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).

https://doi.org/10.52202/079017-0949

An important consequence of this is that static path-based centralities like closeness or betweenness can give misleading results about the true importance of nodes in temporal graphs. To address this issue, temporal generalizations of betweenness and closeness centrality have been defined that are based on the shortest time-respecting paths between pairs of nodes [51, 27, 1, 52]. A major issue of those generalizations is that the calculation of time-respecting paths as well as the resulting centralities is computationally expensive [11, 14, 47]. Addressing this issue, a number of recent works developed methods to approximate temporal betweenness and closeness centralities in temporal graphs [47]. Additionally, few works have used deep (representation) learning techniques to predict computationally expensive path-based centralities in *static* networks [19, 18].

Research Gap and Contributions To the best of our knowledge, no prior works have considered the application of time-aware graph neural networks to predict path-based centralities in temporal graphs. Closing this gap, our work makes the following contributions:

- We introduce the problem of predicting temporal betweenness and closeness centralities of nodes in temporal graphs. We consider a situation where we have access to a training graph as well as ground truth temporal centralities and seek to predict the centralities of nodes in a future observation of the same graph, which does not necessarily contain the same node set.
- To address this problem, we introduce a deep learning method that utilizes De Bruijn Graph Neural Networks (DBGNN), a recently proposed time-aware graph neural network architecture [41] that is based on higher-order graph models of time-respecting paths, which capture correlations in the sequence of time-stamped edges. An overview of our approach in a toy example of a temporal graph is shown in Figure 1.
- We compare our method to a Graph Convolutional Network (GCN), which only considers a static, time-aggregated weighted graph that captures the frequency and topology of edges. Evaluating our approach against two deep learning methods for temporal graphs, we consider the time-aware graph embedding method EVO [6] as well as the Temporal Graph Network (TGN) framework [44]. We further compare our method to ONBRA[47], which efficiently approximates temporal betweenness centralities of nodes to varying degrees of accuracy.
- We experimentally evaluate all models in 13 temporal graphs from biological and social systems. Our results show that the application of the time-aware DBGNN architecture considerable improves the prediction of both betweenness and closeness centrality compared to other static and time-aware graph learning techniques. Our method outperforms ONBRA for the prediction of temporal betweenness centralities in large datasets.

In summary, we show that predicting temporal centralities is an interesting temporal graph learning problem, which could be included in community benchmarks [24]. Moreover, our study highlights the potential of time-aware deep learning architectures for node-level regression tasks in temporal graphs. Finally, our results are a promising step towards an approximation of temporal centralities in large data, with potential applications in social network analysis and recommender systems.

2 Background and Related Work

In the following, we provide the background of our work. We first introduce temporal graphs and define time-respecting paths. We then cover generalizations of path-based centralities for nodes in temporal graphs. We finally discuss prior works that have studied the prediction, or approximation, of path-based centralities both in static and temporal graphs. This will motivate the research gap that is addressed by our work.

Dynamic Graphs and Time-respecting Paths Apart from static graphs $G = (V, E)$ that capture the topology of edges $E \subseteq V \times V$ between nodes V, we increasingly have access to time-stamped interactions that can be modelled as *temporal graphs or networks* [13, 23, 22]. We define a temporal graph as $G^{\mathcal{T}} = (V, E^{\mathcal{T}})$ where V is the set of nodes and $E^{\mathcal{T}} \in V \times V \times \mathbb{R}$ is a set of (possibly directed) time-stamped edges, i.e. an edge $(v, w; t) \in E^{\mathcal{T}}$ describes an interaction from node v to w occurring at time t. In our work, we assume that interactions are *instantaneous*, i.e. $(v, w; t) \in E^{\mathcal{T}}$ does not imply that $(v, w; t') \in E^{\mathcal{T}}$ for all $t' > t$. Hence, we do not specifically consider *growing networks*, where the time-stamp t is the creation of an edge. For a temporal network $G^{\mathcal{T}} = (V, E^{\mathcal{T}})$ it is common to consider a static, time-aggregated and weighted graph representation $G = (V, E)$, where $(v, w) \in E$ iff $(v, w; t) \in E^{\mathcal{T}}$ for some time stamp t and for the edge weights we define $w(v, w) = |\{t \in \mathbb{R} : (v, w; t) \in E^{\mathcal{T}}\}|$, i.e. the number of occurrences of time-stamped edges.

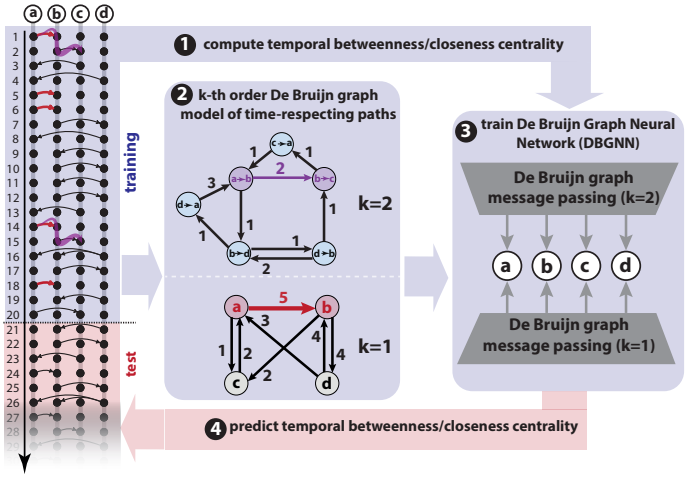


Figure 1: Overview of proposed approach to predict temporal node centralities in a temporal graph: We consider a time-based split in a training and test graph (left). Calculating time-respecting paths in the training split enables us to (1) compute temporal centralities, and (2) fit a k-th order De Bruijn graph model for time-respecting paths. The weighted edges in such a k-th order De Bruin graph capture frequencies of time-respecting paths of length k (see time-respecting path of length one (red) and two (magenta)). (3) We use these centralities and the k-th order models to train a De Bruijn graph neural network (DBGNN), which allows us to (4) predict temporal centralities in the test graph.

An important difference to the static case is that, in temporal graphs, the temporal ordering of edges determines *time-respecting paths* [26, 23, 22]. For temporal graph $G^{\mathcal{T}} = (V, E^{\mathcal{T}})$ we define a *time-respecting path* of length l as node sequence $v_0, \ldots, v_l$ such that the following conditions hold:

$$(i)\ \exists\, t_1, \ldots, t_l\ :\ (v_{i-1}, v_i; t_i) \in E^{\mathcal{T}} \text{ for } i = 1, \ldots, l\,;$$
$$(ii)\ 0 < t_i - t_{i-1} \leq \delta \text{ for some } \delta \in \mathbb{R}.$$

In contrast to definitions of time-respecting paths that only require interactions to occur in ascending temporal order, i.e. $0 < t_i - t_j$ for $j < i$ [26, 4], we also impose a maximum "waiting time" δ [35, 23]. This implies that we only consider time-respecting paths where subsequent interactions occur within a time interval that is often defined by the processes that we study on temporal networks [14, 3]. In line with the definition for static networks, we define a *shortest time-respecting path* between nodes v and w as a (not necessarily unique) time-respecting path of length l such that all other time-respecting paths from v to w have length $l' \geq l$. In static graphs a shortest path from v to w is necessarily a *simple* path, i.e. a path where no node occurs more than once in the sequence $v_1, \ldots, v_l$. This is not necessarily true for shortest time-respecting path, since –due to the maximum waiting time δ– we may be forced to move between (possibly the same) nodes to continue a time-respecting path. Due to the definition of time-respecting paths with limited waiting time δ, we obtain a *temporal-topological* generalization of shortest paths to temporal graphs that accounts for the temporal ordering and timing of interactions. We note that other definitions of *fastest paths* only account for temporal rather than topological distance [35], which we however do not consider in our work.

The definition of time-respecting paths above has the important consequence that the connectivity of nodes via time-respecting paths in a temporal network can be considerably different from paths in the corresponding time-aggregated static network. As an example, for a temporal network with two time-stamped edges $(u, v; t)$ and $(v, w; t')$ the time-aggregated network contains a path from u via v to w, while a time-respecting path from u via v to w can only exist iff $0 < t' - t \leq \delta$. In other words, while connectivity in static graphs is *transitive*, i.e. the existence of edges (or paths) connecting u to v and v to w implies that there exists a path that transitively connects u to w, the same does not hold for time-respecting paths. A large number of works have shown that this difference between paths in temporal and static graphs influences connectivity and reachability [30], the evolution of dynamical processes like diffusion or epidemic spreading [45, 39, 53, 49], cluster patterns [45, 46, 29], as well as the controllability of dynamical processes [40].

