SG Final Symposium
The Complexity of Social and Economic Systems:
From Models to Measures

What makes teams successful?

From Network Science to Causal Graph Learning

Prof. Dr. Ingo Scholtes

Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science
Julius-Maximilians-Universität Würzburg
ingo.scholtes@uni-wuerzburg.de

Chair of Machine Learning for Complex Networks

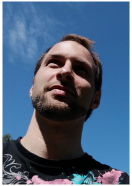
Franziska Heeg

Moritz Lampert

Jan von Pichowski

Lisi Qarkaxhija

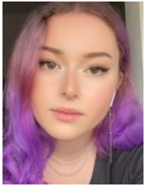
Chester Tan



Dr. Christopher Blöcker

Dr. Vincenzo Perri

Dr. Anatol Wegner



Lola Kohl

Prof. Dr. Ingo Scholtes

What makes teams successful?

- relevant for organizational psychology, software engineering, complex systems theory and industry

image credit: DALL-E generated image

What makes teams successful?

- relevant for organizational psychology, software engineering, complex systems theory and industry
- how can we measure, model, and predict **collective phenomena** in complex social systems?

Max Weber
1864 – 1920

> "Die zunehmende Intellektualisierung und Rationalisierung bedeutet [...] den Glauben daran [...] daß man [...] alle Dinge – im Prinzip – **durch Berechnen beherrschen** könne. Das aber bedeutet: die **Entzauberung der Welt**." ⟶ M Weber: "Wissenschaft als Beruf", 1917

image credit: Ernst Gottmann, Wikimedia Commons, public domain

What makes teams successful?

- relevant for organizational psychology, software engineering, complex systems theory and industry
- how can we measure, model, and predict **collective phenomena** in complex social systems?
- since 1980s: **agent-based models** of collective dynamics in biological, social, and economic systems

Frank Schweitzer
ETH Zürich

The resulting **systemic behavior** [...] often shows consequences that are **hard to predict** [...] we need a more fundamental insight into the **system's dynamics** and how they can be traced back to the structural properties of the underlying interaction network.

→ F Schweitzer et al: "Economic Networks: The New Challenges", Science, 2009

What makes teams successful?

- relevant for organizational psychology, software engineering, complex systems theory and industry
- how can we measure, model, and predict **collective phenomena** in complex social systems?
- since 1980s: **agent-based models** of collective dynamics in biological, social, and economic systems
- since 2000s: focus on **complex networks** of interactions between agents

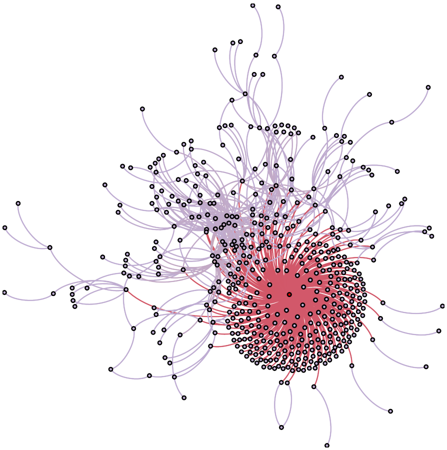
Frank Schweitzer
ETH Zürich

The resulting systemic behavior [...] often shows consequences that are hard to predict [...] we need a more fundamental insight into the system's dynamics and how they can be traced back to the **structural properties of the underlying interaction network**.

→ F Schweitzer et al: "Economic Networks: The New Challenges", Science, 2009

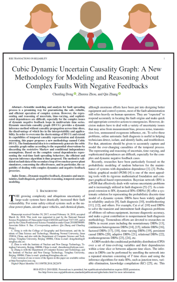
What makes teams successful?

- relevant for organizational psychology, software engineering, complex systems theory and industry
- how can we measure, model, and predict **collective phenomena** in complex social systems?
- since 1980s: **agent-based models** of collective dynamics in biological, social, and economic systems
- since 2000s: focus on **complex networks** of interactions between agents
- since 2010s: application of **machine learning** to complex networks



complex **collaboration network**

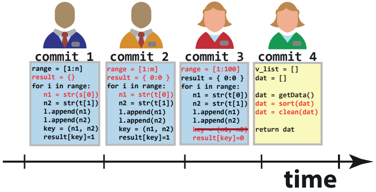
Identifying social factors of “success”



Cubic Dynamic Uncertain Causality Graph: A New Methodology for Modeling and Reasoning About Complex Faults With Negative Feedbacks

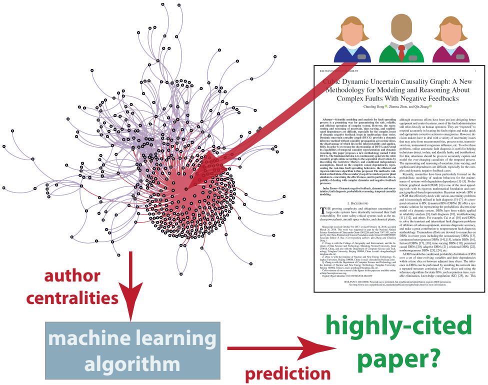
Chunling Dong, Zhenxu Zhou, and Qin Zhang

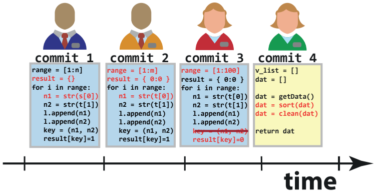
highly-cited paper?



efficient software team?

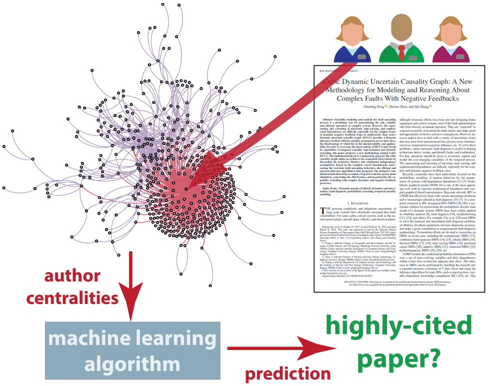
Identifying social factors of “success”

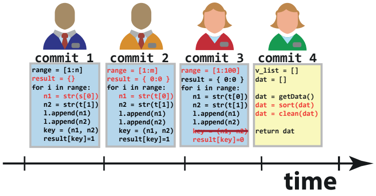




efficient software team?

