.bpa.gov/>.

<sup>2</sup> <https://www.weather.gov/asos/>.

3. **Land cover:** Rapid tree growth or falling trees can damage power lines. To address this issue, we utilize land cover data obtained from the National Historical Geographic Information System (NHCIS). This dataset provides land cover features such as mixed, deciduous, and evergreen forests from the National Land Cover Database (NLCD). We leverage the GISJOIN identifier, which uses the FIPS code to designate counties and states [21].
4. **Lightning:** We obtain lightning information from the National Oceanic and Atmospheric Administration (NOAA) database<sup>3</sup>. As a result, we observe 7,015 lightning strikes from counties in the five states.
5. **Transmission lines:** This study focuses on transmission line outages rather than distribution line outages. We collect transmission line information in the BPA service region of the Northwest U.S. using the BPA map covering over 15,000 miles of transmission lines.
6. **Social sensor:** We gather social media activities using weather-related and power-outage-related keywords. **Twitter “currently known as X”:** first, we collect tweets using snscreape. Snscreape is a Python package designed to scrap historical tweets. We scrap tweets within a 10-mile radius of specified geographic coordinates (latitude and longitude) to capture a broad range of posts from the neighborhood. As a result, we collect 8.5 million relevant tweets about weather and power outage events. **Reddit:** we collect Reddit posts from counties’ subreddits using Reddit API. We collect all posts and comments from counties’ subreddits, followed by filtering the posts and comments using weather and power outage keywords. As a result, we obtained 353,421 posts, of which 95,144K were used after filtering and selection based on the keyword selection.

### 3.2 Modeling of the Spatiotemporal Multiplex Network

We then construct a Hierarchical Spatiotemporal Multiplex Network. Let  $G$  denote a spatiotemporal multiplex network defined as  $G(V, E, L, T)$ , where:  $V = \{v_1, v_2, \dots, v_n\}$  represents the set of vertices (counties),  $E = \{e_1, e_2, \dots, e_m\}$  represents the set of edges,  $L = \{l_1, l_2, l_3, l_4, l_5, l_6\}$  represents the set of layers, and  $T = \{t_1, t_2, \dots, t_k\}$  represents a set of time steps. The edges  $E$  within each layer  $L$  signify a unique type of relationship among the vertices  $V$ . These relationships are described as follows:

1. **Transmission lines layer:** In layer  $l_1$ , two counties  $(u_{l_1}, v_{l_1})$  are linked if they share the same transmission line. The edge weight represents the number of transmission lines shared between the counties.
2. **Power outage layer:** In layer  $l_2$ , two counties  $(u_{l_2}, v_{l_2})$  are linked if both report a power outage on the same date. The edge weight represents the number of shared power outages between the counties.

---

<sup>3</sup> <https://www.ncei.noaa.gov/>.

3. **Weather layer:** In some cases, power outages are closely related to weather conditions; therefore, in layer  $l_3$ , we connect nodes representing two counties  $(u_{l_3}, v_{l_3})$  that share similar weather properties. In this context, we compute the Euclidean distance between each pair of vertices.
4. **Lightning layer:** In layer  $l_4$ , two counties  $(u_{l_4}, v_{l_4})$  are linked if both report a lightning strike on the same day. The edge weight represents the number of shared lightning strikes between the counties.
5. **Land cover layer:** In layer  $l_5$ , two counties  $(u_{l_5}, v_{l_5})$  are linked if they share similar land cover properties. Here, we calculate the Euclidean distance between vertices.
6. **Social sensor layers:** In layer  $l_6$ , two counties  $(u_{l_6}, v_{l_6})$  are linked if both report social media activities during the power outage. The edge weight represents the number of shared social media activities, such as Tweets or Reddit posts, between the counties.

This graph serves as input for the HMN-RTS model. At the end of each day, a new multiplex graph snapshot, timestamped with that day's data is added to capture the interdependencies between counties. We train the model using the snapshot as input, generating embeddings for the county nodes through the proposed method, which is explained in detail in the following subsection. We aim to predict whether a county will experience a power outage. If an outage is anticipated, we estimate the severity by forecasting its duration. The duration prediction is based on county node embeddings and real-time social media data (from Twitter and Reddit) related to weather conditions and power outages.

### 3.3 Proposed Model: Hierarchical Spatiotemporal Multiplex Network (HMN-RTS)

This study evaluates whether a hierarchical multiplex network and multi-modal data approach can enhance the prediction of disruption severity three hours in advance. First, we use an annotated dataset to identify weather-related power outages, noting that the frequency of these outages varies across different states and counties. Each data point is labeled as one if a power outage occurs in a county during a specific time frame and 0 otherwise. Second, to forecast outage duration, we classify the outages into five categories, as outlined in Table 1. The model architecture consists of two phases: (1) prediction of the power outage occurrence followed by (2) prediction of the severity of the power outage by estimating the duration of power outages.

**Power Outage Risk Prediction.** The model uses the multiplex snapshot to generate county node embeddings via a modified version of Node2Vec [8] that incorporates multiple graph layers. These embeddings are then combined with weather data from ASOS and fed into a neural network model consisting of three fully connected layers, with dropout layers in between to mitigate overfitting.

Our dataset is inherently imbalanced, with fewer than two percent of instances classified as outages. We find that the Synthetic Minority Oversampling Technique (SMOTE) [3] is the best technique to address the imbalance issue in the training data. Binary cross entropy is used to train this part of the HMN-RTS architecture. This effectively isolates instances where the model correctly identified outages.

**Power Outage Duration Prediction.** Once a power outage is predicted for a county, the second part of the model aims to predict the duration of the outage three hours ahead through multiclass classification. First, we gather ASOS weather station recordings within each county at 30-minute intervals. Since there may be multiple stations and recordings per station within a county, we aggregate the data by calculating the mean and standard deviation for each feature, which normalizes the weather data and provides a comprehensive view of local conditions.

Next, we collect all Reddit posts from the U.S. Pacific Northwest region and use BERT [4] to generate 768-dimensional embeddings for each post. We apply max pooling for intervals with multiple posts to create a single representative vector. Similarly, for Twitter data, we gather all tweets posted within each 30-minute interval across the U.S. Pacific Northwest and use BERTweet [26] to generate 768-dimensional vectors for each tweet. Again, we use max pooling to aggregate these vectors into a single representative vector for each interval. This approach effectively captures the most significant patterns from social media, providing the model with a concise yet robust input for each 30-minute segment. Finally, we concatenate the weather data, Reddit vectors, Twitter vectors, and multiplex model embeddings, using this combined dataset as input for the model to predict outage duration. The combined input is fed into a multiclass neural network consisting of three fully connected layers with ReLU activations and interspersed dropout layers to reduce overfitting. The model is trained using multiclass cross-entropy loss to optimize its performance in predicting outage durations across the defined classes. The multiclass cross-entropy loss measures the difference between the predicted probabilities  $\hat{y}$  and the actual labels  $y$ .

$$\text{Loss}(\hat{y}, y) = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^C y_{ij} \log(\hat{y}_{ij}) \quad (1)$$

Here  $\hat{y}_{ij}$  is the predicted probability that the  $i$ th sample belongs to the  $j$ th class.  $y_{ij}$  is the actual label for the  $i$ th sample in the  $j$ th class, which is one if the  $i$ th sample belongs to the  $j$ th class and 0 otherwise.  $N$  is the total number of samples in the dataset.  $C$  is the total number of classes.

## 4 Experimental Setup

This study aims to determine if hierarchical multiplex network and multi-modal data method can improve early classification of disruption severity to one of five

categories (short to very long time). We use two years of data for training and testing. We use data from 2021-01-01 to 2022-06-30 for training and evaluate the model on disjoint data from 2022-07-01 to 2022-12-31. The model is optimized using the *Adam* optimizer and trained for 100 epochs with a batch size 32 and a learning rate of 0.0001. We conduct machine learning experiments to predict the duration of disruptions caused by the occurrence of power outages. Our learning approach is supervised machine learning based. We compare the HMN-RTS model vs Neural Network (NN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Reddit and Twitter Multiplex Network (RTMNO) [1]. We choose metrics suitable for a power system setting to evaluate the model's performance. We select metrics appropriate for a power system context to assess model performance.

Given that our classification involves five classes, we use macro-averaging, which assigns equal weight to each class irrespective of the class frequency. We use macro precision and recall assessing the rates of false positives and false negatives. Moreover, we evaluate the model's performance on the test set using the macro F1 scores for each class. Given that  $C$  represents the number of classes, macro F1 can be defined as

$$\text{Macro F1} = \frac{1}{C} \sum_{i=1}^C 2 \cdot \frac{\text{precision}_i \cdot \text{recall}_i}{\text{precision}_i + \text{recall}_i} \quad (2)$$

**Table 1.** Distribution of power outage durations in BPA service territory across five classes of duration in the years 2021-2022.

Class	Duration	Percentage
Class 1	Less than 30 min	63%
Class 2	30 min to 1 h	1%
Class 3	1 to 3 h	8%
Class 4	3 to 6 h	6%
Class 5	Greater than 6 h	22%

## 5 Results and Discussion

The results for different evaluation metrics of NN, RNN, LSTM, RTMNO, and the proposed Hierarchical Multiplex Network model (HMN-RTS) are shown in Table 2. The performance of the HMN-RTS model outperformed the performance of the traditional models. In addition, the HMN-RTS model obtains a higher macro F1 score in all models considered. As a result, the Hierarchical Multiplex Network model achieves a macro F1 score of 0.76. Further, we can

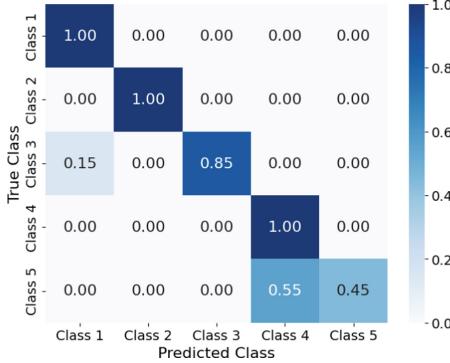
observe that the RTMO model enhances the prediction performance with performance improvement up to 30% versus alternative models. In contrast, despite optimizing the baseline models (NN, RNN, and LSTM), they resulted in macro F1 scores of only 0.16, 0.19, and 0.21, respectively. This can be explained by the limited capabilities of baseline models to capture context compared to our HMN-RTS model.

**Table 2.** Comparison of macro precision, macro recall, and macro F1 score of Neural Network (NN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Reddit and Twitter Multiplex Network (RTMNO), and the Hierarchical Multiplex Network model (HMN-RTS). Outage duration is classified into five classes defined in Table 1, and the models are evaluated on BPA outage data from 2022-07-01 to 2022-12-31.

Model	Macro precision	Macro recall	Macro F1 Score
NN	0.15	0.21	0.16
RNN	0.17	0.22	0.19
LSTM	0.18	0.24	0.21
RTMNO[1]	0.55	0.50	0.51
HMN-RTS	<b>0.83</b>	<b>0.86</b>	<b>0.76</b>

Further, we evaluate the HMN-RTS model performance using a confusion matrix, which illustrates the number of correct and incorrect predictions for each class, as shown in Fig. 2. Note that Class 1 means duration less than 30 min, Class 2 is for duration between 30 mins to 1 h, Class 3 is for duration between 1 to 3 h, Class 4 is for duration between 3 to 6 h, and Class 5 is for duration greater than 6 h. Due to the ordinal nature of our classes, the model can distinguish between classes with small intervals, while the model struggles to distinguish between classes with high intervals. In the HMN-RTS model, the macro F1 score is lower than macro precision and macro recall because it considers both precision and recall in all classes and is sensitive to lower values in either metric. To investigate this phenomenon, we examine the prediction of each class. Table 3 shows the results of the HMN-RTS model per class. While precision and recall are relatively high for most classes, the lower precision 0.14 and recall 0.45 values cause the macro F1 score to drop significantly, reflecting the harmonic mean's sensitivity to lower values.

The second set of experiments performs an ablation study. We create five different versions of HMN-RTS for this experiment. HMN-R is the model that only utilizes Reddit information. It is modified to process Reddit data only. HMN-T is the model that only uses Twitter information. It is adjusted to handle only Twitter data. HMN-S is the model that only employs multiplex network structure. The model is adjusted to handle multiplex network data. HMN-RT is the model that uses both Reddit and Twitter information, modified to process Reddit and Twitter data. Finally, HMN-RTS is the model that employs Reddit,



**Fig. 2.** Normalized confusion matrix of the HMN-RTS model predicting duration of outages for 2022-07-01 to 2022-12-31. Class 1 means duration less than 30 min, Class 2 is for duration between 30 minutes to 1 h, Class 3 is for duration between 1 to 3 h, Class 4 is for duration between 3 to 6 h, and Class 5 is for duration greater than 6 h.

**Table 3.** Comparison of precision, recall, and F1 score for every class using the HMN-RTS model trained on data from 2021-01-01 to 2022-06-30 and evaluated on 2022-07-01 to 2022-12-31.

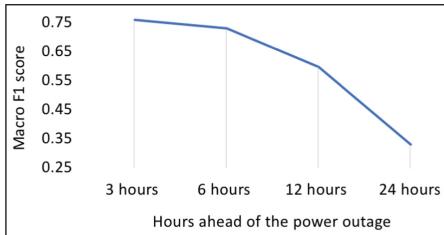
Class	Precision	Recall	F1 Score
Class 1	0.99	1	0.99
Class 2	1	1	1
Class 3	1	0.85	0.92
Class 4	0.145	1	0.25
Class 5	1	0.45	0.62

Twitter, and the multiplex network structure. Table 4 outlines the model's performance. It outperforms all other configurations when the HMN-RTS leverages multi-modal data and the network structure.

The third experiment evaluates the HMN-RTS model's ability to predict outcomes at earlier stages. In power systems, predictions made earlier provide more significant benefits. Figure 3 shows the macro F1 score for early prediction. The macro F1 score indicates the HMN-RTS model's performance in early detection. The X-axis shows the number of hours before the power outage when the prediction is made. We can see that the macro F1 score increases as the prediction time approaches the power outage event. This is expected, as impacts are generally more predictable over shorter periods. We can also observe that the proposed HMN-RTS model can predict power outages with a high macro F1 score up to 6 hours before the event. Furthermore, it can make predictions up to 12 hours in advance with slightly reduced performance, which is still adequate for taking appropriate measures to mitigate the side effects of the outage.

**Table 4.** Ablation study of the HMN-RTS model. Multiplex Network Structure includes layers 1–5 defined in Sect. 3.2 while Reddit and Twitter activities are captured by separate layers.

Settings	Reddit	Twitter	Multiplex Network Structure	Macro precision	Macro recall	Macro F1 Score
HMN-R	✓			0.66	0.73	0.66
HMN-T		✓		0.63	0.79	0.68
HMN-S			✓	0.67	0.80	0.71
HMN-RT	✓	✓		0.83	0.82	0.74
HMN-RTS	✓	✓	✓	<b>0.83</b>	<b>0.86</b>	<b>0.76</b>



**Fig. 3.** The HMN-RTS model performance in early detection of outages at the BPA service territory for 2022-07-01 to 2022-12-31 is indicated by the macro F1 score for a five-class problem formulation defined at Table 1. The X-axis shows the number of hours before the power outage when the prediction is made.

## 6 Conclusion

Power outages can affect daily life, such as transportation and communication. Therefore, predicting the severity of power outages is critical for effective planning. This study proposes a county-level hierarchical multiplex network-based methodology in the U.S. Pacific Northwest. The proposed approach predicts the occurrence of weather-related power outages along with the power outage duration. The HMN-RTS model predicts power outage occurrences and severity across multiple time horizons, including 3, 6, 12, and 24-hour intervals. We use multi-modal data collected over time and space to provide earlier predictions of outage duration. We quantify the benefit of integrating weather-related disruption information with people’s behavior.

Consequently, we consider information derived from social sensors. We assess whether the proposed hierarchical multiplex network method can learn better and enhance the prediction performance compared to other machine learning models. Our hierarchical spatiotemporal multiplex network provides a novel way of encoding online social sensor data into a network-based approach that can improve three hours ahead duration prediction accuracy for power outages with a macro F1 score of 0.76, enabling grid operators to implement outage mitigation strategies promptly. The limitation of this work is that the distribution of user ages impacts how they engage on social media platforms, which should be

examined in a follow-up study. Future research can explore social sensor data from a broader range of age demographics. In addition, a follow-up study is needed to evaluate our model across different geographic locations and extended prediction horizons (e.g., 36 hours).

**Acknowledgment.** This research was sponsored by the U. S. Army Engineer Research and Development Center (ERDC) and was accomplished under Cooperative Agreement Number W9132V-23-2-0002. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Engineer and Development Center (ERDC) or the U.S. Government.

## References

1. Aljurbua, R., Alshehri, J., Alharbi, A., Power, W., Obradovic, Z.: Social media sensors for weather-caused outage prediction based on spatio-temporal multiplex network representation. *IEEE Access* **11**, 125883–125896 (2023)
2. Azad, S., Ghandehari, M.: A study on the association of socioeconomic and physical cofactors contributing to power restoration after hurricane Maria. *IEEE Access* **9**, 98654–98664 (2021)
3. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **16**, 321–357 (2002)
4. Devlin, J.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) (2018)
5. Dokic, T., Pavlovski, M.: Spatially aware ensemble-based learning to predict weather-related outages in transmission. In: The Hawaii International Conference on System Sciences—HICSS, Maui, Hawaii, January 2019 (2019)
6. Doshi, R., Saini, R.D., Kansal, S.: Time duration prediction of electrical power outages. In: Smart Structures in Energy Infrastructure: Proceedings of ICRTE 2021, vol. 2, pp. 31–40. Springer (2022)
7. Eskandarpour, R., Khodaei, A.: Machine learning based power grid outage prediction in response to extreme events. *IEEE Trans. Power Syst.* **32**(4), 3315–3316 (2016)
8. Grover, A., Leskovec, J.: Node2Vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
9. Hou, H., Liu, C., Wei, R., He, H., Wang, L., Li, W.: Outage duration prediction under typhoon disaster with stacking ensemble learning. *Reliab. Eng. Syst. Safety* **237**, 109398 (2023)
10. Huang, W., Deng, Y., Hui, S., Wang, J.: Image inpainting with bilateral convolution. *Remote Sens.* **14**(23), 6140 (2022)
11. Hughes, W., et al.: Assessing grid hardening strategies to improve power system performance during storms using a hybrid mechanistic-machine learning outage prediction model. *Reliab. Eng. Syst. Safety* **248**, 110169 (2024)
12. Jasiūnas, J., Lund, P.D., Mikkola, J., Koskela, L.: Linking socio-economic aspects to power system disruption models. *Energy* **222**, 119928 (2021)
13. Jones, K.F.: Freezing fraction in freezing rain. *Weather Forecast.* **37**(1), 163–178 (2022)

14. Kezunovic, M., Pinson, P., Obradovic, Z., Grijalva, S., Hong, T., Bessa, R.: Big data analytics for future electricity grids. *Electr. Power Syst. Res.* **189**, 106788 (2020)
15. Kiciman, E.: OMG, I have to tweet that! a study of factors that influence tweet rates. In: *Proceedings Of The International AAAI Conference On Web And Social Media*, vol. 6, pp. 170–177 (2012)
16. Kim, J., Lee, J., Seo, J., Ghim, Y.C., Lee, Y., Na, Y.S.: Enhancing disruption prediction through Bayesian neural network in KSTAR. *Plasma Phys. Control. Fusion* **66**(7), 075001 (2024)
17. Lee, J., et al.: Data-driven disruption prediction using random forest in KSTAR. *Fusion Eng. Des.* **199**, 114128 (2024)
18. Liu, H., Davidson, R.A., Apanasovich, T.V.: Statistical forecasting of electric power restoration times in hurricanes and ice storms. *IEEE Trans. Power Syst.* **22**(4), 2270–2279 (2007)
19. Liu, W., Yang, Y., Xu, Q., Xia, Y.: Multi-target prediction model of urban distribution system rainfall-caused outage based on spatiotemporal fusion. *Int. J. Electr. Power Energy Syst.* **146**, 108640 (2023)
20. Ma, H., Lei, X., Li, Z., Yu, S., Liu, B., Dong, X.: Deep-learning based power system events detection technology using spatio-temporal and frequency information. *IEEE J. Emerg. Sel. Topics Circ. Syst.* **13**(2), 545–556 (2023)
21. Manson, S.M.: IPUMS national historical geographic information system: version 15.0 (2020)
22. Mensah, A.F., Dueñas-Osorio, L.: Outage predictions of electric power systems under hurricane winds by Bayesian networks. In: *2014 International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pp. 1–6. IEEE (2014)
23. Mitsova, D., Escaleras, M., Sapat, A., Esnard, A.M., Lamadrid, A.J.: The effects of infrastructure service disruptions and socio-economic vulnerability on hurricane recovery. *Sustainability* **11**(2), 516 (2019)
24. Nateghi, R., Guikema, S.D., Quiring, S.M.: Comparison and validation of statistical methods for predicting power outage durations in the event of hurricanes. *Risk Anal. Int. J.* **31**(12), 1897–1906 (2011)
25. Nateghi, R., Guikema, S.D., Quiring, S.M.: Forecasting hurricane-induced power outage durations. *Nat. Hazards* **74**, 1795–1811 (2014)
26. Nguyen, D.Q., Vu, T., Nguyen, A.T.: BERTweet: a pre-trained language model for English Tweet. arXiv preprint [arXiv:2005.10200](https://arxiv.org/abs/2005.10200) (2020)
27. Otudi, H., Gupta, S., Albarakati, N., Obradovic, Z.: Classifying severe weather events by utilizing social sensor data and social network analysis. In: *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*, pp. 64–71 (2023)
28. Owerko, D., Gama, F., Ribeiro, A.: Predicting power outages using graph neural networks. In: *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 743–747. IEEE (2018)
29. Reed, D.A.: Electric utility distribution analysis for extreme winds. *J. Wind Eng. Ind. Aerodyn.* **96**(1), 123–140 (2008)
30. Robertson, J., Kaptein, M.: Modern statistical methods for HCI, vol. 6. Springer (2016)
31. Streiner, D.V.L.: The case of the missing data: methods of dealing with dropouts and other research vagaries. *Canadian J. Psychiatry* **47**(1), 68–75 (2002)
32. Tanguma, J.: A review of the literature on missing data (2000)

33. Uddin, M.Y.S., Al Amin, M.T., Le, H., Abdelzaher, T., Szymanski, B., Nguyen, T.: On diversifying source selection in social sensing. In: 2012 Ninth International Conference on Networked Sensing (INSS), pp. 1–8. IEEE (2012)
34. Weyrich, P., Ruin, I., Terti, G., Scolobig, A.: Using serious games to evaluate the potential of social media information in early warning disaster management. *Int. J. Disaster Risk Reduction* **56**, 102053 (2021)
35. Xu, Z., Liu, J., Fan, W., Wang, Y., Luan, L., Zhou, K.: An power outage prediction method based on XGBoost with improved objective function. In: 2022 37th Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 1560–1565. IEEE (2022)
36. Yang, F., Wanik, D.W., Cerrai, D., Bhuiyan, M.A.E., Anagnostou, E.N.: Quantifying uncertainty in machine learning-based power outage prediction model training: a tool for sustainable storm restoration. *Sustainability* **12**(4), 1525 (2020)
37. Yang, X., You, Z., Hiller, J., Watkins, D.: Updating and augmenting weather data for pavement mechanistic-empirical design using ASOS/AWOS database in Michigan. *Int. J. Pavement Eng.* **19**(11), 1025–1033 (2018)



# Robustness of Short-Term Memory Models on Networks in Spatial Prisoner's Dilemma

Akihiro Takahara<sup>(✉)</sup> and Tomoko Sakiyama

Soka University, 1-236, Tangi-Choho, Tokyo, Japan  
e23m5324@soka-u.jp, sakiyama@soka.ac.jp

**Abstract.** In this study, the Prisoner's Dilemma is implemented using a network. In the conventional model, the problem is that as the parameters increase, the number of Defectors increases, and Co-operators are eliminated. To solve this problem, we have introduced memory into our model. However, it is well known that Co-operators cannot be maintained simply by introducing memory unless a long past is considered, but it is not realistic to remember the long past accurately. Therefore, we were able to create a model that can maintain Co-operators even with short-term memory by introducing a rule to update to a strategy that has not been experienced much in the past. In particular, this model focuses on one past interaction of a player and adjacent players to consider the neighbour's influence.

**Keywords:** Spatial prisoners dilemma · Memory · Cooperative evolution

## 1 Introduction

Cooperative behaviour is a common phenomenon in groups. However, in conventional spatial game theory (SPD), where Co-operators (C) and Defectors (D) interact in a group and adopt the more rewarding strategy, C is eliminated by D over time [1–3]. One of the objectives of this study is to solve this problem and maintain C. This study focuses on addressing this issue using the Prisoner's Dilemma in game theory.

Recently, a Twisted Prisoner's Dilemma (TPD) model has been developed, incorporating players' memories of past strategies and occasionally disregarding traditional strategy update rules [4]. The motivation for using memory is to reflect real-world dynamics in the model, and many studies have utilized memory. In the study by Dank (2019), two models were proposed: one that aggregates past payoffs and uses them in the present, and another that uses payoffs from a specific point in the past. In this model, it was necessary to consider a long past [5].

Similarly, in the study by Zhiqiang Gou (2023), a model was devised to evaluate past stability. This model suggests that if an individual consistently experiences the same strategy over a defined period, they are considered stable and less likely to rely on nearby strategies. Conversely, if they lack stability, they are more inclined to imitate others' strategies [6]. This study also indicates that long memory is necessary throughout the process. However, as previously mentioned, it is not realistic to remember a long past accurately.

On the other hand, Feng Shu (2019) developed a model to obtain a general strategy within the range of interactions to avoid risks due to the pursuit of high scores. In this model, the strategy with the highest score in both the individual's past and the neighbour's past is used to perform a new virtual interaction, updating the strategy based on past experiences. The study found that using a very short past can contribute to the maintenance of Co-operators [7]. However, if the length of the past considered is extended, Defectors are more likely to increase. Thus, while a short past contributes to maintaining C, considering too short a past is not realistic, and the model becomes more complex.

In a previous model, the frequency of strategy occurrence was calculated based on each player's memory, making it easier for less frequently adopted strategies to be chosen by ignoring traditional strategy update rules of the SPD model. This model assumes that players can only access recent memories, making it more realistic than traditional SPD models. Consequently, we created a model that contributes to maintaining C even if the length of past interactions is varied [4, 8, 9]. Our model can maintain C more effectively compared to Dank and Zhiqiang's models, which require a long past. Furthermore, our model is similar to Feng Shu's (2019) model but avoids complications by using simple memory and contributes to C maintenance even with short or varying lengths of memory. Previously, we used lattice space in our model, but in this paper, we use a random network as an SPD system. Recently, many studies in game theory have utilized networks [10–14]. Similarly, in this paper, we created a new model that can access short memories in the neighbourhood while using the above rules. We will investigate whether the network can also contribute to the maintenance of C.

## 2 Methods

### 2.1 Initial Condition

In this paper, we set the number of players to 1000. The average degree of nodes was pre-configured, and players were randomly connected to other players. Two average degrees of nodes were used: 5 and 50. In this study, a random network is used. In the results, we discuss changes in the average degree of nodes. A smaller average degree increases the likelihood of a player becoming isolated. All players were randomly assigned a strategy of C or D.

### 2.2 SPD Model (Spatial Prisoner's Dilemma Model)

The payoff was arranged as  $T = b$ ,  $R = 1$ ,  $S = P = 0$  as shown in Table 1, where  $T > R > P = S$ . The parameter  $b$  was defined as  $1 < b < 2$  (Nowak & May, 1992).

When the strategy of a connected player is C, a player receives R if the player's strategy is C. When the strategy of a connected player is D, a player receives S if the player's strategy is D. When the strategy of a connected player is C, a player receives P if the player's strategy is D. When the strategy of a connected player is D, a player receives T if the player's strategy is C. All players compare their strategies with all players to which their node is connected according to the rules of the payoff matrix. Then, the

**Table 1.** Payoff matrix

	C	D
C	R(1)	P(0)
D	T(b)	S(0)

payoffs obtained are summed as the player's score. After that, all players compare their strategy with all players to which their node is connected, and update their own strategy to the strategy of the player who has the highest score in the adjacent players. This process of comparison and update is defined as 1 time step. It is repeated for 1000-time steps as 1 trial.

### 2.3 PPD Model (Perverse Prisoner's Dilemma Model)

In this paper, we named the proposed model as the PPD model (Perverse Prisoner's Dilemma model). In this model, same rules with the SPD model are used except under certain conditions regarding the strategy update. The different sub-model of this model from the SPD model is ran when the score of connected players for a player is higher than that player's score and all of those players' strategies are same with each other. However, if there are two or more adjacent players with different strategies and the same maximum score, the players do not update their strategies. If the above situation is satisfied, the player tries to update it strategy in a different manner from so far and uses the past information. Strictly, the player considers the current and previous strategies of their own and the adjacent players. The player counts the number of strategies of C and records it in  $n_C^t(i)$  as the following.

$$n_C^t(i) = \left( \sum_{k=0}^1 \text{if } \text{strategy}(i)^{t-k} = C \right) + \left( \sum_{j \in N(i)} \sum_{k=0}^1 \text{if } \text{strategy}(j)^{t-k} = C \right)$$

where  $(i)$  denotes the player number and  $(j)$  denotes the neighbor number connected to the player. In addition,  $\text{strategy}(i)^{t-k}$  indicates the strategy of the player. Also,  $\text{strategy}(j)^{t-k}$  indicates the strategy of neighbors.  $t$  is the time step. Then, with the following probability, the player updates its strategy to C:

$$1 - (n_C^t(i)/L(i))$$

Also, with the following probability, the player updates its strategy to D:

$$(n_C^t(i)/L(i))$$

The value of  $L(i)$  indicates that the number of strategies what own player and neighbors in the past and present. Therefore, the  $L$  is determined by the player and the number of adjacent players. For example, if there are five adjacent players, the value of L, including the player, is 12, since the present and past strategies are used. However, players

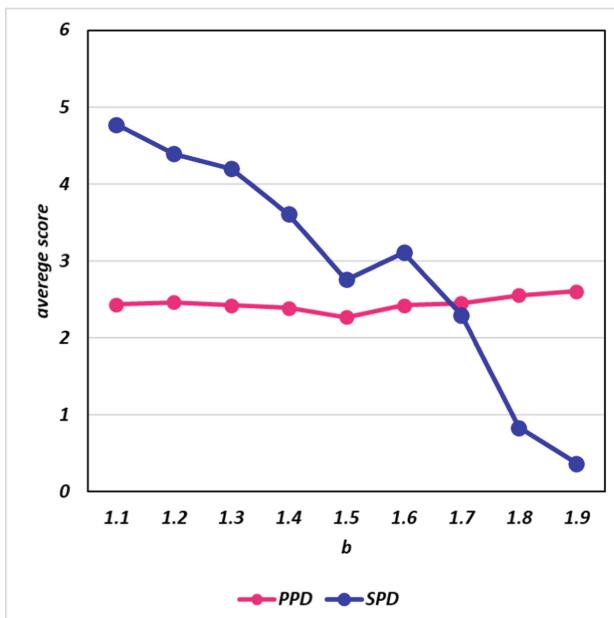
without connections do not use probabilities and do not update their strategies. In this strategy update rule, if there is only one type of strategy in all the referenced past or current strategies, the player is sure to adopt another strategy. Since this model is aimed to overcome the situation by oneself, the more the strategy of the neighbour is biased to one side, it is easier for players with low scores to change the rules for updating their own strategies.

### 3 Results

In this paper, we examined the average score and defector density at the end of each trial, with results averaged over 100 trials. This calculation was done for each value of  $b$ . The results of the average score are shown in Fig. 1. The PPD model scores lower than the SPD model at the lowest value of  $b$ , but as  $b$  increases, the PPD model scores higher than the SPD model.

Figure 2 shows the results for the defector density. The PPD model is stable as shown in Fig. 1. Compared to the SPD model, the PPD model maintains strategy C even as  $b$  increases.

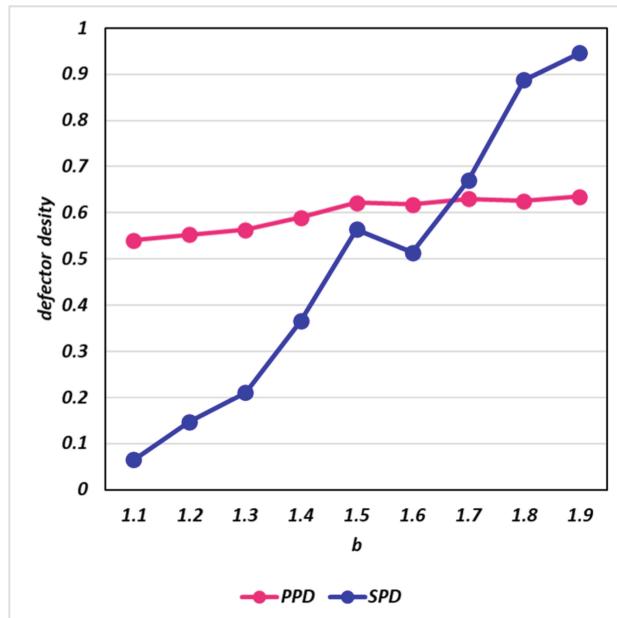
Next, defector densities are shown in Fig. 3 for various average degrees of nodes. Some average degrees of nodes for individuals were set as 1 to 40. As a result, the PPD model has a slightly higher defector density than the SPD model at average degree of nodes of 1. This is the case when the number of connections is extremely small, in which case the interaction of strategies is unlikely to occur in both models. When a



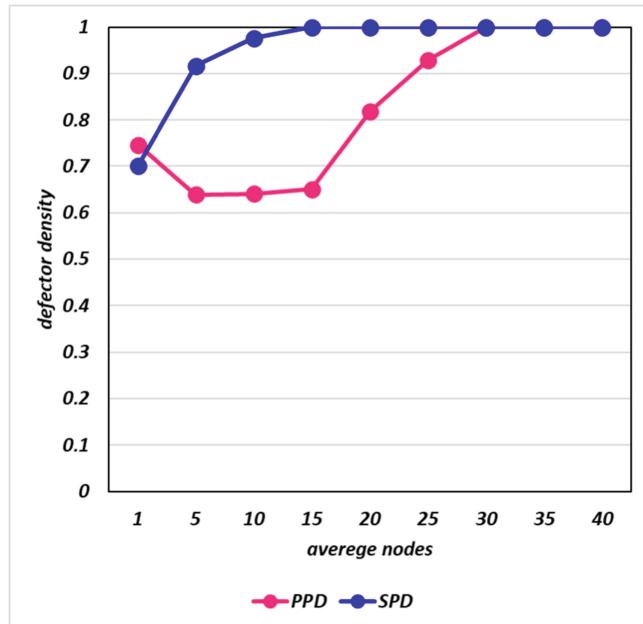
**Fig. 1.** Average score for each model from 100 trials. The average degrees of nodes are 5.

small connection has been formed, if there are three players holding C connected together without interference from D, the situation will stagnate in the SPD model. However, in the PPD model, the PPD rule occurs and D appears. We think that the difference is shown in Fig. 3. For the average degree of nodes after that, the PPD model is able to maintain the average degree of nodes up to 25, compared to the SPD model. However, it seems that the defector density becomes 1 for both models after that. A higher degree of connectivity implies that strategies can spread more easily. In addition, if the node player has D, then the payoff matrix will make D the strongest strategy, and as a result, D is likely to spread fairly quickly.

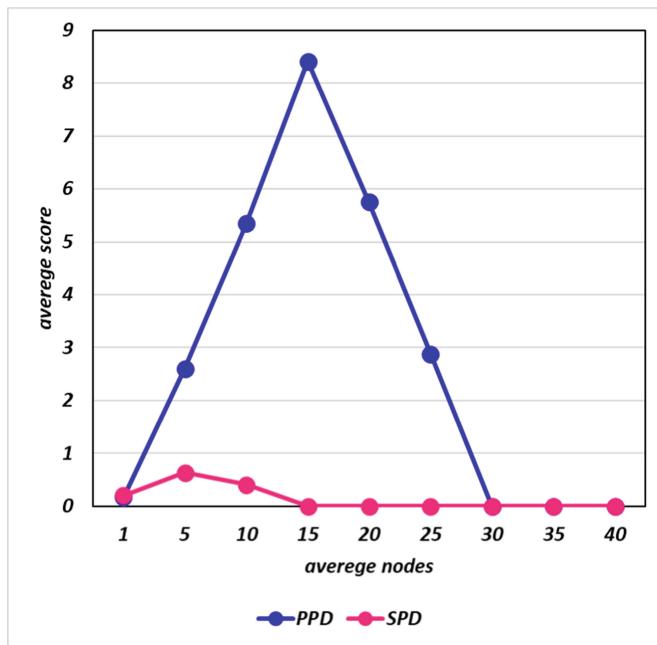
Finally, average score is shown in Fig. 4 for various average degrees of nodes. Some average degrees of nodes for individuals were set as 1 to 40 same as Fig. 3. Similarly, the average score is low when there are few or many average nodes. For the same reason as above, we expect that the score is higher when the number of average nodes is between 5 and 25, since C is maintained. However, it is not very realistic to have extremely few or many connections. Therefore, the PPD model contributes more to the maintenance of C compared to the SPD model.



**Fig. 2.** Defector density for each model from 100 trials. The average degrees of nodes are 5.



**Fig. 3.** Relationship between average degree of nodes and defector density



**Fig. 4.** Relationship between average degree of nodes and average score

## 4 Conclusion

In this study, we introduced the Perverse Prisoner's Dilemma (PPD) model, which is an extension of the traditional Spatial Prisoner's Dilemma (SPD) model. Our model includes memory and probabilistic strategy updates. By allowing players to access short length memories and make decisions based on the strategies, the PPD model shows it can maintain cooperative behavior (C) more effectively than the SPD model.

Our results show that the PPD model performs better than the SPD model in sustaining cooperative strategies across different parameters, the value of  $b$  and the average degree of connectivity among players. Also, the PPD model shows improved stability in network environments with moderate connectivity.