Temporal Centralities Another interesting question is how the time dimension of temporal graphs influences the importance or *centrality* of nodes [27]. To this end, several works have generalized

https://doi.org/10.52202/079017-0949

centrality measures originally defined for static graphs to temporal networks. For our purpose we limit ourselves to generalizations of betweenness and closeness centrality, which are defined based on the shortest paths between nodes. In a static network, a node v has high *betweenness centrality* if there are many shortest paths that pass through v [16] and it has high *closeness centrality* if the overall distance to all other nodes is small [5]. We omit those standard definitions here due to space constraints but include them in appendix H.

Analogously to betweenness centrality for static graphs, for a temporal graph $G = (V, E^T)$ we define the *temporal betweenness centrality* of node $v \in V$ as

$$c_B^{temp}(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

where $\sigma_{s,t}$ is the number of shortest *time-respecting* paths from node s to t.

To calculate *temporal closeness centrality* we define the temporal distance $d(u, v)$ between two nodes $u, v \in V$ as the length of a shortest time-respecting paths from u to v and thus obtain

$$c_C^{temp}(v) = \frac{1}{\sum_{u \in V} d(u, v)}.$$

Even though the definitions above closely follow those for static networks, it has been shown that the temporal centralities of nodes can differ considerably from their counterparts in static time-aggregated networks [27, 29]. These findings highlight the importance of a *time-aware* network analysis, which consider both the timing and temporal ordering of links in temporal graphs.

Approximating Path-based Centralities While path-based centralities have become an important tool in network analysis, a major issue is the computational complexity of the underlying all-pairs shortest path calculation in large graphs. For static networks, this issue can be partially alleviated by smart algorithms that speed up the calculation of betweenness centralities [9]. Even with these algorithms, calculating path-based centralities in large graphs is a challenge. Hence, a number of works considered approaches to calculate fast approximations, e.g. based on a random sampling of paths [42, 2, 21]. Another line of studies either used standard, i.e. not graph-based, machine learning techniques to leverage correlations between different centrality scores [18, 19], or used neural graph embeddings in synthetic scale-free networks to approximate the ranking of nodes [33].

Existing works on the approximation of path-based node centralities in time series data have generally focussed on a fast updating of *static* centralities in *evolving graphs* where edges are added or deleted [7, 43], rather than considering *temporal node centralities*. For the calculation of temporal closeness or betweenness centralities, the need to calculate shortest *time-respecting paths* between all pairs of nodes is computationally challenging: Temporal closeness centrality minimally requires the traversal of all time-stamped edges for all nodes in the graph, which has a time complexity in $O(n \cdot m)$ where n is the number of nodes and m is the number of time-stamped edges in the temporal graph. Building on Brandes' algorithm for static betweenness centrality [9], a fast algorithm for temporal betweenness centrality with complexity $O(n \cdot m \cdot T)$ (where T is the number of different time stamps in the temporal graph) has recently been proposed in [12]. Considering the approximate estimation of temporal betweenness and closeness centrality in temporal graphs, [50] generalizes static centralities to higher-order De Bruijn graphs, which capture the time-respecting path structure of a temporal graph. [47] recently proposed a sampling-based estimation of temporal betweenness centralities. To the best of our knowledge, no prior works have considered the application of deep graph learning to predict temporal node centralities in temporal graphs, which is the gap addressed by our work.

3 A Time-Aware GNN to Predict Temporal Centralities

Here, we first present higher-order De Bruijn graph models for time-respecting paths in temporal networks. We then describe our deep learning architecture to predict temporal centralities.

Higher-Order De Bruijn Graph Models of Time-respecting paths Each time-respecting path gives rise to an ordered sequence $v_0, v_1, \ldots, v_l$ of traversed nodes. Let us consider a k-th order Markov chain model, where $P(v_i|v_{i-k}, \ldots, v_{i-1})$ is the probability that a time-respecting path continues to node v_i, conditional on the k previously traversed nodes. A first-order Markov chain

model can be defined based on the frequencies of edges (i.e. paths of length $k = 1$) captured in a weighted time-aggregated graph, where

$$P(v_i|v_{i-1}) := \frac{w(v_{i-1}, v_i)}{\sum_j w(v_{i-1}, v_j)}.$$

While a first-order model is justified if the temporal graph exhibits no patterns in the temporal ordering of time-stamped edges, several works have shown that empirical data exhibit patterns that require higher-order Markov models for time-respecting paths [45, 49, 46]. To address this, for $k > 1$ we define a k-th order Markov chain model based on frequencies of time-respecting paths of length k as

$$P(v_i|v_{i-k}, \ldots, v_{i-1}) = \frac{w(v_{i-k}, \ldots, v_i)}{\sum_j w(v_{i-k}, \ldots, v_{i-1}, v_j)},$$

where $w(v_0, \ldots, v_k)$ counts the number of time-respecting path $v_0, \ldots, v_k$ in the underlying temporal graph. For a temporal graph $G^{\mathcal{T}} = (V, E^{\mathcal{T}})$, this approach defines a static *k-th order De Bruijn graph model* $G^{(k)} = (V^{(k)}, E^{(k)})$ with

- $V^{(k)} = \{(v_0, \ldots, v_{k-1}) \mid v_0, \ldots, v_{k-1}$ is a time-respecting walk of length $k-1$ in $G^{\mathcal{T}}\}$
- $(u, v) \in E^{(k)}$ iff
  (i) $v = (v_1, \ldots, v_k)$ with $v_i = u_i$ for $i = 1, \ldots, k-1$
  (ii) $u \bigoplus v = (u_0, \ldots, u_{k-1}, v_k)$ is a time-respecting path of length k in $G^{\mathcal{T}}$.

We call this k-th order model a *De Bruijn graph model* of time-respecting paths, since it is a generalization of a k-dimensional De Bruijn graph [10], with the additional constraint that an edge only exists iff the underlying temporal network has a corresponding time-respecting path. For $k = 1$ the first-order De Bruijn graph corresponds to the commonly used static, time-aggregated graph $G = (V, E)$ of a temporal graph G^T, where edge can be considered time-respecting paths of length one and which neglects information on time dimension. For $k > 1$ we obtain *static but time-aware higher-order generalizations of time-aggregated graphs*, which are sensitive to the timing and ordering of time-stamped edges. Each node in such a k-th order De Bruin graph represents a time-respecting path of length $k-1$, while edges represent time-respecting paths of length k. Edge weights correspond to the number of observations of time-respecting paths of length k (cf. fig. 1).

De Bruijn Graph Neural Networks for Temporal Centrality Prediction Our approach to predict temporal betweenness and closeness centrality uses the recently proposed De Bruijn Graph Neural Networks (DBGNN), a deep learning architecture that builds on k-th order De Bruijn graphs [41]. The intuition behind this approach is that, by using message passing in multiple (static) k-th order De Bruijn graph models of time-respecting paths, we obtain a *time-aware learning algorithm* that considers both the graph topology as well as the temporal ordering and timing of interactions.

Our proposed method is summarized in fig. 1. Considering time series data on a temporal graph, we first perform a time-based split of the data into a training and test graph. We then calculate temporal closeness and betweenness centralities of nodes in the training graph and consider a supervised node-level regression problem, i.e. we use temporal centralities of nodes in the training graph to train a DBGNN model. To this end, we construct k-th order De Bruijn graph models for multiple orders k, based on the statistics of time-respecting paths of lengths k. The maximum order is determined by the temporal correlation length (i.e. the Markov order) present in a temporal graph and can be determined by statistical model selection techniques [48].

Using the update rule defined in Eq. (1) of [41], we simultaneously perform message passing in all k-th order De Bruijn graphs. For each k-th order De Bruijn graph this yields a (hidden) representation of k-th order nodes. To aggregate the resulting representation to actual (first-order) nodes in the temporal graph, we perform message passing in an additional bipartite graph, where each k-th order node $(v_0, \ldots, v_{k-1})$ is connected to first-order node v_{k-1} (cf. Eq (2) in [41] and fig. 1). Taking a node regression perspective, we use a final dense linear layer with a single output. We use the trained model to predict the temporal centralities of nodes in the test graph. Since the subset of nodes and edges that are active in the training and test graph can differ, our model must be able to generalize to temporal graphs with different nodes as well as different graph topologies. To address this, we train our models in an inductive fashion by choosing a suitably large number of dimensions for the one-hot encodings during the training phase.

 https://doi.org/10.52202/079017-0949

Compared to [41], we introduce two significant technical advances: first, we adapt DBGNN for a node-level regression task, which, to our knowledge, has not been previously explored. Second, we implement a different training procedure designed for forecasting. Unlike the node classification task in [41], our approach involves training a model on a temporal graph within a training window and subsequently refitting this model to forecast temporal centralities in a future observation, which may include previously unseen nodes and edges. This approach enables our model to generalize to forecasting scenarios involving previously unobserved graph elements, potentially extending its utility to other temporal graph forecasting tasks.

The implementation of our method is based on the Open Source temporal graph learning library pathpyG[1]. The code of our experiments has been permanently archived at Zenodo[2].