Identifying social factors of “success”





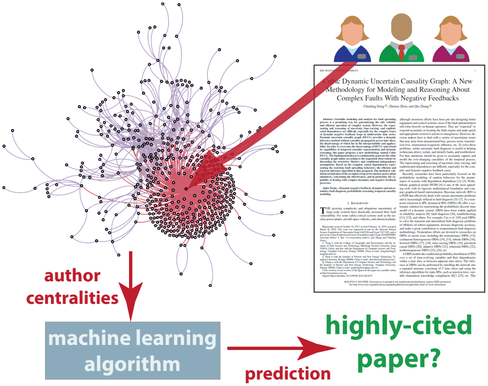
efficient software team?

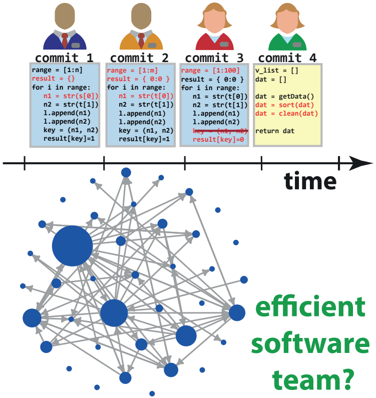
result

authors’ position in collaboration network allows to **predict future citation success of paper** six times better than expected at random

⟶ E Sarigöl, R Pfitzner, I Scholtes, A Garas, F Schweitzer, EPJ Data Science, 2014

Identifying social factors of “success”





result

authors’ position in collaboration network allows to **predict future citation success of paper** six times better than expected at random

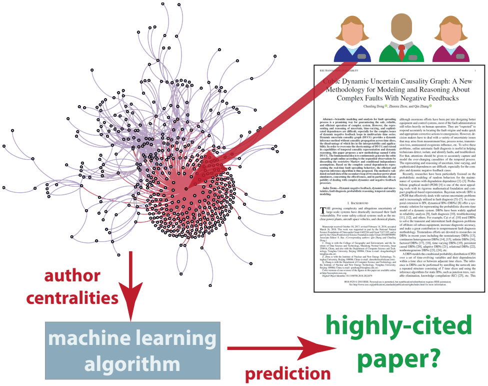
⟶ E Sarigöl, R Pfitzner, I Scholtes, A Garas, F Schweitzer, EPJ Data Science, 2014

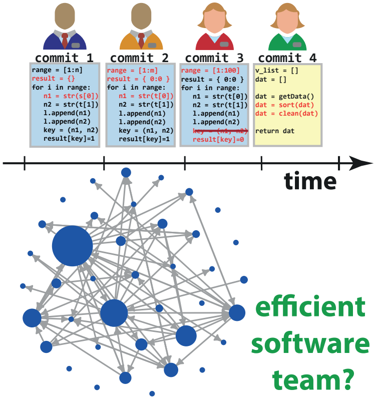
result

structure of coordination network among developers allows to **explain productivity differences across software teams**

⟶ I Scholtes, P Mavrodiev, F Schweitzer, Emp. Softw. Eng., 2016

Identifying social factors of "success"

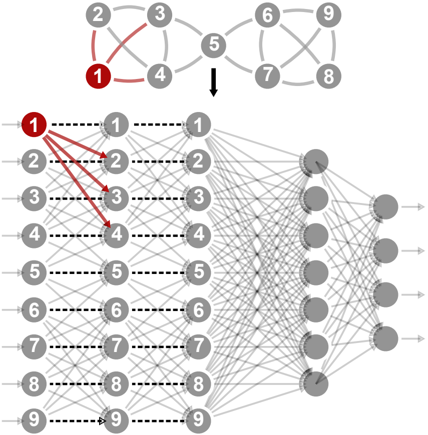




open questions

- can we use **end-to-end deep learning** to model social factors of success in teams?
- how can we leverage high-resolution data on **dynamic collaboration networks**?

Deep learning in complex networks



- **graph convolutional network (GCN)** = neural network architecture for **graph-structured data**

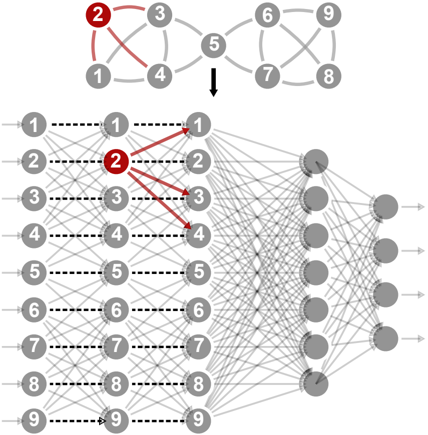
  → T Kipf, M Welling, 2017

Deep learning in complex networks

- **graph convolutional network (GCN)** = neural network architecture for **graph-structured data**

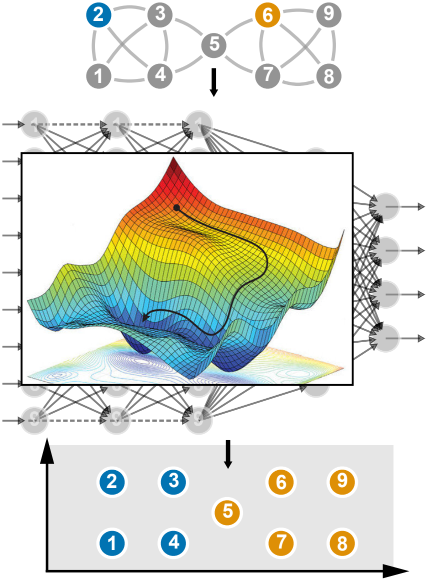
  → T Kipf, M Welling, 2017

- **neural message passing**: use complex network to iteratively update node features based on
  1. differentiable function with (learnable) parameters
  2. neighbor aggregation function
  3. non-linear activation function



Deep learning in complex networks

- **graph convolutional network (GCN)** = neural network architecture for **graph-structured data**
  → T Kipf, M Welling, 2017
- **neural message passing**: use complex network to iteratively update node features based on
  1. differentiable function with (learnable) parameters
  2. neighbor aggregation function
  3. non-linear activation function



end-to-end representation learning

- use **differentiable loss function** to compare model output to ground truth (supervised setting)
- partial derivatives w.r.t. model parameters yield **gradients** that point towards local minimum of loss function
- GPU-accelerated **backpropagation algorithm** to learn “useful” **vector space representation**
  → DE Rumelhart, GE Hinton, RJ Williams, Nature, 1986