Using memory-based strategy updates gives a realistic approach to modeling human behavior, recognizing that people often use past experiences when making decisions. This method reduces the complexity of long-term memory models while still promoting cooperation effectively.

## References

1. Nowak, M.A., May, R.M.: Evolutionary games and spatial chaos. *Nature* **359**(6398), 826–829 (1992)
2. Axelrod, R., Hamilton, W.D.: The evolution of cooperation. *Science* **211**(4489), 1390–1396 (1981)
3. Hauert, C., Doebeli, M.: Spatial structure often inhibits the evolution of cooperation in the snowdrift game. *Nature* **428**(6983), 643–646 (2004)
4. Takahara, A., Sakiyama, T.: Twisted strategy may enhance the evolution of cooperation in spatial prisoner's dilemma. *Physica A* **629**, 129212 (2023)
5. Danku, Z., Perc, M., Szolnoki, A.: Knowing the past improves cooperation in the future. *Sci. Rep.* **9**(1), 262 (2019)
6. Gou, Z., Li, Y.: Prisoner's dilemma game model Based on historical strategy information. *Sci. Rep.* **13**(1), 1 (2023)
7. Shu, F., et al.: Memory-based conformity enhances cooperation in social dilemmas. *Applied Mathematics and Computation* **346**, 480–490 (2019)
8. Takahara, A., Sakiyama, T.: Twisted Strategy Bolsters Minority Cooperator Populations. *ICAART* (1). (2024)
9. Takahara, A., Sakiyama, T.: The Robustness of a Twisted Prisoner's Dilemma for Incorporating Memory and Unlikeliness of Occurrence. *Complexis* (2024)
10. Sakiyama, T.: A power law network in an evolutionary hawk–dove game. *Chaos, Solitons & Fractals* **146**, 110932 (2021)
11. Habib, M.A., Tanaka, M., Tanimoto, J.: How does conformity promote the enhancement of cooperation in the network reciprocity in spatial prisoner's dilemma games? *Chaos, Solitons & Fractals* **138**, 109997 (2020)
12. Allen, B.: Evolutionary dynamics on any population structure. *Nature* **544**(7649), 227–230 (2017). <https://doi.org/10.1038/nature21723>
13. Débarre, F., Hauert, C., Doebeli, M.: Social evolution in structured populations. *Nat. Commun.* **5**(1), 3409 (2014)
14. Maciejewski, W., Feng, F., Hauert, C.: Evolutionary game dynamics in populations with heterogeneous structures. *PLoS Comput. Biol.* **10**(4), e1003567 (2014)



# Hierarchical Structure of Inter-Regional Input-Output Network: A Multilayer Approach

Ardian Maulana<sup>1,2</sup>(✉) , Hokky Situngkir<sup>2</sup> , Pradono<sup>1</sup> , and Adiwan Fahlan Aritenang<sup>1</sup>

<sup>1</sup> Department of Urban and Regional Planning, School of Architecture, Planning, and Policy Development, Institut Teknologi Bandung (ITB), Bandung, Indonesia  
ardianmaulanaeff@gmail.com

<sup>2</sup> Department of Computational Sociology, Bandung Fe Institute, Bandung, Indonesia

**Abstract.** This research aims to elucidate the organizational patterns of inter-regional economic interdependence to enhance our comprehension of the national economy's structure at a regional scale. Employing a multilayer network model, this study represents economic interdependence among Indonesian regions, utilizing the Inter-Regional Input-Output (IRIO) table. Through the application of various metrics, such as degree and strength distribution, assortativity coefficient, and rich club coefficient, to the multilayer IRIO network, we uncover the organizational patterns of economic exchanges between provinces and economic sectors within Indonesia. Our findings demonstrate that a multilayer network approach reveals the heterogeneous and complex structure of the national economy at the regional level. By analyzing the assortativity pattern and global rich-club coefficient, we illustrate that the IRIO network exhibits a hierarchical organization, where significant provincial-sector nodes are interconnected and form dense rich clubs, extending from a few structural cores to peripheral regions.

**Keywords:** Multilayer Network · Interregional Input-Output Table · Hierarchical Organization

## 1 Introduction

Understanding the spatial structure of the national economy at the regional level is essential for resilience and economic development, especially in light of the growth of interregional commerce and the long-term uneven development of the regional economy [1, 2]. Economic structure refers to the pattern of interactions between the components that make up an economic system [3]. Interactions between economic sectors that implicitly have a location dimension give rise to interregional spatial structures, namely the organization of geographic regions based on underlying interactions, whether caused by the movement of people, goods/services, or information [4]. The organizational pattern of these spatial economic structures strongly influences national economic growth and the resilience of the economic system to the propagation of economic shocks [5, 6].

Empirical studies on economic interdependence between regions tend to be carried out in the context of the global economy [7]. Currently, open access to data on interactions

between economic sectors across regions on a national scale, ie. Sub-national Multi/Inter-Regional Input Output [8] provides an opportunity to conduct scientific exploration of statistical and spatial patterns of interregional economic structure.

This study uses a multilayer network model to represent economic interdependence between regions in Indonesia based on the InterRegional Input-Output (IRIO) table. We then apply several measurements, i.e. degree and strength distribution, assortativity coefficient, and global rich club coefficient, to the multilayer IRIO network to reveal the organizational pattern of economic exchange between provinces and economic sectors in Indonesia.

## 2 Data and Method

### 2.1 Data

This study is based on the 2016 Indonesian Inter-Regional Input-Output (IRIO) table published by the Indonesian Central Bureau of Statistics [9]. The table records the economic exchanges among 17 economic sectors and 34 provinces in Indonesia, reflecting their intricate economic relationships and interdependence. Detailed descriptions of sectors and regions are listed in Table 1 and 2. We construct the multilayer input-output network of the Indonesian economy at the sub-national level using the 2016 Indonesian IRIO data.

**Table 1.** List of sectors in the 2016 Indonesian IRIO table

ID	Sectors	ID	Sectors
1	Agriculture, Forestry & Fisheries	10	Information & Communication
2	Mining & excavation	11	Financial Services & Insurance
3	Processing industry	12	Real Estate
4	Proc. of Electricity & Gas	13	Company Services
5	Water Supply, Waste Management, Waste & Recycling	14	Gov. Adm., Defense & Mandatory Social Security
6	Construction	15	Education Services
7	Wholesale & Retail Trade; Car & Motorcycle Repair	16	Health Services & Social Activities
8	Transportation & Warehousing	17	Other Services
9	Prov. of accommodation, food & drink		

### 2.2 Method

We obtain from the intermediate transaction matrix of the IRIO table set of interlayer and intralayer flow of goods and services to build the multilayer input-output network. Inter-layer flows are between pairs of the same or different provinces in different sectors, and

**Table 2.** List of provinces in the 2016 Indonesian IRIO table

ID	Provinces	ID	Province	ID	Provinces
1	Aceh	13	Central Java	25	North Sulawesi
2	North Sumatra	14	Yogyakarta	26	Central Sulawesi
3	West Sumatra	15	East Java	27	South Sulawesi
4	Riau	16	Banten	28	Southeast Sulawesi
5	Jambi	17	Bali	29	Gorontalo
6	South Sumatra	18	West Nusa Tenggara	30	West Sulawesi
7	Bengkulu	19	East Nusa Tenggara	31	Maluku
8	Lampung	20	West Kalimantan	32	North Maluku
9	Bangka Belitung	21	Central Kalimantan	33	West Papua
10	Riau Island	22	South Kalimantan	34	Papua
11	Jakarta	23	East Kalimantan		
12	West Java	24	North Kalimantan		

intralayer flows are between pairs of nodes of different provinces within the same sector. A multilayer network is defined as  $M = (G, C)$  [10–12].  $G$  is a set of weighted directed graphs representing each economic sector layer,  $G = \{G_\alpha; \alpha \in \{1, \dots, k\}\}$ . For layer  $\alpha$ ,  $G_\alpha$  consists of (i)  $V_\alpha$ , a set of nodes  $v_i$  representing provinces, ( $v_i \in V_\alpha; i \in \{1, \dots, N\}$ ); (ii)  $E_\alpha$ , a set of directed edges  $e_{ij}$ , ( $e_{ij} \in E_\alpha; i, j \in \{1, \dots, N\}$ ), between pair of nodes within same layer (intralayer), which indicates the existence of economic exchange between the two provinces.  $C$  is a set of directed edges between pairs of nodes from different layers (interlayer),  $C = \{C_{\alpha\beta} \subseteq V_\alpha \times V_\beta; \alpha, \beta \in \{1, \dots, k\}; \alpha \neq \beta\}$ .

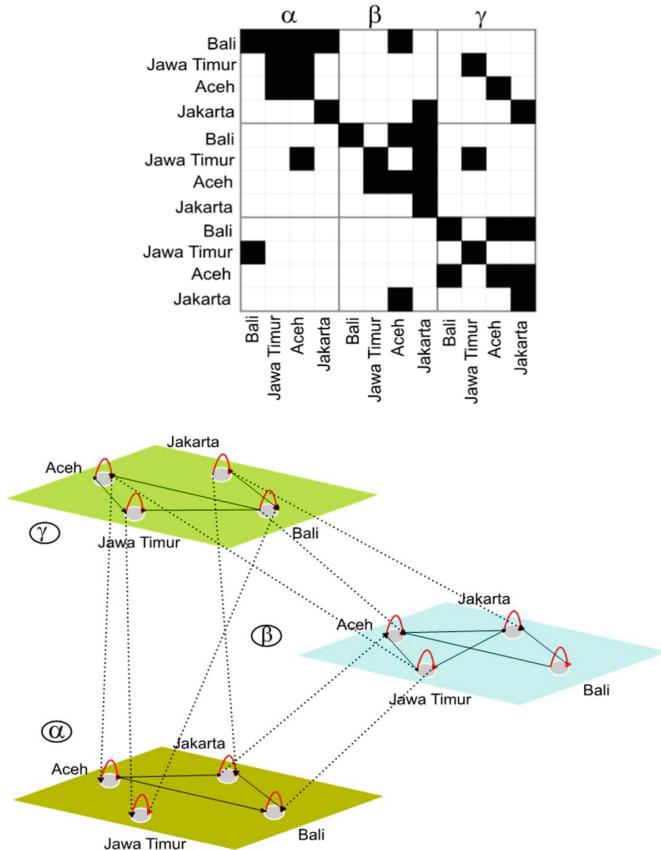
Each edge in the interlayer and intralayer network has a weight that indicates the amount of economic exchange that occurs between two provinces within the same sector or across sectors. We use the intermediate matrix  $Z$  of IRIO to identify and assign weights to the network edges by constructing the supra-adjacency matrix  $W$  as follows,

$$W_{ij}^\alpha = \begin{cases} Z_{ij}^\alpha, & \text{if } e_{ij} \in E_\alpha \\ 0, & \text{Others} \end{cases} \quad (1)$$

where  $W_\alpha = \{W_{ij}^{\alpha\alpha}\} \in \mathbb{R}^{34 \times 34}$  is the intralayer adjacency matrix and  $W_{ij}^{\alpha\alpha}$  is the intralayer weight between nodes  $(i, \alpha)$  and  $(j, \alpha)$ . The elements  $W_{ij}^{\alpha\beta}$  of the interlayer adjacency matrix  $W_{\alpha\beta} = \{W_{ij}^{\alpha\beta}\} \in \mathbb{R}^{578 \times 578}$  corresponding to the set of couplings  $E_{\alpha\beta}$  are defined as

$$W_{ij}^{\alpha\beta} = \begin{cases} Z_{ij}^{\alpha\beta}, & \text{if } e_{ij} \in E_{\alpha\beta} \\ 0, & \text{Others} \end{cases} \quad (2)$$

where  $W_{ij}^{\alpha\beta}$  is the intralayer weight between nodes  $(i, \alpha)$  and  $(i, \beta)$ . The supra-adjacency matrix for the multilayer network is then equal to  $W_\alpha + W_{\alpha\beta}$ . Figure 1 show the schematic of the Indonesian multilayer input-output network and the supra-adjacency matrix.



**Fig. 1.** Illustration of the Indonesian multilayer input-output network and the supra-adjacency matrix

### 3 Result

We apply several indicators, namely degree, strength, assortativity coefficient, and global rich club coefficient, to reveal the interregional spatial structure that emerges from economic exchange between provinces and economic sectors in Indonesia.

#### 3.1 Basic Network Properties

Table 3 shows the basic properties of the multilayer IRIO network. This network has a high density where the ratio of the total number of actual edges to the total number

of possible edges is  $\sim 0.9$ . This is not surprising because each economic sector requires input supplies from various other economic sectors. On average, given that the total nodes are 578, each node receives and supplies goods/services from and to 89 percent of other nodes, with a median in-degree/out-degree of 520.5 and 577. The intralayer and interlayer structures have the same characteristics, showing that economic interactions occur intensively across sectors and regions.

In contrast, the node in/out-strength provides non-trivial information about the intensity of connectivity of the multilayer IRIO network and demonstrates the importance of accounting for the intensity of economic exchange across sectors and locations. The order of magnitude between minimum, maximum, and median values of in/out-strengths are very large, with coefficient of variation of  $\sim 2.8$  dan  $\sim 2.6$ .

**Table 3.** Summary of the basic network properties

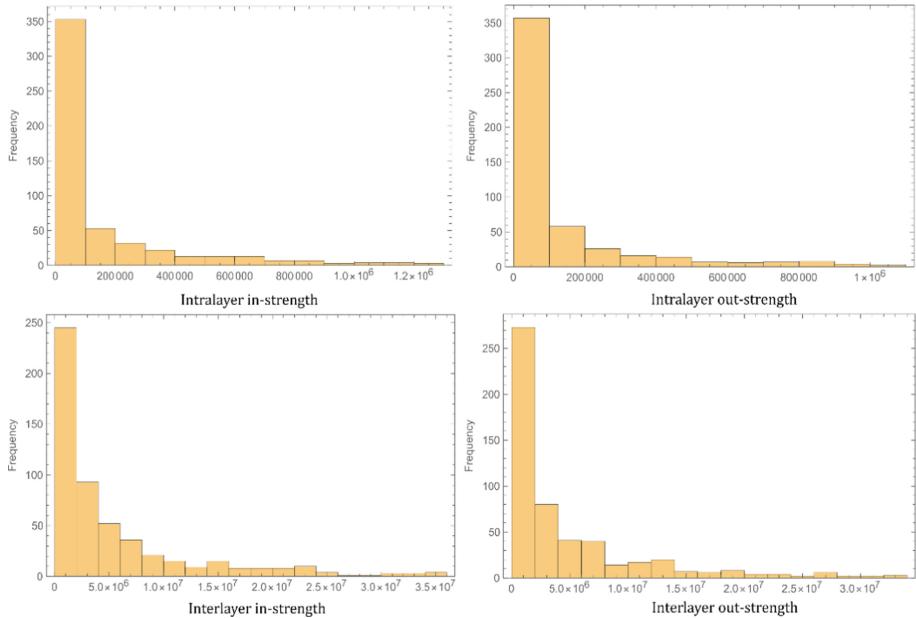
Properties		Multilayer		Intralayer		InterLayer	
		In	Out	In	Out	In	Out
Basic	Node	578	578	578	578	578	578
	Edge	299495	299495	17532	17532	281963	281963
	Density	0.898	0.898	0.92*	0.92*	0.846	0.846
Degree	Min	425	7	0	0	413	6
	Max	553	577	33	33	525	544
	Median	520.5	577	30.3322	30.3322	489	544
Strength	Min	29817.1	8708.21	0.014	0.001	29419.7	8696.79
	Max	4.2E+08	3.2E+08	6.2E+07	4.8E+07	3.7E+08	2.8E+08
	Median	3E+06	2.4E+06	38736.1	25425.8	2.80E+06	2.3E+06

\*Average across Layer

The heterogeneity of strength connectivity among nodes is clearly shown in Fig. 2. The histogram of in- and out-strength in multilayer, intra, and interlayer structures seems to be characterized by heavy tail patterns, suggesting the network might be more vulnerable to perturbations than expected for a random network. Indeed, we find that the empirical in- and out-strength ccdf in all structures are characterized by a log-normal distribution, which is a heavy tail distribution. The log-normal provides an adequate fit to all in- and out-strength where the log-likelihood ratio test between log-normal versus other distributions has a large positive value ( $p < 0.01$ ) [13]. This indicates that large economic exchanges between locations within and/or across sectors are frequent but not as frequent as would be implied by a power law or scale-free distribution.

### 3.2 Assortativity

Assortativity is a fundamental property of complex networks that describes the tendency of nodes to connect with other nodes that have the same (assortative) or different (disassortative) connectivity features [14]. In this study, we use the formula proposed by Yuan



**Fig. 2.** Histogram of intralayer/interlayer in- and out-strength distribution for Indonesian multi-layer input-output network.

et al. [15] to measure the degree of assortativeness in a multilayer IRIO network which is a weighted directed graph.

Table 4 presents a collection of assortativity coefficients for all strength and structural types. Our first observation is that most of the coefficients have positive values, except for the in-out strength type in multilayer and interlayer structures and the in-in type in interlayer structures. This means that the multilayer IRIO network has an assortative structure, showing the propensity of provincial sector nodes to connect with other nodes with comparable levels of strength.

However, if seen from the magnitude of the coefficient, the multilayer structure has weak disassortativity, while the intralayer structure for all types of strength has moderate strong assortativity. Take the out-in assortativity as an example. The positive coefficient suggests that provincial-sector nodes with large inputs are likely to take high transaction volumes from other nodes that have high output in the network. This assortative pattern is more obvious in the intralayer structure, indicating that economic transactions among provincial nodes from the same sector are very closely connected based on similarity of transaction strength. On the other hand, the high volume of transactions between provincial-sector nodes within the province supports the tendency for regional fragmentation [1], resulting in a negligible assortativity pattern in interlayer and multilayer structures.

**Table 4.** Assortativity coefficients for the Indonesian multilayer input-output network

Type	Multilayer	Intralayer	Interlayer
Out-Out	0.09	0.42	0.1
Out-In	0.17	0.42	0.19
In-Out	-0.05	0.35	-0.09
In-In	0.02	0.34	-0.01

### 3.3 Rich Club

The tendency of nodes to interact more intensely among themselves than would be predicted at random is known as the rich-club phenomenon [16]. These nodes form a group of rich nodes that actively and selectively interact with one another while maintain their ties to not-rich nodes. In this study, by using node in/out strength as a richness parameter, we calculate weighted directed rich clubs coefficient as follow [17]:

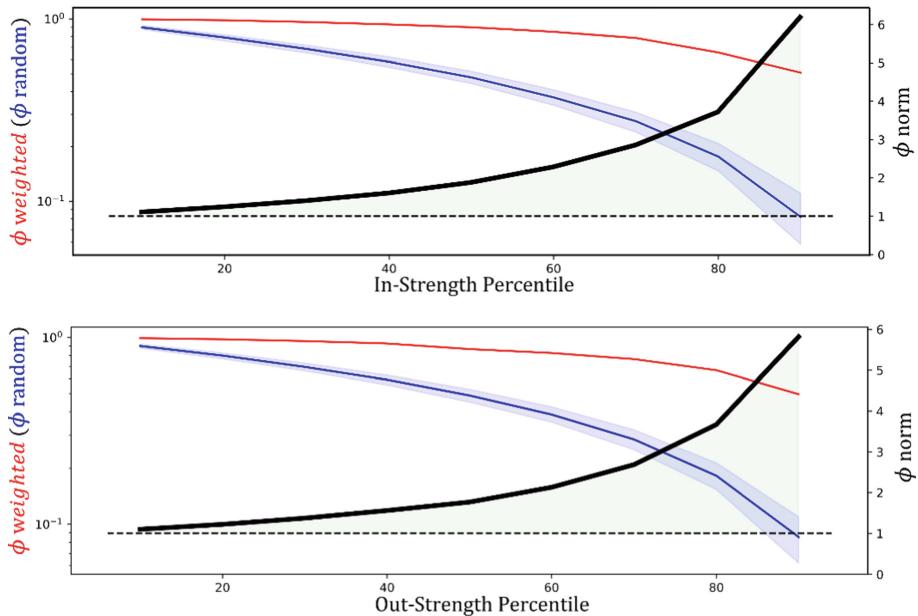
$$\phi_{norm} = \frac{\Phi}{\Phi_{rand}} \quad (3)$$

$$\phi = \frac{W_{>r}}{\sum_{l=1}^{E_{>r}} w_l^{rank}} \quad (4)$$

where  $r$  is the richness parameter,  $W_{>r}$  is sum of these edges weight, and  $w_l^{rank}$  is the  $l^{\text{th}}$  ranked weights on the edges of the netrk,  $\phi_{norm} > 1$  shows the presence of the rich-club phenomenon. Evaluation of the weighted connectedness of a rich club requires normalization through comparison to randomized controls with the same richness sequence.

Rich-club coefficient supplement assortativity measure to provide a detailed description about the presence of dominant subgroup within the network. As shown previously, uneven economic exchange means that some provincial-sector nodes play a key role in the IRIO network. Those prominent nodes form an oligarchic rich group when they preferentially interact with one another while preserving their connections to not-rich nodes [16].

The overall increasing trends of  $\phi_{norm}(s)$ , which are primarily greater than 1, are depicted in Fig. 3. This confirms the presence of rich-club phenomenon in the IRIO network. The rich-club organization of the IRIO network structure becomes more noticeable with the increase of the in- and out-strength level. By considering the assortativity and rich-club properties together, we could infer that the IRIO network has a hierarchical organization structure, in which prominent provincial-sector nodes are not only interconnected but also form dense rich clubs, extending from the few regional cores into peripheral regions. The formation of a rich club consisting of important province-sector nodes implies that these nodes serve as robust collective entities that could offer some resilience to thinterregional economic system in the event that one of its major nodes malfunctions.



**Fig. 3.** Rich-club function for Indonesian multilayer input-output network: In and Out-Strength. The figure show the rich club value for a range of k percentiles:  $\phi$  (red),  $\phi_{random}$  (blue),  $\phi_{lower95\%}$  and  $\phi_{upper95\%}$  (light blue),  $\phi_{norm}$  (black).

## 4 Conclusions

In this study, we build and analyze the multilayer network of intranational supply chains based on the 2016 Indonesia interregional input-output table (IRIO). We show that implementing a multilayer network could uncover the heterogeneous and complex structure of the national economy at the regional level. The distribution of in- and out-strength in multilayer, intra, and interlayer structures is characterized by a log-normal distribution, which is a heavy tail distribution. These findings indicate that multilayer connectivity is an important feature of the IRIO network where simplifying the way we represent the structure would result in substantial loss of information.

We analyze the global network configuration using macroscopic characteristics based on the assortativity coefficient, and global rich-club coefficient and demonstrate the existence of assortativity pattern and the rich-club characteristic in the multilayer IRIO network. These findings indicate that the IRIO network has a hierarchical organization in which prominent provincial-sector nodes are not only interconnected but also form dense rich clubs, extending from the few structural cores into peripheral regions.

In regional planning and development, investigating interregional spatial structures is crucial from both a theoretical and practical standpoint. Our analysis of the spatial structure of interregional input-output networks using a multilayer network framework improves our understanding of the complex organization of national economies at the regional level.

**Acknowledgement.** The material is based upon work supported by the PhD grant from the Indonesia Endowment Fund for Education Agency.

## References

1. Wang, T., Xiao, S., Yan, J., Zhang, P.: Regional and sectoral structures of the Chinese economy: A network perspective from multi-regional input–output tables. *Phys. A Stat. Mech. its Appl.* **581**, 126196 (2021). <https://doi.org/10.1016/j.physa.2021.126196>
2. Gomez, M., Garcia, S., Rajtmajer, S.: Fragility of a multilayer network of intranational supply chains. *Appl. Netw. Sci.* (2020)
3. Thakur, S.: Fundamental economic structure and structural change in regional economies: A methodological approach. *Reg. Dev.* **33**, 9–38 (2011)
4. Rodrigue, J., Comtois, C., Slack, B.: *The geography of transport systems*. Routledge, London (2006)
5. Carvalho, V.: From micro to macro via production networks. *J. Econ. Perspect.* **28**, 23–48 (2014)
6. Blöchl, F., Theis, F., Vega-Redondo, F., Fisher, E.: Vertex centralities in input–output networks reveal the structure of modern economies. *Phys. Rev. E.* **83**, (2011). <https://doi.org/10.1103/PhysRevE.83.046127>
7. Lee, J.-S.: *Trade and Spatial Economic Interdependence: U.S. Interregional Trade and Regional Economic Structure*, (2010)
8. Faturay, F., Lenzen, M., Nugraha, K.: A new subnational multi-region input–output database for Indonesia. *Econ. Syst. Res.* (2017)
9. BPS: Indonesian Inter Regional Input-Output Table Domestic Transactions at Producer's Price By 34 Province and 52 Industry, 2016 (Million Rupiah), [https://www.bps.go.id/id/statistics-table/1/MjEyOCMx/tabel-inter-regional-input-output-indonesia-transaksi-domestik-atas-dasar-harga-produsen-menurut-34-provinsi-dan-52-industri--2016--juta-rupiah-.html](https://www.bps.go.id/id/statistics-table/1/MjEyOCMx/tabel-inter-regional-input-output-indonesia-transaksi-domestik-atas-dasar-harga-produksen-menurut-34-provinsi-dan-52-industri--2016--juta-rupiah-.html), last accessed 2023/02/02
10. Alves, L., Mangioni, G., Rodrigues, F., Panzarasa, P., Moreno, Y.: Unfolding the Complexity of the Global Value Chain: Strength and Entropy in the SingleLayer, Multiplex, and Multi-Layer International Trade Networks. *Entropy* **20**, 909 (2018)
11. Garcia, S., Gomez, M., Rushforth, R., Ruddell, B.L., Mejia, A.: Multilayer Network Clarifies Prevailing Water Consumption Telecouplings in the United States. *Water Resour. Res.* (2021)
12. Bartesaghi, P., Grassi, R., Clemente, G.P., Luu, D.T.: The multilayer architecture of the global input–output network and its properties. *J. Econ. Behav. Organ.* **204**, 304–341 (2022)
13. Alstott, J., Bullmore, E., Plenz, D.: Powerlaw: A Python package for analysis of heavy-tailed distributions. *PLoS One.* **9**, (2014)
14. Newman, M.E.J.: *Networks: An Introduction*. Oxford University Press, New York (2010)
15. Yuan, Y., Yan, J., Zhang, P.: Assortativity coefficients for weighted and directed networks. *J. Complex Networks.* **9**, (2021)
16. Zhou, S., Mondragon, R.: The rich-club phenomenon in the internet topology. *IEEE Commun. Lett.* **8**, 180–182 (2003)
17. Opsahl, T., Colizza, V., Panzarasa, P., Ramasco, J.J.: Prominence and control: The weighted rich-club effect. *Phys. Rev. Lett.* **101**, (2008)

# **Resilience**



# Structural and Flow-Based Approaches to Vulnerability Analysis of Complex Network Systems

Olexandr Polishchuk<sup>(✉)</sup> 

Laboratory of Modeling and Optimization of Complex Systems, Pidstryhach Institute for Applied Problems of Mechanics and Mathematics, National Academy of Sciences of Ukraine,  
Naukova 3B, Lviv 79069, Ukraine  
[od\\_polishchuk@ukr.net](mailto:od_polishchuk@ukr.net)

**Abstract.** A comparative analysis of structural and flow-based approaches to study the vulnerability of complex network systems (NS) from targeted attacks and non-target lesions of various types is carried out. Typical structural and functional scenarios of successive targeted attacks on the most important by certain criteria system elements are considered, and scenarios of simultaneous group attacks on the overriding NS's components are proposed. The problem of system lesions scale from heterogeneous negative influences is investigated. It was confirmed that the flow-based approach allows us to obtain a much more realistic picture of such lesions consequences. It is shown that the scenarios of group targeted attacks built on the basis of NC flow model are more optimal from the point of view of selecting attack targets than structural ones.

**Keywords:** Complex network · Network system · Vulnerability · Targeted attack · Non-target lesion · Core · Centrality · Influence · Betweenness

## 1 Introduction

Humanity is continuously faced with many threats of natural and artificial origin, which affect many real-world complex systems [1, 2]. Caused by them lesions can be external and internal, local, group, or system-wide, and local can develop into group, and group – into system-wide [3]. They can be predictable and unexpected, centralized and decentralized, when the damaged element itself becomes a source of lesion, spread with different speed – from almost instantaneous (cascade phenomena in power grids) to those that last for decades (global warming, the spread of AIDS) [4, 5]. We divide the negative influences on the system into targeted attacks caused by a specific intruder, and non-target lesions generated by factors of natural or artificial origin that do not have such intruder. Despite the diversity of real-world systems and the variety of types of negative influences on them, targeted attacks and non-target lesions can have much common both in the methods of action and the consequences of such influences: the spread of dangerous infectious diseases (DID) and computer viruses, traffic jams and DDoS attacks,

shelling of populated areas and earthquakes, etc. In this article, we will try to show that this similarity allows us to develop universal means of active and passive protection of real-world systems from such influences, as well as evaluating and overcoming of their consequences.

Currently, the main attention of researchers in the theory of complex networks (TCN) is focused on the development of protection strategies against successive targeted attacks on the most structurally important nodes of network systems [6, 7]. Much less attention is paid to the creation of attack scenarios on the operation process of such systems. Undoubtedly, damage to the structure affects this process, but it is possible to destabilize or even stop the system functioning even with an intact structure (during the Russian-Ukrainian war the air transport system of Ukraine with the preserved structure completely stopped its activity only because the threat of downed planes). Another shortcoming of the structural approach to NS vulnerability analysis is the evaluation of lesion scale [8]. Directly damaged objects are actually destroyed elements that must be removed from the system structure. Only the network system nodes adjacent to the directly damaged can reasonably be considered consequentially injured. This approach is quite acceptable for assortative networks [9]. However, for disassortative NSs, which include the majority of real-world man-made systems and elements of which are connected by paths, this approach does not fully reflect the entire set of victims as a result of targeted attack or non-target system lesions. Only objective comprehensive evaluation of real picture of negative influence consequences, which combines all directly damaged and consequentially injured system elements, will make it possible to more accurately classify the scale of this influence and quantify the damage caused by it.

One of the ways to protect the system is neutralization the source of negative influence, in particular, use of active protection means against it (sanctions to the aggressor country, destruction of agricultural pests, cessation of terrorist and hacker groups activities, vaccination [10, 11], etc.). The problem of optimizing the scenarios of such attacks, as well as development of effective countermeasures against non-target lesions, is also paid a little attention so far, although the party that initiates and carries out such active protection means, for example, imposes sanctions against the aggressor country or quarantine measures during epidemics of DID, also bears considerable losses. The purpose of article is application a flow-based approach to vulnerability analysis of NS operation process and demonstration its advantages over structural ones during evaluation the real losses caused by negative influence on the system, as well as optimization the scenarios of targeted attacks on it.

## 2 Structural Approach to Vulnerability Analysis

The mathematical model of complex network (CN)  $G = (V, E)$ , where  $V$  is the set of nodes of network  $G$  and  $E$  is the set of connections (edges) between them, is the adjacency matrix  $\mathbf{A} = \{a_{ij}\}_{i,j=1}^N$ . Here  $N$  is the quantity of elements of the set  $V$  [12]. The value  $a_{ij}$  of matrix  $\mathbf{A}$  is equal to 1 if there is connection between nodes  $n_i$  and  $n_j$  (such nodes are called adjacent),  $i, j = \overline{1, N}$ , and is equal to 0 if there is no such connection. It is also assumed that the nodes do not have loop connections, i.e. the diagonal elements of matrix  $\mathbf{A}$  are zero.

No large-scale real-world complex system can protect all its elements [13, 14]. Therefore, the problem arises of determining those NS nodes that must be protected first of all. To solve this problem, the concept of centrality is used as an importance indicator of node [15, 16]. The main local characteristics of binary directed network node are its input and output degrees [12] or degree centrality. Here, by local we mean a characteristic that describes the properties of element itself or one or another aspect of its interaction with adjacent system elements. The input degree of node determines the quantity of edges that “enter” it, and the output degree – the quantity of edges that “leave” from a given node to adjacent NS nodes. Characteristics of element that describe one or another aspect of its interaction with all other elements of this system will be called global. Global centralities allow us to determine the importance of node in the network at a whole. However, the concept of “importance” can have different meanings, which has led to the appearance of many definitions of the term “centrality”, namely betweenness, closeness, eigenvector, Freeman and many others [16, 17]. At the same time, the value of one centrality may contradict another, and centrality, important for solving one problem, may be insignificant for another [18], which introduces some ambiguity when they are used as importance indicator of element in the system structure.

## 2.1 Typical Scenarios of Successive Targeted Attacks

Usually, an attack in TCN refers to actions aimed at intentionally removing from the system structure a certain number of the most important nodes based on the chosen centrality in order to change the structural network properties [19]. Since the system lesion is usually carried out by successive or simultaneous damage to its elements, the first step in formation of its protection methods consist in construction the so-called scenarios of targeted attacks on the network system [6, 16]. The most effective scenarios of such attacks are formed when their developer “puts himself” in the place of “intruder”, who tries to cause maximum damage to the attacked system with minimal means. The development of each scenario should be preceded by determination of attack success criteria. From a structural point of view, such criteria can be the division of complex network into unconnected components, increase of average length of the shortest path [16], and so on.

The structural scenarios of NS lesions developed so far, which can be divided into two main groups, are based on the use of above-mentioned centralities of nodes in the system structure (generalized degree centrality, as the sum of input and output degrees of node in directed CN, centralities by betweenness, closeness, eigenvalue, etc.) [3, 20]. Each of scenarios of the first group begins with ordering a set of NS nodes according to decreasing values of their centrality of selected type and subsequent consecutive removal of nodes from the structure according to this order until the attack success criterion is met. The scenarios of this group do not involve changing the centrality values of nodes that remained in the network. In the second group of scenarios, it is taken into account that with each removal of node, the structure of NS changes due to the establishment of new connections between the remaining nodes. This requires a new ordering of the sequence of NS nodes according to the changed values of their centrality of selected type. The next step in this scenarios group removes a node from the beginning of newly created list that takes these changes into account. It is obvious that the typical scenarios

described above do not determine specific ways to protect a real-world system, which depend on its type and kind of negative influence, however, these scenarios make it possible to identify the elements that must be primarily protected from the point of view of their importance in the NS structure.

## 2.2 Evaluation of the System Structure Lesions

In general, three types of interrelated problems appear to protect network systems from various negative influences, namely: analysis of real and potential threats and the development of effective means of protection against them *before* lesion, provision of ways to counteract the spread of negative influences and minimize their consequences *during* damage, and evaluation of consequences and restoration of the system structure *after* its lesion. After all, the better the system is protected, the weaker the effect of negative influence, and therefore the smaller its consequences. To solve the first of these problems, the considered above typical scenarios of targeted attacks can be used, and for the second and third, the structural model of network system can be applied. Obviously that all changes occurred in system structure should be immediately reflected in its model. Then the difference between the rank of matrix  $\mathbf{A}$  before lesion and the rank of this matrix during (after) damage determines the quantity of nodes destroyed during (after) it in the source NS structure. The difference between the quantity of non-zero elements of structural model of network system before the lesion and the quantity of non-zero elements of matrix  $\mathbf{A}$  during (after) damage determines the quantity of edges destroyed during (after) it in the source NS structure. Thus, the comparison of structural models of network system makes it possible to draw up a sufficiently objective quantitative picture of the level of damage to the system or its separate components as a result of targeted attack or action of non-target lesion. At the same time, along with the integral damage indicators (the total quantity of destroyed network nodes and edges), the structural model allows us to analyze the lesion of each NS's element. Thus, zeroing the element of matrix  $\mathbf{A}$  indicates the removal (destruction, blocking) of corresponding edge from NS structure, zeroing of all elements of row and column of matrix  $\mathbf{A}$  – the removal of corresponding node from this structure, reducing the generalized degree of node – decreasing the number of its interactions with other system elements. In general, the structural model allows us to reproduce the picture of all directly damaged and part of consequentially injured NS elements. The main drawback of structural approach to evaluation the lesion consequences is that only elements adjacent to the directly damaged nodes of the network system can be considered consequentially injured, i.e. some underestimation of real system losses takes place.

## 2.3 Simultaneous Group Attacks on the System Structure

It is obvious that simultaneous group attacks or non-target system lesions are much more difficult than successive attacks on the most important NS elements, both from the point of view of its protection and overcoming the consequences. We divide simultaneous group negative influences into one-time (the attack on Pearl Harbor on December 7, 1941), repeated (the permanent bombing of London during the Second World War) and

successive (attacks on transformer stations of the Ukrainian power system in 2022–2024). Repeated group attacks are carried out regularly at certain time intervals on the same system objects. Consecutive group attacks differ from repeated ones by changing the targets. A particular danger is that successful sequential group attacks can lead to system-wide NS lesions, for example, a prolonged blackout in the country. In the case of targeted attacks, this separation is often determined by the attacker's ability to launch subsequent massive attacks and the ability of attacked system to effectively defend and counter them. It is clear that each of above types of attack requires the development of specific type of scenarios for its most likely implementation. The simplest scenario of one-time group attack is obviously implemented by attempt to simultaneously defeat a group of the most important NS elements according to determined centrality. The repeated attack scenario is realized by attempt to damage a preselected and previously attacked, but not destroyed, group of network system elements. A sequential group attack scenario involves the consecutive execution of following steps:

- 1) compile a list of groups of NS nodes in order of decreasing indicators of their importance in the system, selected according to a certain feature;
- 2) delete the first group from the created list;
- 3) if the criterion of attack success is reached, then complete the execution of scenario, otherwise go to point 4;
- 4) since the system structure changes as a result of removal of certain group of nodes (and their connections), compile a new list of groups in order of decreasing recalculated indicators of their importance in the NS and proceed to point 2.