4 Experimental Results

With our experimental evaluation we seek to answer the following five research question:

- **RQ1** How does the predictive power of a time-aware DBGNN model compare to that of a standard GCN that only uses the static topology and ignores the time dimension of dynamic graphs?
- **RQ2** How does the performance of the DBGNN model compare to (i) a two step approach that combines the temporal graph embedding EVO [6] with a feed-forward neural network, and (ii) TGN [44], an end-to-end temporal GNN architecture that does, however, not explicitly consider time-respecting paths.
- **RQ3** How do the predictions of the DBGNN architecture compare to the results of ONBRA, a method that aims to approximate temporal betweenness centralities?
- **RQ4** Which speed-up does our prediction method offer compared to the calculation of temporal node centralities?
- **RQ5** Does the DBGNN architecture generate node embeddings that facilitate interpretability?

Experimental setup We experimentally evaluate the performance of the DBGNN architecture by predicting temporal centralities in 13 empirical temporal graphs. We split each temporal graph in training and test graphs, where the training and test graph contain half of the data each. Since a maximum order detection in those data sets yields a maximum of two (see table 11 in appendix F), we limit the DBGNN architecture to $k = 2$. To calculate edge weights of the DBGNN model, we count time-stamped edges as well as time-respecting paths of length two for weights of the first and second-order De Bruijn graph, respectively (cf. fig. 1). Adopting the approach in [41] we use one message passing layer with 16 hidden dimensions for each order k and one bipartite message passing layer with 8 hidden dimensions. We use a sigmoid activation function for higher-order layers and an Exponential Linear Unit (ELU) activation function for the bipartite layer.

As a first time-neglecting baseline model, we use a Graph Convolutional Neural Network (GCN) [28], which we apply to the weighted time-aggregated representation of the temporal graphs. For the GCN model, we use two message passing layers with 16 and 8 hidden dimensions and a sigmoid activation function, respectively. As input features, we use a one-hot encoding (OHE) of nodes for both architectures. In the case of the DBGNN architecture we apply OHE to nodes in all (higher-order) layers. Addressing a node regression task, we use a final dense linear layer with a single output and an ELU activation function, and use mean squared error (MSE) as loss function for both architectures. We train both models based on (ground-truth) temporal centralities in the training data, using 1000 epochs with an ADAM optimizer, different learning rates, and weight decay of $5 \cdot 10^{-4}$. We tested the use of dropout layers for both architectures, but found the results to be worse. In table 16 and table 17 in the appendix we summarize the architecture and report all hyperparameters for both models.

As a second baseline method we use the time-aware graph embedding EVO [6], which models correlations in the sequence of nodes traversed by time-respecting paths. Similar to DBGNN, EVO uses these correlations to produce a single static embedding of nodes that captures both the topology and temporal patterns in the dynamic graph. Different from DBGNN, EVO does not yield an end-to-end centrality prediction approach, i.e. it only produces node embeddings that can then be used for downstream learning tasks. To address centrality prediction, we use 16-dimensional node embeddings produced by EVO for time-respecting paths up to length two. We then train a two-layer feed-forward

[1] see `https://www.pathpy.net` and `https://github.com/pathpy/pathpyG`

[2] `https://doi.org/10.5281/zenodo.10202791`

network with a 16-dimensional input, a hidden layer with eight dimensions, and a single output. We use a Rectified Linear Unit (ReLU) activation function for both the hidden layer and the output layer.

As a third baseline, we use the Temporal Graph Networks (TGN) framework, a recently proposed graph neural network architecture for temporal graphs [44]. Different from DBGNN and EVO the TGN framework does not explicitly consider time-respecting paths but produces a time evolving node embedding that accounts for the sequentiality of interactions and node-wise events. To this end, TGN splits a temporal graph into multiple equally-sized batches of consecutive time-stamped interactions. In each of these batches a message passing algorithm is used to update node representations based on time-stamped edges in the current batch as well as a memory of node representations and messages in previous batches. Within each batch the learnable parameters of a TGN model can be trained using a variety of graph learning tasks such as, e.g., link prediction and node classification.

On the one hand we want batches to be small to obtain a sufficient numbers of batches that we can use to train the model on a given dataset. On the other hand small batch sizes introduce the problem that, due to the small number of time-respecting paths, we cannot calculate meaningful centralities. To address this issue, we calculate the temporal centralities of nodes in a given batch i based on a temporal graph obtained by the batches $i - k + 1$ to i for some k. Adopting this approach we train the TGN model based on the training splits of our data. We then use the trained model to perform a per-batch prediction of temporal centralities for all batches in the test split of our data. For TGN, we chose the training and test set such that both contain the same number of batches. Finally, we average the prediction scores of the test batches to evaluate the performance of the model.

In addition to the deep learning methods above, as a final baseline we include ONBRA, a recently proposed sampling-based method, which can estimate temporal betweenness centralities with varying degrees of fidelity by sampling pairs of nodes and calculating shortest temporal paths between them[47]. By choosing a suitable number of samples, we experimentally adjusted the estimation fidelity of ONBRA, such that the estimation algorithm took approximately the same time as our model. In particular, using the publicly available implementation of the authors[3], we estimate temporal betweenness for shortest δ-restless walks for ten iterations and adjust the number of sampled node pairs. In order to get a meaningful comparison, we only run ONBRA on the test window, on which we predicted the temporal betweenness centralities with the other models.

Data sets We use 13 data sets on temporal graphs from different contexts, including human contact patterns based on (undirected) proximity or face-to-face relations, time-stamped (directed) E-Mail communication networks, as well as antenna interactions between ants in a colony. An overview of the data sets along with a short description, key characteristics and the source is given in table 4 in the appendix. All data are publicly available from netzschleuder [37] and SNAP [31].

Evaluation procedure To evaluate our models, we first fit the pre-trained models to the test graph, i.e. we apply the trained models to the test graph and the trained DBGNN model to the De Bruijn graphs for the test data. We then use the trained models to predict temporal closeness and betweenness centralities and compare those predictions to ground truth centralities. For the calculation of temporal closeness centrality, we calculate shortest time-respecting path distances for a given maximum time difference δ between all pairs of nodes using a variation of Dijkstra's algorithm that traverses all time-stamped edges. For temporal betweenness centrality, we adopt a variation of the algorithm proposed in [12], which we adjusted to account for shortest time-respecting paths with a maximum time difference δ. We will make our code of the temporal centrality calculation available upon acceptance of the manuscript. Figure 1 provides an illustration of our evaluation approach. We use Kendall-Tau and Spearman rank correlation to compare a node ranking based on predicted centralities with a ranking obtained from ground truth centralities. Since both rank correlation measures yielded qualitatively similar results, we only report the Spearman correlation. Since centrality scores are often used to identify a small set of most central nodes, we further calculate the number of hits in the set of nodes with the top ten predicted centralities. Since we repeated each experiment 20 times, we report the mean and the standard deviation of all scores. We repeated all experiments for different learning rates between 0.1 and 0.0001 and report the best mean scores. The associated learning rates as well as all other hyperparameters of the models are reported in the appendix.

Discussion of results The results of our experiments for temporal betweenness and closeness centralities are shown in table 1 and table 2, respectively. Considering **RQ1**, we find that our time-

[3] `https://github.com/iliesarpe/ONBRA`

https://doi.org/10.52202/079017-0949

aware DBGNN-based architecture significantly outperforms a static GCN model for all 13 data sets and for both evaluation metrics for temporal closeness centrality. We further observe a large relative increase of the Spearman rank correlation coefficient ranging between 13 % for ants-2-1 and 241 % for eu-email-dept4. For temporal betweenness centrality, we find that the proposed DBGNN architecture outperforms a GCN-based prediction in terms of Spearman rank correlation for 12 of the 13 data sets, while we observe better performance of the GCN model for a single data set (haggle). For the 12 cases where DBGNN outperforms GCN, we find relative increases in Spearman rank correlation between 3 % (sp-hospital) and 151 % (ants-2-2). For haggle, where a GCN model outperforms a DBGNN-based prediction, the relative increase is 8 %. Additionally, we observe that all methods generally perform better for temporal closeness centrality compared to temporal betweenness centrality. We attribute this to the specific characteristics of those centralities, which are rooted in their definitions. The temporal closeness centrality of a node only depends on the length of shortest time-respecting paths from that node to all other nodes. Moreover temporal closeness centralities likely exhibit strong correlations between neighboring nodes, which specifically favors a prediction based on neural message passing. In contrast, the temporal betweenness centrality of a node is not only influenced by the length of time-respecting paths but also by the specific sequence of traversed nodes. At the same time, depending on the structure of time-respecting paths, two neighboring nodes can have vastly different temporal betweenness centralities. These factors suggest that the prediction of temporal betweenness centrality is a fundamentally more difficult problem than the prediction of temporal closeness centrality.

Table 1: Results for prediction of temporal betweenness centrality. Reported values are arithmetic mean across 20 runs and we also report the standard deviation. Bold values represent the best result for a given data set and metric.