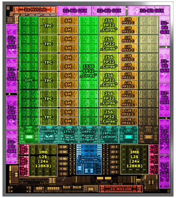
Deep learning in complex networks

- **graph convolutional network (GCN)** = neural network architecture for **graph-structured data**
  → T Kipf, M Welling, 2017
- **neural message passing**: use complex network to iteratively update node features based on
  1. differentiable function with (learnable) parameters
  2. neighbor aggregation function
  3. non-linear activation function

end-to-end representation learning

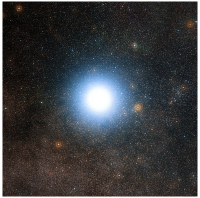
- use **differentiable loss function** to compare model output to ground truth (supervised setting)
- partial derivatives w.r.t. model parameters yield **gradients** that point towards local minimum of loss function
- GPU-accelerated **backpropagation algorithm** to learn "useful" **vector space representation**
  → DE Rumelhart, GE Hinton, RJ Williams, Nature, 1986

Geoffrey E. Hinton
Nobel prize in physics 2024



Nvidia G102 GPU
28.3 billion transistors
40 TeraFLOPs

image credit: Tom's Hardware, Fritzchens Fritz

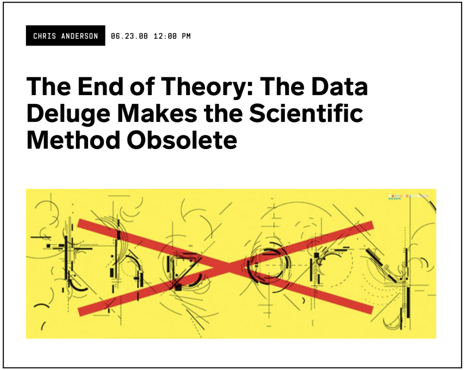


Alpha Centauri
distance approx.
40 billion km

image credit: ESO/DSS 2, CC-BY-SA

The end of theory?

- good **machine learning models** …
  - capture relevant patterns in data
  - **generalize** to unseen data



CHRIS ANDERSON 06.23.08 12:00 PM

The End of Theory: The Data Deluge Makes the Scientific Method Obsolete

"The scientific method is built around testable hypotheses. […] This is the way science has worked for hundreds of years. But **faced with massive data, this approach to science - hypothesize, model, test - is becoming obsolete**."

→ C Anderson: "The End of Theory: The Data Deluge Makes the Scientific Method Obsolete", Wired, 2008

The end of theory?

- good **machine learning models** ...
  - capture relevant patterns in data
  - **generalize** to unseen data
- good **scientific theories** ...
  - describe relevant observed phenomenon
  - make predictions that can be validated
  - help to **understand causal mechanisms**

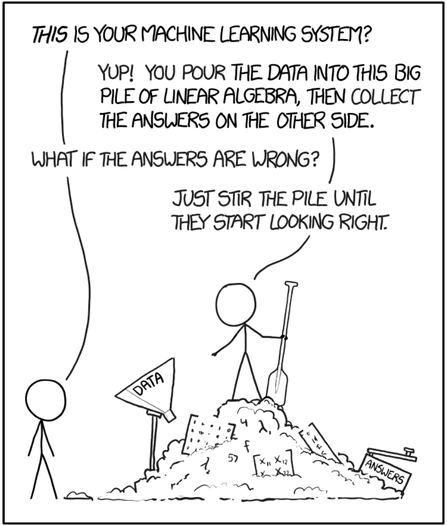


image credit: xkcd.com, Randall Munroe, CC-BY-SA

The end of theory?

- good **machine learning models** …
  - capture relevant patterns in data
  - **generalize** to unseen data
- good **scientific theories** …
  - describe relevant observed phenomenon
  - make predictions that can be validated
  - help to **understand causal mechanisms**
- grand challenge: incorporate **causality** in deep (graph) learning models

Bernhard Schölkopf
MPI for Intelligent Systems

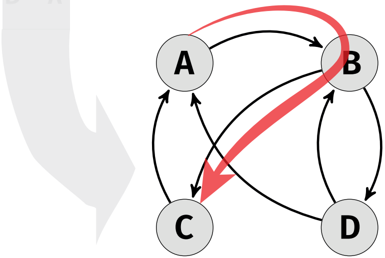
> […] if we compare what machine learning can do to what animals accomplish, we observe that the former is rather bad at some crucial feats where animals excel. […] **causality** […] can make a substantial contribution towards understanding and resolving these issues and thus **take the field to the next level**.
>
> → B Schölkopf: "Causality for Machine Learning", 2019

image credit: Herlinde Koelbl, MPI Tübingen

The arrow of time in networks

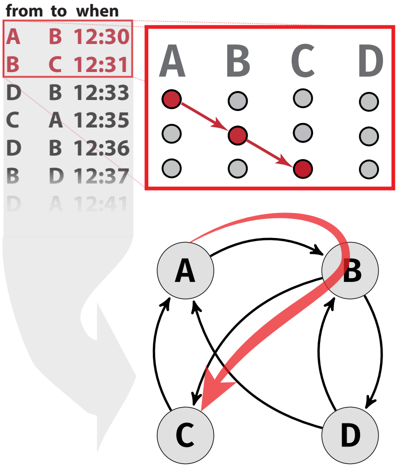
- network science maps and analyzes topology of **possible causal relations** between agents in complex systems
- neural message passing in GCN uses **all possible paths**

from	to
A	B
B	C
D	B
C	A
D	B
B	D
D	A



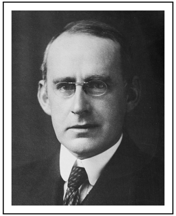
The arrow of time in networks

- network science maps and analyzes topology of **possible causal relations** between agents in complex systems
- neural message passing in GCN uses **all possible paths**



The arrow of time in networks

- network science maps and analyzes topology of **possible causal relations** between agents in complex systems
- neural message passing in GCN uses **all possible paths**
- <u>but:</u> **cause must temporally preceed effects**



Sir Arthur Stanley Eddington

1882 – 1944

> " I shall use the phrase **'time's arrow'** to express this one-way property of time which has **no analogue in space**." → Sir Arthur Eddington