If, during implementation of last scenario, a certain group of nodes contains too many elements that the attacker is unable to damage simultaneously, then such group is divided into the minimum quantity of connected subgroups available for such attacks. In addition, the execution of scenario may terminate when the attacking party has exhausted the resources to continue the attack. It follows from the above scenarios that the main way to increase their effectiveness is to choose the importance indicators of group in network system, the lesion of which will cause it the greatest damage. The most obvious way of such choice is to form a list of NS nodes in order of decreasing the values of their centrality of selected type and form a group from the first nodes of this list, the quantity of which is determined by the ability of attacker to carry out a simultaneous attack on them. The second method is based on the principle of nested hierarchy of the network system [21]. The method proposed by us consists in applying the concept of k-core of CN, as the largest subnet of source network, the centrality of which, according to the generalized structural degree of nodes, is at least  $k > 1$  [22]. This method is based on the use of the most structurally important components of network and obviously fits into the above scenario of successive group attacks. In particular, groups are initially selected for the maximum value  $k$  for a given CN, which is then sequentially reduced until the attack success criterion is met.

### 3 Flow-Based Approach to Vulnerability Analysis

To determine the functional importance indicators of separate NS components, we will use its flow model [3], after all, the most real-world systems are created precisely to ensure the movement of flows through the relevant networks (transport, financial, trade, energy, information, etc.) or the movement of flows directly ensures their vital activity (the movement of blood, lymph, neuroimpulses in a living organism and so on). By a flow that passes through a network edge, we mean a certain positive function correlated to this edge. This function can reflect the flow density at each point of the edge or the volume of flow that is on the edge at current moment of time  $t \geq 0$ , or the total volume of flow that has passed through the edge up to the moment  $t$  for a certain period of time  $T > 0$ , etc. Let us represent the set of flows that pass through the NS edges in the form of flow adjacency matrix  $\mathbf{V}(t)$ , the elements of which are determined by the volumes of flows that have passed through the edges of complex network  $G$  for the period  $[t - T, t]$  up to current moment of time  $t \geq T$ :

$$\mathbf{V}(t) = \{V_{ij}(t)\}_{i,j=1}^N, \quad V_{ij}(t) = \tilde{V}_{ij}(t) / \max_{l,m=1,N} \tilde{V}_{lm}(t), \quad V_{ij}(t) \in [0, 1], \quad i, j = \overline{1, N},$$

in which the values  $\tilde{V}_{ij}(t)$  are equal to the real volumes of flows that passed through the edge  $(n_i, n_j)$ ,  $i, j = \overline{1, N}$ , of the complex network during the time period  $[t - T, t]$ ,  $t \geq T$ . Note, the more extreme the situation in which the system is, the smaller the value  $T$  should be chosen. It is obvious that structure of matrix  $\mathbf{V}(t)$  coincides with the structure of matrix  $\mathbf{A}$ . The elements of flow adjacency matrix are determined on the basis of empirical data about movement of flows through the network system edges. Currently, with the help of modern means of information extraction, such data can be easily obtained for many real-world natural and the vast majority of man-made systems [23]. It is clear that the NS flow model described above is not its mathematical model in the usual sense of the word, but it gives a sufficiently clear quantitative picture about the operation process of network system, allows us to analyze the features and predict the behavior of this process, as well as evaluate its effectiveness and prevent existing or potential threats [24].

#### 3.1 Flow-Based Scenarios of Targeted Attacks

Another, compared to the structural, and often much more effective and easier to implement method of attack consists in destabilizing or stopping the operation process of separate components or the system at a whole without directly destroying its elements – significantly reducing or stopping the movement of flows through the network, creating conditions for critical loading of the paths of movement of these flows, blocking of separate nodes – generators, transistors and/or final receivers of flows, desynchronization of flows movement through the network, etc. The construction of scenarios of successive targeted attacks on the most functionally important system elements is carried out according to the same principles as typical structural ones, with the difference that as importance indicators of NS nodes are used the characteristics that determine the role

of NS elements in the process of its functioning as generators, final receivers and transitors of flows [3]. Structural and functional approaches to building scenarios of targeted attacks on the system can be combined. For example, if there are groups with the same values of a certain type of functional centrality in the sequence of NS nodes, they can be ordered by the values of selected type of structural centrality and vice versa.

On the basis of flow model, we can determine such global characteristics of NS nodes as input and output parameters of their influence on the system, as well as betweenness parameters [3]. Namely, the input (output) strength of influence of the node – final receiver (generator) of flows is equal to the total volume of flows that were received (generated) in this node during the period  $[t - T, t]$ ; the input (output) domain of influence of the node – final receiver (generator) of flows is the set of NS nodes, in which the flows directed to (from) it were generated (finally received) during the period  $[t - T, t]$ ; the input (output) power of influence of the node – final receiver (generator) of flows is equal to the quantity of elements of input (output) domain of influence of this node, respectively. The measure of betweenness of the node is equal to the total volume of flows that passed through it in transit during the time period  $[t - T, t]$ ,  $t \geq T$ ; the domain of betweenness of the node is the set of NS nodes that directed and received flows through this transit node, and the power of betweenness is the quantity of elements that make up the domain of betweenness. The influence and betweenness parameters of the node make it possible to quantify how its damage will impact on the NS operation process, and which, how many elements and to what extent will be affected.

During the development of scenarios of simultaneous group attacks, as a functionally most important component of the network system, the concept of its flow  $\lambda$ -core [25] can be used, as the largest subsystem of source system, elements of flow adjacency matrix of which have values not less than  $\lambda \in [0, 1]$ . It is obvious that the larger the value  $\lambda$ , the more important from a functional point of view NS's components are reflected by its  $\lambda$ -core. They can become one of the primary targets of simultaneous group attack, the scenario of which was given in Sect. 2.3. Similarly, as for NS elements, we can determine the parameters of influence and betweenness of its  $\lambda$ -core, which significantly deepens the analysis of network system lesions.

### 3.2 Evaluation of Consequences of the System Operation Process Lesions

The difference between the sum of elements of the matrix  $\mathbf{V}(t)$  before lesion and the sum of elements of this matrix during (after) lesion can be used as an integral indicator of losses caused to the network system by a certain negative influence. This indicator determines the total decrease of the volume of flows in NS as a result of such influence. At the same time, along with integral indicators, the flow model makes it possible to analyze the lesion of each network node and edge. Thus, the zeroing of matrix  $\mathbf{V}(t)$  element indicates the removal (destruction, blocking) of corresponding edge from the operation process of network system, the zeroing of all elements of the row and column of matrix  $\mathbf{V}(t)$  that correspond to a certain NS's node – the removal of this node from the system operation process (even if it remains intact in the system structure). A decrease of the value of matrix  $\mathbf{V}(t)$  element is a sign of decrease of the volume of flows that pass through the corresponding edge, and decrease of the value of the sum of elements in a row and column that correspond to a certain node of network system indicates a decrease

of the volume of flows that are generated, received and transited through this NS's node. In general, the flow model allows us to reproduce the picture of not only destroyed, but also all consequentially injured nodes and edges of network system and to quantify the level of lesion, which is an additional advantage of this model.

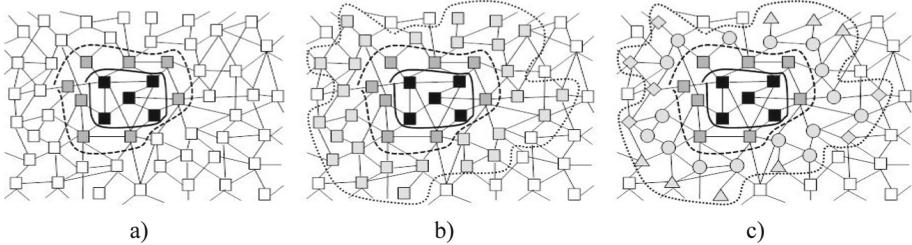
## 4 Analysis of Network System Lesions

The picture of targeted attack consequences on the network, obtained on the basis of structural approach and using its generalized structural degree as importance indicator of a node is shown in Fig. 1a. The directly damaged NS nodes are marked in black; in dark gray – adjacent to directly damaged (consequentially injured) nodes; in white – undamaged nodes of the network system; the continuous curve delimits the directly damaged NS domain; the dashed curve delimits domain of adjacent consequentially injured network system elements.

The consequences of targeted attack on the network system, obtained on the basis of its flow model (in gray are marked nodes, the volume of flows from (to, through) which decreased as a result of attack (consequentially injured nodes); the dotted curve delimits the consequentially injured NS's domain) is shown in Fig. 1b. As follows from this picture, the domain of consequentially injured NS elements determined on the basis of flow model can be much larger than domain of adjacent to directly damaged nodes of network system determined on the basis of its structural model. Thus, the comparison of NC flow models before, during, and after negative influence allows us to draw up a sufficiently objective quantitative picture of the lesion level of network system or its separate components as a result of targeted attack or the action of non-target lesion. The ratio of quantity of directly damaged to the quantity of attacked system elements is an objective indicator of its protection against attacks or lesions of a certain type. The concept of system sensitivity to consequences of negative influence can be determined as the ratio of quantity of directly damaged to the quantity of consequentially injured elements. It is obvious that the closer the value of this indicator is to zero, the more sensitive the system is to negative influences, since a small quantity of directly damaged nodes generates a large quantity of consequentially injured NS's elements.

In general, after the lesion of certain NS node, the combination of domains of its influence and betweenness fully determines the set and quantity of all system elements consequentially injured as a result. In Fig. 1c shows the consequences of targeted attack on the network system based on analysis of influence and betweenness parameters of network system elements (in gray diamonds, triangles and circles are marked the consequentially injured generators, final receivers and transit nodes, respectively).

Comparing Fig. 1a, 1b, and 1c, it can be reasonably concluded that the flow-based approach makes it possible to create a much more realistic picture of lesions consequences caused by a certain negative influence than the structural one. The importance of analysis of influence and betweenness parameters of NS's elements, i.e. generator, final receivers, and flow transit nodes lesions is explained by the fact that they require the search for new suppliers, consumers and alternative paths of flows movement, which is usually a rather difficult problem, especially in the case of global NS lesions.



**Fig. 1.** Consequences of negative influence based on the NS structural model (a), NS flow model (b), and analysis of influence and betweenness parameters of NS's nodes (c)

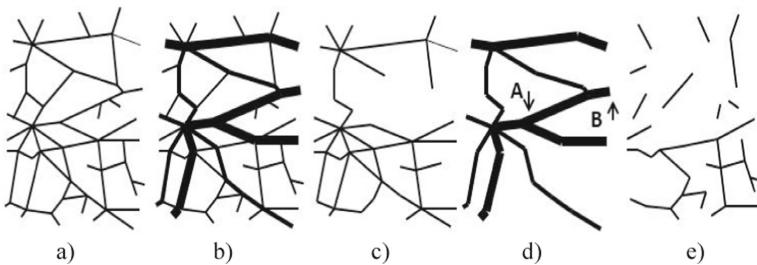
## 5 Optimization of Active Protection Means

It was mentioned above that one of the ways to protect the system is to counterattack the intruder. It is clear that the organizers of such counterattacks also suffer considerable losses. That is, the problem of optimizing attack scenarios arises, namely, how to destroy or block the operation of minimum quantity of nodes of attacked party, to cause it the greatest possible lesion. A similar situation is observed during the development of scenarios for combating the spread of non-target lesions, for example, epidemics of DID. In particular, how to minimize the volume of passenger flows through the transportation network by blocking the smallest possible quantity of nodes that ensure the movement of these flows. Obviously that it is advisable to take into account not only the magnitude of direct negative influence, but also the scale of mediated lesion consequences.

### 5.1 Optimization of Simultaneous Group Attacks Scenarios

Above, for the construction of simultaneous group attacks scenarios, it was proposed to use the concepts of structural  $k$ - and flow  $\lambda$ -core of network system. We will show that the use of flow  $\lambda$ -cores compared to structural  $k$ -cores of NSs is significantly more effective when building scenarios of group targeted attacks, both from the point of view of possible lesion of the most functionally important NS's components, and for the purpose of optimizing these scenarios in terms of the quantity of attack targets. Let's consider the railway transport system (RTS) of the western region of Ukraine. In Fig. 2a shows the structure of this system, and in Fig. 2b – the same structure, but in the form of weighted network, which schematically reflects the volumes of cargo flows that passed through its edges during 2021 (the thickness of lines is proportional to the volumes of flows). Note that this real-world network contains 354 nodes in total, but in Fig. 2a-b, only 29 nodes and 62 edges are displayed (transit nodes with structural degree 2 are not shown, and an edge is considered to be a line that connects two nodes with degree greater than 2). In Fig. 2c contains the structural 4-core of RTS, which includes 12 nodes and 35 edges, and in Fig. 2d is its flow 0.8-core, which contains 4 nodes and 12 edges. One of the main disadvantages of  $k$ -cores compared to flow cores is the possibility of excluding functionally important components of the network system (path A-B in Fig. 2d).

From Fig. 2 also follows that the flow 0.8-core reflects a functionally more important subsystem of RTS and the target of group attack on it is a much smaller quantity of nodes than 4-core of corresponding structure. Easy to see that in both cases, a successful attack on NS nodes selected with the help of  $k$ - and  $\lambda$ -cores will lead to actual termination of its operation process, as it divides RTS into unconnected components (Fig. 2e), but in the second case, the goal of attack is achieved with significantly less efforts (three times in terms of quantity of nodes and edges). Thus, the flow-based approach allows us to build scenarios that are much more optimal from the point of view of attacking side's efforts than the structural one. By analyzing the parameters of influence and betweenness of 0.8-core of given RTS fragment, it was established that all elements of this fragment will be consequentially injured by a successful targeted attack on it.



**Fig. 2.** Examples of structure (a), operation process (b), structural 4-core (c), flow 0.8-core (d), and the addition to flow 0.8-core in the source structure (e) of railway transport system of the Western region of Ukraine

## 5.2 Network Granulation and the Use of Edges as Attack Targets

The scenario of combating non-target lesion (Covid-19 pandemic spreading) was considered in monograph [3], which consisted in granulation of network, that is, its successive division according to the principle of nesting [21] into subnets, the connections between elements of which become significantly weaker or completely blocked, than before the lesion. However, such scenario simultaneously is a very effective way of targeted attack. The main feature of network granulation, as a type of attack, is the destruction or blocking not nodes, but NS's edges. Such approach, which is practically not developed within TCN, is a powerful resource for optimizing scenarios of targeted attacks, especially for the party that uses active protection. At the same time, it is obvious that much easier to block communication between NS nodes than to destroy one of them. Thus, in order to damage the defense-industrial complex of aggressor country, along with the direct destruction of facilities that manufacture high-precision weapons, it is enough to stop supplying these facilities with the modern equipment and components. That is, we can formulate the following optimization problem: how, by blocking the minimum quantity of edges between NS nodes, stop or at least significantly delimit the operation process of maximum number of network system elements. Solution of this problem requires development of models of multidimensional network systems operation process [26], because

manufacture of any high-precision weapon requires supply of many heterogeneous components, that is, ensuring the movement of various types of flows, – from the simplest, for example, UAV propellers or fuselages, to sufficiently complex ones – microcircuits, thermal imagers, software with artificial intelligence elements, and so on. On the other hand, to strike the budgetary sphere of aggressor country in order to reduce its financial possibilities for continuation of the war, along with the freezing of international assets, an embargo on purchase of energy carriers and other minerals can be established. Then we can formulate the following optimization problem: how, by blocking the minimum quantity of edges between NS nodes, to maximally reduce the volumes of flow movement in the system. To solve this problem, it is expedient to use the concept of flow  $\lambda$ -core of network system, choosing as the volume of flow the financial equivalent of its content. That is, for this case as well, the flow-based approach makes it possible to develop more optimal scenarios of targeted attacks on the functioning of complex network systems compared to the structural one.

## 6 Conclusions

The article proposes a flow-based approach to vulnerability analysis of complex network systems, namely, the construction of scenarios of successive and simultaneous group attacks on operation process of the NS and evaluation of consequences of lesions caused by them. For this purpose, as importance indicators of system elements, the parameters of their output and input influence, as generators and final receivers of flows, as well as the parameters of betweenness of network transit nodes, are determined. The advantages of proposed approach over structural ones have been confirmed, both from the point of view of building scenarios of targeted attacks, and evaluation the scale of their consequences. The similarity of both the method of action and consequences of many targeted attacks and non-target lesions of network systems is shown, which makes it possible to develop universal means for active and passive protection against them. To optimize the scenarios of simultaneous group attacks, as a means of active protection against the source of negative influence, the application of flow cores of the NS is proposed. The advantages of such scenarios over structural ones based on the concept of k-core of a complex network are shown on the example of real-world network system. As an additional method for optimizing the scenarios of targeted attacks on operation process of NS, the effect of network granulation and the use of its edges as attack targets is considered. Further steps of our research are the vulnerability analysis and development of structural and functional scenarios of successive, simultaneous group and system-wide attacks on the structure and operation process of multilayer network systems. The peculiarity of such problems is that lesion of one layer as a result of targeted attack or non-target lesion can lead to damage the process of intersystem interactions at a whole.

## References

1. Vitenu-Sackey, P.A., Barfi, R.: The impact of Covid-19 pandemic on the global economy: Emphasis on poverty alleviation and economic growth. *The Economics and Finance Letters* **8**(1), 32–43 (2021)
2. Bluszcz, J., Valente, M.: The Economic Costs of Hybrid Wars: The Case of Ukraine. *Defence and Peace Economics* **33**(1), 1–25 (2022)
3. Polishchuk, O., Yadzhak, M.: Models and methods of comprehensive research of complex network systems and intersystem interactions. *Pidstryhach Institute for Applied Problems of Mechanics and Mathematics, NAS of Ukraine*, Lviv (2023)
4. Sawada, Y., Bhattacharyay, M., Kotera, T.: Aggregate impacts of natural and man-made disasters: A quantitative comparison. *International Journal of Development and Conflict* **9**(1), 43–73 (2019)
5. Wandel, S.: A comparative analysis of approaches to network-dismantling. *Sci. Rep.* **8**(1), 13513 (2018)
6. Nguyen, Q., et al.: Conditional attack strategy for real-world complex networks. *Physica A* **530**, 12156 (2019)
7. Bellingeri, M., Cassi, D., Vincenzi, S.: Efficiency of attack strategies on complex model and real-world networks. *Physica A* **414**, 174–180 (2014)
8. Hansberry, R.L., et al.: How wide is a fault damage zone? Using network topology to examine how fault-damage zones overprint regional fracture networks. *J. Struct. Geol.* **146**, 104327 (2021)
9. Noldus, R., Van Mieghem, P.: Assortativity in complex networks. *Journal of Complex Networks* **3**(4), 507–542 (2015)
10. Brunst, P.W.: Terrorism and the Internet: New Threats Posed by Cyberterrorism and Terrorist Use of the Internet. In: Wade, M., Maljevic, A. (eds.) *A War on Terror?*, pp. 51–78. Springer, New York (2010)
11. North Atlantic Treaty Organization, Hybrid warfare: New threats, complexity and trust as the antidote, NATO Review, last accessed 2021/11/30/
12. Boccaletti, S., et al.: Complex networks: Structure and dynamics. *Phys. Rep.* **424**(4), 175–308 (2006)
13. Proletarsky, A., et al: Decision support system to prevent crisis situations in the socio-political sphere. *Cyber-Physical Systems: Industry 4.0 Challenges*, 301–314 (2020)
14. Waltner-Toews, D., Kay, J.J., Lister, N.: *Ecosystem Approach: Complexity, Uncertainty, and Managing for Sustainability*. Columbia University Press, New York (2008)
15. Saxena, A., Iyengar, S.: Centrality measures in complex networks: A survey. *arXiv*: 2011.07190 (2020)
16. Mariyam, J., Lekha, D.S.: Need for a realistic measure of attack severity in centrality based node attack strategies. In: Benito, R.M., et al. (eds.) *Complex Networks and Their Application X*, pp. 857–866. Springer, Cham (2022)
17. Glenn, L.: Understanding the influence of all nodes in a network. *Sci. Rep.* **5**, 8665 (2015)
18. Krackhardt, D.: Assessing the political landscape: Structure, cognition, and power in organizations. *Adm. Sci. Q.* **35**(2), 342–369 (1990)
19. Albert, R., Jeong, H., Barabasi, A.-L.: Error and attack tolerance of complex networks. *Nature* **406**, 378–382 (2000)
20. Amaral, L.A.N., Ottino, J.M.: Complex networks. *Eur. Phys. J. B* **38**, 147–162 (2004). <https://doi.org/10.1140/epjb/e2004-00110-5>
21. Polishchuk, O., Yadzhak, M.: Network structures and systems: III. Hierarchies and networks. *System research and informational technologies* 4, 82–95 (2018)

22. Dorogovtsev, S.N., Goltsev, A.V., Mendes, J.F.F.: K-core organization of complex networks. *Phys. Rev. Lett.* **96**(4), 040601 (2006)
23. Barabasi, A.-L.: The architecture of complexity. *IEEE Control Syst. Mag.* **27**(4), 33–42 (2007)
24. Polishchuk, D., Polishchuk, O., Yadzhak, M.: Complex evaluation of hierarchically-network systems. arXiv preprint [arXiv:1602.07548](https://arxiv.org/abs/1602.07548) (2016)
25. Polishchuk, O., Yadzhak, M.: Network structures and systems: II. Cores of networks and multiplexes. *System research and informational technologies* 3, 38–51 (2018)
26. Boccaletti, S., et al.: The structure and dynamics of multilayer networks. *Phys. Rep.* **544**(1), 1–122 (2014)



# Sublinear Cuts are the Exception in BDF-GIRGs

Marc Kaufmann<sup>(✉)</sup>, Raghu Raman Ravi, and Ulysse Schaller

ETH Zürich, Zurich, Switzerland

{kamarc,ulysses}@ethz.ch, raravi@student.ethz.ch

**Abstract.** The introduction of geometry has proven instrumental in the efforts towards more realistic models for real-world networks. In Geometric Inhomogeneous Random Graphs (GIRGs), Euclidean Geometry induces clustering of the vertices, which is widely observed in networks in the wild. Euclidean Geometry in multiple dimensions, however, restricts proximity of vertices to those cases where vertices are close in each coordinate. We introduce a large class of GIRG extensions, called BDF-GIRGs, which capture arbitrary hierarchies of the coordinates within the distance function of the vertex feature space. These distance functions have the potential to allow more realistic modeling of the complex formation of social ties in real-world networks, where similarities between people lead to connections. Here, similarity with respect to certain features, such as familial kinship or a shared workplace, suffices for the formation of ties. It is known that - while many key properties of GIRGs, such as log-log average distance and sparsity, are independent of the distance function - the Euclidean metric induces small separators, i.e. sublinear cuts of the unique giant component in GIRGs, whereas no such sublinear separators exist under the component-wise minimum distance. Building on work of Lengler and Todorović, we give a complete classification for the existence of small separators in BDF-GIRGs. We further show that BDF-GIRGs all fulfill a stochastic triangle inequality and thus also exhibit clustering.

**Keywords:** Real-world Networks · Geometric Inhomogeneous Random Graphs · Boolean Distance Functions · Network Robustness · Small Separators · Sparse Cuts · Clustering

## 1 Introduction

Bringing generative graph models closer to applications has driven network science since its inception. This includes the design of models which capture structural properties widespread among real networks. A prominent such model are Geometric Inhomogeneous Random Graphs (GIRGs), which are known to be sparse, small worlds, and whose degrees follow a power-law distribution [2].

---

M.K. and U.S. were supported by the Swiss National Science Foundation [grant number 200021\_192079].

Recent research has shown that they are well-suited to model geometric network features such as the local clustering coefficient as well as closeness and betweenness centrality of real networks [4]. In GIRGs, each vertex comes equipped with a set of coordinates in a geometric ground space and a weight, both drawn independently. Pairs of vertices  $u, v$  are then connected independently with a probability which depends on the product of the vertex weights  $w_u$  and  $w_v$ , and decays as their distance in the ground space increases. The role played by the distance function has been relatively unexplored, but we know that many graph properties, such as diameter and average distance in the giant component, are invariant under this choice [2]. In contrast, it has recently been shown that this distance function influences the speed of rumor spreading [7]. What is more, the robustness of a GIRG, that is, how much of the graph's giant component can be removed before it falls apart (into chunks of comparable size) crucially depends on it [8]. Questions of robustness and fragility such as this one have been widely researched both for random graph models and real-world networks, in the quest of properties that are universal across networks. Robustness can be examined with respect to vertex or edge removal - and removals can be adversarial or random. Understanding which removal strategies are most successful has developed into a research direction of its own [1, 6, 11, 12].

In our present work, we consider robustness with respect to adversarial edge removal. In GIRGs, choosing the Euclidean metric induces a graph which contains small separators, that is, edge cuts of sublinear size, which split the giant component into two connected components of linear size. Using instead the so-called minimum-component distance function (MCD), then in dimensions  $d \geq 2$  Lengler and Todorović have demonstrated that all sublinear separators disappear with high probability<sup>1</sup> [8]. This coincides with the picture that we encounter in Erdős-Rényi random graphs at edge probability  $p = \frac{1+O(1)}{n}$ , as shown by Luczak and McDiarmid [9]. In both cases, the proofs proceed by a two-round exposure of the edges in the graph. After the first, larger, batch of edges is unveiled, the graph already contains a giant component - and this component contains only few sparse cuts. In the second round, one can show that enough edges are sprinkled between the vertices that are only sparsely connected, so that all sublinear cuts will disappear. Compared to Erdős-Rényi graphs, several complications arise. It is, in particular, not possible to sample the edges in two independent batches. Instead, one can produce two "almost independent" batches of edges by unveiling first a number of  $d - 1$  coordinates of the vertices and then unveiling their remaining  $d$ th coordinate. It is unavoidable that the giant component grows substantially, as a constant fraction of the total number of edges in the graph is unveiled this way, creating many new potential vertex bipartitions in the giant component which may yield sublinear cuts. This problem can be addressed by uncovering only coordinate information of bounded-weight vertices within the giant component for the second batch. In their elegant proof, Lengler and Todorović then show that edges incident to this vertex set do not increase the size of the giant component by too much, using an Azuma-Hoeffding type estimate.

---

<sup>1</sup> In dimension  $d = 1$ , the two distance functions coincide.

As we will demonstrate, this procedure can be extended far beyond the case of MCD-GIRGs. We first generalize GIRGs to accommodate a large class of underlying distance functions, which we call Boolean Distance Functions (BDF). Intuitively speaking, this family of distance functions covers arbitrary combinations of minima and maxima of subsets of coordinates. For example, even if two individuals have exactly the same hobby (in this case, the minimum component distance is zero), they may still be unlikely to know each other if they live in different countries. A more refined notion could be for example, the formula  $dist := \min\{dist_{work}, \max\{dist_{hobby}, dist_{residence}\}\}$  which encodes that two individuals are likely to know each other if they either work in closely related fields, or if they have at the same time similar hobbies and locations. This illustrates the potential of BDF for building more realistic models for real-world networks. We remark that the dimension  $d$  of the underlying geometric space is always assumed to be a constant with respect to the number of vertices  $n$ .

In many real-world instances, particularly social networks, connections are formed based on some feature set that will naturally dominate, and not based on an *averaged* similarity as the one captured by Euclidean distances, which weight all features in a symmetric way. One may think of an edge encoding that two people, i.e. vertices, know each other, and closeness in a specific coordinate encoding having a parent in common – or more generally how many generations one needs to go back to find a common ancestor. Even if all other features, e.g. education, age, wealth, domicile, interests, differ greatly, sharing a parent will usually ensure that two people know each other. Sometimes single features are not dominant enough, but being similar in a specific subset of  $k$  features will render the other  $d - k$  features obsolete, i.e. these  $k$  features will suffice to guarantee a lower bound on the connection probability. One such example could be a feature that encodes that two people live in the same town with a population of ca. 100'000 people (which does not suffice for a connection) and another playing table tennis in a club (which on its own also is not sufficient). But it is plausible that playing in a table tennis club and living in the same town jointly ensures a constant lower bound on the connection probability. This is in line with the complex mechanisms underlying the observed formation of social ties in real networks, where similarities with regard to certain features are dominant but the baseline provided by similarity in this regard is combined with similarities or differences in other dimensions [10]. Such more sophisticated situations can be captured with BDF but are completely impossible to encode in the Euclidean setting - and, due to the equivalence of norms on finite-dimensional vector spaces - by any other metric induced by a norm.

The precise definition of Boolean Distance Functions can be found in Definition 2. In this family of GIRGs, there is only one subfamily, which we call Single-Coordinate Outer-Max (SCOM) GIRGs, that contains small separators. These collections of GIRGs exhibit a distance function that can be expressed as the maximum of one singled-out coordinate and an arbitrary “Boolean” combination of the other coordinates. An example of such a SCOM BDF would be  $dist := \max\{dist_{residence}, \min\{dist_{hobby}, dist_{work}\}\}$ . Precisely, we prove the following first main result:

**Theorem 1.** *Let  $G$  be a GIRG induced by a SCOM BDF  $\kappa$  acting on the  $d$ -dimensional torus  $\mathbb{T}^d$ . Then, with probability  $1 - o(1)$ , the giant component of  $G$  has a separator of size  $o(n)$ .*

We note that the separators can be described explicitly. One can consider two hyperplanes that are perpendicular to the singled-out coordinate axis, which split the ground space into two halves. It is then possible to show that a sub-linear number of edges cuts across the hyperplanes, connecting two linear-sized components, hence yielding the natural small separators.

We then proceed to show - leveraging a modified version of the algorithm by Lengler and Todorović [8] - that for all other BDF-GIRGs, a two-round exposure of the vertex coordinates generates a graph where all potential sparse cuts in the giant component are erased in the second round. The two key modifications concern first the partition of the coordinates into batches. Here, the crucial ingredient is an upper bound on the distance function by a minimum of two distance functions which involve each only a subset of (disjoint) coordinates. This then enables an extension of the two-round exposure to arbitrary non-SCOM BDF-GIRGs, by allowing an estimate of the edges contributed by the second exposure round. We also need to adjust the criteria for when edges are inserted. With these additional insights we are able to prove our second main result:

**Theorem 2.** *Let  $G$  be a GIRG induced by a non-SCOM BDF  $\kappa$  acting on the  $d$ -dimensional torus  $\mathbb{T}^d$ . Then, with probability  $1 - o(1)$ , the giant component of  $G$  has no separator of size  $o(n)$ .*

Together, Theorems 1 and 2 provide a complete characterization of the occurrence of sublinear separators in BDF-GIRGs. Finally, one can show that all BDFs satisfy a stochastic version of the triangle inequality. From this, it immediately follows that all BDF-GIRGs exhibit clustering. In order to formally state the theorem, we first require the definition of the clustering coefficient, taken from [8].

**Definition 1.** *For a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  the clustering coefficient of a vertex  $v \in \mathcal{V}$  is defined as*

$$\text{CC}(v) := \begin{cases} \frac{1}{\binom{\deg(v)}{2}} \cdot \#\{\text{triangles in } \mathcal{G} \text{ containing } v\}, & \text{if } \deg(v) \geq 2, \\ 0, & \text{otherwise,} \end{cases}$$

and the (mean) clustering coefficient of  $\mathcal{G}$  is defined as  $\text{CC}(\mathcal{G}) := \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{G}} \text{CC}(v)$ .

Our final main result is then the following:

**Theorem 3.** *Let  $G$  be a GIRG induced by a BDF  $\kappa$  acting on the  $d$ -dimensional torus  $\mathbb{T}^d$ . Then, with probability  $1 - o(1)$ , its clustering coefficient is bounded away from zero, i.e.  $\text{CC}(G) = \Theta(1)$ .*

Its proof is very similar to the proof of Theorem A. 4 in [8] and is omitted due to the space constraints.<sup>2</sup>

In social networks, the clustering coefficient has a natural interpretation: What proportion of the friends of a node are also mutual friends ? In sparse networks, a pair of nodes  $u, v$  having a common friend thus greatly boosts their chances of being directly connected. Recently it has further been shown that, in a different setting, namely for GIRGs whose distance function is induced by an  $L_p$ -norm, the clustering coefficient can be used to estimate the dimension of the underlying ground space [5]. In the rest of the paper, we will say that an event happens *with high probability* (*w.h.p.*) if it happens with probability  $1 - o(1)$ .

## 2 Geometric Inhomogeneous Random Graphs and Underlying Geometric Spaces

In this section, we define Boolean Distance Functions (BDFs) as well as Geometric Inhomogeneous Random Graphs (GIRGs). At the end of the section we give some useful properties of GIRGs.

### 2.1 Definition of Boolean Distance Functions

We start by introducing the notion of Boolean Distance Functions. They define a symmetric and translation-invariant distance between any two points in the  $d$ -dimensional torus  $\mathbb{T}^d := \mathbb{R}^d / \mathbb{Z}^d$ , and hence can be characterized by an even function  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$ . The distance  $dist(x, y)$  between two points  $x, y \in \mathbb{T}^d$  is then given by  $dist(x, y) := \kappa(x - y)$ . For the sake of simplicity, all the properties of the distance  $dist(\cdot, \cdot)$  will be expressed in terms of  $\kappa(\cdot)$ . Note that on the one-dimensional torus, the distance between two points  $x, y \in [0, 1]$  is given by

$$|x - y|_T := \min(|x - y|, 1 - |x - y|).$$

A Boolean Distance Function is then defined recursively as follows.

**Definition 2.** Let  $d \in \mathbb{N}$  be a positive integer, let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$ , and let  $x = (x_1, x_2, \dots, x_d) \in \mathbb{T}^d$  be an arbitrary point. Then  $\kappa$  is a Boolean Distance Function (BDF) if:

- When  $d = 1$ , then  $\kappa(x) = |x|_T$ .
- When  $d \geq 2$ , then there exists a non-empty proper subset  $S \subsetneq [d]$  of coordinates such that

$$\kappa(x) = \max(\kappa_1((x_i)_{i \in S}), \kappa_2((x_i)_{i \notin S})) \quad \text{or} \quad \kappa(x) = \min(\kappa_1((x_i)_{i \in S}), \kappa_2((x_i)_{i \notin S})),$$

where  $\kappa_1 : \mathbb{T}^{|S|} \rightarrow \mathbb{R}_{\geq 0}$  and  $\kappa_2 : \mathbb{T}^{d-|S|} \rightarrow \mathbb{R}_{\geq 0}$  are Boolean Distance Functions.

---

<sup>2</sup> For the sake of brevity we only provide proof sketches throughout the paper, but full proofs are available at arXiv:2405.19369.

In the  $d \geq 2$  case, we call  $\kappa_1$  and  $\kappa_2$  the comprising functions of  $\kappa$ . Moreover, we say that  $\kappa$  is an outer-max BDF if it is defined recursively as the maximum of two other BDFs, and we say  $\kappa$  is an outer-min BDF if it is defined recursively as the minimum of two other BDFs.

Since we only work with the torus geometry in this paper, we will simply write  $|x|$  instead of  $|x|_T$ . Observe that the max-norm  $\kappa(x) := \max_{i \in [d]} |x_i|$  and the minimum component distance (MCD)  $\kappa(x) := \min_{i \in [d]} |x_i|$  are both BDFs.

Given a Boolean Distance Function  $\kappa$ , we denote by  $V_\kappa(r)$  the volume of a ball of radius  $r$  with respect to the distance function induced by  $\kappa$ , i.e.,  $V_\kappa(r)$  is the Lebesgue measure of  $\{x \in \mathbb{T}^d : \kappa(x) < r\}$ . We will sometimes drop the  $\kappa$  index and just write  $V(r)$  for that quantity when the underlying BDF is clear from context.

We now define the notion of depth of a Boolean Distance Function. As we will see in Lemma 1,  $V_\kappa(r)$  is characterized (as a function of  $r$ ) by the depth of  $\kappa$ .

**Definition 3.** Let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a Boolean Distance Function. Then the depth of  $\kappa$ , written  $\mathcal{D}(\kappa)$ , is defined recursively as follows:

- When  $d = 1$ , then  $\mathcal{D}(\kappa) := 1$ .
- When  $d \geq 2$ , then  $\mathcal{D}(\kappa) := \mathcal{D}(\kappa_1) + \mathcal{D}(\kappa_2)$  if  $\kappa$  is outer-max, and  $\mathcal{D}(\kappa) := \min(\mathcal{D}(\kappa_1), \mathcal{D}(\kappa_2))$  if  $\kappa$  is outer-min, where  $\kappa_1$  and  $\kappa_2$  are the comprising functions of  $\kappa$ .

We now define the notion of Single-Coordinate Outer-Max (SCOM) Boolean Distance Function. These are BDFs that can be written as the maximum of a single coordinate and some other BDF acting on the remaining coordinates. This property characterizes whether BDF-GIRGs have small separators or not.

**Definition 4.** Let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a Boolean Distance Function. We say that  $\kappa$  is Single-Coordinate Outer-Max (SCOM) if it can be written as

$$\kappa(x) = \max(|x_k|, \kappa_0((x_i)_{i \neq k}))$$

for some coordinate  $k \in [d]$  and some BDF  $\kappa_0 : \mathbb{T}^{d-1} \rightarrow \mathbb{R}_{\geq 0}$ . In dimension 1, we also say that  $\kappa(x) = |x|$  is a SCOM BDF.

Note that this means that the unique BDF acting on  $d = 1$  dimension is SCOM.

## 2.2 Geometric Inhomogeneous Random Graphs

We now define Geometric Inhomogeneous Random Graphs (GIRGs). This model was initially introduced for the max-norm in [3]. Here, we generalize this definition to cover any Boolean Distance Function as the underlying distance. Throughout the paper, we consider undirected graphs with vertex set  $\mathcal{V} := [n]$  and edge set  $\mathcal{E}$ . A sequence  $(a_i)_{i \in [n]}$  is said to be power-law distributed with exponent  $\beta$  if  $|\{i : a_i \geq k\}| = \Theta(nk^{1-\beta})$ .

**Definition 5.** Let  $\beta \in (2, 3)$ ,  $\alpha > 1$ ,  $d \in \mathbb{N}$ , and let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a Boolean Distance Function. Let  $(w_v)_{v \in \mathcal{V}}$  be a sequence of weights that is power-law distributed with exponent  $\beta$ . A  $\kappa$ -Geometric Inhomogeneous Random Graph ( $\kappa$ -GIRG) is obtained by the following two-step procedure:

1. Every vertex  $v \in \mathcal{V}$  draws independently and uniformly at random a position  $x_v$  in the torus  $\mathbb{T}^d$ .
2. For every two distinct vertices  $u, v \in \mathcal{V}$ , add the edge  $uv \in \mathcal{E}$  independently with some probability  $p_{uv}$  satisfying

$$c_L \cdot \min \left\{ \frac{w_u w_v}{n \cdot V_\kappa(\kappa(x_u - x_v))}, 1 \right\}^\alpha \leq p_{uv} \leq c_U \cdot \min \left\{ \frac{w_u w_v}{n \cdot V_\kappa(\kappa(x_u - x_v))}, 1 \right\}^\alpha$$

for some constants  $c_U \geq c_L > 0$ .