	DBGNN		GCN		EVO		TGN	
Experiment	Spearmanr	hitsIn10	Spearmanr	hitsIn10	Spearmanr	hitsIn10	Spearmanr	hitsIn10
ants-1-1	**0.636** ± 0.063	**6.050** ± 1.234	0.282 ± 0.147	2.350 ± 1.424	0.404 ± 0.084	4.050 ± 0.51	0.263 ± 0.056	3.400 ± 0.424
ants-1-2	**0.655** ± 0.078	**4.750** ± 0.91	0.498 ± 0.157	4.600 ± 1.392	0.339 ± 0.312	3.150 ± 1.496	0.257 ± 0.084	3.725 ± 0.411
ants-2-1	0.284 ± 0.073	**3.550** ± 0.887	0.161 ± 0.126	2.200 ± 1.196	0.096 ± 0.089	2.150 ± 0.366	**0.309** ± 0.017	2.111 ± 0.192
ants-2-2	0.466 ± 0.239	4.000 ± 1.451	0.185 ± 0.287	2.100 ± 1.832	**0.599** ± 0.042	**4.400** ± 1.957	0.357 ± 0.088	3.050 ± 0.574
eu-email-dept4	0.322 ± 0.062	3.900 ± 1.252	-0.047 ± 0.244	1.950 ± 1.432	**0.486** ± 0.111	**4.750** ± 1.293	0.292 ± 0.019	4.164 ± 0.378
eu-email-dept2	0.383 ± 0.071	2.900 ± 1.41	0.240 ± 0.089	**4.000** ± 1.487	**0.503** ± 0.094	1.550 ± 0.759	0.225 ± 0.023	3.274 ± 0.348
eu-email-dept3	**0.532** ± 0.068	5.700 ± 0.979	0.408 ± 0.1	**6.400** ± 0.883	0.504 ± 0.034	3.750 ± 1.743	0.236 ± 0.096	3.151 ± 0.568
sp-workplace	**0.588** ± 0.065	**4.350** ± 1.04	0.441 ± 0.103	3.450 ± 0.887	0.294 ± 0.072	3.400 ± 0.94	0.077 ± 0.02	1.963 ± 0.064
sp-hypertext	**0.839** ± 0.017	6.300 ± 0.865	0.786 ± 0.021	**6.400** ± 0.503	0.622 ± 0.061	3.800 ± 0.696	0.260 ± 0.048	2.574 ± 0.545
sp-hospital	**0.832** ± 0.03	**8.000** ± 1.257	0.804 ± 0.041	6.950 ± 0.945	0.695 ± 0.067	4.600 ± 0.681	0.522 ± 0.076	6.463 ± 0.402
haggle	0.626 ± 0.023	5.650 ± 1.04	**0.680** ± 0.003	**5.850** ± 0.366	0.630 ± 0.102	2.250 ± 0.716	0.628 ± 0.013	3.302 ± 0.191
manufacturing-email	**0.744** ± 0.106	**3.750** ± 1.251	0.404 ± 0.14	1.700 ± 0.865	0.578 ± 0.111	1.850 ± 0.489	0.320 ± 0.113	1.824 ± 0.265
sp-highschool-2013	**0.661** ± 0.03	**3.500** ± 1.469	0.465 ± 0.055	0.850 ± 0.875	0.267 ± 0.056	1.350 ± 0.587	0.114 ± 0.007	1.224 ± 0.08

Table 2: Results for prediction of temporal closeness centrality. Reported values are arithmetic mean across 20 runs and we also report the standard deviation. Bold values represent the best result for a given data set and metric.

	DBGNN		GCN		EVO		TGN	
Experiment	Spearmanr	hitsIn10	Spearmanr	hitsIn10	Spearmanr	hitsIn10	Spearmanr	hitsIn10
ants-1-1	**0.900** ± 0.017	7.300 ± 0.733	0.805 ± 0.011	**7.850** ± 0.366	0.622 ± 0.045	3.800 ± 0.523	0.283 ± 0.038	2.780 ± 0.54
ants-1-2	**0.944** ± 0.006	**6.650** ± 0.587	0.702 ± 0.003	6.000 ± 0.0	0.339 ± 0.312	3.150 ± 1.496	0.305 ± 0.03	3.000 ± 0.51
ants-2-1	**0.974** ± 0.004	**8.600** ± 0.503	0.861 ± 0.004	7.150 ± 0.366	0.117 ± 0.025	3.150 ± 0.988	0.325 ± 0.032	3.080 ± 0.46
ants-2-2	**0.964** ± 0.004	**8.050** ± 0.394	0.662 ± 0.025	7.300 ± 0.47	0.722 ± 0.057	5.550 ± 1.356	0.373 ± 0.072	3.680 ± 0.642
eu-email-dept4	**0.972** ± 0.003	**8.800** ± 0.41	0.285 ± 0.157	2.200 ± 2.042	0.486 ± 0.111	4.750 ± 1.293	0.664 ± 0.026	5.084 ± 0.566
eu-email-dept2	**0.968** ± 0.006	**8.550** ± 0.605	0.563 ± 0.007	3.000 ± 0.0	0.503 ± 0.094	1.550 ± 0.759	0.574 ± 0.096	3.519 ± 0.876
eu-email-dept3	**0.992** ± 0.001	**8.850** ± 0.366	0.653 ± 0.009	5.000 ± 0.0	0.504 ± 0.034	3.750 ± 1.743	0.465 ± 0.025	3.187 ± 0.277
sp-workplace	**0.893** ± 0.009	**7.900** ± 0.553	0.639 ± 0.002	6.000 ± 0.0	0.388 ± 0.07	3.000 ± 0.649	0.164 ± 0.061	2.692 ± 0.312
sp-hypertext	**0.977** ± 0.004	**7.750** ± 0.55	0.809 ± 0.001	7.000 ± 0.0	0.622 ± 0.061	3.800 ± 0.696	0.360 ± 0.037	3.022 ± 0.607
sp-hospital	**0.918** ± 0.006	**7.850** ± 0.489	0.744 ± 0.002	5.300 ± 0.47	0.695 ± 0.067	4.600 ± 0.681	0.509 ± 0.058	5.649 ± 0.335
haggle	**0.948** ± 0.005	**9.300** ± 0.47	0.393 ± 0.001	4.950 ± 0.759	0.630 ± 0.102	2.250 ± 0.716	0.559 ± 0.021	3.242 ± 0.331
manufacturing-email	**0.971** ± 0.002	**7.900** ± 0.641	0.556 ± 0.004	3.750 ± 0.444	0.716 ± 0.084	2.550 ± 1.146	0.496 ± 0.028	2.258 ± 0.573
sp-highschool-2013	**0.925** ± 0.002	**7.800** ± 0.41	0.540 ± 0.002	2.000 ± 0.0	0.276 ± 0.026	2.900 ± 0.308	0.166 ± 0.041	1.776 ± 0.165

Considering **RQ2**, in table 1 and table 2 we observe that our proposed DBGNN-based method outperforms both time-aware graph learning techniques TGN and EVO in the majority of data sets, both for temporal closeness and temporal betweenness. For temporal betweenness centrality, DBGNN outperforms EVO in all but four data sets (ants-2-2, eu-email-dept4, eu-email-dept2, haggle). For the nine data sets where DBGNN outperforms EVO, we find relative performance increases in Spearman rank correlation of up to 139 % (sp-highschool-2013). Results for temporal closeness are even more pronounced, DBGNN outperforming EVO on all data sets, with performance increases ranging from 35 % (ants-2-2) to 732 % (ants-2-1). This is likely due to DBGNN providing end-to-end learning

based on time-respecting paths, as opposed to the two-step approach where we use EVO embeddings as input to a subsequent neural network.

We further find that DBGNN outperforms TGN in all tested cases except for temporal betweenness in the ants-2-1 data set. We attribute this to the fact that DBGNN explicitly models patterns in the sequence of nodes traversed by time-respecting paths, which are the basis for the definition of betweenness and closeness centrality. The worse performance of TGN can be explained by the fact that the TGN architecture does not use time-respecting paths for the message passing algorithm. This makes it – despite being a time-aware technique that accounts for the temporal evolution of graphs – a bad choice for temporal graph learning tasks that depend on time-respecting paths.

Considering **RQ3**, for the ONBRA method to approximate temporal betweenness centrality, we find that (i) our method provides a considerably higher performance in terms of Spearman rank correlation for large data sets, and (ii) generally lower mean absolute error across all data sets. Moreover, ONBRA failed to return results for three data sets where our method shows high performance. In table 7 in appendix C we report the Spearman rank correlation of the results across all data sets, as well as the MAE scores and the time the model took to calculate the estimated centralities. We chose the samples for the ONBRA algorithm such that the time required for the estimation of the centralities approximately matches the inference time for the DBGNN model.

Table 3: Speed-up of the time required for fitting our pretrained model and inference of temporal closeness and betweenness centrality compared to the time required to calculate temporal closeness and betweenness centrality in the validation set.