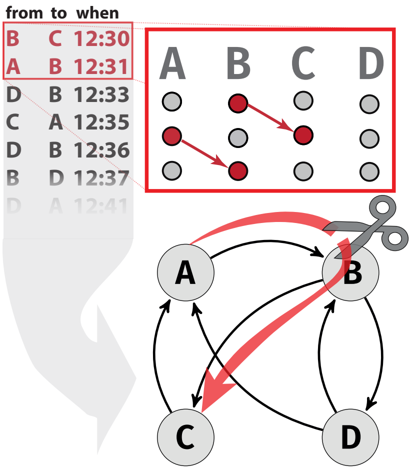


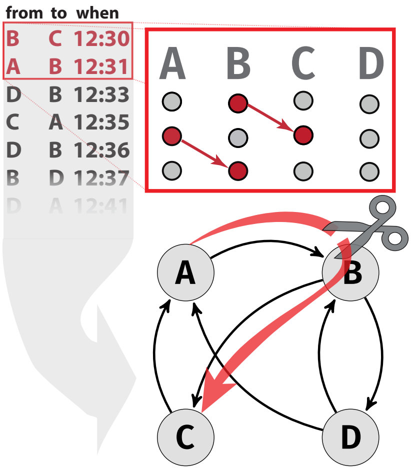
image credit: public domain

The arrow of time in networks

- network science maps and analyzes topology of **possible causal relations** between agents in complex systems
- neural message passing in GCN uses **all possible paths**
- <u>but:</u> **cause must temporally preceed effects**

Networks, time, and causality at the Chair of Systems Design

- temporal correlation measure → R Pfitzner et al., PRL 2013
- predicting diffusion speed → I Scholtes et al., Nature Comm 2014
- temporal centralities → I Scholtes, N Wider, A Garas, EPJ B 2016
- multi-order model selection → I Scholtes, SIGKDD 2017
- anomaly detection for temporal data → T LaRock et al., SIAM Data Mining 2020
- controllability of temporal networks → Y Zhang et al., JoP Complexity 2021
- generative models for path data → C Gote et al., Applied Network Science 2023

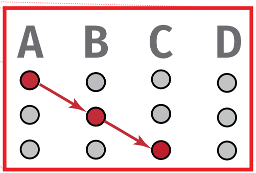


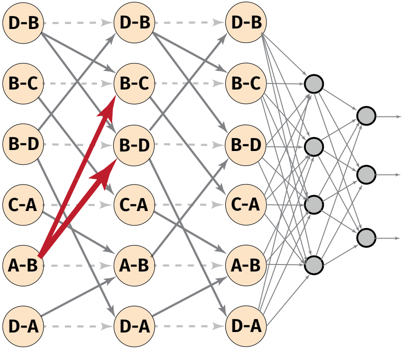
state-of-the-art (temporal) graph neural networks
ignore arrow of time in time series data

Towards deep “causal” graph learning

- **De Bruijn graph neural network (DBGNN)** = deep learning architecture using **higher-order De Bruijn graphs**

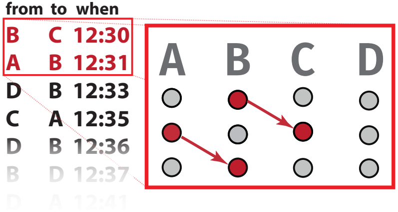
from	to	when
A	B	12:30
B	C	12:31
D	B	12:33
C	A	12:35
D	B	12:36
B	D	12:37

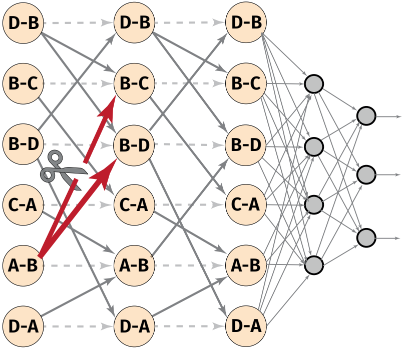




Towards deep "causal" graph learning

- **De Bruijn graph neural network (DBGNN)** = deep learning architecture using **higher-order De Bruijn graphs**
- <u>idea:</u> use neural message passing, but **restrict messages to follow arrow of time**
- we use statistical learning to infer **parsimonious message passing architecture**
  - ⟶ I Scholtes, SIGKDD 2017
  - ⟶ L Petrovic, I Scholtes, WWW 2022
  - ⟶ J von Pichowski, V Perri, L Qarkaxhija, I Scholtes, arXiv 2406.16552





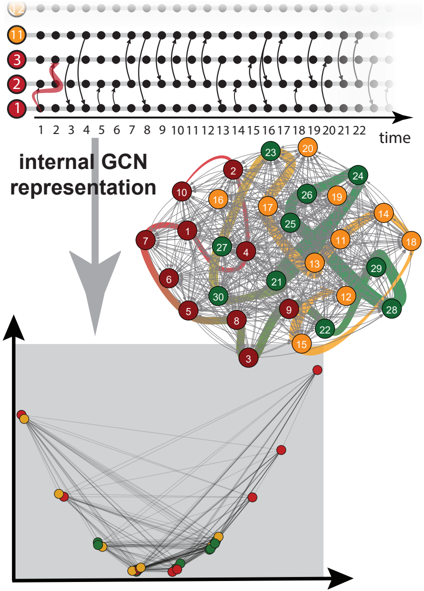
Towards deep "causal" graph learning

- **De Bruijn graph neural network (DBGNN)** = deep learning architecture using **higher-order De Bruijn graphs**
- idea: use neural message passing, but **restrict messages to follow arrow of time**
- we use statistical learning to infer **parsimonious message passing architecture**

  → I Scholtes, SIGKDD 2017

  → L Petrovic, I Scholtes, WWW 2022

  → J von Pichowski, V Perri, L Qarkaxhija, I Scholtes, arXiv 2406.16552



causality-aware graph representation learning

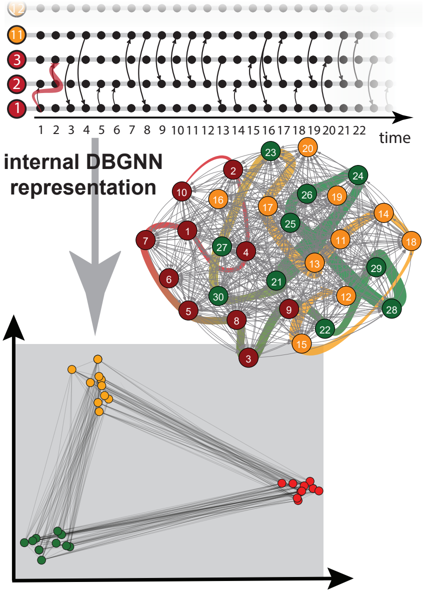
- gradient descent optimization yields **static vector space representation of temporal network** that captures ...
  - topology of interactions between nodes