We remark that [2] uses a very general definition of GIRG, and in particular our definition of  $\kappa$ -GIRG (Definition 5) fits in their geometric setting. Hence,  $\kappa$ -GIRGs have the following properties:

- Their degree distribution follows a power-law distribution, with the same exponent  $\beta$  as their weight sequence [2, Theorem 2.1].
- They have a unique connected component of linear size, which we call the *giant component* of the graph, and all other connected components are at most poly-logarithmic in size [2, Theorem 2.2].
- The average distance between two vertices in the giant component is  $\Theta(\log \log n)$  [2, Theorem 2.3].

### 3 Properties of Boolean Distance Functions

In this section, we provide key propositions about Boolean Distance Functions. These propositions will be used for deriving our main results, but we believe they are also interesting results on their own. Remember that  $V_\kappa(r)$  denotes the volume of a ball of radius  $r$ , with the distances measured with respect to  $\kappa$ . We start by analyzing how  $V_\kappa(r)$  behaves as  $r \rightarrow 0$ . When  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  is the max-norm we have  $V_\kappa(r) = \Theta(r^d)$ , while for the MCD  $\kappa(x) = \min_{i \in [d]} |x_i|$  we have  $V_\kappa(r) = \Theta(r)$ . The following proposition generalizes this to arbitrary BDFs.

**Proposition 1.** Let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a Boolean Distance Function of depth  $D(\kappa)$  (see Definition 3). Then  $V_\kappa(r) = \Theta(r^{D(\kappa)})$  as  $r \rightarrow 0$ .

The proof proceeds by induction on the dimension  $d$ . The induction step consists of writing  $V_\kappa(r)$  as an integral and, using the recursive structure of the BDF  $\kappa$ , writing this integral as the product of two integrals, splitting into two different cases depending on whether  $\kappa$  is outer-max or outer-min.

In the next proposition, we upper-bound a BDF  $\kappa$  by a max-norm over a subset of the coordinates that has size exactly  $D(\kappa)$ .

**Proposition 2.** Let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a Boolean Distance Function. Then there exists a subset  $S \subseteq [d]$  of the coordinates with  $|S| = \mathcal{D}(\kappa)$  such that  $\kappa(x) \leq \max_{i \in S} |x_i|$  for all  $x \in \mathbb{T}^d$ .

The proof is again by induction on the dimension. For the induction step, we have (by the induction hypothesis) two subsets  $S_1, S_2 \subseteq [d]$  such that the max-norm on these subsets of coordinates is an upper bound for the comprising functions of  $\kappa$ : taking  $S$  to be the union of  $S_1$  and  $S_2$  if  $\kappa$  is outer-max, respectively the set of smallest cardinality among  $S_1, S_2$  if  $\kappa$  is outer-min, completes the induction step.

The next proposition allows us to upper-bound any non-SCOM outer-max BDF by an outer-min BDF acting on the same set of coordinates.

**Proposition 3.** Let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a non-SCOM outer-max Boolean Distance Function. Then there exists an outer-min BDF  $\kappa' : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  such that  $\mathcal{D}(\kappa) = \mathcal{D}(\kappa')$  and  $\kappa(x) \leq \kappa'(x)$  for all  $x \in \mathbb{T}^d$ .

The proof also proceeds by induction on  $d$ , with the key inequality being

$$\max(\min(x_1, x_2), \min(x_3, x_4)) \leq \min(\max(x_1, x_3), \max(x_2, x_4)).$$

Combining the three previous propositions, we get the following result, which is one of the main building blocks for the proof of Theorem 2.

**Proposition 4.** Let  $\kappa : \mathbb{T}^d \rightarrow \mathbb{R}_{\geq 0}$  be a non-SCOM Boolean Distance Function. There exist disjoint subsets  $S_1, S_2 \subseteq [d]$  with  $\min(|S_1|, |S_2|) = \mathcal{D}(\kappa)$  such that, with  $\kappa'((x_i)_{i \in S_1 \cup S_2}) := \min(\max_{i \in S_1} |x_i|, \max_{i \in S_2} |x_i|)$ , it holds for all  $x \in \mathbb{T}^d$  that  $\kappa(x) \leq \kappa'((x_i)_{i \in S_1 \cup S_2})$ . Moreover,  $\kappa'$  is a BDF and there exists a constant  $c > 0$  such that  $V_\kappa(r) \leq cV_{\kappa'}(r)$  for all  $r \geq 0$ .

## 4 Proof Outline

### 4.1 Small Separators in Single-Coordinate-Outer-Max GIRGs

Our first main result states that GIRGs induced by BDFs that are SCOM have natural sub-linear separators, which, as it turns out, run along the singled-out coordinate axis. The key proof idea is to partition the ground space along the singled-out coordinate axis into two half-spaces of equal volume, ensuring that each half-space contains a linear number of the vertices of the giant component. We then upper-bound the number of edges crossing the separating hyperplanes. Each pair of vertices will contribute an edge intersecting one of the two hyperplanes if and only if the vertices lie in different half-spaces and are connected by an edge. The joint probability of this event can be computed using the law of iterated probability. One can show that the number of crossing edges is  $o(n)$  with high probability. Thus the two subgraphs can be disconnected through the removal of the  $o(n)$  crossing edges, yielding Theorem 1. Note that the max-norm is also a SCOM BDF, hence our result extends the result proved for max-norms in [3].

## 4.2 Robustness of Non-Single-Coordinate-Outer-Max GIRGs

Our second main result states that GIRGs induced by BDFs that are not SCOM are robust, i.e., they do not contain separators of sub-linear size in the giant component. First, we restate a lemma that bounds the number of small cuts in connected graphs [9]. This lemma is the inspiration for the proof of robustness of MCD-GIRGs [8] and will also be used in our proof of Theorem 2.

**Lemma 1 (Lemma 7 in [9]).** *For any  $\varepsilon > 0$  there exists  $\eta_0(\varepsilon) > 0$  and  $n_0$  such that for all  $n \geq n_0$ , and for all connected graphs  $G$  with  $n$  vertices, there are at most  $(1 + \varepsilon)^n$  many bipartitions of  $G$  with at most  $\eta_0 n$  cross-edges.*

**Edge Insertion Criteria.** To extend the two-round edge exposure procedure from MCD-GIRGs to general  $\kappa$ -GIRGs, we will use the upper bounds in terms of an outer-min distance function for the BDF at hand that were stated in Sect. 3. Our overarching goal will be to expose at first only a subset of the coordinates of each vertex, insert some of the edges based on this partial information, then reveal the remaining coordinates which will lead to the creation of additional edges. More concretely, our aim will be to construct for each pair of vertices  $u, v$  a pair of independent random variables  $(Y_{uv}^1, Y_{uv}^2)$ , inserting edges in the first round if  $Y_{uv}^1 < p_{uv}$  and in the second round if  $Y_{uv}^2 < p_{uv}$ . The careful modification of the distribution of these two random variables will allow us to emulate in two rounds the one-round sampling procedure where an edge is inserted if  $Y_{uv} < p_{uv}$ , with  $Y_{uv}$  drawn uniformly at random in the interval  $[0, 1]$ .

Consider therefore a  $\kappa_0$ -GIRG for some non-SCOM BDF  $\kappa_0$ . By Proposition 4, we can find disjoint subsets  $S_1, S_2 \subseteq [d]$ , with  $\mathcal{D}(\kappa_0) = \min(|S_1|, |S_2|)$  such that the outer-min BDF given by

$$\kappa((x_i)_{i \in S_1 \cup S_2}) := \min\left(\max_{i \in S_1} |x_i|, \max_{i \in S_2} |x_i|\right) =: \min(\|(x)_{S_1}\|_\infty, \|(x)_{S_2}\|_\infty)$$

is an upper bound for  $\kappa_0(x)$ . This implies in particular that there exists a  $\kappa$ -GIRG for some sufficiently small choice of constants  $c'_U, c'_L$  such that for every  $u, v \in \mathcal{V}$ ,  $p_\kappa(u, v) \leq p_{\kappa_0}(u, v)$ , where the former denote the connection probabilities of vertices  $u$  and  $v$  in a  $\kappa$ -GIRG respectively in a  $\kappa_0$ -GIRG. Without loss of generality, we will assume that  $S_1 \subseteq \{1, \dots, d-m\}$  and  $S_2 = \{d-m+1, \dots, d\}$ . Additionally, we can assume that  $m = |S_2| \leq |S_1|$ , which also means that  $D := \mathcal{D}(\kappa_0) = \mathcal{D}(\kappa) = m$ . We will modify the original algorithm given in [8] by splitting the sampling process into the first  $d-m$  and the last  $m$  coordinates (in the original approach by Lengler and Todorović, the split was between the first  $d-1$  coordinates and the last coordinate).

Let  $Y_{uv}$  be a uniform random variable on the interval  $[0, 1]$ , and define two iid random variables  $Y_{uv}^1, Y_{uv}^2$  distributed as follows:

$$\mathbb{P}[Y_{uv}^1 < c] = \mathbb{P}[Y_{uv}^2 < c] = 1 - \sqrt{1-c}.$$

Then the following also hold:

$$\begin{aligned} c/2 &\leq \mathbb{P}[Y_{uv}^1 < c] = \mathbb{P}[Y_{uv}^2 < c] \leq c, \\ \mathbb{P}[\min(Y_{uv}^1, Y_{uv}^2) < c] &= \mathbb{P}[Y_{uv} < c] = c. \end{aligned}$$

Let

$$p_{uv}(c, x) := c \cdot \min \left\{ 1, \left( \frac{w_u w_v}{n|x|^D} \right)^\alpha \right\},$$

and define the Edge Insertion Criterion ([EIC](#)) as

$$Y_{uv} =: \min(Y_{uv}^1, Y_{uv}^2) < p_{uv}(c_L, \kappa_0(x_u - x_v)). \quad (\text{EIC})$$

By our choice of  $c'_L$ , we have the following lower bound:

$$\begin{aligned} p_{uv}(c_L, \kappa_0(x_u - x_v)) &\geq p_{uv}(c'_L, \kappa(x_u - x_v)) \\ &= \max(p_{uv}(c'_L, \|(x_u - x_v)_{S_1}\|_\infty), p_{uv}(c'_L, \|(x_u - x_v)_{S_2}\|_\infty)). \end{aligned}$$

Thus, we obtain the two following sufficient conditions for edge insertion:

$$Y_{uv}^1 < p_{uv}(c'_L, \|(x_u - x_v)_{S_1}\|_\infty), \quad (\text{LB1})$$

$$Y_{uv}^2 < p_{uv}(c'_L, \|(x_u - x_v)_{S_2}\|_\infty). \quad (\text{LB2})$$

Crucially, notice that if we insert the edges according to [\(LB1\)](#) first, we obtain a GIRG (with respect to the max-norm on  $\mathbb{T}^{|S_1|}$ ), which means that it satisfies all the properties mentioned in [Sect. 2.2](#). In particular, after the insertion of the first batch of edges, our graph will already contain a (unique) giant component.

We next describe an algorithm that is central in proving the robustness of a  $\kappa_0$ -GIRG. It follows closely the algorithm used in [\[8\]](#), but we need to modify it to make the proof work in our more general setting. The main modifications are as follows. Firstly, we use the updated [\(EIC\)](#), [\(LB1\)](#) and [\(LB2\)](#). Secondly, instead of first sampling the first  $d - 1$  coordinates and then the last coordinate, we first sample the first  $d - m$  coordinates and then sample the last  $m$  coordinates.

**Sampling Algorithm.** In this section we describe the procedure for uncovering the edges of the  $\kappa_0$ -GIRG. We fix some constant  $\delta \in (0, 1)$ . The algorithm can be decomposed into 6 phases.

**Phase 1.** We start by sampling  $Y_{uv}^1$  for all pairs of vertices  $u, v \in \mathcal{V}$ . Additionally, for every vertex  $u \in \mathcal{V}$ , we sample the first  $d - m$  coordinates of its position  $(x_{ui})_{1 \leq i \leq d-m}$  independently and uniformly at random from  $[0, 1]$ . This is sufficient to determine the graph induced by the edge insertion criterion [\(LB1\)](#), which we refer to as  $G_1$ . The graph  $G_1$  has a unique linear-sized component (the giant) with at least  $s_{max}n$  vertices w.h.p. for some constant  $s_{max} \in (0, 1)$ ; we will assume that this holds for the rest of the proof. We denote this giant by  $K_{max}^1$ .

**Phase 2.** Let  $B'$  be a constant such that at least half the vertices of  $K_{max}^1$  have weight less than  $B'$ . Next, we sub-sample  $F' \subseteq \mathcal{V}$  by including every vertex (not just those in the giant) with weight less than  $B'$  into  $F'$  independently with probability  $4f/s_{max}$  for a constant  $0 < f < (s_{max}/12) \cdot \min\{\delta, s_{max}\}$  to be determined later.

**Phase 3.** Now set  $F := F' \cap K_{max}^1$ . It is straightforward to see that by the choice of parameters it holds that  $2fn \leq \mathbb{E}[|F|] \leq \mathbb{E}[|F'|] \leq 4fn/s_{max}$ . Additionally, by Chernoff's bounds, we have that  $fn \leq |F| \leq |F'| \leq 6fn/s_{max}$  w.h.p.; we will assume that this holds for the rest of the proof.

The final three phases are split up into  $n$  steps in total (one step for each vertex). In each step, we draw the last  $m$  coordinates of some vertex and potentially add some incident edges according to the edge insertion criterion (EIC). The order in which the vertices are treated is as follows - first the vertices that are not in  $K_{max}^1$  (Phase 4), then the remaining vertices that are not in  $F$  (Phase 5), and finally the vertices that are in  $F$  (Phase 6). Thus, if we order the vertices as  $u_1, u_2, \dots, u_n$  in order to have  $K_{max}^1 = \{u_i \mid |\mathcal{V} \setminus K_{max}^1| < i \leq n\}$  and  $F = \{u_i \mid |\mathcal{V} \setminus F| < i \leq n\}$ , then the  $k$ th step can be described as follows:

- Draw  $(x_{ki})_{d-m < i \leq m}$  each independently and uniformly at random from  $[0, 1]$ . (Note that we denote  $x_{u_k}$  by  $x_k$ )
- For all  $1 \leq j < k$ , sample  $Y_{jk}^2$  independently.
- For all  $1 \leq j < k$ , add an edge between  $u_j$  and  $u_k$  if (EIC)s satisfied.

**Phases 4 - 6.** Perform steps 1 to  $|\mathcal{V} \setminus K_{max}^1|$  (Phase 4), then steps  $|\mathcal{V} \setminus K_{max}^1| + 1$  to  $|\mathcal{V} \setminus F|$  (Phase 5). Finally perform steps  $|\mathcal{V} \setminus F| + 1$  to  $n$  (Phase 6).

We denote the resulting graph after Phase  $2+i$  for  $i = 2, 3, 4$  by  $G_i$  and the corresponding giant component that contains  $K_{max}^1$  by  $K_{max}^i$ . In the remainder we will show that the last phase destroys all sublinear cuts present in the giant component of  $G_3$ , while adding only a small number of vertices to the giant.

**Proof of Theorem 2.** First, we observe, recorded in Lemma 2, that the neighbors of a given vertex cannot be too concentrated in a small region. To help quantify this we define the notion of cells. For some  $M > 0$ , we partition  $[0, 1]$  into  $M$  sub-intervals of equal length  $I_j := [j/M, (j+1)/M]$  for  $0 \leq j < M$ . We define a  $M$ -cell (or just cell if  $M$  is clear from context) to be a region of the form  $\mathbb{T}^{d-m} \times I_{j_{d-m+1}} \times I_{j_{d-m+2}} \times \dots \times I_{j_d}$ , where  $0 \leq j_{d-m+1}, j_{d-m+2}, \dots, j_d < M$ . It is easy to see that the entire space  $\mathbb{T}^d$  is partitioned into  $M^m$  cells. We call such a partition an  $M$ -cell partition. Note that the cell which contains a given vertex  $v$  can be completely characterized by the last  $m$  coordinates of  $v$ . Lemma 2 can now be derived by observing that a set  $S$  of  $rn$  cells has volume  $rn \cdot M^{-m} \in [rl2^{-m}, rl]$  and thus contains an expected number of vertices between  $rnl2^{-m}$  and  $rnl$ .

**Lemma 2.** Let  $M = \lceil (n/l)^{1/m} \rceil$  for some constant  $l \in (0, 1]$ , and consider the  $M$ -cell partition of  $\mathbb{T}^d$ . Then, for every  $\delta \in (0, 1)$  and every  $l$ , there exists a constant  $r(\delta, l, m) > 0$  such that the following property holds - with probability  $1 - e^{-\Theta(n)}$ , there is no set  $S$  of  $rn$  cells (of the considered partition) such that there are at least  $\delta n/2$  vertices in the cells of  $S$ .

Intuitively, Lemma 2 says that since the positions of vertices are randomly chosen, they must be somewhat spread out in the last  $m$  coordinates. Thus, we must expect that when we uncover vertices later in the ordering, they are close enough to previously uncovered vertices to have a good chance of edge formation. Indeed, one can show a constant lower bound on the probability of

edge formation in every step of phases 5 and 6. The remainder of the proof now closely models that of Theorem 3.2 in [8]. Due to Lemma 1, which restricts the number of sparse bipartitions in a connected graph, there are no small cuts in  $K_{max}^3$ . Then, we can show that in the last phase the giant component does not grow too much, more precisely that  $|K_{max}^4| \leq |K_{max}^3| + 3\delta n$ . Now any sparse cut in  $K_{max}^4$  would induce a sparse cut in  $K_{max}^3$ , which does not exist. Hence no sparse cut can exist in  $K_{max}^4$  either. Since any  $\kappa$ -GIRG w.h.p. has a unique giant component, this yields Theorem 2.

## References

1. Bellingeri, M., Lu, Z.M., Cassi, D., Scotognella, F.: Analyses of the response of a complex weighted network to nodes removal strategies considering links weight: the case of the Beijing urban road system. *Mod. Phys. Lett. B* **32**(05), 1850067 (2018)
2. Bringmann, K., Keusch, R., Lengler, J.: Average distance in a general class of scale-free networks with underlying geometry. arXiv preprint [arXiv:1602.05712](https://arxiv.org/abs/1602.05712) (2016)
3. Bringmann, K., Keusch, R., Lengler, J.: Geometric inhomogeneous random graphs. *Theoret. Comput. Sci.* **760**, 35–54 (2019)
4. Dayan, B., Kaufmann, M., Schaller, U.: Expressivity of geometric inhomogeneous random graphs-metric and non-metric, pp. 85–100 (2024). [https://doi.org/10.1007/978-3-031-57515-0\\_7](https://doi.org/10.1007/978-3-031-57515-0_7)
5. Friedrich, T., Göbel, A., Katzmann, M., Schiller, L.: A simple statistic for determining the dimensionality of complex networks (2023)
6. Iyer, S., Killingback, T., Sundaram, B., Wang, Z.: Attack robustness and centrality of complex networks. *PloS One* **8**(4), e59613 (2013)
7. Kaufmann, M., Lakis, K., Lengler, J., Ravi, R.R., Schaller, U., Sturm, K.: Rumour spreading depends on the latent geometry and degree distribution in social network models. arXiv preprint [arXiv:2408.01268](https://arxiv.org/abs/2408.01268) (2024)
8. Lengler, J., Todorović, L.: Existence of small separators depends on geometry for geometric inhomogeneous random graphs. arXiv preprint [arXiv:1711.03814](https://arxiv.org/abs/1711.03814) (2017)
9. Luczak, M., McDiarmid, C.: Bisecting sparse random graphs. *Random Struct. Algorithms* **18**, 31–38 (2001). [https://doi.org/10.1002/1098-2418\(200101\)18:13.0.CO;2-1](https://doi.org/10.1002/1098-2418(200101)18:13.0.CO;2-1)
10. Mcpherson, M., Smith-Lovin, L., Cook, J.: Birds of a feather: homophily in social networks. *Annu. Rev. Sociol.* **27**, 415–444 (2001). <https://doi.org/10.1146/annurev.soc.27.1.415>
11. Tian, L., Bashan, A., Shi, D.N., Liu, Y.Y.: Articulation points in complex networks. *Nat. Commun.* **8**(1), 14223 (2017)
12. Wandelt, S., Sun, X., Feng, D., Zanin, M., Havlin, S.: A comparative analysis of approaches to network-dismantling. *Sci. Rep.* **8**(1), 13513 (2018)

## **Structural Network Measures**



# Elucidation of the Relationship Between the Box and Cluster Dimensions Using a Small-World Fractal Tree Model

Nobutoshi Ikeda<sup>(✉)</sup>

Tohoku Seikatsu Bunka Junior College, 1-18-2 Niji-no-Oka, Izumi-ku,  
Sendai 981-8585, Japan  
[ikeda.nobu@nifty.com](mailto:ikeda.nobu@nifty.com)

**Abstract.** Fractality is an important feature prevalent in real networks. However, a fundamental problem regarding the relationship between the fractal box dimension  $d_b$  and the cluster dimension  $d_c$  has remained unsolved. In this study, we develop an algorithm that generates the small-world scale-free fractal tree graph and perform a thorough exploration of the relationship between  $d_b$  and  $d_c$  using model networks generated by this algorithm. We derive an explicit relation that relates  $d_b$  to  $d_c$  in terms of the mean distance and diameter of networks. We show the validity of this relation through the investigation of the small-world scale-free fractal tree graph using a scale transformation that enables a direct comparison between the mean box and cluster sizes used to define the fractal dimensions. Furthermore, we show that the same analysis can be applied to real networks that exhibit fractal properties.

**Keywords:** fractal dimension · box-counting method · cluster-growing method · small-world · fractal tree

## 1 Introduction

Fractality is one of the properties commonly found in real networks, such as biological and social networks [1]. These common properties are of fundamental interest because they suggest the existence of universal mechanisms that govern the evolution of complex networks. However, much remains unknown about the origins and properties of fractal networks [2].

There are two representative methods for defining fractal networks. The first one, the cluster-growing method, is defined using the mean distance distribution measured from many randomly chosen seed vertices [3] as follows:

$$\langle M_c(l_c) \rangle = \langle k+1 \rangle l_c^{d_c}, \quad (1)$$

where  $\langle M_c(l_c) \rangle$  is the average number of vertices that can be reached from randomly chosen seed vertices in at most  $l_c$  steps, and  $\langle k+1 \rangle$  denotes the mean degree added by 1. We refer to  $\langle M_c(l_c) \rangle$  as the mean cluster size. If (1) holds in

a finite range of  $l_c$ , the graph is regarded as a fractal network with fractal cluster dimension  $d_c$ . In general, the relation (1) can be observed in a very narrow range of  $l_c$ , because, as  $l_c$  exceeds  $\langle d \rangle$ ,  $M_c(l_c)$  contains most of the nodes in the graph, and thus  $M_c(l_c)$  approaches its maximum value, the number of vertices  $V$ . The other method, the box-counting method, is defined using the number of boxes needed to tile the graph [1],

$$N_b(l_b) \sim l_b^{-d_b}, \quad (2)$$

where  $N_b(l_b)$  is the minimum number of boxes with a diameter smaller than an integer  $l_b$  needed to tile the graph. If (2) holds in a finite range of  $l_b$ , the graph is regarded as a fractal network with fractal box dimension  $d_b$ . Ideally, (2) holds in a range  $2 \leq l_b \leq D + 1$ , where  $D$  is the diameter of the whole graph, because  $N_b(D + 1) = 1$ .

As explained in the first report on the box-counting method [1], there is a natural reason why  $d_c$  and  $d_b$  can take on different values. The average size of boxes used in the box-counting method, which can be estimated as

$$\langle B(l_b) \rangle = V/N_b \sim l_b^{d_b}, \quad (3)$$

may be different from the cluster size  $\langle M_c(l_c) \rangle$  because different clusters used in the cluster-growing method can share many of the same vertices, including the hub vertex, while different boxes used in the box-counting method do not share the same vertices. The difference between  $\langle M_c(l_c) \rangle$  and  $\langle B(l_b) \rangle$  is expected to become more noticeable, especially for small-world and scale-free networks. However, this is only a qualitative explanation for the difference between  $d_c$  and  $d_b$ . It has not yet been clarified to what extent and under what conditions the value of  $d_c$  differs from the value of  $d_b$ .

Fractal network growth models are useful for investigating this problem because it provides data across the entire evolution of a graph, whereas most data for real networks provides information only for a specific time. In our previous studies on fractal network models, we have already shown that the difference between  $d_c$  and  $d_b$  can be attributed to the difference between the behaviors of the mean shortest path length  $\langle d \rangle$  and the diameter of the graph  $D$  with respect to the number of vertices  $V$  in the graph. The investigations were performed on growth network models based on random walks [4, 5] and the generalized  $(u, v)$ -flowers [6]. According to those studies, the value of  $d_c$  must change with respect to  $V$ , corresponding to the small-world behavior of the graph  $\langle d \rangle \sim \log V$ , whereas the value of  $d_b$  remains constant, corresponding to the non-small-world behavior of the diameter  $D \sim V^{1/d_b}$ . The consistency between  $d_c$  and  $d_b$  is supported only when  $\langle d \rangle$  exhibit the same non-small-world behavior as  $D \sim V^{1/d_b}$ .

In this study, we develop an algorithm that produces a fractal tree graph and investigate the relationship between  $d_c$  and  $d_b$  using it. In contrast to the previous models, the proposed algorithm has an advantage in that the values of fractal dimensions can be intentionally changed by setting only a few tunable parameters. In contrast, previous models using random walkers required many parameters and numerous calculations to obtain one result. The proposed algorithm seems to have the drawback that it only generates tree graphs with clustering

coefficients of zero. However, it would not be difficult to increase the clustering coefficient without damaging fractal properties by appropriately adding short-range links, because it is known that adding short-range links into fractal trees hardly damages their fractal property [7].

The algorithm proposed in this study is a revised version of fractal tree algorithms reported in our early work. The first version was devised by mimicking the mechanism by which the movements of random walkers generate the fractal properties of networks [8]. Subsequently, attempts were made to improve the algorithm to generate clearer fractal properties [9]. However, those algorithms were only able to generate fractal networks for a limited range of fractal box dimensions. In this study, we update the algorithm to produce fractal trees in a wider range of fractal dimensions compared to those of the previous models.

Note that “small-world property” refers to the behavior of the mean distance between vertices  $\langle d \rangle \sim \log V$ , not to the diameter of the graph. As mentioned above, and will be shown in Sect. 3, the small-world property in the context of the mean distance  $\langle d \rangle \sim \log V$  can coexist with the non-small-world property in the context of the diameter  $D \sim V^{1/d_b}$ . In this paper, we aim to elucidate the relation between the values of  $d_c$  and  $d_b$  using a model network with those properties. The investigation is also performed on some real networks.

The remainder of this paper is organized as follows. In the next section, we provide a detailed description of the small-world scale-free fractal tree algorithm. Additionally, we examine the dependence of the fractal box dimension on the power-law exponent  $\gamma$  describing the degree distribution. In Sect. 3, we introduce a plot using a scale transformation to compare directly the behaviors of  $\langle M_c(l_c) \rangle$  and  $\langle B_b(l_b) \rangle$ . A theoretical relation between  $d_c$  and  $d_b$  is verified using the plots. In Sect. 4, we apply the same analysis to four real networks. Section 5 summarizes the results.

## 2 Small-World Scale-Free Fractal Tree Algorithm

A tree graph can generally be developed from a root vertex by successively adding vertices with one edge. The fractal tree algorithm proposed in this study is essentially characterized by the following additional three assumptions. First, the tree graph is developed by attaching branches consisting of several vertices, not by attaching single vertices. Second, the probability that a branch attaches to a vertex at a distance  $l$  from the root vertex is proportional to  $M_l l^{-\alpha}$ , where  $M_l$  is the number of vertices at a distance  $l$  from the root vertex. Third, in each layer (a set of vertices at the same distance from the root vertex), the new branch preferentially attaches to a vertex with a probability proportional to the number of offspring of the vertex.

The first assumption was introduced in our previous work in ref. [9], where the length distribution of the branches was assumed to be subject to a geometric distribution. The modification introduced in the present study is to assume that the length distribution  $P(n)$  of the branches is subject to the power-law form distribution

$$P(n) \sim n^{-\gamma_n}. \quad (4)$$

As will be shown in this paper, this revision dramatically improves the reproducibility of the fractal properties of resulting graphs.

The second and third assumptions were first introduced in ref. [8] to reproduce fractal and scale-free properties similar to those found in networks generated by random walks. An important point regarding the algorithm is that the value of the power-law exponent  $\gamma$  that describes the scale-free property of the degree distribution  $V_k \sim k^{-\gamma}$ , where  $V_k$  denotes the number of vertices with degree  $k$ , is adjustable using the relation [8]

$$\frac{N_1}{N_2} \simeq \frac{8}{(\gamma + 3)(\gamma - 2)}, \quad (5)$$

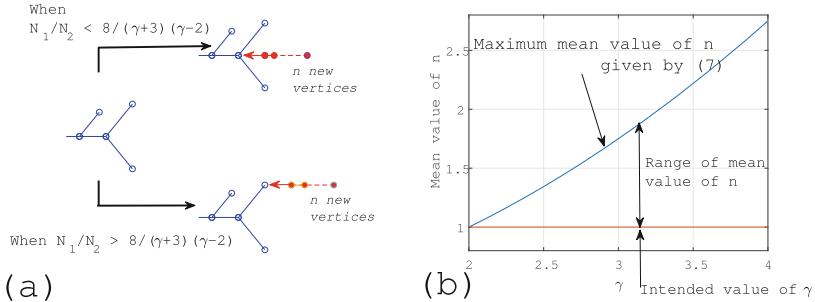
where  $N_1$  is the number of terminal vertices, and  $N_2$  is the number of other vertices. To fix the value of  $\gamma$ , we need to devise a way in the algorithm to keep the ratio  $N_1/N_2$  at a fixed value when each branch links to the existing tree structure.

Specifically, the growing fractal tree algorithm is as follows (most of the following is quoted from ref. [9], except for the revision mentioned above).

### Small-world scale-free fractal tree generation algorithm

1. At initial time  $t = 0$ , a root vertex with degree  $k_{\text{root}}$  is prepared as an initial graph  $G_{t=0}$  (In the numerical calculation, we used  $k_{\text{root}} = 3$  at  $t = 0$ ). At this stage,  $M_0 = 1$ ,  $M_1 = k_{\text{root}}$ , and  $M_l = 0$  for  $l \geq 2$ , where  $M_l$  is the number of vertices at a distance  $l$  from the root vertex (the number of vertices in the  $l$ -th layer). The maximum integer  $l$  that satisfies  $M_l > 0$ ,  $l_{\max}$ , is 1.
2. At each time step  $t$ , one of the following two operations is performed on  $G_t$ .
  - (a) If  $G_t$  satisfies the condition  $N_1/N_2 \leq 8/(\gamma + 3)(\gamma - 2)$ , an integer  $l = 0$  is selected with the probability  $k_{\text{root}}/\sum_{i \in G_t} s_i$ , where  $s_i$  is the number of offspring of the vertex  $i$ . Otherwise, with a probability of  $1 - k_{\text{root}}/\sum_{i \in G_t} s_i$ , an integer is selected from  $l = 1, 2, \dots, l_{\max}$  with a probability proportional to  $M_l l^{-\alpha}$  (In this study, the value of  $\alpha$  is selected as 0.5.). The selected  $l$  is referred to as  $l_s$ . Then, a new branch is linked to one of the vertices in the  $l_s$ -th layer with a probability proportional to  $s_i$  (top of Fig. 1(a)), unless all the vertices in the  $l_s$ -th layer have no offspring. If all the vertices in the  $l_s$ -th layer have no offspring, the new branch is randomly linked to one of the vertices in the  $l_s$ -th layer.
  - (b) If  $G_t$  satisfies the other condition  $N_1/N_2 > 8/(\gamma + 3)(\gamma - 2)$ , an integer is selected from  $l = 1, 2, \dots, l_{\max}$  with a probability proportional to  $M_l l^{-\alpha}$  (the probability that the number  $l = 0$  is assumed to be 0). The selected  $l$  is referred to as  $l_s$ . Then, a new branch is linked randomly to one of the vertices without offspring in the  $l_s$ -th layer (bottom of Fig. 1(a)), unless all the vertices in the  $l_s$ -th layer have offspring. If all the vertices in the  $l_s$ -th layer have offspring, the new branch is linked to a vertex  $i$  in the  $l_s$ -th layer with a probability proportional to  $s_i$ .

3. An integer 1 is added to  $M_{l_s+1}, M_{l_s+2}, \dots, M_{l_s+n}$ , where  $n$  is the number of vertices in the newly added branch, and to  $t$  ( $t \rightarrow t + 1$ ). If  $l_s = 0$ , then an integer 1 is added to  $k_{\text{root}}$  ( $k_{\text{root}} \rightarrow k_{\text{root}} + 1$ ). If  $l_{\max} < l_s + n$ , then  $l_{\max}$  is replaced by  $l_s + n$ .
4. Repeat, starting with the second step.



**Fig. 1.** (a) How to preserve the ratio  $N_1/N_2$ , assuming that a new branch consisting of  $n$  vertices attaches to an existing tree graph. Top: When  $N_1/N_2 \leq 8/(\gamma+3)(\gamma-2)$ , the branch is intended to attach to a vertex with offspring. In this case, the increment of the number of terminal vertices  $\Delta N_1$  is equal to 1, and the increment of the number of other vertices  $\Delta N_2$  is equal to  $n - 1$ . This process enlarges the ratio  $N_1/N_2$ . Bottom: When  $N_1/N_2 > 8/(\gamma+3)(\gamma-2)$ , the branch is intended to attach to a terminal vertex. In this case,  $\Delta N_1 = 1 - 1 = 0$  and  $\Delta N_2 = n$ . This process reduces the ratio  $N_1/N_2$ . If the former process is selectively performed, the ratio  $N_1/N_2$  takes the maximum value  $1/(\bar{n} - 1)$  on average. (b) Limitation on the range of the mean branch length  $\bar{n}$  for a fixed value of  $\gamma$ . The solid line denotes the maximum mean branch length (7) that can produce an intended value of  $\gamma$ .

It should be noted here that for a given mean length of branches  $\bar{n}$ , which is determined by (4), there is a limitation on the range of possible values of  $\gamma$ . As shown in Fig. 1(a),  $N_1/N_2$  never exceeds  $1/(\bar{n} - 1)$  once a certain value of  $\bar{n}$  is given. According to (5),  $\gamma$  changes from  $\infty$  to  $\gamma_{\min}$ , corresponding to a change in  $N_1/N_2$  from 0 to the maximum value  $1/(\bar{n} - 1)$ , where

$$\gamma_{\min} = \frac{-1 + \sqrt{32\bar{n} - 7}}{2}. \quad (6)$$

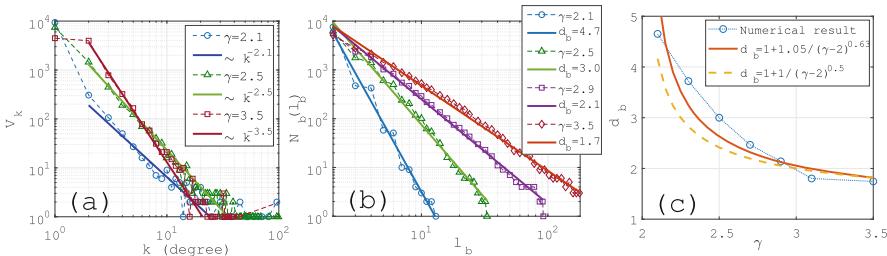
In other words, for a fixed value of  $\bar{n}$ , the possible values of  $\gamma$  is limited to the range  $\gamma_{\min} < \gamma < \infty$ . Conversely, for a given value of  $\gamma$ , the possible value of  $\bar{n}$  is limited to the range  $1 < \bar{n} < \bar{n}_{\max}$  (see Fig. 1(b)). The value of  $\bar{n}_{\max}$  is given by solving (6),

$$\bar{n}_{\max} = \frac{\gamma^2 + \gamma + 2}{8}. \quad (7)$$

In the following calculations, we used the value of  $\bar{n}$  given by  $\bar{n} = (1+r\bar{n}_{\max})/(r+1)$  for each value of  $\gamma$ . We have mainly used  $r = 2$ , but the value of  $r$  was occasionally selected to obtain stable results for the fractal analysis ( $r = 4$  and  $r = 1$  were used for  $\gamma = 3.5$  and  $\gamma = 2.1$ , respectively).

First, we show that the above algorithm certainly generates scale-free fractal tree graphs. Figure 2(a) shows the typical numerical results for the degree distribution. They confirm that the algorithm produces the scale-free properties  $V_k \sim k^{-\gamma}$  with the intended values of the power-law exponent  $\gamma$ . Figure 2(b) shows the results for the fractal analysis using the box-counting method (2). They show that the value of the fractal box dimension  $d_b$  varies with the power-law exponent  $\gamma$ . The numerical calculations using various  $\gamma$  showed that the relation between  $\gamma$  and  $d_b$  is roughly approximated by a functional form,  $d_b \simeq 1 + \frac{1}{(\gamma-2)^{0.63}}$  (Fig. 2(c)). This functional form is only a suggestion based on the assumption that  $d_b \rightarrow \infty$  ( $\gamma \rightarrow 2+$ ) and  $d_b \rightarrow 1$  ( $\gamma \rightarrow \infty$ ). Furthermore, the result contains uncertainty due to statistical insufficiencies and difficulty in determining the exact value of  $d_b$  by a simple fitting to the data. However, this result is similar to that obtained by the original version of the fractal tree algorithm [9], which suggested that  $d_b \simeq 1 + \frac{1}{(\gamma-2)^{1/2}}$ .