Experiment	Closeness			Betweenness		
	Fitting+Inference	Centrality	Speed-Up	Fitting+Inference	Centrality	Speed-Up
ants-1-1	0.019	0.068	3.478	0.019	0.288	14.865
ants-1-2	0.012	0.056	4.510	0.012	0.107	8.721
ants-2-1	0.007	0.029	4.127	0.007	0.045	6.376
ants-2-2	0.010	0.055	5.352	0.010	0.116	11.137
eu-email-dept4	0.066	1.201	18.175	0.066	1.476	22.369
eu-email-dept2	0.080	1.500	18.720	0.080	1.883	23.502
eu-email-dept3	0.017	0.191	11.420	0.018	0.309	17.274
sp-workplace	0.058	1.577	27.299	0.058	4.961	86.154
sp-hypertext	0.207	5.908	28.570	0.207	100.517	485.728
sp-hospital	0.465	15.289	32.908	0.464	125.977	271.461
haggle	0.069	0.516	7.431	0.069	1.032	14.910
manufacturing-email	0.339	4.843	14.267	0.339	5.762	16.982
sp-highschool-2013	2.086	91.147	43.701	2.085	2247.218	1077.554

Addressing **RQ4**, a potential advantage of our method is that it facilitates *predictions* of temporal centrality node rankings that are much faster than the actual *calculation* of temporal centralities. Highlighting this, in table 3 we report the time needed (i) to fit our pretrained model to the test data, and (ii) to infer the temporal centrality prediction. While our approach requires to fit a k-th order De Bruijn graph model in the test data, this procedure only requires to calculate time-respecting paths of exactly length k, which is a simpler problem than the calculation of all shortest time-respecting paths. We compare the combined time of those two steps to the time required to calculate temporal closeness and betweenness centrality in the test graphs, for which we used the fastest known algorithms mentioned in section 2. The results show that our approach provides speed-up factors ranging from approx. 3.5 to 43.7 for temporal closeness and from approx. 6.4 to 1077 for temporal betweenness.

The corresponding speed-up tables for GCN, TGN and EVO can be found in appendix D. Being a much simpler model, the static GCN model provide a higher speed-up (but considerably worse performance). Similarly, EVO provides higher speed-ups but worse predictions. We finally note that the temporal TGN model yields lower speed-ups than our method, despite giving worse predictions.

A potential criticism of our method could be that the size of a higher-order De Bruijn graph model can be considerably larger than a first-order graph, possibly making training and inference computationally expensive. To address this concern, in appendix B we report both the training and inference times of all models across all 13 data sets. We find that both the training and inference times of the DBGNN architecture are actually comparable to those of a static GCN. We attribute this to the fact that the DBGNN architecture provides a compact, *static but time-aware* De Bruijn graph representation of potentially large time series, rather than requiring a representation of all time-stamped edges. We further find that DBGNN has considerably lower training costs than both TGN or EVO.

Considering **RQ5**, another aspect of our approach to use a time-aware but *static* graph neural network is that the hidden layer activations yield *static* embeddings that are based on the *causal topology* of

 https://doi.org/10.52202/079017-0949

temporal graphs. This causal topology is influenced by (i) the topology of links, and (ii) their timing and temporal ordering. To explain the favorable performance of our model compared to a static GCN, we hypothesize that nodes for which our model learns similar embeddings also have more similar temporal centralities, compared to embeddings generated by a GCN model. To test this, we apply a dimensionality reduction to the node activations generated by the last 8-dimensional bipartite layer in the DBGNN architecture, comparing it to the representation obtained from (i) the last message passing layer of a GCN model and (ii) an EVO embedding. In fig. 4 in appendix I we show the resulting embeddings for one representative prediction of temporal closeness and betweeness in the eu-email-dept4 data, where the color gradient highlights ground truth node centralities in the test data. The plot shows that the time-aware DBGNN architecture better captures the ranking of nodes compared to a time-neglecting GCN as well as the EVO embeddings. We again attribute this to the end-to-end learning approach provided by DBGNN compared to the two-step approach of EVO.

5 Conclusion

In summary, we investigate the problem of predicting temporal betweenness and closeness centralities in temporal graphs. We use a recently proposed time-aware graph neural network architecture, which relies on higher-order De Bruijn graph models of time-respecting paths. An empirical study in which we compare our approach with a time-neglecting static graph neural network demonstrates the potential of our method. We find that our approach considerably outperforms other time-aware graph learning techniques that (i) either do not consider time-respecting paths, or (ii) do not provide an end-to-end approach where the learning of node representations is integrated with the prediction task. A comparative analysis in 13 empirical temporal graphs highlights differences between static and temporal centralities that are likely due to the underlying temporal patterns, and shows that our model is generally better at predicting temporal closeness compared to betweenness. A scalability analysis reveals that our prediction approach provides a considerable speed-up compared to the exact calculation of temporal node centralities, yielding speed-up factors between 3.5 and 1077. We finally investigate (static) embeddings produced by the last message passing layer of our architecture and show that they better capture temporal centralities compared to GCN.

Open questions and future work Our work necessarily leaves open questions that should be addressed in future work. Rather than optimizing the predictive performance of our model, the focus of the present work was to highlight the potential of time-aware graph neural networks for temporal centrality prediction. We thus have not performed an exhaustive optimization of hyperparameters such as, e.g., the maximum time difference δ, the maximum order k of the De Bruijn graphs used in the DBGNN architecture, or the number and width of graph convolutional layers. While we do report optimal values across three learning rates for all models, a more thorough investigation of the influence of those hyperparameters is future work. Moreover, we did not utilize additional node features like, e.g., node degrees, static centralities, or node embeddings that could further improve our results. Another aspect that we have not studied in our work is the impact of the size of the training data, i.e. how little training data is sufficient to predict temporal centralities with reasonable accuracy, and where the trade-offs in the choice of the training size are. An interesting further question is whether our approach could be adapted to support a fully *inductive* setting, i.e. to train our model on a set of dynamic graphs and then use the trained model to predict temporal centralities in other, previously unseen networks. Similarly, for some data sets with non-stationary temporal patterns it could be beneficial to train a De Bruijn Graph Neural Network based on a sliding window approach, hence adjusting the model (and predictions) as time progresses. Such a combination of the concepts behind TGN and DBGNN could yield better results in a number of practical settings.

We believe that our work is of high practical relevance for applications of knowledge discovery and machine learning in time-stamped relational data. For dynamic social network analysis, our approach allows to quickly estimate temporal centralities whose calculation is computationally expensive. More generally, our study highlights the potential of *compact, static but time-aware graph neural network architectures* for node-level regression in temporal graphs.

Acknowledgments

IS acknowledges support by the Swiss National Science Foundation (SNF), grant no. 176938 and the German Federal Ministry of Education and Research, grant no. 031L0311A (TissueNet).

References

[1] A. Alsayed and D. J. Higham. Betweenness in time dependent networks. *Chaos, Solitons & Fractals*, 72:35–48, 2015.

[2] D. A. Bader, S. Kintali, K. Madduri, and M. Mihail. Approximating betweenness centrality. In *Algorithms and Models for the Web-Graph: 5th International Workshop, WAW 2007, San Diego, CA, USA, December 11-12, 2007. Proceedings 5*, pages 124–137. Springer, 2007.

[3] A. Badie-Modiri, M. Karsai, and M. Kivelä. Efficient limited-time reachability estimation in temporal networks. *Phys. Rev. E*, 101:052303, May 2020. doi: 10.1103/PhysRevE.101.052303. URL `https://link.aps.org/doi/10.1103/PhysRevE.101.052303`.

[4] A. Badie-Modiri, M. Karsai, and M. Kivelä. Efficient limited-time reachability estimation in temporal networks. *Physical Review E*, 101(5):052303, 2020.

[5] A. Bavelas. Communication patterns in task-oriented groups. *The journal of the acoustical society of America*, 22(6):725–730, 1950.

[6] C. Belth, F. Kamran, D. Tjandra, and D. Koutra. When to remember where you came from: Node representation learning in higher-order networks. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 222–225, 2019.

[7] E. Bergamini, H. Meyerhenke, and C. L. Staudt. Approximating betweenness centrality in large evolving networks. In *2015 Proceedings of the Seventeenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 133–146. SIAM, 2014.

[8] B. Blonder and A. Dornhaus. Time-ordered networks reveal limitations to information flow in ant colonies. *PloS one*, 6(5):e20298, 2011.

[9] U. Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[10] N. D. Bruijn. A combinatorial problem. In *Nederl. Akad. Wetensch., Proc. 49*, pages 461–467, 1946.

[11] S. Buß, H. Molter, R. Niedermeier, and M. Rymar. Algorithmic aspects of temporal betweenness. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 2084–2092, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403259. URL `https://doi.org/10.1145/3394486.3403259`.

[12] S. Buß, H. Molter, R. Niedermeier, and M. Rymar. Algorithmic aspects of temporal betweenness. *Network Science*, 12(2):160–188, 2024. doi: 10.1017/nws.2024.5.