Towards deep “causal” graph learning

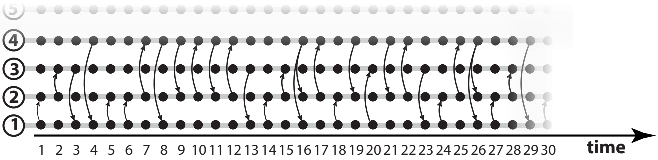
- **De Bruijn graph neural network (DBGNN)** = deep learning architecture using **higher-order De Bruijn graphs**
- idea: use neural message passing, but **restrict messages to follow arrow of time**
- we use statistical learning to infer **parsimonious message passing architecture**
  → I Scholtes, SIGKDD 2017
  → L Petrovic, I Scholtes, WWW 2022
  → J von Pichowski, V Perri, L Qarkaxhija, I Scholtes, arXiv 2406.16552

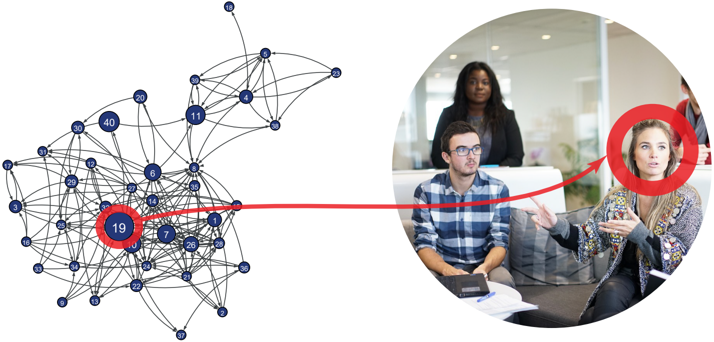
causality-aware graph representation learning

- gradient descent optimization yields **static vector space representation of temporal network** that captures …
  - topology of interactions between nodes
  - “causality” due to temporal order of interactions
- increases node classification performance by **up to 22 %** compared to state-of-the-art → L Qarkaxhija, V Perri, I Scholtes, PMLR 2022



Who is “important” in a team?

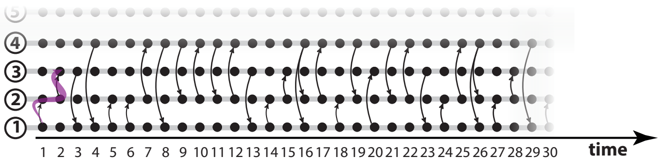


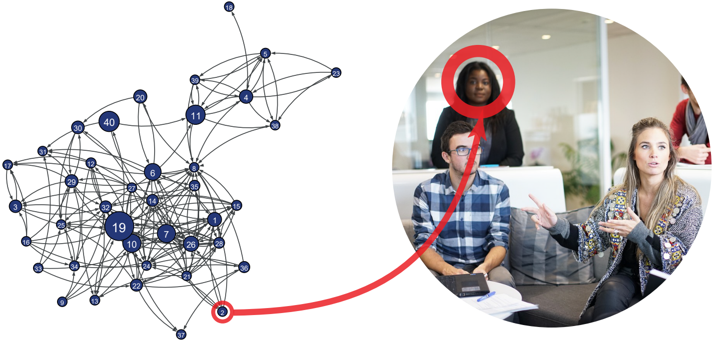


challenge

- **temporal node centralities** substantially differ from static centrality measures

Who is “important” in a team?



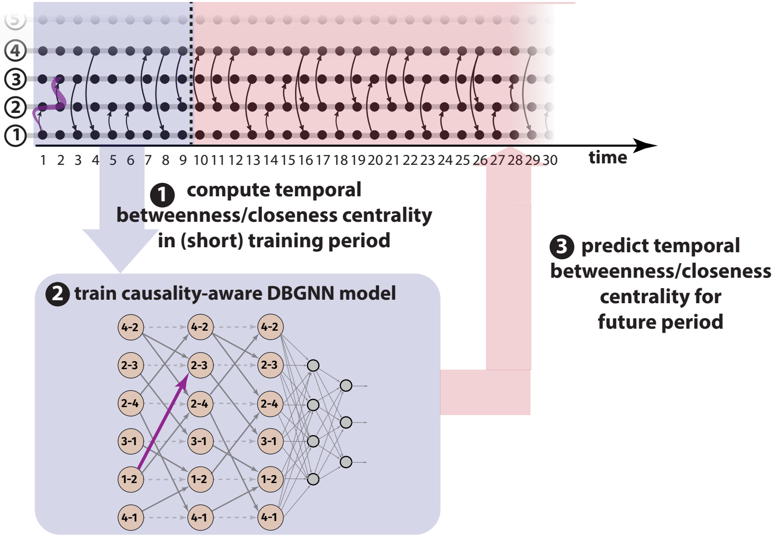


challenge

- **temporal node centralities** substantially differ from static centrality measures
- but: computing **temporal centralities** is **prohibitively expensive**

example: 2,247 s for temporal betweenness in data set with 327 nodes and 188,000 time-stamped edges

Who is "important" in a team?



challenge

- **temporal node centralities** substantially differ from static centrality measures
- but: computing **temporal centralities** is **prohibitively expensive**

example: 2,247 s for temporal betweenness in data set with 327 nodes and 188,000 time-stamped edges

idea

train causality-aware DBGNN model for **regression of temporal centralities**

Who is "important" in a team?