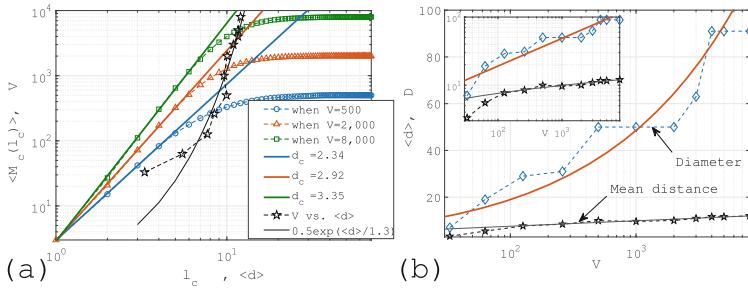
The novel feature provided by introducing the length distribution of branches (4) is a clear realization of fractal properties regardless of the value of  $d_b$ . In contrast to older algorithms [8,9], the present version is able to produce a clear fractal relation even when  $\gamma$  approaches or exceeds a value of 3 (see Fig. 2(b)). This improvement in the realization of fractal properties enables us to examine the relationship between  $d_b$  and  $d_c$  in a wider range of values of  $d_b$ .



**Fig. 2.** Typical numerical results for the degree distribution and the box-counting method. (a) Degree distributions for different values of  $\gamma$  ( $\gamma = 2.1, 2.5$ , and  $3.5$ ). The solid lines indicate the behaviors  $\sim k^{-\gamma}$  expected from the algorithm. (b) Results for the box-counting method applied to cases  $\gamma = 2.1, 2.5, 2.9$ , and  $3.5$ . The fitted lines indicate slopes obtained by fitting the form  $N_b(l_b) \sim l_b^{-d_b}$  to the data. The calculations were performed on the graphs for  $V = 8,000$ . (c) Numerical results for  $\gamma$  versus  $d_b$  (circles). The fitted line  $d_b = 1 + \frac{1.05}{(\gamma-2)^{0.63}}$  and  $d_b = 1 + \frac{1}{(\gamma-2)^{0.5}}$  suggested by ref. [9] are superimposed using solid and dashed lines, respectively.

### 3 Relation Between the Box and Cluster Dimensions

The algorithm described in the previous section produces a small-world fractal graph in the context of the cluster-growing method. The small-world property  $V \sim e^{\langle d \rangle / \beta}$  may seem to contradict the fractal property  $\langle M_c(l_c) \rangle \sim l_c^{d_c}$ . However, this apparent contradiction can be resolved by introducing a size-dependent fractal cluster dimension [4–6] as follows.



**Fig. 3.** Result of the cluster-growing method and the small-world property in the context of the mean distance. (a) The result of the cluster-growing method when  $\gamma = 2.9$ . In (a),  $\langle M_c(l_c) \rangle$  is plotted for different  $V$  ( $V = 500$  (circles), 2,000 (triangles), and 8,000 (squares)). The  $V$ -dependence of  $\langle d \rangle$  (stars) and the fitted line  $V = 0.5e^{\langle d \rangle / 1.3}$  (solid line) are superimposed. (b) A typical behaviors of the mean distance  $\langle d \rangle$  (stars) and diameter  $D$  (diamonds) of the small-world scale-free fractal tree when  $\gamma = 2.9$ . The fitted lines,  $\langle d \rangle \simeq 2.93 + 1.010 \log V$  and  $D \sim V^{0.41}$  are superimposed (solid lines). The inset is the log-log plot.

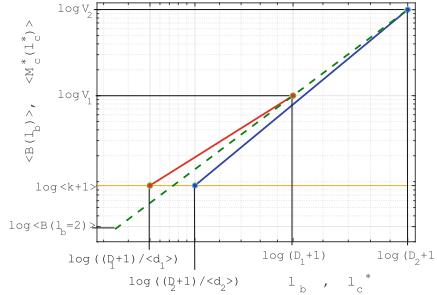
Figure 3(a) shows a typical result of the cluster-growing method when applied to the model network using  $\gamma = 2.9$ . Comparing the results for  $\langle M_c(l_c) \rangle$  at each  $V$  with the behavior of  $V$  versus the mean distance  $\langle d \rangle$ , it is found that the power-law relation observed in  $\langle M_c(l_c) \rangle$  is approximately limited to the range  $l_c \leq \langle d \rangle$ . If we use this observation, the cluster dimension is calculated as

$$\begin{aligned} d_c &= (\log V - \log \langle k+1 \rangle) / (\log \langle d \rangle - \log 1) \\ &= \log (V / \langle k+1 \rangle) / \log \langle d \rangle, \end{aligned} \quad (8)$$

because, in the log-log plane ( $\log l_c$  versus  $\langle M_c(l_c) \rangle$ ), the increment in the horizontal direction is  $\log \langle d \rangle - \log 1$ , and the increment in the vertical direction is  $\log V - \log \langle k+1 \rangle$ .

Equation (8) shows that if the mean distance exhibits a non-small-world behavior  $\langle d \rangle \sim V^{1/d_d}$ , the value of  $d_c$  approaches a constant value  $d_d$  as  $V$  increases. However, if the mean distance exhibits the small-world behavior  $\langle d \rangle \sim \beta \log V$ , (8) indicates that  $d_c$  must increase with respect to  $V$ . The plots of  $V$  versus  $\langle d \rangle$  in Figs. 3(a) and (b) indicate the latter situation. Furthermore, the coordinate  $(\langle d \rangle, \langle k+1 \rangle \langle d \rangle^{d_c})$  is close to the coordinate  $(\langle d \rangle, 0.5e^{\langle d \rangle / 1.3})$  that

represents the trajectory of the small-world behavior (see Fig. 3(a)). This result certainly shows that  $d_c$  increases corresponding to the small-world behavior of  $V \sim e^{\langle d \rangle / \beta}$ . Note that, in contrast to this result, the diameter  $D$  of the tree graph shown in Fig. 3(b) indicates a non-small-world behavior  $D \sim V^{1/d_b}$ .



**Fig. 4.** Illustration of the mean box  $\langle B(l_b) \rangle$  and cluster  $\langle M_c^*(l_c^*) \rangle$  sizes using the transformation (9). The number of vertices  $V$  and the diameter  $D$  are assumed to increase from  $V_1$  and  $D_1$  to  $V_2$  and  $D_2$ , respectively. The slope of  $\log \langle B(l_b) \rangle$  with respect to  $\log l_b$  (dashed line),  $d_b$ , is given by  $(\log V_2 - \log V_1)/(\log(D_2 + 1) - \log(D_1 + 1))$ , because this value is constant when the ideal condition  $D + 1 \sim V^{1/d_b}$  holds. If the fractal network has the small-world property where  $D$  and  $\langle d \rangle$  increase differently with respect to  $V$ , the starting point of  $\log \langle M_c^*(l_c^*) \rangle$ ,  $(\log [\min(l_c^*)], \log \langle M_c^*(\min(l_c^*)) \rangle) = (\log [(D + 1)/\langle d \rangle], \log \langle k + 1 \rangle)$  moves to the right as  $V$  increases. Consequently,  $d_c$ , the slope of  $\log \langle M_c^*(l_c^*) \rangle$  (solid lines), which is determined by the ratio  $\log(V/\langle k + 1 \rangle)/\log \langle d \rangle$ , increases with  $V$ .

To compare directly the mean cluster size  $\langle M_c(l_c) \rangle$  with the mean box size  $\langle B(l_b) \rangle$ , it is better to use a scale transformation because  $l_c$  and  $l_b$  are different metrics. In the box-counting method, the fractal relation holds in the range  $2 \leq l_b \leq D + 1$ , whereas in the cluster-growing method, it is limited to at most the range  $1 \leq l_c \leq \langle d \rangle$ . Therefore, it is convenient to rescale the distance  $l_c$  as

$$l_c^* = l_c(D + 1)/\langle d \rangle. \quad (9)$$

Using (9), the fractal relation (1) is rewritten as

$$\log \langle M_c^*(l_c^*) \rangle = d_c \log l_c^* + \log \langle k + 1 \rangle + d_c \log [(\langle d \rangle / (D + 1))], \quad (10)$$

where the possible values of  $l_c^*$  are limited to the range  $(D + 1)/\langle d \rangle \leq l_c^* < D + 1$ .

Figure 4 shows a diagram of how the relation between  $\log \langle M_c^* \rangle$  and  $\log l_c^*$  (10) changes with respect to  $V$ . As shown in the figure, the slope of  $\log \langle M_c^* \rangle$  in the log-log plot,  $d_c$ , is determined by the ratio of  $\log V - \log \langle k + 1 \rangle$  to  $\log \langle d \rangle$ . This was previously shown by (8). The important point to be observed in Fig. 4 is that if  $D + 1$  and  $\langle d \rangle$  exhibit a similar non-small-world behavior  $\sim V^{1/d_b}$ , the coordinate  $(\log [(D + 1)/\langle d \rangle], \log \langle k + 1 \rangle)$  becomes a fixed point, whereas if  $\langle d \rangle$  exhibits the small-world behavior  $\langle d \rangle \sim \log V$ , it moves to the right as  $V$

increases. In contrast, the slope of  $\langle B \rangle$  in the log-log plot  $d_b$  does not depend on  $V$  because  $d_b$  is determined by  $(\log V_2 - \log V_1)/(\log(D_2 + 1) - \log(D_1 + 1))$ . This value is constant if  $D + 1$  exhibits non-small-world behavior  $\sim V^{1/d_b}$ . As shown above, the difference between  $d_b$  and  $d_c$  originates from the difference between the behaviors of  $D + 1$  and  $\langle d \rangle$ .

If the ideal fractal conditions  $\langle k + 1 \rangle \langle d \rangle^{d_c} \simeq V$  and  $N_b(l_b = D + 1) = 1$  hold,  $d_c$  can explicitly be related to  $d_b$ . Noting that the fractal box dimension  $d_b$  is a slope in the log-log plot plane,

$$\begin{aligned} -d_b &= \frac{\log 1 - \log N_b(l_b = 2)}{\log(\tilde{D} + 1) - \log 2} \\ &= \frac{\log \langle B(l_b = 2) \rangle - \log V}{\log(\tilde{D} + 1) - \log 2}, \end{aligned} \quad (11)$$

where (3) was used, and the diameter  $D$  was replaced by a virtual diameter  $\tilde{D}$  that makes the fitted function  $\log N_b \sim \log l_b^{-d_b}$  zero by substitution of  $l_b = \tilde{D} + 1$ . The mean box size  $\langle B(l_c = 2) \rangle$  was also estimated by  $N_b(l_b = 2) \sim 2^{-d_b}$ , whose coefficient was obtained by fitting the functional form to the numerical data. Solving equations (8) and (11) with respect to  $d_c$  by eliminating  $\log V$  from them, we obtain

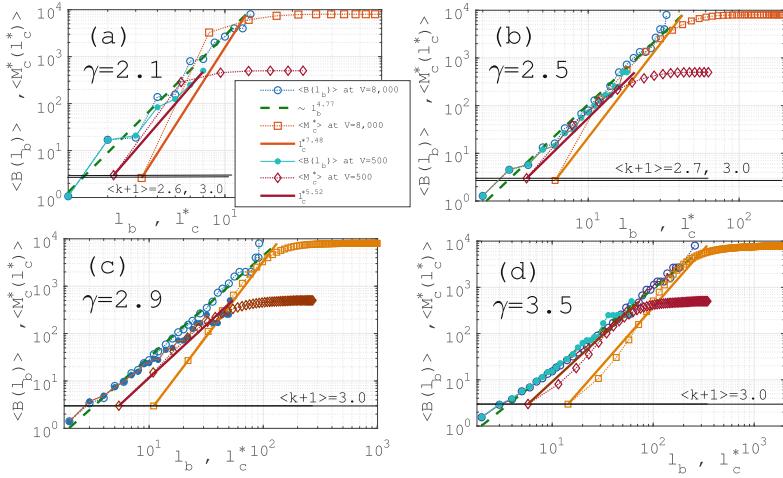
$$d_c = \frac{d_b \log[(\tilde{D} + 1)/2] + \log(\langle B(l_b = 2) \rangle / \langle k + 1 \rangle)}{\log \langle d \rangle}. \quad (12)$$

Note that the actual value of  $d_c$  may vary from (12) depending on the incompleteness of the ideal fractal conditions mentioned above.

Figure 5 shows the numerical results comparing  $\langle B(l_b) \rangle$  to  $\langle M_c^*(l_c^*) \rangle$  using the scale transformation (9) for the fractal tree graphs. The solid lines in the figure represent the prediction for  $\langle M_c^*(l_c^*) \rangle$  using (10), into which the value of  $d_c$  predicted by (12) has been substituted. These predictions exhibit the small-world property of the graphs; they show that the starting point of (10),  $(\log[(D + 1)/\langle d \rangle], \log \langle k + 1 \rangle)$  moves to the right as  $V$  increases. In contrast, the behavior of the mean box-size  $\langle B(l_b) \rangle$  is invariant as  $V$  increases.

Figures 5(a) and (b) show the results when  $\gamma$  is small, that is, when  $d_b$  is larger than 2. In these cases, the range of  $l_c^*$  that supports the power-law form  $(D + 1)/\langle d \rangle \leq l_c^* < D + 1$  is so narrow that the power-law relation is not clearly observed. This is because the mean distance  $\langle d \rangle$  is small. Although the prediction using (10) and (12) approximately describes the actual behaviors of  $\langle M_c^*(l_c^*) \rangle$ , the behaviors of  $\log \langle M_c^*(l_c^*) \rangle$  at  $l_c^* \simeq (D + 1)/\langle d \rangle$  indicate slopes larger than those predicted by (12). In other words, the strict behavior of  $\log \langle M_c^*(l_c^*) \rangle$  with respect to  $\log l_c^*$  is not described by a straight line but by a convex upward function with respect to  $\log l_c^*$  in the log-log plot plane.

As  $\gamma$  increases further, the power relationship  $\langle M_c^*(l_c^*) \rangle$  begins to clearly emerge (Figs. 5(c) and (d)). These results show that  $\langle d \rangle$  is large enough to support a sufficient width of  $l_c^*$ , where the power-law relation holds. As a result, the prediction using (10) and (12) explains the actual behavior of  $\langle M_c^*(l_c^*) \rangle$  with

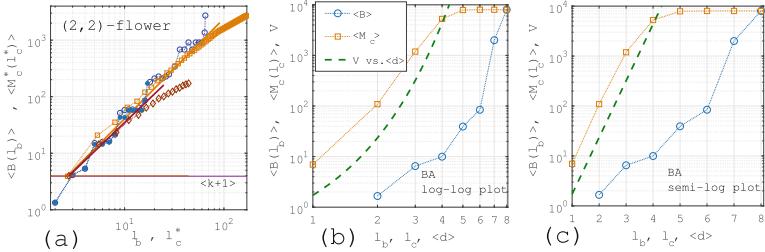


**Fig. 5.** Comparison of mean box sizes  $\langle B(l_b) \rangle$  (circles) and mean cluster sizes  $\langle M_c^*(l_c^*) \rangle$  at  $V = 500$  (diamonds) and  $V = 8,000$  (squares) for different values of  $\gamma$ : (a)  $\gamma = 2.1$ , (b)  $\gamma = 2.5$ , (c)  $\gamma = 2.9$  and (d)  $\gamma = 3.5$ . The behaviors of  $\langle B(l_b) \rangle$  are approximated when  $V = 8,000$  (dashed lines) by (a)  $d_b = 4.77$ , (b)  $d_b = 3.00$ , (c)  $d_b = 2.14$ , and (d)  $d_b = 1.74$ . The slopes predicted by (12) (solid lines) are (a)  $d_c = 5.52$  ( $d_b = 4.14$ ,  $\langle B(2) \rangle = 1.55$ ,  $\tilde{D} = 7.06$  and  $\langle d \rangle = 2.53$ ) and  $d_c = 7.48$  ( $\langle B(2) \rangle = 1.30$ ,  $\tilde{D} = 11.44$  and  $\langle d \rangle = 2.93$ ), (b)  $d_c = 3.10$  ( $d_b = 2.64$ ,  $\langle B(2) \rangle = 1.09$ ,  $\tilde{D} = 19.3$  and  $\langle d \rangle = 5.21$ ) and  $d_c = 4.11$  ( $\langle B(2) \rangle = 0.85$ ,  $\tilde{D} = 41.2$  and  $\langle d \rangle = 7.00$ ), (c)  $d_c = 2.23$  ( $d_b = 1.78$ ,  $\langle B(2) \rangle = 1.43$ ,  $\tilde{D} = 53.1$  and  $\langle d \rangle = 9.96$ ) and  $d_c = 3.18$  ( $\langle B(2) \rangle = 1.02$ ,  $\tilde{D} = 130.9$  and  $\langle d \rangle = 12.0$ ), and (d)  $d_c = 2.02$  ( $d_b = 1.67$ ,  $\langle B(2) \rangle = 1.25$ ,  $\tilde{D} = 71.9$  and  $\langle d \rangle = 12.7$ ) and  $d_c = 2.49$  ( $\langle B(2) \rangle = 1.02$ ,  $\tilde{D} = 341.3$  and  $\langle d \rangle = 23.9$ ) for  $V = 500$  and  $V = 8,000$ , respectively.

considerable accuracy. However, as  $\gamma$  increases further beyond 3 and the value of  $d_b$  decreases to less than 2, the actual behavior of  $\log \langle M_c^*(l_c^*) \rangle$  begins to show a convex downward functional form (Fig. 5(d)). This result shows that, due to the large gap between the behaviors of  $\langle d \rangle$  and  $\tilde{D}$ , it becomes increasingly difficult to maintain the fractal property when  $d_b$  is much smaller than 2.

To highlight the characteristics of the small-world scale-free fractal trees, two other network models were examined using the same method. Figure 6(a) shows the result for the  $(2, 2)$ -flower, a typical non-small-world fractal graph in which  $\langle d \rangle \sim D \sim V^{1/d_b}$  holds exactly [10]. As shown in Fig. 6(a), in non-small-world fractal networks, the behaviors of  $\langle M_c^*(l_c^*) \rangle$  and  $\langle B(l_b) \rangle$  are consistent. In contrast to the small-world scale-free fractal trees, the starting point of (10),  $(\log [(D+1)/\langle d \rangle], \log \langle k+1 \rangle)$ , is fixed with respect to an increase in  $V$ . This is because the relation  $\langle d \rangle \sim D \sim V^{1/d_b}$  holds. As a result, the difference between  $d_b$  and  $d_c$  will disappear as  $V$  becomes large. Figures 6(b) and (c) show results for the BA model [11], a typical small-world non-fractal network in which  $\langle d \rangle \sim D \sim \log V$  holds. In this case, the scale transformation (9) is invalid because there are no power-law relationships in either  $\langle B(l_b) \rangle$  or  $\langle M(l_c) \rangle$  (Fig. 6(b)). The behavior

of  $\langle M(l_c) \rangle$  is described by the small-world behavior of  $\langle d \rangle$ , whereas  $\langle B(l_b) \rangle$  is not (Fig. 6(c)). This is because  $\langle d \rangle$  and  $D$  are similarly proportional to  $\log V$  but with different proportionality factors.

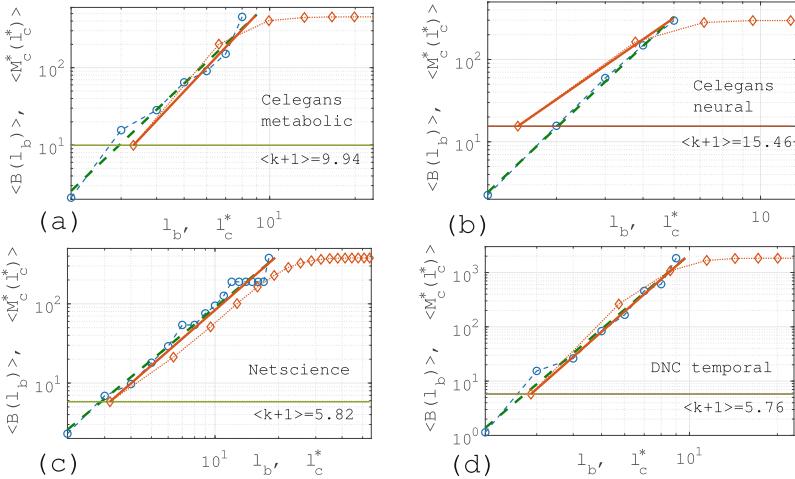


**Fig. 6.** Comparison between the mean box sizes  $\langle B(l_b) \rangle$  (circles) and mean cluster sizes  $\langle M_c^*(l_c^*) \rangle$  (diamonds and squares) and  $\langle M_c(l_c) \rangle$  (squares) for (a) the (2, 2)-flower with  $d_b = 2$  and (b and c) a BA model ( $V = 8,000$ ) ((c) is a semi-log plot of (b)). In (a), the slopes (solid lines) predicted by (12) are  $d_c = 1.6844$  at  $V = 172$  and  $d_c = 1.7729$  at  $V = 2732$ . In (b) and (c), dashed lines indicate the small-world behaviors of  $V \sim e^{\langle d \rangle / \beta}$ .

## 4 Real Networks

The analysis in the previous section was also applied to four real networks: the metabolic network of *Caenorhabditis elegans* (C-elegans) ( $V = 453$ ) [12], the neural network of C-elegans ( $V = 297$ ) [13], a network of co-authorships in the area of network science (the largest component having  $V = 379$  of a network with  $V = 1461$ ) [14], and the network of e-mails in the 2016 DNC e-mail leak (the largest component having  $V = 1833$  of a network with  $V = 1891$ ). This empirical data is available in the Koblenz Network Collection (KONECT) Project [15].

Figure 7 shows the results for the mean box size  $\langle B(l_b) \rangle$  and mean cluster size  $\langle M_c^*(l_c^*) \rangle$  for these real networks. All results show that the prediction (12) succeeded in explaining the behavior of each  $\langle M_c^*(l_c^*) \rangle$ . The results shown in Fig. 7(b) exhibit the characteristics of small-world fractal networks, because the values of  $d_b$  and  $d_c$  are significantly different. However, the other results apparently show the characteristics of non-small-world fractal networks because the values of  $d_b$  and  $d_c$  are consistent with considerable accuracy. However, there is no way to determine with certainty whether or not the coordinate  $(\log [(D + 1)/\langle d \rangle], \log \langle k + 1 \rangle)$  is fixed because the data corresponds to a particular  $V$ . In fact, even for the results of the small-world fractal networks shown in Fig. 3, the values of  $d_b$  and  $d_c$  are not very different when  $V$  is small. In particular,  $\log \langle M_c^* \rangle$  in Fig. 7(a) shows a convex upward functional form, and the asymptotic slope at  $l_c^* = 6 \sim 10$  ( $l_c = 1 \sim 2$ ) is much larger than  $d_b$ . This situation is similar to that found in Figs. 5(a) and (b). However, considering the consistency between  $d_c$  and  $d_b$ , the results, except for Fig. 7(b), are suggestive of the non-small-world property.



**Fig. 7.** The mean box sizes  $\langle B(l_b) \rangle$  (circles) and mean cluster sizes  $\langle M_c^*(l_c^*) \rangle$  (diamonds) were calculated for (a) the metabolic network of C-elegans ( $V = 453$ ) [12], (b) the neural network of C-elegans ( $V = 297$ ) [13], (c) a network of co-authorships in the area of network science ( $V = 379$ ) [14], and (d) the network of e-mails in the 2016 DNC e-mail leak ( $V = 1833$ ). The behaviors of  $\langle B(l_b) \rangle$  are approximated (dashed lines) by (a)  $d_b = 3.49$ , (b)  $d_b = 4.47$ , (c)  $d_b = 2.21$  and (d)  $d_b = 4.57$ . The slopes predicted by (12) (solid lines) are (a)  $d_c = 3.90$  ( $\langle B(2) \rangle = 2.56$ ,  $\tilde{D} = 7.8$  and  $\langle d \rangle = 2.66$ ), (b)  $d_c = 3.29$  ( $\langle B(2) \rangle = 2.42$ ,  $\tilde{D} = 4.87$  and  $\langle d \rangle = 2.46$ ), (c)  $d_c = 2.32$  ( $\langle B(2) \rangle = 2.58$ ,  $\tilde{D} = 18.20$  and  $\langle d \rangle = 6.04$ ), and (d)  $d_c = 4.74$  ( $\langle B(2) \rangle = 1.38$ ,  $\tilde{D} = 8.65$  and  $\langle d \rangle = 3.37$ ).

## 5 Summary

We have improved the prior small-world scale-free fractal tree algorithm by introducing a power-law form length distribution of branches, and we used it to clarify the relation between the box dimension  $d_b$  and the cluster dimension  $d_c$  of fractal networks. The improved algorithm allows us to investigate small-world fractal networks in a wide range of  $d_b$ . We derived a general relation that relates  $d_b$  and  $d_c$  using the mean distance, diameter, mean degree, and the mean box size at  $l_b = 2$ . It was previously known that the difference between the behaviors of  $\langle d \rangle$  and  $D$  causes the difference between  $d_b$  and  $d_c$ . However, the explicit relation between  $d_b$  and  $d_c$  was first derived in this study. An analysis using a scale transformation enables direct comparison between the mean box size  $\langle B(l_b) \rangle$  and the mean cluster size  $\langle M_c(l_c) \rangle$ . This analysis graphically shows how the difference between  $d_b$  and  $d_c$  increases as the number of vertices  $V$  grows. The theoretical relation between  $d_b$  and  $d_c$  was confirmed by an application to the small-world scale-free fractal trees. We applied the same analysis to four real networks. All results showed that the theoretical prediction for  $d_c$  explains the behavior of  $\langle M_c(l_c) \rangle$  with fairly good accuracy. Although it is difficult to determine whether or not the network has the small-world property only from the

relation between  $d_b$  and  $d_c$ , the method is applicable to any fractal networks, including non-small-world fractal networks.

## References

1. Song, C., Havlin, S., Makse, H.A.: Self-similarity of complex networks. *Nature* **433**, 392–395 (2005). <https://doi.org/10.1038/nature03248>
2. Zakar-Polyák, E., Nagy, M., Molontay, R.: Towards a better understanding of the characteristics of fractal networks. *Appl. Netw. Sci.* **8**, 17 (2023). <https://doi.org/10.1007/s41109-023-00537-8>
3. Csányi, G., Szendrői, B.: Fractal-small-world dichotomy in real-world networks. *Phys. Rev. E* **70**, 016122 (2004). <https://doi.org/10.1103/PhysRevE.70.016122>
4. Ikeda, N.: Growth model for fractal scale-free networks generated by a random walk. *Phys. A* **521**, 424–434 (2019). <https://doi.org/10.1016/j.physa.2019.01.043>
5. Ikeda, N.: Fractal networks induced by movements of random walkers on a tree graph. *Phys. A* **537**, 122743 (2020). <https://doi.org/10.1016/j.physa.2019.122743>
6. Ikeda, N.: Fractality and the small-world property of generalised  $(u,v)$ -flowers. *Chaos, Solitons Fractals* **137**, 109837 (2020). <https://doi.org/10.1016/j.chaos.2020.109837>
7. Kim, J.S., Goh, K.-I., Salvi, G., Oh, E., Kahng, B., Kim, D.: Fractality in complex networks: critical and supercritical skeletons. *Phys. Rev. E* **75**, 016110 (2007). <https://doi.org/10.1103/PhysRevE.75.016110>
8. Ikeda, N.: Stratified structure of fractal scale-free networks generated by local rules. *Phys. A* **583**, 126299 (2021). <https://doi.org/10.1016/j.physa.2021.126299>
9. Ikeda, N.: Properties of model networks generated by modified small-world scale-free fractal tree algorithm. In: Vlachos, D. (eds.) *Mathematical Modeling in Physical Sciences. ICMSQUARE 2023*. Springer Proceedings in Mathematics & Statistics, vol 446, pp. 245–254. Springer, Cham. (2024). [https://doi.org/10.1007/978-3-031-52965-8\\_20](https://doi.org/10.1007/978-3-031-52965-8_20)
10. Rozenfeld, H. D., Havlin, S., ben-Avraham, D.: Fractal and transfractal recursive scale-free nets. *New J. Phys.* **9**, 175 (2007). <https://doi.org/10.1088/1367-2630/9/6/175>
11. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. *Science* **286**, 509–512 (1999). <https://doi.org/10.1126/science.286.5439.509>
12. Jeong, H., Tombor, B., Albert, R., Oltvai, Z.N., Barabási, A.-L.: The large-scale organization of metabolic networks. *Nature* **407**, 651–654 (2000). <https://doi.org/10.1038/35036627>
13. Watts, D.J., Strogatz, S.H.: Collective dynamics of ‘small-world’ networks. *Nature* **393**, 440–442 (1998). <https://doi.org/10.1038/30918>
14. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* **74**, 036104 (2006). <https://doi.org/10.1103/PhysRevE.74.036104>
15. Kunegis, J.: KONECT: the Koblenz network collection, In: *Proceedings of the 22nd International Conference World Wide Web*, pp. 1343–1350 (2013). <https://doi.org/10.1145/2487788.2488173>



# Cell Signaling Characterization for Spatial Transcriptomics (ST) Data Using Network Analysis

Azka Javaid<sup>(✉)</sup> and H. Robert Frost

Department of Biomedical Data Science, Geisel School of Medicine, Dartmouth College Hanover, Hanover, NH 03755, USA  
[azka.javaid@dartmouth.edu](mailto:azka.javaid@dartmouth.edu)

**Abstract.** We describe a network analysis-based cell-cell communication method for Spatial Transcriptomics (ST) data. For each evaluated ligand-receptor interaction, we define a fully connected, directed and weighted network model where nodes represent the individual ST locations with directed edge weights set to the product of the reduced rank reconstructed expression values for the ligand at the source location and cognate receptor at the target location divided by the squared distance between the locations. Using this network, we compute the weighted in-degree centrality to quantify signaling activity of the target ligand-receptor interaction at each location. Our method is validated for three interactions on a real ST dataset against five different cell-cell communication strategies. We report that our method captures the simultaneous expression heterogeneity in both the ligand and the receptor and generates biologically plausible cell communication profiles for the Wnt3-Fzd1, Ephb1-Efnb3 and Ptprc-Cd22 interactions. An important finding of this work is the importance of building network models for ST data using a low dimensional embedding of the gene-level data.

**Keywords:** Network analysis · Spatial Transcriptomics · Centrality

## 1 Introduction

Spatial Transcriptomics (ST) data enables simultaneous quantification of mRNA and spatial location via spatially barcoded mRNA-binding oligonucleotides [1]. In comparison to single-cell RNA-sequencing (scRNA-seq) and bulk RNA-seq technologies that perform transcriptomic profiling on cells from a dissociated tissue, ST performs transcriptomic profiling on the intact tissue so is able to preserve the spatial organization of a tissue, which is critical for identifying tissue domains and characterizing cell signaling.

Analysis of cell signaling for scRNA-seq data has been the focus of significant recent research. Methods and databases such as CellPhoneDB [2] provide a repository of ligands, receptors and their interactions based on public and manual

annotations, accounting for the idea that receptors can form heteromeric molecular complexes. CellChat [3] similarly curates signaling molecule interactions and accounts for structural composition of multimeric ligand-receptor pairs while also accounting for agonists/antagonists and co-stimulatory/inhibitory receptors. Similarly, methods like SoptSC [4] and NicheNet [5] leverage information from downstream intracellular interactions to encode cell signaling.

Cell-cell communication methods has also been developed for ST data using a range of analytical approached including deep learning-based models. SpaTalk [6], for example, scores a ligand-receptor target signaling network using a graph network and knowledge graph model. Similarly, GCNG [7] uses a Graph Convolutional Neural networks for Genes to convolve spatial information with gene expression data. In addition to deep learning-based methods, several network analysis-based methods have been developed to estimate cell-cell communication for ST data. Giotto [8], for example, discusses representation of a spatial neighborhood network where each node represents a cell and an edge connects each pair of neighboring cells. Giotto constructs a spatial grid by connecting cells in physical proximity to each other. This spatial grid is then used to identify spatial genes, perform cell-type spatial interaction enrichment analysis and identify spatial domains. COMMOT [9], in comparison, performs cell-cell communication inference for ST data by solving an optimization problem via optimal transport on a cell-cell communication network through predefined ligand-receptor pairs.

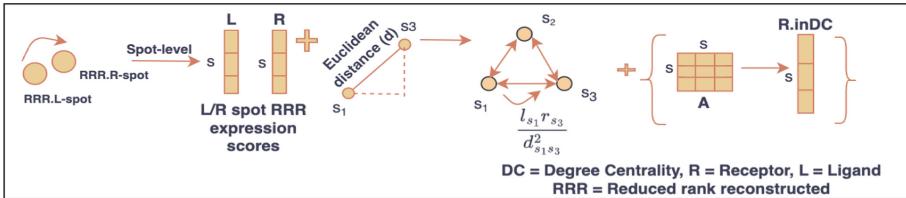
A key limitation of most existing ST-based network analysis methods is that they do not directly incorporate spatial distance into the network construction, which can otherwise account for how the ligand-receptor interaction dynamics transform over the tissue given adjacency and spatial proximity of cells to each other. A second key limitation of existing deep learning-based cell-cell communication methods for ST data is that they are not usually practical or interpretive enough to be implemented by medical practitioners. As compared with supervised, deep learning models, reduced rank reconstruction (RRR)-based strategies offer a more interpretive alternative to addressing cell-cell communication. For example, we previously found that the class of RRR techniques provides superior performance for the task for cell surface receptor abundance estimation relative to other classes of methods [10].

Herein, we propose a novel degree centrality-based method for cell-cell interaction characterization based on product of reconstructed ligand-receptor expression weighted by inverse squared distance. We compare our method against the RRR and the raw counts-based ligand and receptor expression, the CytoSignal-estimated [11] and against the COMMOT-estimated expression values on real ST data. Our solution provides a simpler, and arguably, more interpretable strategy to estimate cell-cell interactions for target ligand-receptor pairs.

## 2 Method Overview

Our method performs RRR of the underlying gene expression ST counts matrix followed by a ligand-receptor network construction and computation of weighted

degree centrality to quantify cell-cell communication. The method schematic is shown in Fig. 1.



**Fig. 1.** Ligand-receptor interaction characterization for spot (s)-level ST data.

## 2.1 Reduced Rank Reconstruction Computation

We first perform reduced rank reconstruction (RRR) on the  $m \times n$  ST matrix that holds gene expression counts for  $m$  cells and  $n$  genes using the *randomizedRRR* function from the *SPECK* [10] package. The *randomizedRRR* function performs both normalization via Seurat's [12–16] *NormalizeData* function and computes the rank followed by RRR using the randomized Singular Value Decomposition with the *rsvd* function from the *rsvd* package [17]. The rank for the RRR is selected using the standard deviation of the non-centered sample principal components, which are subsequently used in a rate of change computation. Each successive standard deviation value is compared to the last to select the rank for which the absolute value of the rate of change between consecutive values is at least 0.01 for at least two value pairs. Overall, the RRR aims to achieve a less sparse representation of otherwise sparse ST data.

## 2.2 Ligand-Receptor Network Construction

We next construct a fully connected, weighted and directed ligand-receptor activity network combining reconstructed expression scores with the physical location of each profiled location in the tissue. This network is stored as an adjacency matrix  $A$  where the nodes represent the individual tissue locations while the edges represent connections between two locations weighted by inverse squared physical distance. Physical distance between each pair of spots (i.e. spots  $s_1$  and  $s_3$ ) is computed using the Euclidean distance ( $d_{s1,s3}^2$ ). The computed normalized and reconstructed ligand and receptor gene expression scores as well as the Euclidean distances are integrated via a product of the ligand and receptor scores for the source and target nodes. Lastly, the product of the ligand and receptor scores are multiplied by inverse distance squared. For example, for a spot  $s_1$  and spot  $s_3$ , the ligand expression score for  $s_1$  is multiplied by the receptor expression score for  $s_3$  (i.e.  $(l_{s1} * r_{s3})/d_{s1,s3}^2$ ). Similarly, ligand expression score for  $s_3$  is multiplied by receptor expression score for  $s_1$  (i.e.  $(l_{s3} * r_{s1})/d_{s1,s3}^2$ ). This

interaction score is computed between each candidate ligand and receptor and vice-versa, resulting in a fully connected network representation.

### 2.3 Weighted Degree Centrality Computation

Following the RRR and the ligand-receptor network construction, we next compute the weighted in-degree centrality for each location. We compute the weighted in-degree centrality by calculating the sums of the weights over the columns. The sum of the weighted edges over the columns allows us to account for the incoming receptor signal as opposed to the outgoing ligand signal. Importantly, this is a local centrality measure as opposed to a global centrality measure like the eigenvector centrality. This preference for the local centrality measure is because the eigenvector centrality is conventionally used to measure the transitive influence of nodes where a high eigenvector score signifies that nodes are themselves connected to highly influential nodes. In the context of real ligand-receptor networks, however, this assumption of transitive influence does not hold, i.e., binding of a receptor by a ligand does not typically result in increased expression of the ligand in that cell.

## 3 Evaluation

We evaluated our method against five ligand-receptor interaction characterization strategies. For two of the five comparative methods (Ligand and Receptor Scores and our Degree Centrality-based methods), we performed RRR on the  $m \times n$  ST gene expression matrix.

1. Ligand and Receptor using Counts: We first compared our method against the raw ligand and receptor spatial counts. For this task, we set the ligand and the receptor expression values to the unnormalized spatial counts.
2. Ligand and Receptor using RRR Scores: We next compared our method against the RRR ligand and receptor expression values. For this task, we performed RRR using the *randomizedRRR* function from the *SPECK* package with default arguments of 100 for the *rank.range.end*, 0.01 for the *min.consec.diff* and 2 for the *rep.consec.diff* parameters. Please see the [2.1](#) section in the Methods for further details.
3. COMMOT using Counts: We next compared our method against the ligand-receptor cell communication scores generated using the *tl.spatial.communication* function from the *commot* package. For this task, we performed basic preprocessing as suggested by the authors (i.e., normalization) on the ST data and specified the ligand-receptor pairs of interest. Then we specified the default spatial distance constraint of 200 in the *tl.spatial.communication* function and extracted the corresponding receiver scores (i.e., the *commot-user.database-sum-receiver* attribute).
4. CytoSignal using Counts: We next compared our method against the ligand-receptor imputed cell communication values generated using the *inferIntrScore* function from the *CytoSignal* package computed on the raw, unnormalized Spatial counts.