[13] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408, 2012.

[14] A. Casteigts, A.-S. Himmel, H. Molter, and P. Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802, 2021.

[15] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.

[16] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[17] M. Génois and A. Barrat. Can co-location be used as a proxy for face-to-face contacts? *EPJ Data Science*, 7(1):1–18, 2018.

 https://doi.org/10.52202/079017-0949

[18] F. Grando and L. C. Lamb. Estimating complex networks centrality via neural networks and machine learning. In *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-17, 2015*, pages 1–8. IEEE, 2015. doi: 10.1109/IJCNN.2015.7280334. URL https://doi.org/10.1109/IJCNN.2015.7280334.

[19] F. Grando, L. Z. Granville, and L. C. Lamb. Machine learning in network centrality measures: Tutorial and outlook. *ACM Comput. Surv.*, 51(5), oct 2018. ISSN 0360-0300. doi: 10.1145/3237192. URL https://doi.org/10.1145/3237192.

[20] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864. ACM, 2016. doi: 10.1145/2939672.2939754. URL https://doi.org/10.1145/2939672.2939754.

[21] M. Haghir Chehreghani. An efficient algorithm for approximate betweenness centrality computation. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1489–1492, 2013.

[22] P. Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88:1–30, 2015.

[23] P. Holme and J. Saramäki. Temporal networks. *Phys. Rep.*, 519(3):97 – 125, 2012. doi: 10.1016/j.physrep.2012.03.001. URL http://www.sciencedirect.com/science/article/pii/S0370157312000841.

[24] S. Huang, F. Poursafaei, J. Danovitch, M. Fey, W. Hu, E. Rossi, J. Leskovec, M. Bronstein, G. Rabusseau, and R. Rabbany. Temporal graph benchmark for machine learning on temporal graphs. *arXiv preprint arXiv:2307.01026*, 2023.

[25] L. Isella, J. Stehlé, A. Barrat, C. Cattuto, J.-F. Pinton, and W. Van den Broeck. What's in a crowd? analysis of face-to-face behavioral networks. *Journal of theoretical biology*, 271(1): 166–180, 2011.

[26] D. Kempe, J. Kleinberg, and A. Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 504–513, 2000.

[27] H. Kim and R. Anderson. Temporal node centrality in complex networks. *Physical Review E*, 85(2):026107, 2012.

[28] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[29] R. Lambiotte, M. Rosvall, and I. Scholtes. From networks to optimal higher-order models of complex systems. *Nature physics*, 15(4):313–320, 2019.

[30] H. H. K. Lentz, T. Selhorst, and I. M. Sokolov. Unfolding accessibility provides a macroscopic approach to temporal networks. *Phys. Rev. Lett.*, 110:118701, Mar 2013. doi: 10.1103/PhysRevLett.110.118701. URL http://link.aps.org/doi/10.1103/PhysRevLett.110.118701.

[31] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[32] R. Mastrandrea, J. Fournet, and A. Barrat. Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PloS one*, 10(9):e0136497, 2015. URL http://www.sociopatterns.org/datasets/high-school-dynamic-contact-networks/.

[33] M. R. F. Mendonça, A. Barreto, and A. Ziviani. Approximating network centrality measures using node embedding and machine learning. *IEEE Trans. Netw. Sci. Eng.*, 8(1):220–230, 2021. doi: 10.1109/TNSE.2020.3035352. URL https://doi.org/10.1109/TNSE.2020.3035352.

[34] M. Nurek and R. Michalski. Combining machine learning and social network analysis to reveal the organizational structures. *Applied Sciences*, 10(5):1699, 2020.

[35] R. K. Pan and J. Saramäki. Path lengths, correlations, and centrality in temporal networks. *Physical Review E*, 84(1):016105, 2011.

[36] A. Paranjape, A. R. Benson, and J. Leskovec. Motifs in temporal networks. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 601–610, 2017.

[37] T. Peixoto. Netzschleuder. http://networks.skewed.de/, 2024.

[38] L. V. Petrovic and I. Scholtes. Paco: Fast counting of causal paths in temporal network data. In J. Leskovec, M. Grobelnik, M. Najork, J. Tang, and L. Zia, editors, *Companion of The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 521–526. ACM / IW3C2, 2021. doi: 10.1145/3442442.3452050. URL https://doi.org/10.1145/3442442.3452050.

[39] R. Pfitzner, I. Scholtes, A. Garas, C. J. Tessone, and F. Schweitzer. Betweenness preference: Quantifying correlations in the topological dynamics of temporal networks. *Phys. Rev. Lett.*, 110: 198701, May 2013. doi: 10.1103/PhysRevLett.110.198701. URL http://link.aps.org/doi/10.1103/PhysRevLett.110.198701. https://doi.org/10.1103/PhysRevLett.110.198701.

[40] M. Pósfai and P. Hövel. Structural controllability of temporal networks. *New Journal of Physics*, 16(12):123055, 2014.

[41] L. Qarkaxhija, V. Perri, and I. Scholtes. De bruijn goes neural: Causality-aware graph neural networks for time series data on dynamic graphs. In *Learning on Graphs Conference*, pages 51–1. PMLR, 2022.

[42] M. Riondato and E. M. Kornaropoulos. Fast approximation of betweenness centrality through sampling. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 413–422, 2014.

[43] M. Riondato and E. Upfal. Abra: Approximating betweenness centrality in static and dynamic graphs with rademacher averages. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(5):1–38, 2018.

[44] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

[45] M. Rosvall, A. V. Esquivel, A. Lancichinetti, J. D. West, and R. Lambiotte. Memory in network flows and its effects on spreading dynamics and community detection. *Nature communications*, 5(1):4630, 2014.

[46] V. Salnikov, M. T. Schaub, and R. Lambiotte. Using higher-order markov models to reveal flow-based communities in networks. *Scientific reports*, 6(1):23194, 2016.

[47] D. Santoro and I. Sarpe. Onbra: Rigorous estimation of the temporal betweenness centrality in temporal networks. In *Proceedings of the ACM Web Conference 2022*, pages 1579–1588, 2022.

[48] I. Scholtes. When is a network a network?: Multi-order graphical model selection in pathways and temporal networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, CA, August 2017*, KDD '17, pages 1037–1046, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3098145. URL http://doi.acm.org/10.1145/3097983.3098145. http://doi.acm.org/10.1145/3097983.3098145.

[49] I. Scholtes, N. Wider, R. Pfitzner, A. Garas, C. J. Tessone, and F. Schweitzer. Causality-driven slow-down and speed-up of diffusion in non-markovian temporal networks. *Nature Communications*, 5:5024, September 2014. doi: 10.1038/ncomms6024. URL http://www.nature.com/ncomms/2014/140924/ncomms6024/full/ncomms6024.html. https://doi.org/10.1038/ncomms6024.

[50] I. Scholtes, N. Wider, and A. Garas. Higher-order aggregate networks in the analysis of temporal networks: path structures and centralities. *The European Physical Journal B*, 89(3):61, 2016. ISSN 1434-6036. doi: 10.1140/epjb/e2016-60663-0. URL http://dx.doi.org/10.1140/epjb/e2016-60663-0. http://dx.doi.org/10.1140/epjb/e2016-60663-0.

[51] J. Tang, M. Musolesi, C. Mascolo, V. Latora, and V. Nicosia. Analysing information flows and key mediators through temporal centrality metrics. In *Proceedings of the 3rd Workshop on Social Network Systems*, pages 1–6, 2010.

[52] I. Tsalouchidou, R. Baeza-Yates, F. Bonchi, K. Liao, and T. Sellis. Temporal betweenness centrality in dynamic graphs. *International Journal of Data Science and Analytics*, 9:257–272, 2020.

[53] P. Vanhems, A. Barrat, C. Cattuto, J.-F. Pinton, N. Khanafer, C. Régis, B.-a. Kim, B. Comte, and N. Voirin. Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. *PloS one*, 8(9):e73970, 2013. URL http://www.sociopatterns.org/datasets/hospital-ward-dynamic-contact-network/.