dataset	model	temporal betweenness Spearman	temporal betweenness Speedup	temporal closeness Spearman	temporal closeness Speedup
sociopatterns hospital	GCN[1]	0.804		0.744	
	TGN[2]	0.522		0.509	
sociopatterns hypertext	GCN[1]	0.786		0.809	
	TGN[2]	0.260		0.360	
sociopatterns highschool	GCN[1]	0.540		0.540	
	TGN[2]	0.166		0.166	
manufacturing email	GCN[1]	0.404		0.556	
	TGN[2]	0.320		0.496	

challenge

- **temporal node centralities** substantially differ from static centrality measures
- but: computing **temporal centralities** is **prohibitively expensive**

example: 2,247 s for temporal betweenness in data set with 327 nodes and 188,000 time-stamped edges

idea

train causality-aware DBGNN model for **regression of temporal centralities**

1 → T Kipf, M Welling, ICLR, 2017 2 → E Rossi et al., arXiv:2006.10637, 2020

Who is “important” in a team?

dataset	model	temporal betweenness Spearman	Speedup	temporal closeness Spearman	Speedup
sociopatterns hospital	GCN[1]	0.804		0.744	
	TGN[2]	0.522		0.509	
	DBGNN	**0.832**		**0.918**	
gain		**+3.4%**	**271 x**	**+ 23.4 %**	**33 x**
sociopatterns hypertext	GCN[1]	0.786		0.809	
	TGN[2]	0.260		0.360	
	DBGNN	**0.839**		**0.977**	
gain		**+6.7%**	**485 x**	**+ 20.7 %**	**28 x**
sociopatterns highschool	GCN[1]	0.540		0.540	
	TGN[2]	0.166		0.166	
	DBGNN	**0.661**		**0.925**	
gain		**+22.4%**	**1077 x**	**+ 71.3 %**	**43 x**
manufacturing email	GCN[1]	0.404		0.556	
	TGN[2]	0.320		0.496	
	DBGNN	**0.744**		**0.971**	
gain		**+84.1%**	**17 x**	**+ 74.6 %**	**14 x**

1 → T Kipf, M Welling, ICLR, 2017

2 → E Rossi et al., arXiv:2006.10637, 2020

Using Time-Aware Graph Neural Networks to Predict Temporal Centralities in Dynamic Graphs

Franziska Heeg
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilans-Universität, Würzburg
franziska.heeg@uni-wuerzburg.de

Ingo Scholtes
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilans-Universität, Würzburg
ingo.scholtes@uni-wuerzburg.de

Abstract

Node centralities play a pivotal role in network science, social network analysis, and recommender systems. In temporal data, static path-based centralities like closeness or betweenness can give misleading results about the true importance of nodes in a temporal graph. To address this issue, temporal generalizations of betweenness and closeness have been defined that are based on the shortest time-respecting paths between pairs of nodes. However, a major issue of those generalizations is that the calculation of such paths is computationally expensive. Addressing this issue, we study the application of De Bruijn Graph Neural Networks (DBGNN), a time-aware graph neural network architecture, to predict temporal path-based centralities in time series data. We experimentally evaluate our approach in 13 temporal graphs from biological and social systems and show that it considerably improves the prediction of betweenness and closeness centrality compared to (i) a static Graph Convolutional Neural Network, (ii) an efficient sampling-based approximation technique for temporal betweenness, and (iii) two state-of-the-art time-aware graph learning techniques for dynamic graphs.

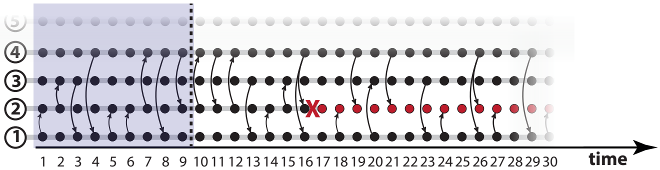
1 Motivation

Node centralities are important in the analysis of complex networks, with applications in network science, social network analysis, and recommender systems. An important class of centrality measures are *path-based centralities* like, e.g. betweenness or closeness centrality [5, 16], which are based on the shortest paths between all nodes. While centralities in static networks are important, we increasingly have access to time series data on temporal graphs with time-stamped edges. Due to the timing and ordering of those edges, the paths in a static time-aggregated representation of such time series data can considerably differ from *time-respecting paths* in the corresponding temporal graph. In a nutshell, two time-stamped edges $(u, v; t)$ and $(v, w; t')$ only form a time-respecting path from node u via v to w iff for the time stamps t and t' we have $t < t'$, i.e. time-respecting paths must minimally respect the arrow of time. Moreover, we often consider scenarios where we need to additionally account for a *maximum time difference* δ between time-stamped edges, i.e. we require $0 < t' - t \leq \delta$ [22]. Several works have shown that temporal correlations in the sequence of time-stamped edges can significantly change the *causal* topology of a temporal graph, i.e. which nodes can influence each other via time-respecting paths, compared to what is expected based on the static topology [30, 35, 39].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

→ F Heeg, I Scholtes, NeurIPS 2024

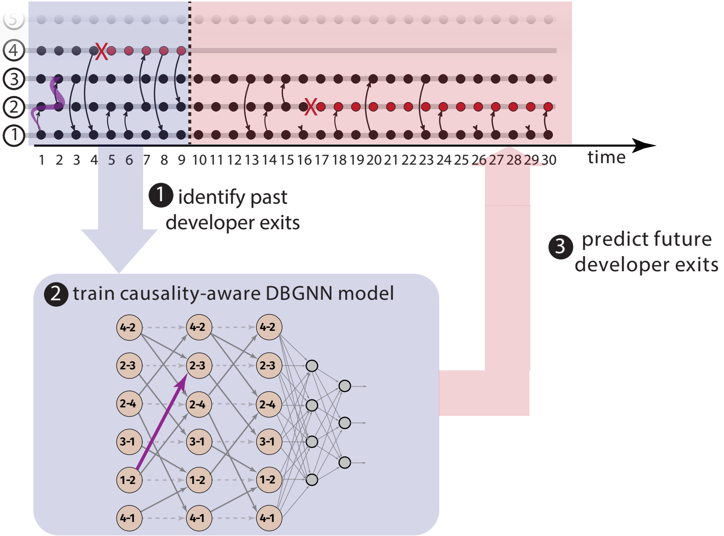
Who will leave the team?



challenge

- **exit of central developer** can be existential threat for software teams
- can we **predict exit of key team members** before they happen?