5. CytoSignal using SCT normalized data: We next compared our method against the *inferIntrScore* function-generated data from the *CytoSignal* package computed on the SCT normalized data as calculated using the *SCTtransform* function from the *Seurat* package.

### 3.1 Real ST Data

We apply our method consisting of the RRR ligand-receptor network construction and weighted in-degree centrality computation to three interactions in the mouse brain. For this task, we analyze the posterior slice of the 10x Visium mouse brain consisting of 48,721 genes and 3,353 cells. We next examine the Wnt Family Member 3 (Wnt3)-Frizzled Class Receptor 1 (Fzd1) interaction since Wnt3 is implicated in many biological processes including regulation of neurogenesis and repression of neural tumors [18, 19]. Second, we examine the EPH Receptor B1 (Ephb1)-Ephrin B3 (Efnb3) interaction since Ephb1 is required for the regulation of proper axon guidance and processes such as cell migration and synapse formation [20]. Third, we examine the Protein Tyrosine Phosphatase Receptor Type C (Ptprc)-Cd22 interaction since CD45 deficiency has been suggested to accelerate cerebral amyloidosis in Alzheimer’s diseased mice [21].

## 4 Results and Discussion

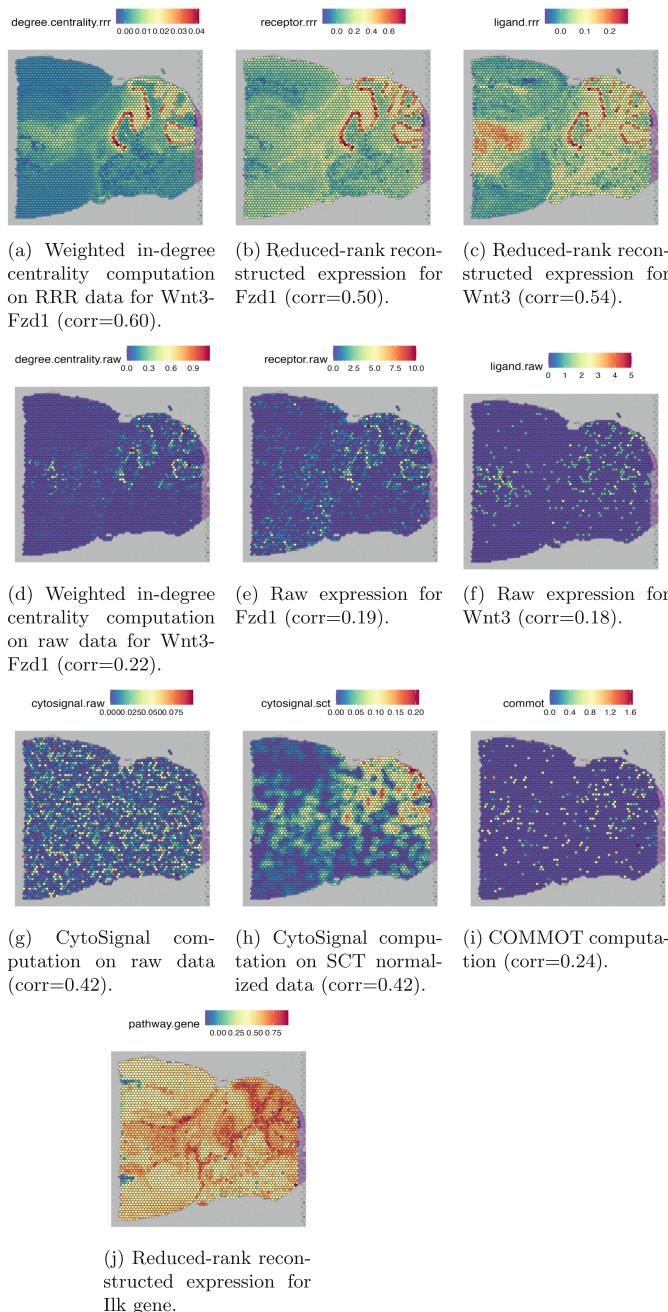
### 4.1 Application on a Real ST Dataset

We first plot the estimated Wnt3-Fzd1 signaling activity on top of tissue histology using the *SpatialFeaturePlot* function from the *Seurat* package. As compared to the raw counts-based ligand (Fig. 2f) and receptor (Fig. 2e) expression scores, RRR ligand (Fig. 2c) and receptor (Fig. 2b) expression values, output from the CytoSignal method based on both the raw (Fig. 2g) and the SCT normalized data (Fig. 2h) and output from the COMMOT method (Fig. 2i), the results from our method are more spatially specific to the cerebellar cortex and the thalamus regions of the mouse brain. As indicated by Fig. 2a, the signaling activity scores returned by our degree centrality-based technique more closely align with the folds of the cerebellar cortex region and the spatial region corresponding to the thalamus. This close alignment to the cerebellar cortex and the thalamus regions for the signaling activity estimated by our method contrasts with the RRR ligand and receptor expression values and with the CytoSignal results. The CytoSignal results in particular are more heterogeneous than the estimates generated using our method and are not highly specific and more generically spread throughout the mouse brain tissue. In addition to regional specificity, we quantified correspondence of the signaling estimates generated by our degree centrality-based technique with the expression of the integrin-linked kinase (Ilk) gene, which is known to regulate the transcription of Fzd1 [22], using the Spearman rank correlation coefficient. As indicated, we observed a correlation of 0.6 between the signaling estimates generated using our method and the RRR expression corresponding to the Ilk gene (Fig. 2j), which is higher

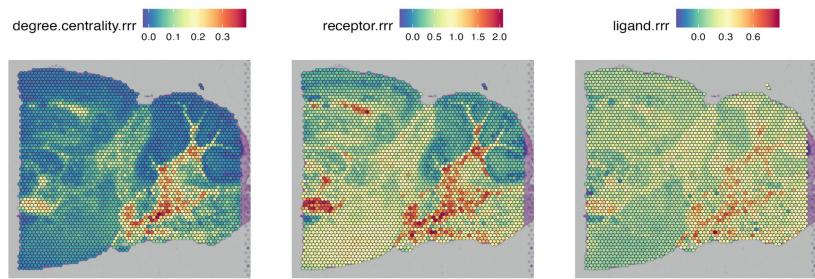
than the correlation between each of the comparative methods and the RRR Ilk gene expression. In addition to regional specificity and quantified correspondence with Ilk, the cell communication profiles generated using our method are biologically plausible as Wnt3 has conventionally been found to be expressed during the mouse cerebellum postnatal development [19] and, additionally, in the dorsal and the ventral thalamus during development [23].

We next visualize the results for the estimated signaling activity of the Ephb1-Efnb3 interaction on top of tissue histology. For this task, we only compare the raw counts-based ligand and receptor expression, the RRR-based ligand and receptor expression and COMMOT expression against the computed weighted in-degree centrality since the CytoSignal method did not generate a score for this interaction. As shown by results in Fig. 3a, the RRR-based degree centrality results show a clear specificity of the interaction to the hindbrain region (i.e., regions of the cerebellum, the medulla oblongata and the pons). This contrasts with the RRR-ligand (Fig. 3c), the RRR-receptor (Fig. 3b), the raw-ligand (Fig. 3f), the raw-receptor (Fig. 3e) and the output from the COMMOT method (Fig. 3g) which instead show more expression heterogeneity throughout the mouse brain tissue regions. We next quantified the correspondence between the signaling estimates generated by our degree centrality-based method and comparative methods against the RRR protein tyrosine kinase 2 (Ptk2) or the Focal adhesion kinase (Fak) gene expression, which is known to decrease in the presence of binding of Ephb1 to Efnb3 in striatal neurons [24]. As shown, we observed a negative correlation of 0.16 between our signaling estimates and the RRR expression corresponding to the Ptk2 gene (Fig. 3h), which is lower than the rank correlation between the signaling profiles generated by comparative methods and the RRR expression corresponding to the Ptk2 gene. In addition to regional specificity and quantified correspondence with Ptk2, our result has biological plausibility as Ephb1 is known to be involved in hindbrain tangential cell migration [25].

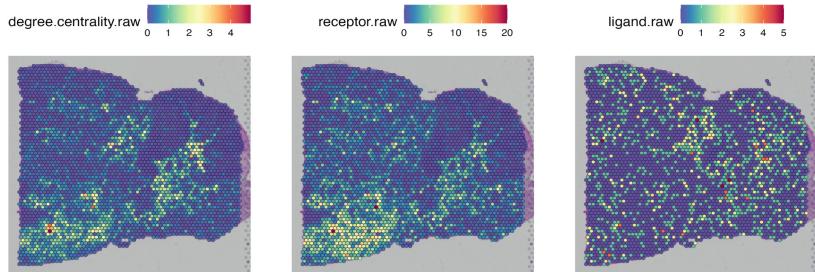
Lastly, we visualize the results for the estimated signaling activity corresponding to the Ptprc-Cd22 interaction. Similar to the results for the Ephb1-Efnb3 interaction, we do not quantify results for the CytoSignal method since it did not generate scores for this interaction. To quantify the regional specificity of the signaling profiles generated by our degree centrality-based method and the comparative methods, we compute the rank correlation between each technique and the RRR expression corresponding to the Src Proto-Oncogene, Non-Receptor Tyrosine Kinase (Src) gene, which is known to represent the best-defined substrate of CD45 [26]. As shown, we observed a positive correlation of 0.38 between our degree centrality-based cell signaling estimates (Fig. 4a) and the RRR expression corresponding to the Src gene (Fig. 4h), which is higher than the correlations between comparative methods and the RRR expression corresponding to the Src gene.



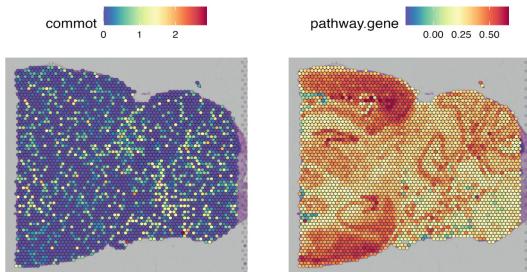
**Fig. 2.** Comparison of weighted degree centrality, ligand and receptor raw and RRR expression, CytoSignal application on both raw and SCT-normalized data and COM-MOT application against the Ilk gene expression on the posterior mouse brain slice for the Wnt3-Fzd1 interaction.



(a) Weighted in-degree centrality computation on RRR data for Ephb1-Efnb3 (corr=-0.16).  
(b) Reduced-rank reconstructed expression for Efnb3 (corr=0.037).  
(c) Reduced-rank reconstructed expression for Ephb1 (corr=-0.15).

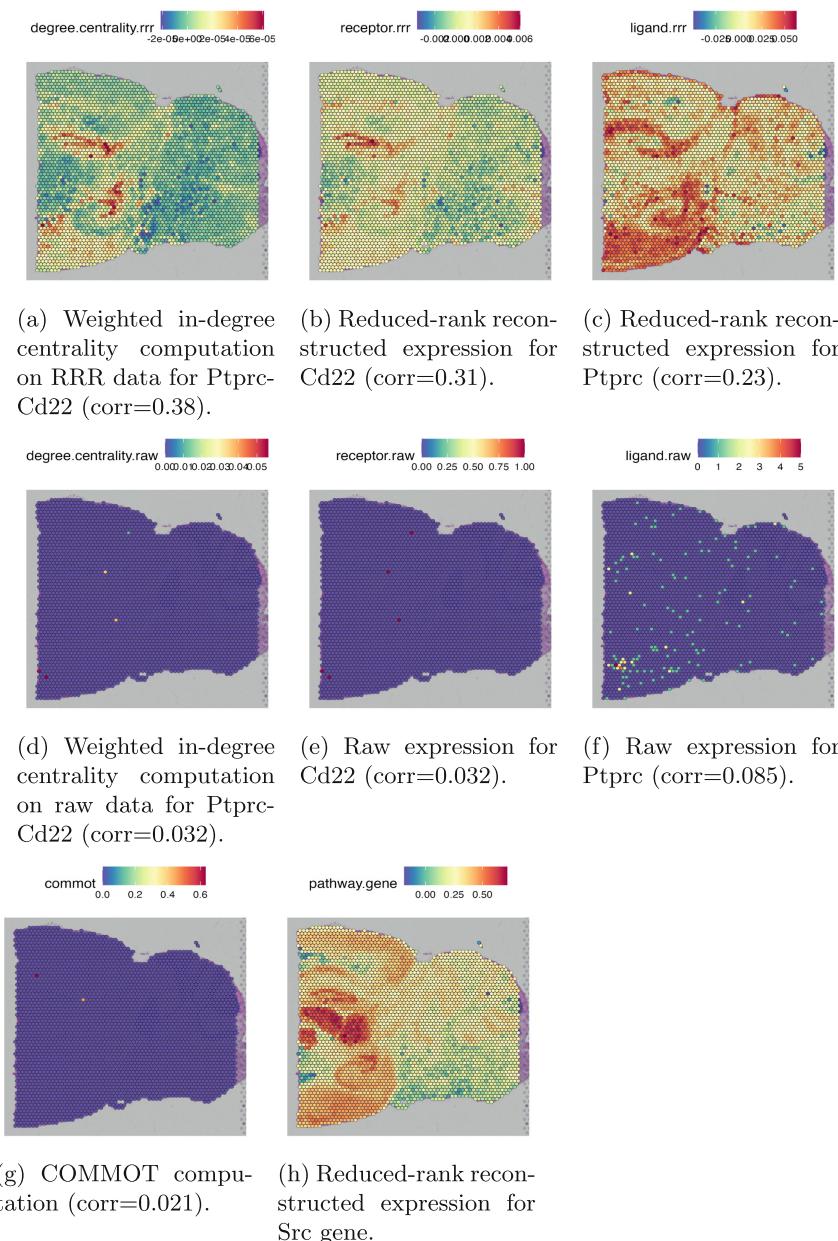


(d) Weighted in-degree centrality computation on raw data for Ephb1-Efnb3 (corr=0.26).  
(e) Raw expression for Efnb3 (corr=0.26).  
(f) Raw expression for Ephb1 (corr=0.10).



(g) COMMOT computation (corr=0.11).  
(h) Reduced-rank reconstructed expression for Ptk2 gene.

**Fig. 3.** Comparison of weighted degree centrality, ligand and receptor raw and RRR expression and COMMOT application against the Ptk2 gene expression on the posterior mouse brain slice for the Ephb1-Efnb3 interaction.



**Fig. 4.** Comparison of weighted degree centrality, ligand and receptor raw and RRR expression and COMMOT application against the Src gene expression on the posterior mouse brain slice for the Ptprc-Cd22 interaction.

## 5 Future Directions

In this paper, we propose a novel network analysis-based cell-cell communication analysis method for ST data. We validate our network analysis model on real ST data for the Wnt3-Fzd1, Ephb1-Efnb3 and Ptprc-Cd22 interactions against five variations of existing cell-cell communication strategies. An important contribution of our method is that we directly incorporate the ST distance between spots in our network to perform cell signaling characterization. Second, we conclude that RRR of ST data is important in generating relatively accurate estimates of cell signaling, as quantified using correlation against expression corresponding to a gene on the interaction pathway.

Our method has several limitations. First, while we validate our method against a pathway gene expression for three interactions, we do not perform a systematic validation for several interactions incorporating simultaneous expression from multiple downstream genes. We plan to address this limitation by leveraging our SCAPE (Single cell transcriptomics-level Cytokine Activity Prediction and Estimation) method [27], that leverages the CytoSig [28] database to perform cytokine activity estimation for scRNA-seq data, to construct a multilayer network definition that incorporates exposure to multiple ligands at once. We can validate the spatial distribution of this multilayer network model by scoring it against Reactome-based pathways, for example. Additionally, while we only evaluate our method for Visium ST data, we plan to extend it for high definition single cell-resolution ST data using a distance thresholding mechanism, which highly weights expression from cells in proximity to each other. Lastly, while we don't show the results due to space constraints, we independently validate our findings using a simulation model.

## References

1. Stähl, P.L., et al.: Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. *Science* **353**(6294), 78–82 (2016). Publisher: American Association for the Advancement of Science. <https://doi.org/10.1126/science.aaf2403>
2. Efremova, M., Vento-Tormo, M., Teichmann, S.A., Vento-Tormo, R.: Cell-PhoneDB: inferring cell-cell communication from combined expression of multi-subunit ligand-receptor complexes. In: *Nature Protocols*, vol. 15, no. 4, pp. 1484–1506 (2020). Publisher: Nature Publishing Group. <https://doi.org/10.1038/s41596-020-0292-x>
3. Jin, S., et al.: Inference and analysis of cell-cell communication using CellChat. In: *Nature Communications*, vol. 12, no. 1, p. 1088 (2021). Publisher: Nature Publishing Group. <https://doi.org/10.1038/s41467-021-21246-9>
4. Wang, S., Karikomi, M., MacLean, A.L., Nie, Q.: Cell lineage and communication network inference via optimization for single-cell transcriptomics. *Nucleic Acids Res.* **47**(11), e66 (2019). <https://doi.org/10.1093/nar/gkz204>
5. Browaeys, R., Saelens, W., Saeys, Y.: NicheNet: modeling intercellular communication by linking ligands to target genes. In: *Nature Methods*, vol. 17, no. 2, pp. 159–162 (2020). Publisher: Nature Publishing Group. <https://doi.org/10.1038/s41592-019-0667-5>

6. Shao, X., et al.: Knowledge-graph-based cell-cell communication inference for spatially resolved transcriptomic data with SpaTalk. In: *Nature Communications*, vol. 13, no. 1, p. 4429 (2022). Publisher: Nature Publishing Group. <https://doi.org/10.1038/s41467-022-32111-8>
7. Yuan, Y., Bar-Joseph, Z.: GCNG: graph convolutional networks for inferring gene interaction from spatial transcriptomics data. *Genome Biol.* **21**(1), 300 (2020). <https://doi.org/10.1186/s13059-020-02214-w>
8. Dries, R., et al.: Giotto: a toolbox for integrative analysis and visualization of spatial expression data. *Genome Biol.* **22**(1), 78 (2021). <https://doi.org/10.1186/s13059-021-02286-2>
9. Cang, Z., et al.: Screening cell-cell communication in spatial transcriptomics via collective optimal transport. In: *Nature Methods*, vol. 20, no. 2, pp. 218–228 (2023). Publisher: Nature Publishing Group. <https://doi.org/10.1038/s41592-022-01728-4>
10. Javaid, A., Frost, H.R.: SPECK: an unsupervised learning approach for cell surface receptor abundance estimation for single-cell RNA-sequencing data. *Bioinf. Adv.* **3**(1), vbad073 (2023). <https://doi.org/10.1093/bioadv/vbad073>
11. Liu, J., et al.: CytoSignal detects locations and dynamics of ligand-receptor signaling at cellular resolution from spatial transcriptomic data. In: Pages: 2024.03.08.584153 Section: New Results (2024). <https://doi.org/10.1101/2024.03.08.584153>
12. Hao, Y., et al.: Dictionary learning for integrative, multimodal and scalable single-cell analysis. *Nat. Biotechnol.* (2023). <https://doi.org/10.1038/s41587-023-01767-y>
13. Hao, Y., et al.: Integrated analysis of multimodal single-cell data. *Cell* (2021). <https://doi.org/10.1016/j.cell.2021.04.048>
14. Stuart, T., et al.: Comprehensive integration of single-cell data. *Cell* **177**, 1888–1902 (2019). <https://doi.org/10.1016/j.cell.2019.05.031>
15. Butler, A., Hoffman, P., Smibert, P., Papalexi, E., Satija, R.: Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.* **36**, 411–420 (2018). <https://doi.org/10.1038/nbt.4096>
16. Satija, R., Farrell, J.A., Gennert, D., Schier, A.F., Regev, A.: Spatial reconstruction of single-cell gene expression data. *Nat. Biotechnol.* **33**, 495–502 (2015). <https://doi.org/10.1038/nbt.3192>
17. Erichson, N.B., Voronin, S., Brunton, S.L., Kutz, J.N.: Randomized Matrix Decompositions Using R. *J. Stat. Softw.* **89**, 1–48 (2019). <https://doi.org/10.18637/jss.v089.i11>
18. Coyle-Rink, J., Valle, L.D., Sweet, T., Khalili, K., Amini, S.: Developmental expression of Wnt signaling factors in mouse brain. *Cancer Biol. Ther.* **1**(6), 640–645 (2002). Publisher: Taylor & Francis. <https://doi.org/10.4161/cbt.313>
19. Anne, S.L., et al.: WNT3 inhibits cerebellar granule neuron progenitor proliferation and medulloblastoma formation via MAPK activation. *PLOS ONE* **8**(11), e81769 (2013). Publisher: Public Library of Science. <https://doi.org/10.1371/journal.pone.0081769>
20. Assali, A., Chenaux, G., Cho, J.Y., Berto, S., Ehrlich, N.A., Cowan, C.W.: EphB1 controls long-range cortical axon guidance through a cell non-autonomous role in GABAergic cells. *Development* **151**(5), dev201439 (2024). <https://doi.org/10.1242/dev.201439>
21. Zhu, Y., et al.: CD45 deficiency drives amyloid- $\beta$  peptide oligomers and neuronal loss in Alzheimer's disease mice. *J. Neurosci.* **31**(4), 1355–1365 (2011). Publisher: Society for Neuroscience Section: Articles. <https://doi.org/10.1523/JNEUROSCI.3268-10.2011>

22. Han, S., Pang, M.-F., Nelson, C.M.: Substratum stiffness tunes proliferation downstream of Wnt3a in part by regulating integrin-linked kinase and frizzled-1. *J. Cell Sci.* **131**(8), jcs210476 (2018). <https://doi.org/10.1242/jcs.210476>
23. Salinas, P.C., Nusse, R.: Regional expression of the Wnt-3 gene in the developing mouse forebrain in relationship to diencephalic neuromeres. *Mech. Dev.* **39**(3), 151–160 (1992). [https://doi.org/10.1016/0925-4773\(92\)90042-I](https://doi.org/10.1016/0925-4773(92)90042-I)
24. Rudolph, J., Gerstmann, K., Zimmer, G., Steinecke, A., DÖding, A., Bolz, J.: A dual role of EphB1/ephrin-B3 reverse signaling on migrating striatal and cortical neurons originating in the preoptic area: should I stay or go away?. *Front. Cell. Neurosci.* **8**, 185 (2014). <https://doi.org/10.3389/fncel.2014.00185>
25. Smith, A., Robinson, V., Patel, K., Wilkinson, D.G.: The EphA4 and EphB1 receptor tyrosine kinases and ephrin-B2 ligand regulate targeted migration of branchial neural crest cells. *Curr. Biol.* **7**(8), 561–570 (1997). [https://doi.org/10.1016/S0960-9822\(06\)00255-7](https://doi.org/10.1016/S0960-9822(06)00255-7)
26. Barashdi, M.A.A., Ali, A., McMullin, M.F., Mills, K.: Protein tyrosine phosphatase receptor type C (PTPRC or CD45). *J. Clin. Pathol.* **74**(9), 548–552 (2021). Publisher: BMJ Publishing Group Section: Molecules in pathogenesis. <https://doi.org/10.1136/jclinpath2020-206927>
27. Javaid, A., Frost, H.R.: Single cell transcriptomics-level Cytokine Activity Prediction and Estimation (SCAPE), pp. 2023.10.17.562739 Section: New Results (2023). <https://doi.org/10.1101/2023.10.17.562739>
28. Jiang, P., et al.: Systematic investigation of cytokine signaling activity at the tissue and single-cell levels. *Nat. Methods* **18**(10), 1181–1191 (2021). Publisher: Nature Publishing Group. <https://doi.org/10.1038/s41592-021-01274-5>



# Exploiting Vertex-Cut Partitioning in Distributed Graph Generation

Furkan Atas and Mehmet Burak Akgun()

TOBB ETU, Ankara, Turkey  
[{furkanatas,m.akgun}](mailto:{furkanatas,m.akgun}@etu.edu.tr)@etu.edu.tr

**Abstract.** Graphs sub-structures have strong influence on the characteristics of real-world complex networks. Moreover, graph generation algorithms must capture and replicate these characteristics to remain representative of their real-world counterparts. The ever-increasing size of data collected from observed networks introduces another layer of complexity -scalability constraints- to the graph generation methods which already have multiple objective functions to satisfy. This paper introduces a novel distributed graph generation methodology that employs vertex partitioning to distribute the graph across multiple execution units. By design, algorithm faithfully replicates the exact Joint Degree Matrix(JDM) characteristics of the original graph. We conducted an extensive profiling study of the algorithm to analyze the effect of execution unit count and partition count on its runtime performance. The results demonstrate that, with just a few compute servers, the algorithm can efficiently scale to handle hundreds of millions of vertices within a reasonable time frame.

**Keywords:** JDM · distributed computing · graph generation

## 1 Introduction

Generating synthetic data is a critical consideration across numerous research and industrial sectors. In machine learning, synthetic data is needed to increase the number of data samples and ensure data diversity (also known as data augmentation), while privacy (anonymity) is the primary concern for IoT [1] and medical data [2]. In some domains, both of them may be required as in network analysis.

To evaluate an internet protocol algorithm, it's crucial to have diverse network examples. This allows for detecting potential bottlenecks prior to real-world deployment. For a social media company, their network consists entirely of sensitive private data. When developing a mining algorithm, they must ensure that the data remains concealed from third parties.

A data generation tool, which also aids in anonymization, should be utilized to replicate the original data's characteristics and produce new samples that preserve these features. Without this, accurate testing becomes impossible. For

testing to be reliable, the data must reflect its genuine attributes. In the case of social network algorithms, using an internet network for testing is not feasible. This principle applies universally across all downstream tasks.

Mathematically, a network is represented as a graph, and in a broader sense, network generation can be considered as the process of graph generation which is a well studied research area back to 1990s [3] and beyond. Up until today, production has primarily been carried out in two different ways: Traditional and Machine Learning - Deep Learning (ML-DL) based generation. Past works for both traditional and ML based methods will be discussed in related works in detail but we will look at some advantages and disadvantages of these models. Traditional methods focus on a specific property of a real-world graph and aim to generate a graph that aligns with the chosen characteristic. However, these methods cannot accommodate other properties. [4]. Machine learning-based models implicitly capture the characteristics of the provided data [5], allowing most features of the data to be represented in a synthetic sample. However; these methods can not scale well for a large graph generation due to training concerns whereas it is not a problem for traditional hand-crafted settings. For example; Liao et al. [6] assert that their approach surpasses earlier deep generative models in terms of node count, though their generated graph samples contain only 5k nodes

More recently, generated and processed data-sets scaled up to petabyte sizes in many domains ( $10^{15}$  bytes) [7,8]. To cope with such a large amount of data, some algorithms are designed in a distributed fashion. Graph pattern matching [9], graph mining [10] are also some examples of those algorithms. To use large synthetic graphs in places where enormous graphs are need, distributed graph generation becomes inevitable. Non-distributed methods designed to run on a single compute node will not be able to generate such large graphs in a reasonable amount of time.

In this study, we propose a distributed graph generation algorithm centered on joint degree distribution to enhance accuracy and scalability in both traditional and machine learning generation methods. By concentrating on joint degree distribution, we can also indirectly address other graph properties without having to define them explicitly. Our approach seeks to merge the advantages of both implicit and traditional generation techniques. Through the use of distributed computation, we tackle scalability challenges, and with the 2k method, we aim to achieve a higher accuracy level than conventional methods.

## 2 Related Work

Previous work in the field has been studied under various taxonomies in the literature [11, 12]. However, we are going to examine under three categories; i.e., traditional methods, ML-DL and distributed methods.

### 2.1 Traditional Methods

Synthetic data should reflect the traits of real data to ensure its applicability in research. Chakrabarti et al. [13] developed a recursive algorithm to generate real-

istic graphs. Leskovec et al. [14] indicate that preferential attachment [15] and some other generators target only one characteristic of the real networks, not enough representation capability. To further improve the realistic generation, they produces a comprehensive generation model utilizing kronocker product. For some special case, Rmat and Erdos-Renyi graphs are obtained. Most of the networks have power-low degree distribution also called as scale-free network [15]. Guided by this fact, many researchers generates graphs preserving power-law degree distribution either from scratch or using prescribed degree sequences [16–19]. Mahadevan et al. [20] present a generation framework based on the probability distribution of the dk series, where  $1k$  corresponds to the degree distribution,  $2k$  represents the joint degree distribution, and higher levels like  $3k$  capture more complex relationships. They implemented up to  $2k$  from scratch and applied a rewiring technique for the  $3k$  series. [21] says although power-low degree distribution is observed in many real life networks, assortativity in networks differ from context to context, therefore joint degree distribution should be used in generation. Also, a recent study on random graphs [22] says that production based on degree sequences is not sufficient to reflect real graph properties. Nagy et al. [23] made a network classification using both real networks and their synthetic counterparts generated by 4 different generation models. They say that  $2k$  generation models is the state-of-the-art models to keep the structural properties of networks.

Building on these studies, it has been determined that the representational power of  $2k$  sequences needs to be supplemented with additional techniques. To generate more accurate graphs,  $2k$  sequences are enhanced with supplementary metrics. For instance, [24] examines the randomness of real networks by generating random networks aimed at replicating the original ones. They point out that  $2k$  statistics are insufficient to capture the global properties of a graph. While  $3k$  statistics could address this, they demand significant computational resources. As a result, incorporating global features in the generation process becomes necessary. Certain real networks, like brain networks with weakly connected components, remain difficult to represent accurately. To address these limitations, the authors propose the  $2.1k$  and  $2.5k$  algorithms. Gjoka et al. [25] implement the  $2k\_simple$  algorithm linear running time in number of edges which directly targets joint degree matrix (JDM) of a given graph. Besides, they extends this algorithm with two different network properties. Tillman et al. [26] implement  $2k\_simple$  algorithm in [25] as well. However, this work provides additional algorithms utilizing new network metrics. Besides they handle the generation problem for directed graphs. Several studies have leveraged the  $2k$  series for specific types of graphs, taking advantage of its capabilities to accurately model graph characteristics relevant to those specific structures. Boroojeni et al. [27] uses  $2k$  algorithm for bipartite networks.

## 2.2 ML-DL Methods

Due to the long training time, DL based methods suffer from low number of nodes. Some of them directly targets molecular graph generation witch requires

number of nodes smaller than 200 for most of the cases. MolGAN [28] improve the objective reinforcement [29] idea using graph structure instead of string based generation. By using GNN [30] in discriminator and reward networks, they leverage the graph structure and address the node permutation issue present in sequential generation models like GraphRNN [31] and the model proposed by Li et al. [32]. However, their experiments are conducted at most 9-nodes graph while GraphRNN generates 500-nodes graphs. GRAN [6] generates nodes in a sequential manner. On the other hand, edges are generated as blocks. They combine both one-shot and sequential generation. They achieve 5k-nodes. Recently, diffusion models [33] rise in generative ai area due to their high quality performance in image tasks. After this success, diffusion models are applied to graph domain, Vignac et al. [34]. However, their scalability performance is a concern as in previous DL methdos.

### 2.3 Distributed Methods

[12] emphasizes the need for scalable generation to produce massive graphs within a reasonable time frame. The study explores various generation techniques and parallel algorithms, analyzing their performance on both CPU and GPU architectures. Atas et al. [35] distributes the 1k generation algorithm across a cluster of computing nodes. Allendorf et al. [36] proposes a parallel Markov Chain sampling generation algorithm utilizing prescribed degree sequences. Park et al. [37] propose a new method called recursive vector model better than RMAT [13] and Kroncker [14]. They distribute the generation process across a cluster to tackle scalability challenges. Funke et al. [38] use communication free paradigm to obtain parallelism. Their work parallelizes a series of well-known generation models, allowing them to scale up to an enormous number of vertices by utilizing a supercomputer. Jung et al. [39] proposes Stochastic Kronecker Signed Graph (SKSG) method on top of basic Kroncker multiplication to generate signed graphs. Using SKSG, Balansing is introduced to make the generation distributed over a Spark cluster. Park et al. [40] make preferential attachment model multi-threaded and reaches trillion scale in number of edges within a small cluster of PC.

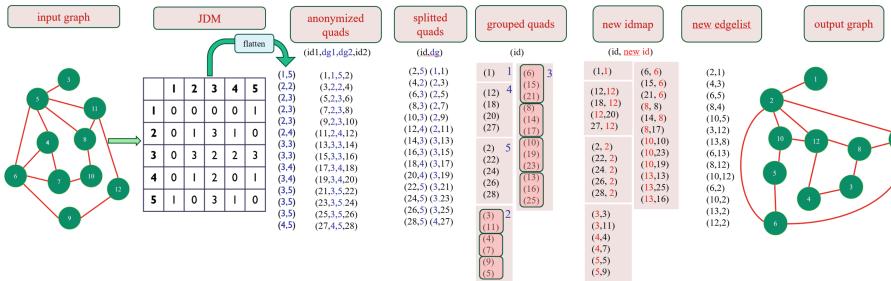
The aforementioned parallel methods offer certain advantages within their respective scopes. However, these methods rely either on degree sequences or other techniques, and none have successfully parallelized 2k generation using both a multi-threaded approach and distributed computing. Scaling to graphs with trillions of nodes is a desired capability for large-scale generators, but its equally important that the generated graph retains properties as realistic as possible to those of real-world networks. In response to this need, we present a distributed 2k generation algorithm capable of producing massive graphs with greater representative power. Since previous research has shown that 2k alone may be insufficient in certain cases, future versions of the algorithm could be updated to incorporate additional properties, as suggested by references [24, 26].

### 3 Algorithm Design

dk-series represents a family of algorithms designed for systematic analysis of graphs with the aim of targeting specific characteristics. 0k focuses only on the average degree while 1k refers to degree distribution and 2k refers to joint degree distribution. One can extract lower level characteristic from upper level ones. For example average degree can be extracted from degree distribution as well as degree distribution can be computed from the joint degree matrix. In the rest of the paper, the following notation is used: For a graph  $G = (V, E)$ ,  $V$  refers to node set while  $E$  refers to edge set. Total number of nodes equals  $n = |V|$ , total number of edges equals  $m = |E|$ . Degree of a node  $u$  is represented by  $\deg(u)$

Joint degree distribution is a probability distribution that reflects the mutual probability of two different degree nodes being connected. Namely as  $k$  value increases, the constraints increases rapidly from a scalar to tensors. Joint Degree Matrix (JDM) is a two-dimensional matrix where the size equals to the maximum degree in the graph. A cell with an integer value  $s$  at the index  $(k, l)$  indicates that there are  $s$  edges in the graph where one endpoint is a  $k$ -degree node and the other endpoint is an  $l$ -degree node.

Tillman et al. [26] implements 2k algorithm as follows: generate all nodes with free stubs (half edges, not connected to any other nodes). While traversing JDM, for each cell, they add a new edge between nodes whose degrees are equal to cell index. At each iteration a new edge is added, number of unconnected nodes with free stubs decreases.



**Fig. 1.** Distributed graph generation flow diagram

Compared to Tillman algorithm, our algorithm is a bit tricky. Figure 1 illustrates the generation and Algorithm 1 summarized the below explanations. We do not use GraphX [41] because of its partitioning strategy. Instead, we rely on Spark's core data abstraction, the Resilient Distributed Dataset (RDD) [42]. Due

to the immutable characteristic of RDD, we can not iterate over each cell one by one. Since each cell in JDM contains some number of edges whose end degrees are specified, we flatten the JDM to obtain (degree, degree) list. Using the list, we generate *anonymized quad* data type. For a given entity of flattened JDM (degree1, degree2), its corresponding *anonymized quad* is (sourceid1, degree1, degree2, sourceid2) where degree 1 and 2 refers degree information of nodes respectively. Since we have no node id information in JDM originally, we generate anonymized *quads* whose sourceids are generated with an indexing method in Spark. Each sourceid is different from the others. Total number of edges in graph is equal to sum of the values in each cell either upper half diagonal or lower half diagonal for undirected graphs. Therefore, total number of *anonymized quads* equals to total number of edges in the graph.

*anonymized quads* are split into two parts, with each part being referred to as a *split quad*. Every single piece represent a node and its degree information. Certainly, some of the nodes are identical since they belong to same graph. After being split, *anonymized quads* with the same degree can be matched with each other. To elaborate further, consider two split quads that match. If one has a degree of 2 and the other a degree of 3, merging them would result in a node with both degrees, which causes a contradiction. The same degree *split quads* are grouped. Number of split *quads* can be divided by their groups' degree. At this point, we break distributed paradigm and copy the RDD data into local arrays. Each anonymized split *quad* matches to its final node id. Every group divided into subgroups whose sizes are equal to degree information of that group. In every subgroup, a node id of a split *quad* is assigned to id of each split quad. The randomness in our algorithm is implemented here. Subgrouping is made randomly. Using this id map, anonymized quads transformed into new edge list via a basic map procedure. This procedure is implemented as in Algorithm 1. Algorithm outputs new graph as an edge list. Since we use an original graph, we do not face realizable problem in [26].

The running time of distributed algorithm is solely depended on number of edges, how huge the network in number of nodes is not important. This is because, we form *quads* for each individual edges. After splitting the quads, the amount of data doubles, but this has no impact on the asymptotic analysis. Subsequently, a grouping step takes place. Rather than using distributed computation, we employ a standard iterative process during the generation of the new id map. For each group, the running time is bounded by its length,  $\Theta(\text{gruoplength})$ . Sum of each groups running time equal to total number of split *quads*,  $\Theta(2*m)$ . Therefore it became  $\Theta(m)$ . Number of groups is at most the maximum degree of the graph.

$$2 * m = \sum_{i=1}^{d_{max}} t_i \quad (1)$$

where,

$$d_{max} = \max(\deg(u)) \quad \text{for all } u \in V$$

$$t_i = \text{number of split quads in } i \text{ degree group}$$

Equation 1 shows the sum of all split *quads* on each group equals to total number of edges  $m$  multiplied by 2.