A Details on empirical datasets

Table 4: Overview of time series data sets used in the experiment evaluation

data set	Description	Ref	Nodes	Edges	Temporal Edges	Directed	δ
ants-1-1	Ant Antenna interactions, colony 1 - filming 1	[8]	89	947	1,911	True	30 sec
ants-1-2	Ant Antenna interactions, colony 1 - filming 2	[8]	72	862	1,820	True	30 sec
ants-2-1	Ant Antenna interactions, colony 2 - filming 1	[8]	71	636	975	True	30 sec
ants-2-2	Ant Antenna interactions, colony 2 - filming 2	[8]	69	769	1,917	True	30 sec
company-emails	E-Mail exchanges in manufacturing company	[34]	167	5,784	82,927	True	60 mins
eu-email-dept2	E-Mail exchanges in EU institution (dept 2)	[36]	162	1,772	46,772	True	60 mins
eu-email-dept3	E-Mail exchanges in EU institution (dept 3)	[36]	89	1,506	12,216	True	60 mins
eu-email-dept4	E-Mail exchanges in EU institution (dept 4)	[36]	142	1,375	48,141	True	60 mins
sp-hospital	Face-to-face interactions in a hospital	[53]	75	1,139	32,424	False	60 mins
sp-hypertext	Face-to-face interactions at conference	[25]	113	2,498	20,818	False	60 mins
sp-workplace	Face-to-face interactions in a workspace	[17]	92	755	9,827	False	60 mins
sp-highschool	Face-to-face interactions in a highschool	[32]	327	5,818	188,508	False	60 mins
haggle	Human proximity recorded by smart devices	[15]	274	2,899	28,244	False	1 min

B Comparison of training and inference times across models

In the following, we investigate the training and inference times of all models for all of the 13 data sets. We specifically compare both the training and the inference times of our DBGNN-based architecture with those of the baseline methods GCN, EVO, and TGN. For EVO, the training time is dominated by the time required to compute embeddings, while the time required to train the subsequent feed-forward neural network is negligible. The inference time for EVO is exclusively based on the inference time of the feed-forward neural network. The results for betweenness and closeness centrality are shown in table 5 and table 6, respectively. Both tables show that the computational requirements of the DBGNN and GCN model are comparable, both during training and inference. The training of the EVO and TGN-based models requires substantially more time. For EVO, this is due to the computational complexity of the embedding calculation, which requires to simulate random walks for the underlying node2vec embedding [20]. For TGN, this is due to the necessity to calculate ground truth centralities separately for each batch used in the per-batch training procedure.

Table 5: Training and inference time for betweenness centrality in seconds

	Train				Test			
	DBGNN	GCN	EVO	TGN	DBGNN	GCN	EVO	TGN
ants-1-1	6.3023	5.1324	1.3095	35.7600	0.0026	0.0021	0.0037	0.1140
ants-1-2	6.0507	5.1753	1.4937	31.9830	0.0022	0.0019	0.0038	0.1010
ants-2-1	6.0295	5.1794	1.4455	4.9620	0.0023	0.0019	0.0040	0.0340
ants-2-2	6.3356	5.2399	1.3833	17.2250	0.0023	0.0018	0.0037	0.0590
eu-email-dept4	6.0213	5.0822	1.2874	386.1770	0.0024	0.0019	0.0035	0.7480
eu-email-dept2	6.0819	5.1082	1.2877	730.1510	0.0025	0.0019	0.0036	1.3380
eu-email-dept3	6.0829	5.2514	1.2815	129.3010	0.0024	0.0020	0.0033	0.3520
sp-workplace	6.4887	5.2195	1.4293	27.6400	0.0030	0.0019	0.0038	0.0960
sp-hypertext	6.0371	5.2276	1.2776	86.9330	0.0027	0.0020	0.0037	0.1970
sp-hospital	5.9835	5.0898	1.2724	166.2600	0.0025	0.0020	0.0036	0.3190
haggle	6.0887	5.1281	1.2939	110.9990	0.0028	0.0020	0.0036	0.2520
manufacturing-email	6.3259	5.2400	1.5247	567.5910	0.0030	0.0021	0.0036	1.0010
sp-highschool-2013	6.5781	4.7907	1.5916	1360.5520	0.0036	0.0020	0.0044	2.2450

C ONBRA results

The results of the ONBRA model for the betweenness centrality are shown in table 7

D Speed-up

Tables 8, 9 and 10 show the speed-ups of the prediction model GCN, TGN and EVO compared to the exact calculation of the node centralities

https://doi.org/10.52202/079017-0949

Table 6: Training and inference time for closeness centrality in seconds

	Train				Test			
	DBGNN	GCN	EVO	TGN	DBGNN	GCN	EVO	TGN
ants-1-1	6.3638	5.4887	1.3095	33.7800	0.0026	0.0021	0.0037	0.1100
ants-1-2	6.2833	5.4537	1.4937	18.9370	0.0025	0.0021	0.0038	0.0620
ants-2-1	6.3401	5.5003	1.4455	16.7240	0.0024	0.0020	0.0040	0.0550
ants-2-2	6.3093	5.4556	1.3833	33.8390	0.0026	0.0020	0.0037	0.1060
eu-email-dept4	6.3398	5.4577	1.2874	694.4370	0.0025	0.0021	0.0035	1.3430
eu-email-dept2	6.3222	5.4565	1.2877	371.8710	0.0027	0.0022	0.0036	0.8190
eu-email-dept3	6.3126	5.4473	1.2815	129.2170	0.0028	0.0022	0.0033	0.3340
sp-workplace	6.5001	5.5629	1.4293	52.0420	0.0029	0.0020	0.0038	0.1300
sp-hypertext	6.5624	5.5350	1.2776	94.0060	0.0030	0.0021	0.0037	0.2140
sp-hospital	6.5881	5.5314	1.2724	171.0810	0.0032	0.0021	0.0036	0.3350
haggle	6.6309	5.5206	1.2939	116.0810	0.0030	0.0021	0.0036	0.2640
manufacturing-email	6.5784	5.6821	1.5247	801.6310	0.0032	0.0022	0.0036	1.4180
sp-highschool-2013	6.5746	5.6309	1.5916	1879.3700	0.0030	0.0022	0.0044	3.2890

Table 7: MAE and Spearman rank correlation for ONBRA estimation of betweenness centrality

	MAE	Spearmanr	Time
ants-1-1	261.31289 $\pm$ 0.00151	0.95429 $\pm$ 0.0062	0.0181 $\pm$ 0.00031
ants-1-2	34.95047 $\pm$ 0.00076	0.8344 $\pm$ 0.03744	0.0111 $\pm$ 0.00011
ants-2-1	4.08689 $\pm$ 0.00019	0.56815 $\pm$ 0.10243	0.0068 $\pm$ 7e-05
ants-2-2	45.82778 $\pm$ 0.00011	0.99184 $\pm$ 0.00178	0.42462 $\pm$ 0.01373
eu-email-dept4	4.64783 $\pm$ 0.00011	0.39658 $\pm$ 0.05351	0.06626 $\pm$ 0.00334
eu-email-dept2	3.23876 $\pm$ 0.00012	NaN	0.07984 $\pm$ 0.00226
eu-email-dept3	2.80843 $\pm$ 0.00013	0.39598 $\pm$ 0.06837	0.01875 $\pm$ 0.0023
sp-workplace	94.92572 $\pm$ 0.00498	0.48874 $\pm$ 0.13345	0.15363 $\pm$ 0.38683
sp-hypertext	NaN	NaN	NaN
sp-hospital	NaN	NaN	NaN
haggle	13.73402 $\pm$ 0.00026	0.40592 $\pm$ 0.10224	0.06889 $\pm$ 0.00925
manufacturing-email	84.03212 $\pm$ 0.00071	0.65181 $\pm$ 0.0414	0.24526 $\pm$ 0.02451
sp-highschool-2013	831.53583 $\pm$ 0.0007	0.48636 $\pm$ 0.0479	2.93658 $\pm$ 0.56495

E Scalability

The computational complexity of our model is linear in the number of time-respecting paths of length two in the temporal graph. This number can be bounded above by the number of paths of length two in the (static) graph, which can be theoretically bounded by $n \cdot \lambda_1^2$, where n is the number of nodes and λ_1 is the largest eigenvalue of the adjacency matrix of the (undirected) static graph. We note that for the fully connected graph with the special case of $\lambda_1 = n$ we obtain an upper bound n^3 for the number of time-respecting paths of length two. This corresponds to all length three sequences of n nodes. For more details see [38].

F Additional results

In the following, we provide additional experimental results, namely the optimal order of a k-th order De Bruijn graph model, inferred using the statistical model selection approach from [48] (table 11), additional results for the number of hits among the top-ranked nodes for betweenness and closeness centrality (table 12 and table 13). We also provide the MAE scores in tables 14 and 15 for the betweenness and closeness centrality across all models.

G Details on Hyperparameters and Computational Resources

In table 16 - table 18 we provide further details on the neural network architecture that we used for our experiments with DBGNN, GCN and EVO. For DBGNN and GCN, we tested different learning rates between 0.1 and 0.001 using an ADAM optimizer with a weight decay of $5 \cdot 10^{-4}$.