Who will leave the team?



challenge

- **exit of central developer** can be existential threat for software teams
- can we **predict exit of key team members** before they happen?

idea

use causality-aware DBGNN to detect **temporal interaction patterns** that are indicative for imminent exit

Who will leave the team?

dataset	model	Balanced Accuracy
facebook react-native	GCN[1]	72.41 ± 0.02
airbnb pay service	GCN[1]	61.12 ± 3.75
alphagov enzyme	GCN[1]	58.98 ± 0.94
keras	GCN[1]	54.25 ± 0.57

1 → T Kipf, M Welling, ICLR, 2017

challenge

- **exit of central developer** can be existential threat for software teams
- can we **predict exit of key team members** before they happen?

idea

use causality-aware DBGNN to detect **temporal interaction patterns** that are indicative for imminent exit

Who will leave the team?

dataset	model	Balanced Accuracy
facebook react-native	GCN[1]	72.41 ± 0.02
	DBGNN	**79.02** ± 0.03
gain		**+ 9.1%**
airbnb pay service	GCN[1]	61.12 ± 3.75
	DBGNN	**70.79** ± 2.47
gain		**+ 15.8%**
alphagov enzyme	GCN[1]	58.98 ± 0.94
	DBGNN	**72.46** ± 0.2
gain		**+ 22.9%**
keras	GCN[1]	54.25 ± 0.57
	DBGNN	**95.57** ± 0.0
gain		**+ 76.2%**


Using Social Comparison Theory to Predict Developer Departures in Open Source Communities

Lisi Qarkaxhija*
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science
Julius-Maximilians-Universität
Würzburg, Deutschland
lisi.qarkaxhija@uni-wuerzburg.de

Christoph Gote*†‡
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science
Julius-Maximilians-Universität
Würzburg, Deutschland
christoph.gote@uni-wuerzburg.de

Bernhard Sendhoff
Honda Research Institute Europe
Offenbach am Main, Deutschland
bernhard.sendhoff@honda-ri.de

Ingo Scholtes‡
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science
Julius-Maximilians-Universität
Würzburg, Deutschland
ingo.scholtes@uni-wuerzburg.de

ABSTRACT

A clear and well-documented LaTeX document is presented as an article formatted for publication by ACM in a conference proceedings or journal publication. Based on the "acmart" document class, this article presents and explains many of the common variations, as well as many of the formatting elements an author may use in the preparation of the documentation of their work.

CCS CONCEPTS

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

KEYWORDS

Do, Not, Us, This, Code, Put, the, Correct, Terms, for, Your, Paper

ACM Reference Format:
Lisi Qarkaxhija, Christoph Gote, Bernhard Sendhoff, and Ingo Scholtes. 2018. Using Social Comparison Theory to Predict Developer Departures in Open Source Communities. In *Proceedings of 46th International Conference on Software Engineering (ICSE 2024)*. ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

*Both authors contributed equally to this research
†also with Chair of Systems Design, ETH Zurich
‡also with Data Analytics Group, Department of Informatics, University of Zurich, Zurich, CH

1 INTRODUCTION

Social comparison, a ubiquitous human behaviour, involves individuals evaluating certain aspects of themselves by comparing them to others. This behaviour is not just confined to the realm of social media. Instead, it permeates various facets of life, spanning work performance, academic achievements, and physical appearance. Within the context of social networks, metrics like degree centrality, which gauges the number of connections a node possesses, or betweenness centrality, measuring a node's role in bridging different groups, can foster comparisons that signal popularity or influence. Social comparison can serve as a powerful motivator, propelling individuals to set and achieve goals, thus enhancing their performance. It also offers a means of self-enhancement through comparisons with less fortunate counterparts. However, beneath the surface of this seemingly benign behaviour lie potential pitfalls, including the generation of negative emotions like envy, guilt, and regret, which can diminish psychological well-being. Moreover, frequent social comparisons can trigger destructive behaviours, such as deceit and blame-shifting. This intricate interplay of social comparison in the context of social networks sets the stage for exploring its multifaceted impact on individuals in other social systems.

In software development teams, issues with the community structure are referred to as community smells[1,4]. Community smells can lead to social debt [1], negatively impact software teams, development processes, outcomes, and organizations by increasing social debt and negatively impacting team performance and productivity [4]. Community smells have further been shown to lead to technical debt in software development teams [34, 52]. In addition, community smells can predict if and which developers stop contributing to a project [25, 26].

Akin to the community smells discussed above, social comparison has been shown to can lead to negative emotions and decreased well-being, which could indirectly impact software development teams by reducing team cohesion and motivation [6] Therefore, it is important for software development teams to be mindful of the

→ L Qarkaxhija, C Gote, B Sendhoff, I Scholtes, in preparation

1 → T Kipf, M Welling, ICLR, 2017

Application perspective

- social factors in software teams introduce severe risks in **software supply chains**

recent example

social engineering attack to install backdoor into fundamental Linux library `xz` that is largely maintained by single developer

⟶ CVE-2024-3094

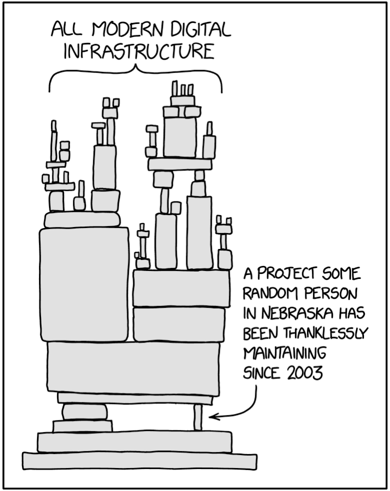


image credit: Randal Munroe, xkcd.com, CC BY-NC 2.5

Application perspective

- social factors in software teams introduce severe risks in **software supply chains**

DATEV

software campus

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

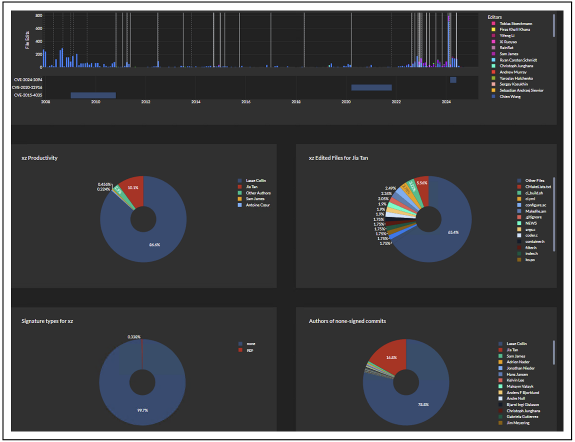
recent example

social engineering attack to install backdoor into fundamental Linux library `xz` that is largely maintained by single developer

⟶ CVE-2024-3094

Industry project Software Campus 3.0

- BMBF-funded **industry project with major software company DATEV eG**, Nürnberg
- online platform to **analyze software projects based on repository data**
- built around **temporal graph learning library** `pathpyG`



Application perspective

- social factors in software teams introduce severe risks in **software supply chains**

GEFÖRDERT VOM

Bundesministerium für Bildung und Forschung

recent example

social engineering attack to install backdoor into fundamental Linux library `xz` that is largely maintained by single developer

→ CVE-2024-3094

Industry project Software Campus 3.0

- BMBF-funded **industry project with major software company DATEV eG**, Nürnberg
- online platform to **analyze software projects based on repository data**
- built around **temporal graph learning library** `pathpyG`
- helps stakeholders to **monitor and assess socio-technical risk factors** in (Open Source) software dependencies

Thank you!