---

**Algorithm 1.** Distributed 2k Generation Algorithm

---

```

 $JDM \leftarrow InputGraph$
 $AnonymizedQuads \leftarrow JDM$
 $SplitQuads \leftarrow AnonymizedQuads$

 $GroupedSplitQuads \leftarrow SplitQuads$

 $Idmap \leftarrow GroupedSplitQuads$

 $NewEdgeList \leftarrow IdMap, AnonymizedQuads$

```

---

## 4 Experiments

### 4.1 Experimental Setup

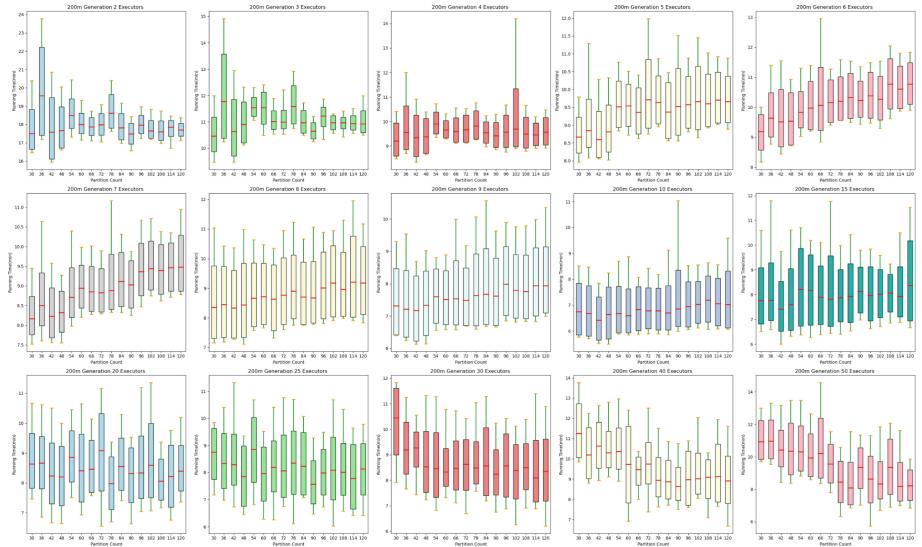
To feed input graph, we run a one node Hdfs cluster. To run the algorithm, we set up a Spark cluster containing 5 commodity computers. Total number of cores in the system becomes 180, total memory reaches up to 0,8 TB. For each executor, we employ 3 cores and 10 GB memory. Total number of executors can reach up to 60, however we observed 50 executors at most. Machines are connected together via a 10Gbps switch.

### 4.2 Dataset

As input data we use Friendster social network employed Yang et al. [43] from Snap Dataset [44]. There are 65 Million nodes and 1,8 billion edges. This data is used for community research but we do not care about communities. Therefore we use only edgelist file given in the website. A sampling algorithm is employed for different size graphs for testing purposes. We prune the original graph so that we have sub-graphs with edges of multiples of 100 million. The reason for this is the running time of distributed algorithm depends on  $m$ . Thus we can test the performance with a stable scale. Considering adjacency matrix representation, let  $X$  be the number of nodes in sampling, the first  $X$  column and row is sampled with the last node with its all edges.

### 4.3 Partition Count Analysis

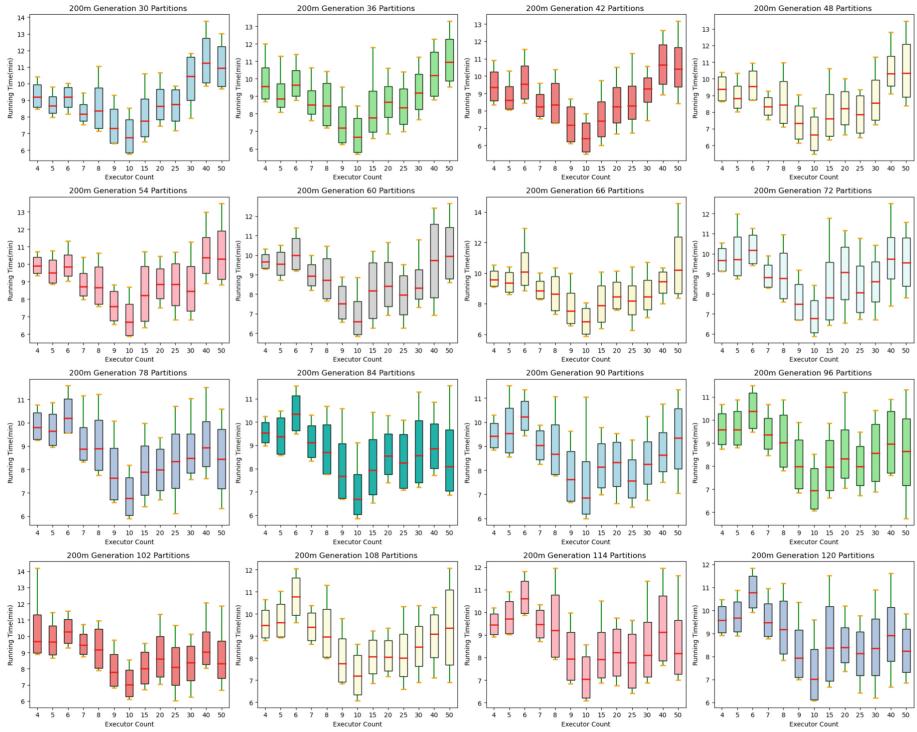
A partition represents a unit of data. For distributed computation, it is essential to distribute the data across multiple machines. Therefore data is separated into partitions and each executor performs the computation on the partition given to it. In this experiment, we examine the impact of increasing the number of partitions on the running time. The more partitions there are, the smaller the data size in each partition since total datasize (for example an edgelist file in our case) is constant. For the rest, we use partition count (pc) to represent number of partitions.



**Fig. 2.** For different number of executors, partition count trace results 200 m Edge Graph Generation

### 4.4 Executor Count Analysis

In this experiment, we analyze how the running time varies with different number of executors. For the rest, executor count is abbreviated as ec. Graph generation with 200 m experiment are conducted 10 times. In Fig. 3, we observe the trace of executor numbers for various data partitions. As the number of executors increases, there is a decrease in running time up to some point, then the increase in ec does not affect the running time considerably up to some point. The further increase in ec causes the increase in the running time, a point where we lose the advantage of distributed computing, the inflection point.



**Fig. 3.** For different number of partitions, executor count trace results 200 m Edge Graph Generation

## 5 Conclusion

In this work, our goal was to scale up the graph generation with joint degree matrix for all types of graphs and networks. In a distributed environment, we scale basic 2k generation algorithm. In the literature, there are two generation methods, traditional and ml based methods. In traditional ways, generation scales very well but only targeted properties are preserved. ML based methods captures implicit characteristics but do not scale well. Our work scales the generation size million and even billion edge size providing implicit characteristics. The distributed algorithm, although it does not preserve implicit features as well as ML-based methods, is better than traditional methods. In experiments, we use subgraph of a real social network. The effects of number of partitions and executors are observed. We observe small changes in partition trace but in executor trace we determine an inflection point. After this point increase in executor number losses the advantage of distributed environment. For the future, we will look at higher sized networks with more hardware to see the inflection point change. In addition, we will explore ways to improve the algorithm by adding extra features to 2k.

**Acknowledgements.** This work was supported in part by a Saleh A. Kamel Graduate Fellowship at TOBB ETU.

## References

1. Antoniou, A., Storkey, A., Edwards, H.: Data augmentation generative adversarial networks. arXiv preprint [arXiv:1711.04340](https://arxiv.org/abs/1711.04340) (2017)
2. Anderson, J.W., Kennedy, K.E., Ngo, L.B., Luckow, A., Apon, A.W.: Synthetic data generation for the Internet of Things. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 171–176 (2014)
3. Dick, R.P., Rhodes, D.L., Wolf, W.: TGFF: task graphs for free. In: Proceedings of the Sixth International Workshop on Hardware/Software Code-sign.(CODES/CASHE'98), pp. 97–101 (1998)
4. Guo, X., Zhao, L.: A systematic survey on deep generative models for graph generation. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 5370–5390 (2022)
5. Bojchevski, A., Shchur, O., Zügner, D., Gunnemann, S.: NetGAN: generating graphs via random walks. In: International Conference on Machine Learning, pp. 610–619 (2018)
6. Liao, R., et al.: Efficient graph generation with graph recurrent attention networks. *Adv. Neural Inf. Process. Syst.* **32** (2019)
7. Qi, C.: Big data management in the mining industry. *Int. J. Miner. Metall. Mater.* **27**(2), 131–139 (2020). <https://doi.org/10.1007/s12613-019-1937-z>
8. Bouhenni, S., Yahiaoui, S., Nouali-Taboudjemat, N., Kheddouci, H.: A survey on distributed graph pattern matching in massive graphs. *ACM Comput. Surv. (CSUR)* **54**, 1–35 (2021)
9. Ma, S., Cao, Y., Huai, J., Wo, T.: Distributed graph pattern matching. In: Proceedings of the 21st International Conference on World Wide Web, pp. 949–958 (2012)
10. Teixeira, C.H., et al.: Arabesque: a system for distributed graph mining. In: Proceedings of the 25th Symposium on Operating Systems Principles, pp. 425–440 (2015)
11. Drobyshevskiy, M., Turdakov, D.: Random graph modeling: a survey of the concepts. *ACM Comput. Surv. (CSUR)* **52**, 1–36 (2019)
12. Penschuck, M., et al.: Recent advances in scalable network generation. arXiv preprint [arXiv:2003.00736](https://arxiv.org/abs/2003.00736) (2020)
13. Chakrabarti, D., Zhan, Y., Faloutsos, C.: R-MAT: a recursive model for graph mining. In: Proceedings of the 2004 SIAM International Conference on Data Mining, pp. 442–446 (2004)
14. Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C.: Realistic, Mathematically Tractable Graph Generation and Evolution, Using Kronecker Multiplication. In: Jorge, A.M., Torgo, L., Brazdil, P., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 133–145. Springer, Heidelberg (2005). [https://doi.org/10.1007/11564126\\_17](https://doi.org/10.1007/11564126_17)
15. Barabasi, A.-L., Albert, R.: Emergence of scaling in random networks. *Science* **286**, 509–512 (1999)
16. Milo, R., Kashtan, N., Itzkovitz, S., Newman, M.E., Alon, U.: On the uniform generation of random graphs with prescribed degree sequences. arXiv preprint cond-mat/0312028 (2003)

17. Berger, A., Müller-Hannemann, M.: Uniform sampling of digraphs with a fixed degree sequence. In: Thilikos, D.M. (ed.) WG 2010. LNCS, vol. 6410, pp. 220–231. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-16926-7\\_21](https://doi.org/10.1007/978-3-642-16926-7_21)
18. Blitzstein, J., Diaconis, P.: A sequential importance sampling algorithm for generating random graphs with prescribed degrees. *Internet Math.* **6**, 489–522 (2011)
19. Fosdick, B.K., Larremore, D.B., Nishimura, J., Ugander, J.: Configuring random graph models with fixed degree sequences. *SIAM Rev.* **60**, 315–355 (2018)
20. Mahadevan, P., Krioukov, D., Fall, K., Vahdat, A.: Systematic topology analysis and generation using degree correlations. *ACM SIGCOMM Comput. Commun. Rev.* **36**, 135–146 (2006)
21. Stanton, I., Pinar, A.: Sampling graphs with a prescribed joint degree distribution using Markov chains. In: 2011 Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments (ALENEX), pp. 151–163 (2011)
22. Van Koevering, K., Benson, A., Kleinberg, J.: Random graphs with prescribed k-core sequences: a new null model for network analysis. In: Proceedings of the Web Conference 2021, pp. 367–378 (2021)
23. Nagy, M., Molontay, R.: Network classification-based structural analysis of real networks and their model-generated counterparts. *Netw. Sci.* **10**, 146–169 (2022)
24. Orsini, C., et al.: Quantifying randomness in real networks. *Nat. Commun.* **6**, 8627 (2015)
25. Gjoka, M., Tillman, B., Markopoulou, A.: Construction of simple graphs with a target joint degree matrix and beyond. In: 2015 IEEE Conference on Computer Communications (INFOCOM), pp. 1553–1561 (2015)
26. Tillman, B., Markopoulou, A., Gjoka, M., Buttsc, C.T.: 2k+ graph construction framework: targeting joint degree matrix and beyond. *IEEE/ACM Trans. Netw.* **27**, 591–606 (2019)
27. Borojeni, A.A., Dewar, J., Wu, T., Hyman, J.M.: Generating bipartite networks with a prescribed joint degree distribution. *J. Complex Netw.* **5**, 839–857 (2017)
28. De Cao, N., Kipf, T.: MolGAN: An implicit generative model for small molecular graphs. arXiv preprint [arXiv:1805.11973](https://arxiv.org/abs/1805.11973) (2018)
29. Guimaraes, G.L., Sanchez-Lengeling, B., Outeiral, C., Farias, P.L.C., Aspuru-Guzik, A.: Objective-reinforced generative adversarial networks (organ) for sequence generation models. arXiv preprint [arXiv:1705.10843](https://arxiv.org/abs/1705.10843) (2017)
30. Wu, Z., et al.: A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **32**, 4–24 (2020)
31. You, J., Ying, R., Ren, X., Hamilton, W., Leskovec, J.: GraphRNN: generating realistic graphs with deep auto-regressive models. In: International Conference on Machine Learning, pp. 5708–5717 (2018)
32. Li, Y., Vinyals, O., Dyer, C., Pascanu, R., Battaglia, P.: Learning deep generative models of graphs. arXiv preprint [arXiv:1803.03324](https://arxiv.org/abs/1803.03324) (2018)
33. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. *Adv. Neural. Inf. Process. Syst.* **33**, 6840–6851 (2020)
34. Vignac, C., et al.: DiGress: Discrete denoising diffusion for graph generation. arXiv preprint [arXiv:2209.14734](https://arxiv.org/abs/2209.14734) (2022)
35. Furkan Atas, M.B.A.: in Complex Networks and Their Applications, Book of Abstracts, pp. 557–559 (2021). <https://2021.complexnetworks.org/>
36. Allendorf, D., Meyer, U., Penschuck, M., Tran, H.: Parallel global edge switching for the uniform sampling of simple graphs with prescribed degrees. *J. Parallel Distrib. Comput.* (2023)

37. Park, H., Kim, M.-S.: TrillionG: a trillion-scale synthetic graph generator using a recursive vector model. In: Proceedings of the 2017 ACM International Conference on Management of Data, pp. 913–928 (2017)
38. Funke, D., et al.: Communication-free massively distributed graph generation. *J. Parallel Distrib. Comput.* **131**, 200–217 (2019)
39. Jung, J., Park, H.-M., Kang, U.: BalanSiNG: fast and scalable generation of realistic signed networks. In: EDBT, pp. 193–204 (2020)
40. Park, H., Kim, M.-S.: LineageBA: a fast, exact and scalable graph generation for the Barabási-Albert model. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE), pp. 540–551 (2021)
41. Gonzalez, J.E., et al.: GraphX: graph processing in a distributed dataflow framework. In: 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14), pp. 599–613 (2014)
42. Zaharia, M., et al.: Resilient distributed datasets: a fault-Tolerant abstraction for In-Memory cluster computing. In: 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12), pp. 15–28 (2012)
43. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. In: Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, pp. 1–8 (2012)
44. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford Large Network Dataset Collection <http://snap.stanford.edu/data> (2014)



# Flow-Model Capacity for Large Scale Quantum Complex Networks

Ricardo Quintas<sup>(✉)</sup> and Bruno Coutinho

Instituto de Telecomunicações, Lisbon, Portugal  
[ricardoquintas7@gmail.com](mailto:ricardoquintas7@gmail.com)

**Abstract.** A quantum network has the potential to integrate quantum technologies on a large scale, creating the so-called quantum internet. In this paper, we examine the connectivity of a large-scale quantum network in relation to a target distribution rate and end-to-end fidelity. We consider a network where each link consists of standard quantum photonic channels, characterized by a trade-off between rate and fidelity allowing the nodes in the network to be connected. The network employs a simple entanglement distribution protocol without error correction or purification. We analyze different network topologies, specifically Scale-Free and Erdős-Rényi networks, and provide analytical solutions for Erdős-Rényi in the thermodynamic regime and semi-analytical solutions for Scale-Free and Erdős-Rényi networks in the finite-size regime. Our results show a phase transition in network connectivity as a function of the target distribution rate, which can be continuous or discontinuous depending on the network topology and size. Furthermore, our findings indicate that Erdős-Rényi networks are more prone to discontinuous phase transitions than Scale-Free.

**Keywords:** Quantum Networks · Quantum Internet · Large-Scale Quantum Complex Networks · Flow Model · Entanglement Generation

## 1 Introduction

Quantum technologies have recently attracted significant attention from researchers, starting with Shor's groundbreaking work [12, 13], similarly, other work has also followed in expanding the interest in the field [2, 3, 8]. Since then, extensive research has focused on developing technologies that leverage quantum properties, one of which is the quantum network. This network has the potential to expand quantum technologies from a local scale, like quantum computation, to a large-scale distributed quantum computation framework. Recent studies have demonstrated the feasibility and benefits of such expansion [5, 7, 15, 16]. While the quantum network offers advantages like truly secure encryption, several challenges must be addressed before it can become a reality.

These challenges include the decoherence of entanglement a valuable resource used in quantum technologies, where two or more particles share a unique correlation without any classical counterpart, and the inability to use classical repeater

techniques due to the “No Cloning Theorem” [14]. One way to address the latter issue is by employing a quantum repeater protocol, which involves entanglement generation and swapping, and in more complex protocols, also purification. Each operation can be seen in more detail in Munro et al. [9]

With the use of a quantum repeater, we will employ the connectivity metric proposed by Coutinho et al. [7]. This metric will allow us to assess how well-connected portions of the network are concerning network resources, specifically the rate (number of qubits per second) and number of qubits. We will study the connectivity of the network for two models without the use of purification protocols and solely focusing on entanglement generation and swapping.

The Erdős-Rényi (ER) and Scale-Free (SF) models more specifically the Bollobás variant of the Barabási-Albert model [1], with different sizes  $N = \{10^3, 10^4, 10^5\}$  and different average degrees  $\lambda = \{6, 8\}$ . Initially, we will work in a theoretical scenario using closed formulas provided by the entanglement protocols and the ER network. Later, we will progress to a numerical setting where we introduce and compare the results.

## 2 Entanglement Generation

Quantum links in a quantum network are often characterized based on two parameters: rate of distribution and fidelity. The rate of distribution measures how many entangled qubits can be generated and distributed per unit time through such a link. Fidelity measures how close each entangled pair is to a maximally entangled qubit pair, often being used as a measure of the entangled qubit pairs. In [6], the authors proposed a simple entanglement generation protocol using two NV-centers (nitrogen-vacancy centers in diamond) placed inside photonic cavities at the ends of a photonic fiber. Laser pulses are used to create entangled NV-center pairs. This setup is commonly used for entanglement generation between two stations. The fidelity( $F$ ) of the entangled qubit pairs generated in this setup is given by

$$F = \frac{1}{2} + \frac{e^{-P_{\text{em}}(1-\epsilon)}}{2} - \frac{P_{\text{dark}}}{P} - a. \quad (1)$$

Here,  $P$  is the success probability,  $P_{\text{em}}$  is the emission probability, which increases with the duration of the laser pulse, and  $\epsilon$  is the collection efficiency (the probability that the cavities collect a photon). In this paper, we fix  $\epsilon = 0.5$ .

The formula also includes  $P_{\text{dark}}$  and “ $a$ ”, representing the dark count probability (the probability that a detector counts a photon even when none was emitted) and a limitation on the maximum fidelity that can be achieved with this setup, respectively. For simplicity, we assume an ideal setting with no dark counts and no maximum fidelity limit, so we do not consider these parameters in our work. Additionally, the authors provided a formula for the success probability of generating a qubit:

$$P = \frac{1 - e^{-P_{\text{em}} \frac{\epsilon}{2}}}{2}, \quad (2)$$

These equations establish a relationship between the achieved fidelity and the success probability. Consider a scenario where  $n$  entangled qubit pairs are generated per unit time, and  $n$  is large. In this case, the entanglement generation rate  $r$  can be approximated by  $r = np$ . By rearranging these terms, we can express the success probability as a function of fidelity, yielding,

$$\gamma = \frac{1}{2} \left( 1 + (1 - 2P)^{\frac{2-2\epsilon}{\epsilon}} \right) \iff \gamma = \frac{1}{2} \left( 1 + \left( 1 - 2\frac{r}{n} \right)^{\frac{2-2\epsilon}{\epsilon}} \right), \quad (3)$$

This model highlights a clear trade-off between fidelity and rate, while this is a notable drawback, we continue to use the model because it provides a closed formula for our scenario.

### 3 Flow Model and Werner States

In quantum networks, when working with entanglement, it is common practice to distribute entanglement throughout the network. To optimize this distribution, we employ a flow model proposed by [5] that allows us to maximize the distribution rate. In this context, flow represents the maximum entanglement distribution rate. We consider an idealized model where flow is conserved, meaning the amount of flow entering a node equals the amount leaving. In such models, the maximum entanglement generation rate along a path is bottlenecked by the smallest flow in the path,  $P'$ , i.e.,

$$r_P = \min_{e \in P'} r_e \quad (4)$$

where  $e$  represents a link belonging to  $P'$ . We adopt a generalized definition of entanglement distribution rate that takes into account the fidelity constraint. Specifically, we consider the maximum entanglement distribution rate as a function of target fidelity. For this, we will consider Werner states defined as  $\rho_w = \gamma|\Phi^+\rangle\langle\Phi^+| + (1 - \gamma)I_4$ , where  $\gamma = (4F - 1)/3$ , and  $\gamma$  is the amplitude probability found in qubits. These states have a valuable property: when traveling through a path in the network, the  $\gamma$  of the end-to-end state connecting the end nodes is equal to the product of each  $\gamma_e$  along the path, resulting in an overall decrease in fidelity [4]. This product of states is achieved through an operation called swapping, where it becomes possible for distant entangled nodes to create a new entangled node, where the resulting fidelity decreases according to the number of swaps. Since we consider high-fidelity states, making it straightforward to determine the fidelity of our states. Combining all of these elements, we arrive at

$$r_{P'}(\gamma) \leq \min_{e \in P'} r_e(\gamma_e) \quad (5)$$

$$\gamma_{P'} = \prod_{e \in P'} \gamma_e \quad (6)$$

Here,  $r_{P'}$  represents the maximum possible entanglement distribution rate between nodes using path  $P$ , for a given fidelity  $F$ . This model links fidelity with concepts from the original flow model.

## 4 Connectivity of the Network

To evaluate network connectivity, we will use a metric proposed by Coutinho et al. [7]. This connectivity metric considers not only whether a path exists between two nodes, but also if such a path confers a connection with sufficient quality, typically represented by its fidelity. In their original work, the authors use a pruning process starting with the network diameter ( $D$ ), which represents the maximum distance between any two nodes in the network. They iteratively remove links that lack sufficient resources to transmit entanglement across the diameter at a specified resource level. As links are removed, the diameter may increase, leading to further link removals. This process continues until the remaining links ensure that any two randomly chosen nodes in the network can communicate with the required quality of service (qos).

We implement this metric by considering all the maximum lengths of the link between two nodes in the network that may used,  $l_e$ . We then start a pruning process by removing all the links that are smaller than a free parameter designated by  $l$ , where we want to find the smallest  $l$  such that it's greater or equal than the diameter of the network. Mathematically we consider then the probability of a given link being greater than this free parameter  $p = P(l_e > l)$ , this  $l$  serves as a lower bound of the links we can retain to achieve a connected component where any two nodes can communicate with each other called the “Communication Core”

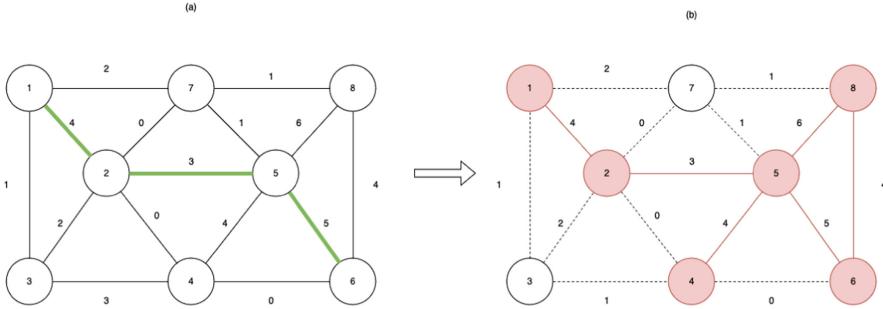
This process is illustrated in Fig. 1. In part (a), we start with a network having a diameter of 3, indicated by the green line. The numbers on the edges represent the entangled pairs, which are our resources. Links with fewer than 3 entangled pairs are removed. In part (b), after removing these links, we are left with the largest connected component (shown in red), where we can ensure that any two randomly chosen nodes can communicate with each other with a certain qos. This example demonstrates the application of the algorithm.

## 5 Results

### 5.1 Length Function

As discussed in Sect. 4, to employ the connectivity metric, we need a length function  $l$ . To derive this function, we use the equations from Sect. 2 along with the properties of Werner States.

Our goal is to ensure that the end-to-end fidelity meets or exceeds our target fidelity ( $\gamma_T = 0.99$ ). This guarantees that a connection between two nodes can be established with the desired rate and fidelity. By using the ideas from Eqs. 5 and 6 we can begin constructing the length function by considering that the



**Fig. 1.** Example of the connectivity metric employed. The figure shows how links with insufficient resources are removed to ensure that the remaining structure, depicted in red, allows any two randomly chosen nodes to communicate with each other at the diameter distance.

fidelities across the links in the network are the same and that we perform swapping operations through a path we can derive a formula for the links that meet these requirements.

$$\gamma^l = \gamma_T \iff l = \frac{\ln(\gamma_T)}{\ln(\gamma)}, \quad (7)$$

In this context,  $\gamma$  is given by Eq. 3, and  $l$  represents the length of a link obtained through multiple swaps.

With the equation for the link length in hand, we can now calculate the probability that a link selected at random  $l_e$  is usable within our formulation, denoted as  $p = P(l_e > l)$ .

This approach provides us with a formula for the length of our network, incorporating variables that depend on quantum parameters.

$$\frac{\ln(\gamma_T)}{\ln\left(\frac{(\frac{2r}{\ln(p)\langle n \rangle} + 1)^{\frac{2-2\epsilon}{\epsilon}} + 1}{2}\right)} = l(\ln(p), r). \quad (8)$$

We introduce a variable substitution by letting  $\ln(p) = -x$  to simplify our analysis, where  $p$  is the probability of the occupied links  $\langle n \rangle$  the average number of qubits.

We assume that the number of qubits  $n_i$  in the links follow an exponential distribution with parameter  $1/\langle n \rangle$ . To find the probability of sending an average of one qubit, we use  $\langle n \rangle = 1/P$ , where  $P$  is the probability of achieving a state with the desired fidelity. We designate this “new”  $\langle n \rangle$  as  $\langle n_0 \rangle$  to indicate that it represents the probability of sending one qubit on average. By multiplying  $\langle n_0 \rangle$  by a constant, we can easily determine the probability of sending  $\langle n \rangle = c\langle n_0 \rangle$ .

For the remainder of this section, we focus on the ER network model for our theoretical analysis and modulation. Later, we will also introduce the SF

model in our numerical results. We have chosen the ER model for its closed-form expressions and theoretical tractability, despite its lack of some complex network properties.

In the literature [1, 10], the diameter of an (ER) network is typically estimated using the formula  $\ln(N)/\ln(c)$ , where  $c$  is the average degree. However, Riordan et al. [11] propose a more rigorous estimation by introducing an additional term to the conventional formula. Their proposed diameter is given by  $D = \ln(N)/\ln(c) - 2/\ln(c^*)$ , where  $c^*$  is the solution to the equation  $c^*e^{c^*} = ce^{-x}e^{-ce^{-x}}$ . The solution  $c^*$  is related to the Lambert W function.

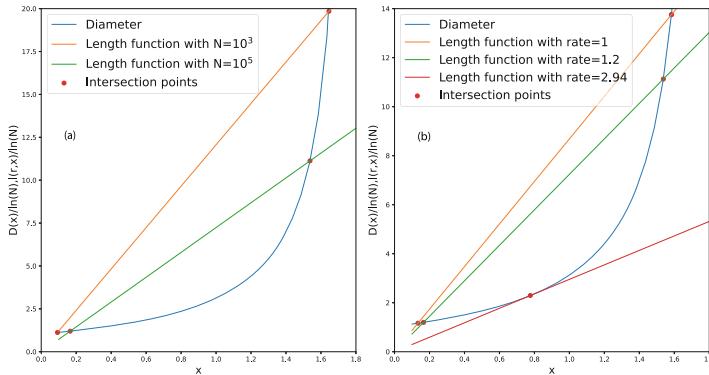
For an ER network with average degree  $\lambda$  and with links remaining with probability  $p = e^{-x}$ , the diameter formula becomes:

$$D(\lambda, x) = \frac{\ln(N)}{-x + \ln(\lambda)} - \frac{2 \ln(N)}{\ln(-W(-\lambda e^{-x} - \lambda e^{-x}))} \quad (9)$$

We are interested in finding the intersection of these two diameter

$$D(\lambda, x) = l(x, r) \frac{1}{\ln(N)} \quad (10)$$

The alteration in the term  $\ln(N)$  may not initially seem to affect the solution Eq. 10, but allows us to maintain the same equation for the diameter for all scenarios of different sizes of the network.



**Fig. 2.** In (a) we have an example of the changes in the length function with different network sizes and  $\lambda = 6$ , (b) shows the effect of different rates in the length function  $\lambda = 6$  and  $N = 10^5$ .

As shown in Fig. 2(a), the growth of the length function becomes less pronounced as the number of nodes in the network increases. Additionally, by using the variable substitution  $\ln(p) = -x$ , we can determine the percentage of links that remain occupied.

The first intersection always yields more occupied links than the second regardless of the size of the network. Therefore, for the remainder of this work, we will focus solely on the first intersection point.

## 5.2 Maximum Value of the Rate

In the previous section, we deduced the equation for our length function. As stated in Sect. 1 we are interested in seeing the highest rate we can achieve in a network therefore we want to be able to know the highest rate we can achieve. As we can observe, the difference in rates affects the solutions, leading to behavior similar to a change in the slope of a linear equation. Figure 2(b), shows that there will be a point where an intersection no longer exists, we will leverage the linear behavior to find this point. With this behavior we can deduce an equation to find the highest rate achievable, we can do this by finding the point where  $D'(x)x$  is tangent to  $D(x)$ ,

$$D(x) = D'(x)x \iff \frac{D(x)}{x} = \frac{-\epsilon \ln(\gamma_T)\langle n \rangle}{(1-\epsilon)2r \ln(N)}. \quad (11)$$

From Fig. 2 (b) we saw that as our rate gets higher the slope is decreased and for the highest value of the rate we can see that our tangent function has reached the minimum of its slope. So we need to find

$$x_0 = \min_{0 < x \leq x_c} \frac{D(x)}{x}, \quad (12)$$

let's call this minimum  $x_0$ , now we want to know for what value of  $r$  this happens

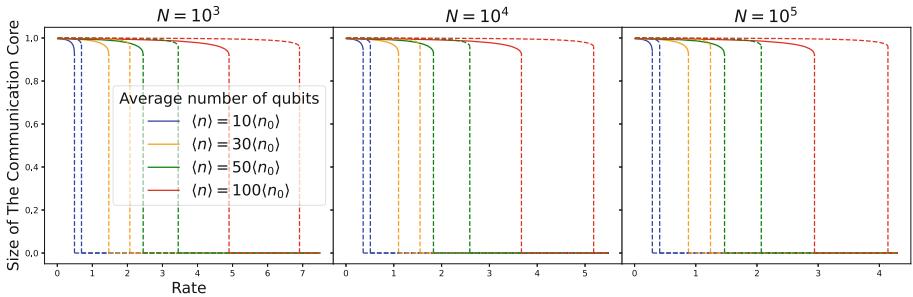
$$x_0 = \frac{-\epsilon \ln(\gamma_T)\langle n \rangle}{(1-\epsilon)2r \ln(N)} \iff r_c = \frac{-\epsilon \ln(\gamma_T)\langle n \rangle}{(1-\epsilon)2x_0 \ln(N)}. \quad (13)$$

With this equation, we can determine the maximum value for  $r$ , this value is found by using an asymptotic length function simplified by using Taylor expansions. The formula is presented in a general form, intended to apply to various network models beyond the (ER) network. In Eq. 12, we introduce an upper limit,  $x_c$ , for the values that  $x$  can take. This upper limit represents the threshold beyond which the network transitions into the disconnected regime. We can visualize this transition in Fig. 2(b) as the diameter reaches its maximum value and then starts transitioning into the disconnected regime.

## 5.3 Theoretical Communication Core

The images in Fig. 3 illustrate the results of constructing the communication core using the described concepts. We analyzed networks of different sizes, average degrees, and average numbers of qubits sent, denoted as  $\langle n \rangle = c\langle n_0 \rangle$ , where  $c \in \{10, 30, 50, 100\}$ . The first set of images shows networks with a Poisson distribution parameter  $\lambda = 6$ , while the second set (dashed) depicts networks with  $\lambda = 8$ .

From these analyses, we observe that the highest achievable rate decreases as the network size increases. In particular, for  $\langle n \rangle = 10\langle n_0 \rangle$ , the average rate does not reach 1, indicating that we are unable to send a single qubit per second successfully. In contrast, for  $\langle n \rangle = 30\langle n_0 \rangle$ , we begin to achieve a rate of at least 1.



**Fig. 3.** Connectivity of the communication core for various rates, network sizes, and average number of qubits for an ER network. The dashed lines represent the scenario with  $\lambda = 8$  while the full lines represent  $\lambda = 6$ . The vertical lines aid the visualization of the discontinuous phase transitions.

This trend aligns with the expectation that larger networks struggle more with achieving end-to-end fidelity, resulting in fewer successfully transmitted qubits. Additionally, as the rate increases, network connectivity decreases, as shown by the shrinking size of the communication core. This outcome is anticipated because maintaining higher qubit rates within the network becomes increasingly challenging.

Within each run, the average number of qubits sent increases, leading to a higher rate.

The images also show that changes in the Poisson parameter have an effect, though not as pronounced as other factors. As the parameter increases, the diameter function shifts to the right and grows more slowly, increasing the highest achievable rate.

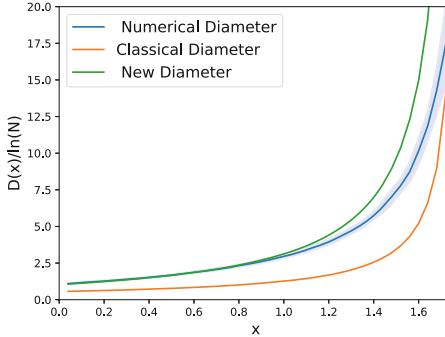
The figures reveal discontinuities in the plots, primarily due to considering only the first connected solution.

#### 5.4 Numerical Communication Cores

The results observed so far are promising and quite interesting. We will now obtain numerical results for the communication cores, which will allow us to introduce the (SF) network model more specifically the Bollobás variant of the Barabási-Albert model. This model was not previously used due to the lack of a closed formula for the diameter. However, using statistical methods, we can estimate the diameter based on the occupied links, providing the diameter size for a specific model in terms of  $\ln(p) = -x$ .

In the following sections, we will present and compare the results from these models, exploring any differences and their causes. We will also compare the ER model results from this chapter with those found in Sect. 5.3. While Eq. 10 provided a diameter formula independent of network size, this does not apply to numerical data. We will investigate any significant differences and focus on rates greater than 1. In Fig. 4, a notable distinction is evident in the diameters that are

used. The conventional equation did not offer as accurate an approximation as the equation seen in 10 when compared to the numerical results we had obtained for the diameter. The new formula itself provided a good approximation of the numerical results, but only up to a certain point. With the assistance of this semi-analytical approach, we can construct a more realistic communication core for our models.

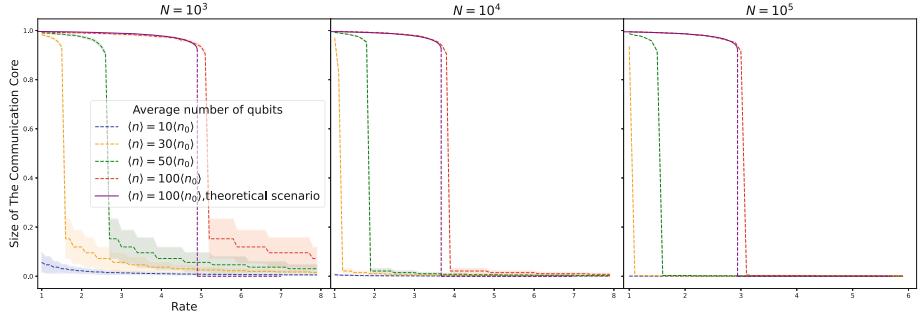


**Fig. 4.** A comparison of the three diameters considered for an ER model with  $N = 10^5$  and  $\lambda = 6$  shows a significant difference between the classical diameter and the one used in this work. However, the numerical and theoretical diameters in this study differ only slightly for small values of  $x$ . The error region in these graphs represents the standard deviation based on 100 independent runs.

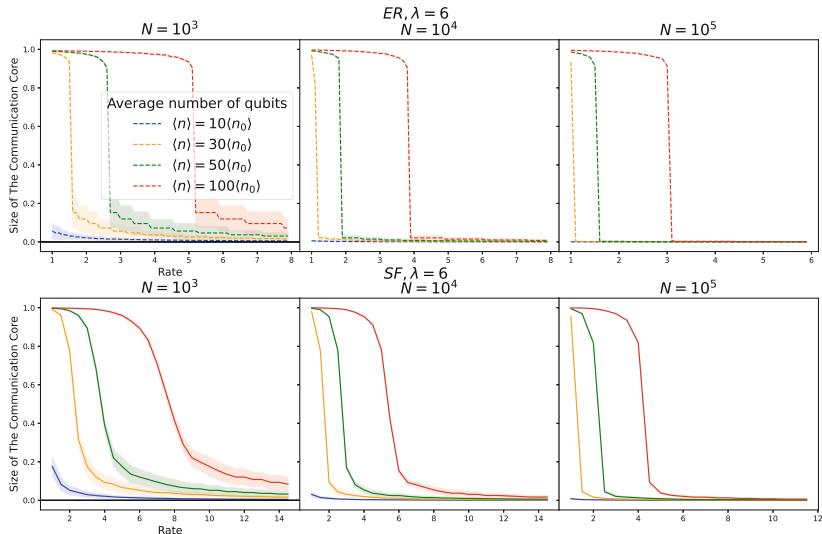
Returning to the ER case and applying the concepts discussed earlier, we obtain the following results as seen in Fig. 5 for the communication core in our model. Upon initial examination, these results differ from the theoretical results. In this particular scenario, when considering an average degree of  $\langle n \rangle = 10\langle n_0 \rangle$ , there is an absence of communication cores in the connected regime. This distinction is attributed to our adoption of a more realistic model, focusing exclusively on rates greater than or equal to 1. Consequently, it becomes apparent that this scenario consistently falls within the disconnected regime.

This adjustment in perspective also has a note-worthy impact primarily on the  $\langle n \rangle = 30\langle n_0 \rangle$  scenario. In the case of a network size of  $N = 10^3$ , a communication core with connectivity still exists. However, for the remaining scenarios with network sizes of  $N = 10^4$  and  $N = 10^5$ , there is a point of high connectivity that diminishes rapidly with a slight increase in the rate.

In contrast, the scenarios  $\langle n \rangle = 50\langle n_0 \rangle$  and  $\langle n \rangle = 100\langle n_0 \rangle$  exhibit subtle deviations from the theoretical case presented in Fig. 3. Despite these apparent distinctions, the conclusions drawn in Sect. 5.3 regarding the connectivity of the communication cores remain unchanged. As stated in Fig. 5, we drew comparisons between the theoretical scenario and the semi-analytical there isn't a significant difference between the two results. As we increase the network size, this slight difference decreases further. For example, when comparing scenarios



**Fig. 5.** A comparison between the theoretical and semi-analytical results for the ER model is made for each respective average number of qubits and sizes. The comparisons focus on  $\langle n \rangle = 100\langle n_0 \rangle$ , where the higher connectivity allows for easier comparisons. This scenario also shows a decrease in differences between the two approaches as the network size increases. The dashed vertical lines illustrate the discontinuous phase transition occurring and the dashed scenarios showcase the semi-analytical scenario. The error region in these graphs represents the standard deviation based on 100 independent runs.



**Fig. 6.** Connectivity of the communication core for various rates, network sizes, and average number of qubits. The first row of the graph is regarding an ER model with  $\lambda = 6$ , the second row is for an SF network with  $\lambda = 6$ . There are clear differences between the two models, where the SF model yields better results and allows to have a connected component for larger values of the rate. The error region in these graphs represents the standard deviation based on 100 independent runs.

with  $N = 10^3$ , the difference between theory and numerical results is noticeable but not pronounced. However, as the network size increases to  $N = 10^5$ , this difference becomes even less significant. If the network size continues to grow, this difference would reduce to nearly zero. The lack of significant difference between both scenarios is primarily due to the minimal difference between the diameter obtained from the data and the theoretical diameter given by Eq. 10.

Similarly, for the SF case, we obtain the following communication core:

We can see that the outcome here allows us to draw the same conclusions and challenges as in Sect. 5.3, it can also be seen that the SF model allows for larger rates than its ER counterpart and the phase transition is less abrupt to the loss of connectivity with an increase in the rate as observed by its smoother decrease compared to the ER model which presents an abrupt change from the connected regime to the disconnected. One of the main reasons for this is the diameters of both models are quite different in terms of their occupied links and the properties of the SF model such as preferential attachment among others (Fig. 6).

## 6 Conclusions

In this work, we examine how the connectivity of a quantum network varies with the target distribution rate, without purification or error correction. We focus on a quantum network where each link is a standard quantum photonic channel, and entanglement is distributed using a conventional swapping operation. We evaluate the network's connectivity based on its “communication core”, considering both Erdős-Rényi (ER) and Scale-Free (SF) network models. We observed the existence of a phase transition, meaning there is a maximum rate beyond which the network's connectivity drops to nearly zero. Our findings reveal that different network structures exhibit varying types of phase transitions in response to changes in target distribution rates. Specifically, the ER network shows a discontinuous phase transition in both theoretical and semi-analytical scenarios across all network parameters considered. In contrast, the SF network is less prone to such discontinuous phase transitions. This suggests that a quantum network with an SF topology is less likely to experience abrupt changes in connectivity compared to an ER network. The entanglement generation and distribution protocol used in this study is based on the current quantum distribution technology being developed today and does not require extensive use of quantum gates. This makes it more likely to be implemented in real-world scenarios within the next few years, though it is known to be inefficient for long-distance entanglement distribution. Our approach is versatile and can be combined with other generation and distribution protocols, paving the way for the study of more complex network models, including those incorporating error correction and purification protocols that significantly enhance entanglement distribution efficiency.

**Acknowledgments.** The authors thank FCT - Fundação para a Ciência e Tecnologia for its support through national funds and, when applicable, co-funded EU funds, via the projects UIDB/50008/2020 with DOI identifier <https://doi.org/10.54499/UIDB/50008/2020>, and project reference QuNetMed 2022.05558.PTDC with DOI identifier <https://doi.org/10.54499/2022.05558.PTDC>.

## References

1. Barabási, A.L., Pósfai, M.: Network Science. Cambridge University Press, Cambridge (2016)
2. Bennett, C.H., Brassard, G.: Quantum cryptography: public key distribution and coin tossing. *Theoret. Comput. Sci.* **560**, 7–11 (2014)
3. Bennett, C.H., Brassard, G., Popescu, S., Schumacher, B., Smolin, J.A., Wootters, W.K.: Purification of noisy entanglement and faithful teleportation via noisy channels. *Phys. Rev. Lett.* **76**(5), 722 (1996)
4. Brand, S., Coopmans, T., Elkouss, D.: Efficient computation of the waiting time and fidelity in quantum repeater chains. *IEEE J. Sel. Areas Commun.* **38**(3), 619–639 (2020)
5. Chakraborty, K., Elkouss, D., Rijssman, B., Wehner, S.: Entanglement distribution in a quantum network: a multicommodity flow-based approach. *IEEE Trans. Quant. Eng.* **1**, 1–21 (2020)
6. Childress, L., Taylor, J., Sørensen, A.S., Lukin, M.D.: Fault-tolerant quantum repeaters with minimal physical resources and implementations based on single-photon emitters. *Phys. Rev. A* **72**(5), 052,330 (2005)
7. Coutinho, B.C., Munro, W.J., Nemoto, K., Omar, Y.: Robustness of noisy quantum networks. *Commun. Phys.* **5**(1), 105 (2022)
8. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC 1996, pp. 212–219. Association for Computing Machinery, New York, NY, USA (1996). <https://doi.org/10.1145/237814.237866>. <https://doi.org/10.1145/237814.237866>
9. Munro, W.J., Azuma, K., Tamaki, K., Nemoto, K.: Inside quantum repeaters. *IEEE J. Sel. Top. Quant. Electron.* **21**(3), 78–90 (2015)
10. Newman, M.: Networks: An Introduction. OUP Oxford (2010)
11. Riordan, O., Wormald, N.: The diameter of sparse random graphs. *Comb. Probab. Comput.* **19**(5–6), 835–926 (2010)
12. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science, pp. 124–134. IEEE (1994)
13. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **41**(2), 303–332 (1999)
14. Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. *Nature* **299**(5886), 802–803 (1982)
15. Wu, A.K., Tian, L., Coutinho, B.C., Omar, Y., Liu, Y.Y.: Structural vulnerability of quantum networks. *Phys. Rev. A* **101**(5), 052,315 (2020)
16. Zhuang, Q., Zhang, B.: Quantum communication capacity transition of complex quantum networks. *Phys. Rev. A* **104**(2), 022,608 (2021)



# Topological Features vs Structural Features in Drug Interactions

Dimitrios Vogiatzis<sup>1,2</sup>(✉)

<sup>1</sup> AI Laboratory, The American College of Greece - Deree, Athens, Greece  
dimitrv@acg.edu

<sup>2</sup> National Centre for Scientific Research, “Demokritos”, Athens, Greece

**Abstract.** We compare two types of topological features on networks to discover commonalities and differences. The first type concerns the well known degree, eigenvector and core centrality features. The second type of features are relatively new in network analysis, and are the product of persistent homology. In particular, we examine the H<sub>0</sub>, H<sub>1</sub>, and H<sub>2</sub> features. H<sub>0</sub> refers to connected components, H<sub>1</sub> to cycles and H<sub>2</sub> to 2-d cycles. The study focuses on comparisons of the two types of features on the ego-network of nodes. The experiments are performed on two biomedical networks, one derived from the Drug-Bank and the other concerns interactions of drugs for rare diseases.

**Keywords:** persistent homology · graph topological features · centrality measures · biomedical networks

## 1 Introduction

The characterization of a network, i.e. the pattern formed by the linking of nodes is very interesting, at very least from the viewpoint of link prediction. In other words the form of data plays a very important role in machine learning. The hypothesis is that in some “regions” carry more information than others, thus affecting any sort of prediction.

It is possible to characterize regions of the network based on various methods such as clique and community detection, k-core decomposition and others. In the current work we examine the application of topological features that stem from algebraic topology and in particular from persistent homology. The idea is that such features discover “structural holes” that persist at different scales and in that sense they are important rather than noise artifacts. The knowledge obtained from such features can contribute into data modeling, graph summarization, and also into feature creation for prediction models.

The other type of features are the widely used in network analysis, i.e. the degree, eigenvector and core centralities. We will name them structural features to distinguish them from the topological features generated by persistent homology. We aim to discover possible correlations between structural and topological features.

The comparison of topological features is motivated by the very practical problem of drug interaction prediction. In that the efficacy of a drug can be improved when co-administered with other drugs [6]. Also, it is very common for patients that suffer from commodities to follow a multiple drug scheme. Yet, there can be adverse effects in those combinations, which occasionally might be toxic. It is an especially serious problem as the number of drugs that people get for a disease continues to increase [7, 11]. While many drug interactions have been discovered, there are potentially new ones that could be predicted with computational methods, before their laboratory confirmation [5]. Thus predicting Drug-Drug Interactions (DDIs) is important for the well-being of patients. The problem of drug interaction prediction with computational methods is usually reduced to the problem of link prediction in a network of interactions.

Naturally the study of different types of topological features and their relations can be generalized to networks that stem from different domains, at least provided they follow near power-law distributions as the current networks under study.

The current work extends a previous work [8] where we tried to discover network regions where drug interaction prediction succeeds or fails. The characterization was in terms of structural and chemical properties of the underlying nodes.

**Contribution.** In this work we make the following contributions:

- Discovered topological features, H0, H1, H2 in two biomedical networks.
- Examined structural versus topological features at the level of individual nodes but also at the level of edges.
- Performed experiments in a network extracted from Drug-Bank, which describes drug-interactions. Also performed experiments on another network that us about drug interactions on Rare-Diseases.

The rest of the paper is structured as follows: In Sect. 2, we present an introduction to Topological Data Analysis (TDA) and in particular to Persistent Homology which is used in the current analysis. In Sect. 3 we provide a literature review on TDA in machine learning. In Sect. 4 we describe the methodology that we followed. A presentation of the datasets is provided in Sect. 5. The experimental results are discussed in Sect. 6 and conclusions and future work suggestions are drawn in Sect. 7.

## 2 Background and Literature Review

*Topological Data Analysis (TDA)* studies the shape and structure of data. It borrows techniques from algebraic topology to extract meaningful insights.

The starting point is a set  $S \subset R^n$  of data points. In TDA “thickening” of a point refers to treating a data point as neighborhood of a certain radius. This is known as the *Victoris-Rips filtration*. The radius is denoted by  $\epsilon$ .

A central concept in TDA is the *simplex*. A simplex in 0 dimensions is a point, in 1 dimension is a line segment, in 2 dimensions is a triangle and in 3 dimensions is a tetrahedron. The simplex definition can be extended to higher dimensions. Based on a Vietoris-Rips filtration, simplices can be produced for various radii. A  $k$  simplex is made of  $k + 1$  distinct vertices. A *d-face* is a subset of the  $k$  vertices of a simplex.  $d$  denotes the dimension of the face. For example, a *0-face* is a vertex, a *1-face* is an edge, a *2-face* is a triangular face. A  $k$ -simplex is bounded by its  $k - 1$  faces, which constitute the boundary of that simplex.

A *simplicial complex* is a combination of simplices. Formally, a set of simplices forms a complex if the following two properties hold:

- Each face of all simplices that make-up the complex should be part of the complex.
- The intersection of a pair of simplices is face that belongs to the two simplices.

In a simplicial complex a *k-chain* is the sum of  $k$ -simplices. A *k-cycle* is a  $k$ -chain with a zero boundary, i.e. a boundary is a closed loop. Generally speaking the boundary of an edge is a map from edges to vertices denoted as:  $\delta C_1 = C_0$ , where  $C_1$  is the set of combinations of directed edges,  $C_0$  is the set of combinations of vertices, and  $\delta$  denotes the mapping.  $C_0$  is the set of 0-dimensional edges,  $C_1$  is the set of 1-dimensional chains. This definition can be extended to higher dimensions also.

A *homology group* denoted by  $H_1$  represents *cycles*, which are chains with zero boundary. In general, the homology groups denoted by  $H_k$  are defined as the group of  $k$ -cycles modulo the  $k$ -boundaries. Formally,

$$H_n = \frac{\ker(\delta_n)}{\text{im}(\delta_{n+1})} \quad (1)$$

where  $\ker$  is the kernel of the boundary map  $\delta_n$  and  $\text{im}$  is the image of the boundary map  $\delta_{n+1}$ . The  $\ker(\delta_n)$  is the set of all  $n$  dimensional holes or cycles in the simplicial complex. The  $\text{im}(\delta_{n+1})$  represents the set of all  $n$ -chains that can be filled in or “bounded” by higher-dimensional simplices.

*Persistent Homology* is a method that analyzes the evolution of topological features in a dataset across different scales. In the context of graphs, persistent homology can be used to study the connectivity and clustering properties of the graph at various resolutions. This is achieved by the Vietoris-Rips filtration for different radii, where homological features are computed for each radius (i.e. H0, H1, H2). In particular different radii change the topology, and as a result the simplicial set. Features that persist over a wide range of “thicknesses” are considered significant and likely represent the underlying structure of the data, while those that appear and disappear quickly are considered noise.

### 3 Literature Review

Topological Data Analysis is essentially a data modeling technique and as such aims to “compress” a data set with a view of providing an understanding of

the data. The field has much older and diverse roots stemming from algebraic topology (see [12] for a quite thorough and theoretical approach.) Recently several useful ideas from algebraic topological have been applied to data analysis, creating the field of Topological Data Analysis (TDA). For instance [4,9], and [10] are introductions aimed at data scientists.

There are already many applications of TDA in data science. A corpus of articles from mathematics and physics was analysed with a view of studying the evolution of conceptual co-occurrences [13]. It was discovered that structural holes tend to represent closely related concepts (from mathematics or physics), whereas the “death” of a hole could potentially represent an advancement in the field. TDA analysis has been used to obtain the shape of data, and subsequently to guide the design of deep learning architectures. This has been applied in image recognition with Convolution Neural Networks [1]. Another application of TDA, closer to the current work, is for drug re-purposing [15]. In that, proteins were represented as graphs, and topological similarities between them were discovered. This is an essential step in finding compatible pairs of proteins to target,,

The benefits of TDA in machine learning has not been fully established yet. In a recent publication, the authors have shown that TDA based models do not outperform graph based machine learning methods [14]. Moreover, computational time in extracting TDA features is significant. However, TDA is robust against outliers, and graph perturbations. The experiments were performed on medical and biological networks.

Concluding, the evidence is that TDA can discover complex patterns in data [3], such as loops and voids in graphs. Moreover, TDA is robust in noisy and high-dimensional data sets [2]. TDA can be used in link prediction in the feature engineering phase. Thus homological features at the level of nodes or edges can be computed. For instance, the number of H0, H1 and H2 features can be computed for a subgraph that is centered around a node. Then the number of each homological feature can be used in prediction models. In such a way TDA can create high level features which can also contribute towards the interpretability of machine learning models.

This paper tries to investigate how TDA and in particular the persistent homological features H0, H1 and H2 are related to more traditional structural network features like node degree, and eigenvector centralities, as well as to k-cores. The experiments were performed on two networks of drug-drug interactions, one from Drug-Bank and the other was about drug interactions for Rare-Diseases.

## 4 Methodology

The aim of this study is to investigate possible relations between topological features derived by persistent homology. But it also aims to discover relations between topological and structural features in two networks related to the biomedical domain.

The *topological features* we focus on are the H0, H1, and H2 that are discovered by persistent homology. H0, H1 and H2 are computed for a whole graph. Naturally, the *birth* and *death* of these features is extracted to denote the “life time” and thus the importance of the said features.

H0 features correspond to connected components, H1 features correspond to cycles (i.e. to paths that start and end at the same node). Finally, H2 correspond to cavities (or voids). There are two remarks to make, first the “lifetime” of the said features is computed. This is important because it represents their significance. Second, initially in a simple graph there are no H2 features. H2 features are built by “surfaces” which are produced at later stages of the TDA feature computation. As mentioned in Sect. 2, the Vietoris-Rips filtration builds an increasing radius around each node. Thus “surfaces” or 2-d cycles can be created in the process, which bound the voids.

We are also interested in computing H0, H1, and H2 for the k-neighborhood of each node also known as the ego-network. The motivation is that such topological features, at the node level can be used in link prediction, and node labeling tasks among others. Finally, we compute the number topological features of the edges by aggregating the topological features of the corresponding nodes, and computing the maximum, minimum, difference and averages. Due to lack of space, the result on edge topological features are not included in the current work.

The *structural features* are node specific, and we considered degree, eigenvector and the k-core centralities. A node belongs to a k-core if it is linked to at least k other nodes.

## 5 Data Acquisition and Pre-processing

The first data set was acquired from Drug-Bank [16] (version 5.1.9), a drug interactions repository that is manually curated. The data included 14,624 drug entries and 1,389,184 unique drug-drug interactions (DDIs), which were used to build the interactions graph. Only the drugs that have at least one known interaction were kept. After the data cleaning, and samping down to facilitate computations, the final graph was comprised of 4,000 nodes and 98,902 edges.

The second data set concerned interactions between drugs for rare diseases and other drugs. First, 214,446 articles were retrieved from PubMed.<sup>1</sup> The articles were related to the rare diseases as specified in the EU funded project SIM-PATHIC.<sup>2</sup> Subsequently the data were analyzed with MetaMap<sup>3</sup> and SemRep<sup>4</sup> to obtain triplets, i.e. named entities and their relation. Then we kept the triplets only with the ‘Interacts-with’ relation. This resulted in a graph with 7,364 nodes

---

<sup>1</sup> <https://pubmed.ncbi.nlm.nih.gov/>.

<sup>2</sup> <https://eatris.eu/projects/simpathic-accelerating-drug-repurposing-for-rare-neurological-neurometabolic-and-neuromuscular-disorders-by-exploiting-similarities-in-clinical-and-molecular-pathology/>.

<sup>3</sup> <https://lhncbc.nlm.nih.gov/ii/tools/MetaMap.html>.

<sup>4</sup> [https://lhncbc.nlm.nih.gov/ii/tools/SemRep\\_SemMedDB\\_SKR/SemRep.html](https://lhncbc.nlm.nih.gov/ii/tools/SemRep_SemMedDB_SKR/SemRep.html).

and 1,642,808 edges. Finally, the data were sampled down to a graph of 2,500 nodes and 187,527 edges. The two data sets as used in the experiments are summarized in Table 1. It should be noted that the density of Drug-Bank graph is 1.3%, whereas the Rare-Diseases is 6%. It should be noted that both graphs are unweighted.

## 6 Experiments and Results

The topological data analysis was performed with the Giotto-tda library<sup>5</sup> while structural data were extracted with the NetworkX library.<sup>6</sup> The number of topological features, as well as their *life span* derived from *persistent homology* for both networks are summarized in Table 2. The table displays the counts of H0 (connected components), H1 (1-d cycles), and H2 (2-d cycles) for each dataset, along with the corresponding *birth* and *death* times of these features. It can be observed that in both data sets the H0 features are born in the beginning. Essentially, each H0 feature is initialized as a node. However as the Vietoris-Rips filtration proceeds and in particular when radius becomes 1 (which corresponds to 1 hop expansion), some features die out because they become on large component. However, there are also other H0 features that never perish, i.e. death time is  $\infty$ . These number 94 and 478 for the Drug-Bank and the Rare-Diseases respectively. They correspond to the connected components of the underlying networks. H1 and H2 features appear later on in the Vietoris-Rips filtration (when radius becomes 1) and they persist.

**Table 1.** Structural features of the data sets

	Drug-Bank	Rare-Diseases
nodes	4,000	2,500
edges	108,107	187,527
connected components	95	479
avg clustering coefficient	0.187	0.403
diameter of largest comp.	2	11
[min degree, max degree]	[0, 2760]	[0, 883]
[min core, max core]	[0, 205]	[0, 252]
[min eigen, max eigen]	[5.68e-27, 0.077]	[7.86e-31, 0.067]

The *ego-network* of each node is obtained, which is essentially all the nodes and corresponding edges that are up to  $l$  hops away, ( $l = 2$  in the current experiments). Subsequently TDA features are computed for each ego-network.

<sup>5</sup> <https://giotto-ai.github.io/gtda-docs/0.5.1/library.html>.

<sup>6</sup> <https://networkx.org/>.

**Table 2.** Birth and death for H0, H1, and H2

#	#(birth, death)	#(birth, death)
Drug-Bank		
H0 = 3999	3905 (0,1)	94 (0,∞)
H1 = 187	187 (1,∞)	
H2 = 35	35 (1,∞)	
Rare-Diseases		
H0 = 2499	2021 (0,1)	478 (0,∞)
H1 = 770	770 (1,∞)	
H2 = 115	115 (1,∞)	

In the first experiment, we explored the *pearson correlation* of the number of structural features per node with the number of topological features for the Drug-Bank and Rare-Diseases. The results are displayed in Tables 3 and 4 respectively. It should be noted that in the sparser Drug-Bank graph there are negative correlations, notably among H0 with H1 and H2. On the other hand in the much denser Rare-Diseases set, all pairs of features are positively correlated. The two networks differ apart from the density also in terms of the content of their connected components. In Drug-Bank 2.3% of the nodes are in connected components of size 1, compared to 17.24% of the nodes in Rare-Diseases.

**Table 3.** Drug-Bank: Centralities and TDA feature correlations for nodes

	Degrees	Eigen	Cores	H0	H1	H2
Degrees		0.92	0.88	-0.60	0.86	0.82
Eigen	0.92		0.95	-0.63	0.90	0.87
Cores	0.88	0.94	—	-0.73	0.92	0.91
H0	-0.60	-0.63	-0.73		-0.73	-0.73
H1	0.86	0.90	0.92	-0.73		0.93
H2	0.82	0.87	0.91	-0.73	0.93	

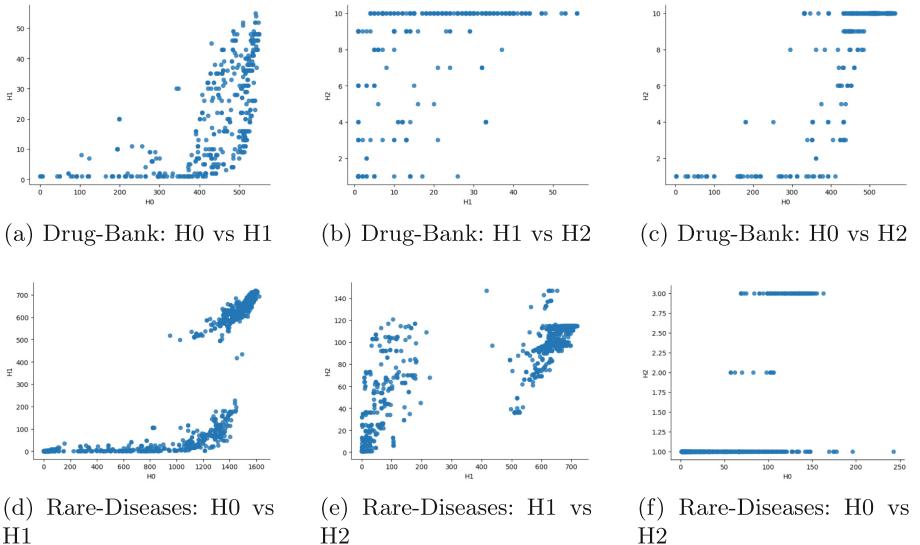
In Fig. 1 the *interrelations* of H0, H1, and H2 features are displayed. The pattern is quite similar for the H0 vs H1, and for the H0 vs H2. With significant differences in the H1 vs H2.

In Fig. 2 the *eigen vector* centrality versus the number of H0, H1 and H2 features is displayed. In both data sets the number of H0 features remains almost constant irrespective of the eigenvector centrality, with the exception of small values of the eigenvector centrality. The H1, which essentially represents cycles, varies among the two data sets.

Finally, regarding the H2 features, which represent 2-d cycles, both data sets exhibit a similar pattern. In particular, after a certain eigenvector centrality

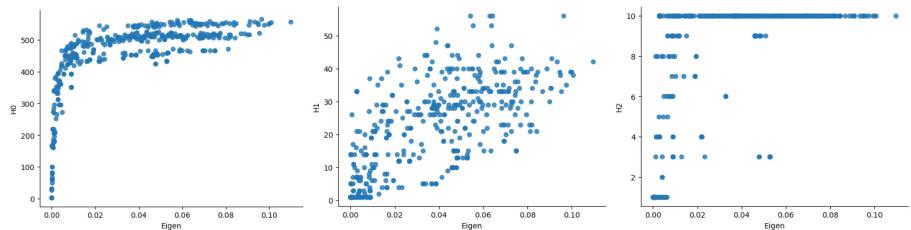
**Table 4.** Rare Diseases: Centralities and TDA feature correlations for nodes

	Degrees	Eigen	Cores	H0	H1	H2
Degrees		0.99	0.91	0.72	0.79	0.81
Eigen	0.99		0.92	0.71	0.81	0.81
Cores	0.91	0.92		0.83	0.87	0.91
H0	0.72	0.71	0.83		0.81	0.89
H1	0.79	0.81	0.87	0.81		0.88
H2	0.81	0.81	0.91	0.89	0.88	

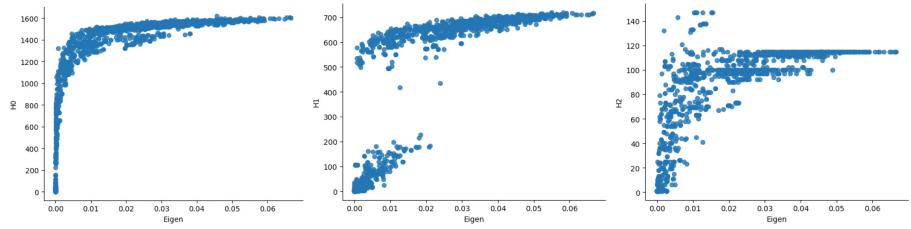
**Fig. 1.** Drug-Bank, Rare-Diseases: Topological features relations

threshold the number of H2 features is almost constant. It should be noted the rare diseases set has far more H2 features.

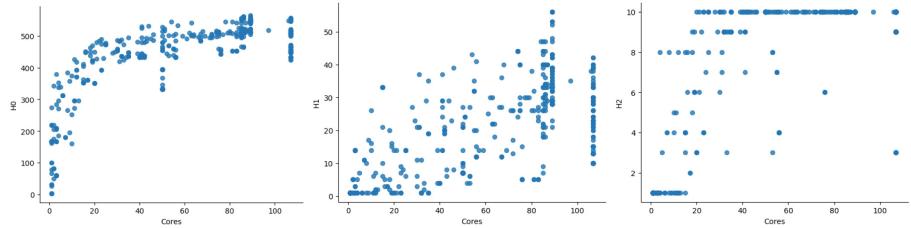
In Fig. 3 the *core centrality* versus the number of H0, H1 and H2 features is displayed. In both data sets the number of H0 features with respect to core centrality exhibit a vey similar pattern. On the other hand, H1 features exhibit a different pattern. But H2 features are quite similar. First it should be noted again that in Rare-Diseases there are more H2 features, and in addition the bulk of H2 is concentrated after a certain level of core centrality.



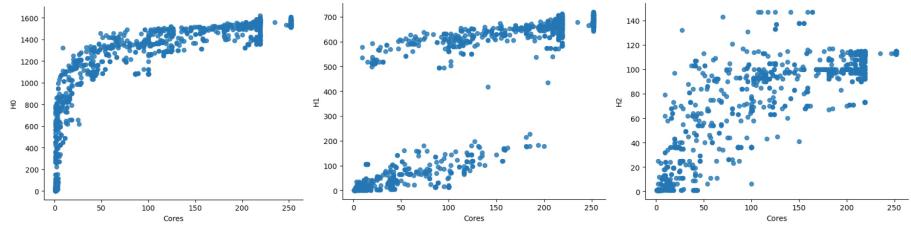
(a) Eigen vs H0 node features (b) Eigen vs H1 node features (c) Eigen vs H2 node features



(d) Eigen vs H0 node features (e) Eigen vs H1 node features (f) Eigen vs H2 node features

**Fig. 2.** Drug-Bank & Rare-Diseases: TDA features vs Eigenvector Centrality

(a) Cores vs H0 node features (b) Cores vs H1 node features (c) Cores vs H2 node features



(d) Cores vs H0 node features (e) Cores vs H1 node features (f) Cores vs H2 node features

**Fig. 3.** Drug-Bank & Rare-Diseases: TDA features vs Core Centrality

## 7 Conclusions and Future Work

In the current work we compared different structural and topological features on two networks derived from biomedical data, i.e. from Drug-Bank and from Rare-

Diseases. The motivation was to find interesting and interpretable features that may ultimately facilitate the problem of link prediction or node characterization.

On the one hand, we examined well established structural features, like the eigen-vector and the core centralities. On the other hand, we extracted features based on persistent homology, a topological data analysis technique. The focus was to discover commonalities in topological features among different graphs and relations between structural and topological features.

Certain homology features, in particular the H0 and H2 have a common pattern in the two data sets we examined. Also the eigen-vector centrality and the core centralities when compared against a common relation to H0 and H2 seem to have a common pattern in the two data sets.

A limiting factor in the experiments was the computation time of topological features with the giotto-tda library. Also, the NetworkX library is expected to present limitations once the networks become large enough.

Another issue that needs to be addressed in future is the existence of multi-relational networks in the biomedical domain. Normally, TDA features are computed on single relational networks and multi-relational networks could be handled as a collection of single relational networks. However there is a line of work that tries to cope with the correlations of the different relations [17].

**Acknowledgements.** The author would like to thank the RTIN at the American College of Greece, Deree for their support.

## References

1. Carlsson, G., Gabrielsson, R.B.: Topological Approaches to Deep Learning. In: Baas, N.A., Carlsson, G.E., Quicke, G., Szymik, M., Thaule, M. (eds.) Topological Data Analysis. AS, vol. 15, pp. 119–146. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-43408-3\\_5](https://doi.org/10.1007/978-3-030-43408-3_5)
2. Carlsson, G., Vejdemo-Johansson, M.: Topological Data Analysis with Applications. Cambridge University Press (2021)
3. Chazal, F., Cohen-Steiner, D., Guibas, L.J., Oudot, S.: The stability of persistence diagrams revisited (2008)
4. Chazal, F., Michel, B.: An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Front. Artif. Intell.* **4**, 667,963 (2021)
5. Cheng, F., Kovács, I.A., Barabási, A.L.: Network-based prediction of drug combinations. *Nat. Commun.* **10**(1), 1–11 (2019)
6. Crystal, A.S., et al.: Patient-derived models of acquired resistance can identify effective drug combinations for cancer. *Science* **346**(6216), 1480–1486 (2014)
7. Kantor, E.D., Rehm, C.D., Haas, J.S., Chan, A.T., Giovannucci, E.L.: Trends in prescription drug use among adults in the united states from 1999–2012. *JAMA* **314**(17), 1818–1830 (2015)
8. Kefalas, G., Vogiatzis, D.: Network structure versus chemical information in drug-drug interaction prediction. In: International Conference on Complex Networks and Their Applications, pp. 402–414. Springer (2022)
9. Munch, E.: A user’s guide to topological data analysis. *J. Learn. Anal.* **4**(2), 47–61 (2017)

10. Postol, M.S.: Algebraic topology for data scientists. arXiv preprint [arXiv:2308.10825](https://arxiv.org/abs/2308.10825) (2023)
11. Qato, D.M., Wilder, J., Schumm, L.P., Gillet, V., Alexander, G.C.: Changes in prescription and over-the-counter medication and dietary supplement use among older adults in the united states, 2005 vs 2011. *JAMA Intern. Med.* **176**(4), 473–482 (2016)
12. Rotman, J.J.: An iNtroduction to Algebraic Topology, vol. 119. Springer Science & Business Media (2013). <https://doi.org/10.1007/978-1-4612-4576-6>
13. Salnikov, V., Cassese, D., Lambiotte, R., Jones, N.S.: Co-occurrence simplicial complexes in mathematics: identifying the holes of knowledge. *Appl. Netw. Sci.* **3**(1), 1–23 (2018). <https://doi.org/10.1007/s41109-018-0074-3>
14. Taiwo, F.M., Islambekov, U., Akcora, C.G.: Explaining the power of topological data analysis in graph machine learning. arXiv preprint [arXiv:2401.04250](https://arxiv.org/abs/2401.04250) (2024)
15. Tarín-Pelló, A., Suay-García, B., Forés-Martos, J., Falcó, A., Pérez-Gracia, M.T.: Computer-aided drug repurposing to tackle antibiotic resistance based on topological data analysis. *Comput. Biol. Med.* **166**, 107,496 (2023). <https://doi.org/10.1016/j.combiomed.2023.107496>. URL <https://www.sciencedirect.com/science/article/pii/S0010482523009617>
16. Wishart, D.S., et al.: DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Res.* **46**(D1), D1074–D1082 (2017). <https://doi.org/10.1093/nar/gkx1037>. <https://doi.org/10.1093/nar/gkx1037>
17. Yuvaraj, M., Dey, A.K., Lyubchich, V., Gel, Y.R., Poor, H.V.: Topological clustering of multilayer networks. *Proc. Nat. Acad. Sci.* **118**(21), e2019994,118 (2021)

# Author Index

## A

- Agarwal, Nitin 233, 271  
Akgun, Mehmet Burak 406  
Alharbi, Abdulrahman 320  
Aljurbua, Rafaat 320  
Alshehri, Jumanah 320  
Al-Taweel, Ahmed 233  
Amure, Ridwan 271  
Anne, Lahari 166  
Aritenang, Adiwan Fahlan 342  
Atas, Furkan 406

## B

- Bagheri, Meshkat Shariat 3  
Banerjee, Dip Sankar 127  
Bernad, Andhika 309  
Brayner, Angelo 140

## C

- Capozzi, Arthur 29  
Carvalho, Álvaro 115  
Chacko, George 103, 166  
Chalehchaleh, Razieh 258  
Chen, Tzu-Yi 294  
Chepovskiy, A. A. 83  
Cinelli, Matteo 282  
Cohen, Sarel 207  
Coutinho, Bruno 418  
Crespi, Noel 258

## D

- Davardoust, Hadi 3  
de Vink, Elze 43  
Digra, Siya 103

## E

- Eshun, Jasmine 246

## F

- Farahbakhsh, Reza 258  
Feng, Daniel Wang 103  
Ficara, Annamaria 181  
Filho, Raimir Holanda 115  
Fionda, Valeria 221  
Fornasiero, Marco 29  
Frost, H. Robert 394  
Fushimi, Takayasu 194

## G

- Ganzach, Yoav 207  
Ghassemi Parsa, Mohsen 3  
Ghosh, Shrabani 71  
Gupta, Shelly 320

## H

- Hu, Jiarui 294

## I

- Ikeda, Nobutoshi 381

## J

- Javaid, Azka 394

## K

- Kaufmann, Marc 366  
Keiluhu, Fausto 309  
Kobti, Ziad 16  
Koike, Rinto 194  
Kothapalli, Kishore 127  
Kour, George 207

## L

- Licari, Claudia 181

## M

- Madi, Saif Aldeen 152  
Magnani, Matteo 91  
Marco, Niccolò Di 282

Maulana, Ardian 309, 342  
Menezes, Ronaldo 140  
Moncalvo, Dario 29  
Moraes, Gustavo 140

**N**  
Ng, Kin Wai 246

**O**  
Obradovic, Zoran 320

**P**  
Panayiotou, Georgios 91  
Park, Minhyuk 57, 103, 166  
Pecile, Giulio 282  
Pirrò, Giuseppe 152  
Polishchuk, Olexandr 353  
Popov, V. A. 83  
Pradono 342

**Q**  
Quadri, Abiodun 233  
Quattrociocchi, Walter 282  
Quintas, Ricardo 418

**R**  
Rahmatikargar, Bahareh 16  
Ravi, Raghu Raman 366  
Ricci, Valeria 29  
Rinaldi, Gabriele 181

Ronchiadin, Silvia 29  
Ruffo, Giancarlo 29

**S**  
Sahu, Subhajit 127  
Sakiyama, Tomoko 335  
Saldanha, Emily 246  
Satel, Sanad 207  
Saxena, Akrati 43  
Schaller, Ulysse 366  
Situngkir, Hokky 309, 342

**T**  
Tabatabaei, Yasamin 57  
Takahara, Akihiro 335  
Toyama, Takumu 194

**V**  
Vilella, Salvatore 29  
Vogiatzis, Dimitrios 430  
Vu-Le, The-Anh 103, 166

**W**  
Warnow, Tandy 57, 103, 166  
Wedell, Eleanor 57  
Wendt, Nathan 246

**Z**  
Zade, Hossein Rafiee 3  
Zadeh, Pooya Moradian 16  
Zare, Hadi 3  
Zinger, Eyal 207