De Bruijn goes Neural: Causality-Aware Graph Neural Networks for Time Series Data on Dynamic Graphs

Lisi Qarkaxhija
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilians-Universität Würzburg, DE
lisi.qarkaxhija@uni-wuerzburg.de

Vincenzo Perri
Data Analytics Group
Department of Informatics
University of Zurich, CH
perri@ifi.uzh.ch

Ingo Scholtes*
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilians-Universität Würzburg, DE
ingo.scholtes@uni-wuerzburg.de

Abstract

We introduce De Bruijn Graph Neural Networks (DBGNNs), a novel time-aware graph neural network architecture for time-resolved data on dynamic graphs. Our approach accounts for temporal-topological patterns that unfold in the causal topology of dynamic graphs, which is determined by *causal walks*, i.e. temporally ordered sequences of links by which nodes can influence each other over time. Our architecture builds on multiple layers of higher-order De Bruijn graphs, an iterative line graph construction where nodes in a De Bruijn graph of order k represent walks of length $k-1$, while edges represent walks of length k.

We develop a graph neural network architecture that utilizes De Bruijn graphs to implement a message passing scheme that follows a non-Markovian dynamics, which enables us to learn patterns in the causal topology of a dynamic graph. Addressing the issue that De Bruijn graphs with different orders k can be used to model the same data set, we further apply statistical model selection to determine the optimal graph topology to be used for message passing. An evaluation in synthetic and empirical data sets suggests that DBGNNs can leverage temporal patterns in dynamic graphs, which substantially improves the performance in a supervised node classification task.

1 Introduction

Graph Neural Networks (GNNs) [1, 2] have become a cornerstone for the application of deep learning to data with a non-Euclidean, relational structure. Different flavors of GNNs have been shown to be highly efficient for tasks like node classification, representation learning, link prediction, cluster detection, or graph classification.

The popularity of GNNs is largely due to the abundance of data that can be represented as graphs, i.e. as a set of *nodes* with pairwise connections represented as *links*. However, we increasingly have access to *time-resolved data* that not only capture which nodes are connected to each other, but also when and in which temporal order those connections occur. A number of works in computer science, network science, and interdisciplinary physics have highlighted how the *temporal dimension* of dynamic graphs, i.e. the timing and ordering of links, influences the *causal topology* of networked systems, i.e. which nodes can possibly influence each other over time [3–5]. In a nutshell, if an undirected link (a, b) between two nodes a and b occurs *before* an undirected link (b, c), node a can causally influence node c via node b. If the temporal ordering of those two links is reversed, node a cannot influence node c via b due to the directionality of the arrow of time. This simple example

*also with Data Analytics Group, Department of Informatics, University of Zurich, Zurich, CH

Preprint. Preliminary work.

→ L Qarkaxhija, V Perri, I Scholtes, Proc. of Learning on Graphs, 2022

PATHPYG

www.pathpy.net

CAIDAS

ML4NETS
MACHINE LEARNING FOR COMPLEX NETWORKS

Using Time-Aware Graph Neural Networks to Predict Temporal Centralities in Dynamic Graphs

Franziska Heeg
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilans-Universität, Würzburg
franziska.heeg@uni-wuerzburg.de

Ingo Scholtes
Chair of Machine Learning for Complex Networks
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilans-Universität, Würzburg
ingo.scholtes@uni-wuerzburg.de

Abstract

Node centralities play a pivotal role in network science, social network analysis, and recommender systems. In temporal data, static path-based centralities like closeness or betweenness can give misleading results about the true importance of nodes in a temporal graph. To address this issue, temporal generalizations of betweenness and closeness have been defined that are based on the shortest time-respecting paths between pairs of nodes. However, a major issue of those generalizations is that the calculation of such paths is computationally expensive.

Addressing this issue, we study the application of De Bruijn Graph Neural Networks (DBGNN), a time-aware graph neural network architecture, to predict temporal path-based centralities in time series data. We experimentally evaluate our approach in 13 temporal graphs from biological and social systems and show that it considerably improves the prediction of betweenness and closeness centrality compared to (i) a static Graph Convolutional Neural Network, (ii) an efficient sampling-based approximation technique for temporal betweenness, and (iii) two state-of-the-art time-aware graph learning techniques for dynamic graphs.

1 Motivation

Node centralities are important in the analysis of complex networks, with applications in network science, social network analysis, and recommender systems. An important class of centrality measures are *path-based centralities* like, e.g. betweenness or closeness centrality [5, 16], which are based on the shortest paths between all nodes. While centralities in static networks are important, we increasingly have access to time series data on temporal graphs with time-stamped edges. Due to the timing and ordering of those edges, the paths in a static time-aggregated representation of such time series data can considerably differ from *time-respecting paths* in the corresponding temporal graph. In a nutshell, two time-stamped edges $(u, v; t)$ and $(v, w; t')$ only form a time-respecting path from node u via v to w iff for the time stamps t and t' we have $t < t'$, i.e. time-respecting paths must minimally respect the arrow of time. Moreover, we often consider scenarios where we need to additionally account for a *maximum time difference* δ between time-stamped edges, i.e. we require $0 < t' - t \leq \delta$ [22]. Several works have shown that temporal correlations in the sequence of time-stamped edges can significantly change the *causal* topology of a temporal graph, i.e. which nodes can influence each other via time-respecting paths, compared to what is expected based on the static topology [30, 35, 39].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

→ F Heeg, I Scholtes, NeurIPS, 2024