Table 8: Speed-up of GCN

	Betweenness			Closeness		
	Inference time	Centrality	Speed-up	Inference time	Centrality	Speed-up
ants-1-1	0.0021	0.2876	134.7530	0.0021	0.0678	32.2630
ants-1-2	0.0019	0.1070	55.5940	0.0021	0.0558	26.2340
ants-2-1	0.0019	0.0450	24.3330	0.0020	0.0293	14.6190
ants-2-2	0.0018	0.1156	65.1200	0.0020	0.0554	27.0050
eu-email-dept4	0.0019	1.4759	766.7580	0.0021	1.2011	577.7060
eu-email-dept2	0.0019	1.8833	990.9080	0.0022	1.5004	674.2110
eu-email-dept3	0.0020	0.3086	156.1860	0.0022	0.1911	85.8470
sp-workplace	0.0019	4.9610	2576.6020	0.0020	1.5769	807.7110
sp-hypertext	0.0020	100.5170	50256.4670	0.0021	5.9080	2778.1840
sp-hospital	0.0020	125.9772	62201.1900	0.0021	15.2888	7367.6280
haggle	0.0020	1.0321	515.9920	0.0021	0.5155	248.3130
manufacturing-email	0.0021	5.7621	2710.2200	0.0022	4.8435	2196.1540
sp-highschool-2013	0.0020	2247.2180	1136946.6840	0.0022	91.1466	41897.8210

Table 9: Speed-up of TGN

	Betweenness			Closeness		
	Inference time	Centrality	Speed-up	Inference time	Centrality	Speed-up
ants-1-1	0.1140	0.2876	2.5230	0.1100	0.0678	0.6160
ants-1-2	0.1010	0.1070	1.0600	0.0620	0.0558	0.8990
ants-2-1	0.0340	0.0450	1.3240	0.0550	0.0293	0.5330
ants-2-2	0.0590	0.1156	1.9590	0.1060	0.0554	0.5220
eu-email-dept4	0.7480	1.4759	1.9730	1.3430	1.2011	0.8940
eu-email-dept2	1.3380	1.8833	1.4080	0.8190	1.5004	1.8320
eu-email-dept3	0.3520	0.3086	0.8770	0.3340	0.1911	0.5720
sp-workplace	0.0960	4.9610	51.6780	0.1300	1.5769	12.1300
sp-hypertext	0.1970	100.5170	510.2390	0.2140	5.9080	27.6080
sp-hospital	0.3190	125.9772	394.9130	0.3350	15.2888	45.6380
haggle	0.2520	1.0321	4.0960	0.2640	0.5155	1.9530
manufacturing-email	1.0010	5.7621	5.7560	1.4180	4.8435	3.4160
sp-highschool-2013	2.2450	2247.2180	1000.9880	3.2890	91.1466	27.7130

For the feed forward neural network applied to the 16-dimensional embeddings generated by EVO, we used a fully connected network with ReLu activation functions, and input layer with 16 dimensions and a hidden layer with eight dimensions. We trained the model for 2000 epochs testing different learning rates between 0.01 and 0.0001 with an ADAM optimizer with weight decay of $5 \cdot 10^{-4}$.

As hyperparameters of the TGN architecture, we used learning rates between 0.01 and 0.0001 with an ADAM optimizer with weight decay of $5 \cdot 10^{-4}$ and 200 epochs. We further tested window sizes $k \in \{3, 5, 7, 10\}$, batch sizes between 100 and 600 and memory dimensions between 50 and 70.

All experiments were run on two dedicated workstation machines. The first workstation had an AMD Ryzen 9 7950X 16-core CPU with 32 GB of RAM and Nvidia RTX 4090 GPU. The second machine was equipped with an AMD Ryzen 9 7900X 12-Core CPU with 32 GB of RAM and Nvidia RTX 4080 GPU.

H Static vs temporal centralities

Let $G = (V, E)$ be a (static) graph, where V is a set of vertices or nodes and $(v, w) \in E$ are potentially directed edges or links from node v to w. Let us further consider weighted graphs, where we have a function $w : E \to \mathbb{N}$ that assigns integer weights to edges. In a static network $G = (V, E)$, we define a path (or walk) of length l from v_0 to v_l as any sequence of nodes $v_0, \ldots, v_l$ iff $(v_{i-1}, v_i) \in E$ for $i = 1, \ldots, l$. If every node occurs only once in the sequence, we call the sequence a *simple* path. A shortest path between two nodes v and w is a (not necessarily unique) path of legth l such that all other paths from v to w have length $l' \geq l$.

In static networks, shortest paths between pairs of nodes allow us to define *path-based nodes centralities*, which can be used to identify influential nodes. Here, we briefly introduce two important path-based centrality measures, namely *betweenness and closeness centrality*. For static networks

https://doi.org/10.52202/079017-0949

Table 10: Speed-up of EVO

	Betweenness			Closeness		
	Inference time	Centrality	Speed-up	Inference time	Centrality	Speed-up
ants-1-1	0.0037	0.2876	76.8720	0.0037	0.0678	18.1150
ants-1-2	0.0038	0.1070	28.4500	0.0038	0.0558	14.8230
ants-2-1	0.0040	0.0450	11.2570	0.0040	0.0293	7.3250
ants-2-2	0.0037	0.1156	31.1560	0.0037	0.0554	14.9260
eu-email-dept4	0.0035	1.4759	424.8480	0.0035	1.2011	345.7490
eu-email-dept2	0.0036	1.8833	517.6840	0.0036	1.5004	412.4260
eu-email-dept3	0.0033	0.3086	93.6080	0.0033	0.1911	57.9480
sp-workplace	0.0038	4.9610	1302.1110	0.0038	1.5769	413.8860
sp-hypertext	0.0037	100.5170	26984.4380	0.0037	5.9080	1586.0440
sp-hospital	0.0036	125.9772	35100.9190	0.0036	15.2888	4259.9130
haggle	0.0036	1.0321	286.0590	0.0036	0.5155	142.8800
manufacturing-email	0.0036	5.7621	1594.8320	0.0036	4.8435	1340.5670
sp-highschool-2013	0.0044	2247.2180	514590.7920	0.0044	91.1466	20871.6800

Table 11: Result of detection of optimal order based on likelihood ratio test.

data set	K_{opt} train	K_{opt} val
ants-1-1	2	2
ants-1-2	2	2
ants-2-1	1	1
ants-2-2	2	2
company-emails	2	2
eu-email-4	1	1
eu-email-2	2	2
eu-email-3	1	1
sp-hospital	2	2
sp-hypertext	2	2
sp-workplace	2	2
sp-highschool	2	2
haggle	2	2

Table 12: Results for hitsIn5 and hitsIn10 for prediction of temporal betweenness centrality and learning rate for which each experiment performed best

	DBGNN			GCN		
Experiment	lr	hitsIn30	hitsIn5	lr	hitsIn30	hitsIn5
ants-1-1	0.001	17.55 ± 1.468	2.45 ± 0.887	0.001	14 ± 2.34	0.75 ± 0.786
ants-1-2	0.001	20.7 ± 1.867	2.1 ± 0.553	0.001	18.6 ± 1.984	1.7 ± 0.801
ants-2-1	0.001	17.3 ± 1.342	1.85 ± 0.813	0.001	14.6 ± 2.137	0.8 ± 0.834
ants-2-2	0.100	18.3 ± 1.867	1.05 ± 0.605	0.001	14.8 ± 3.915	0.55 ± 0.686
eu-email-dept4	0.010	14.2 ± 3.205	1.2 ± 1.152	0.100	9.8 ± 5.217	0.3 ± 0.733
eu-email-dept2	0.010	14.65 ± 1.785	1.4 ± 1.046	0.010	15.1 ± 2.315	1.3 ± 0.801
eu-email-dept3	0.010	18.1 ± 5.418	2.75 ± 1.209	0.100	17.75 ± 1.713	3.1 ± 0.912
sp-workplace	0.100	20.1 ± 1.586	1.85 ± 0.489	0.001	16.25 ± 1.517	1.1 ± 0.852
sp-hypertext	0.100	21.6 ± 1.353	2.1 ± 0.718	0.100	20.6 ± 0.821	3.2 ± 0.523
sp-hospital	0.010	24.9 ± 1.071	2.45 ± 0.51	0.010	24.65 ± 0.671	2.25 ± 0.55
haggle	0.001	25.75 ± 1.333	1.1 ± 0.852	0.001	26 ± 0.0	0 ± 0.0
manufacturing-email	0.010	19.05 ± 2.188	1.05 ± 0.826	0.100	13.7 ± 2.342	0.15 ± 0.489
sp-highschool-2013	0.100	13.7 ± 1.922	1.3 ± 0.865	0.001	7.2 ± 2.118	0.25 ± 0.444

without temporal interactions the betweenness centrality of a node v is calculated as

$$c_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

where $\sigma_{s,t}$ is the number of the shortest paths between nodes s and t and $\sigma_{s,t}(v)$ is the number of such paths that pass through node v. In other words the node is considered central if there are many

Table 13: Results for hitsIn5 and hitsIn10 for prediction of temporal closeness centrality and learning rate for which each experiment performed best

	DBGNN			GCN		
Experiment	lr	hitsIn30	hitsIn5	lr	hitsIn30	hitsIn5
ants-1-1	0.100	25.7 ± 0.865	3.45 ± 0.51	0.001	25.65 ± 0.587	2 ± 0.0
ants-1-2	0.010	27.2 ± 0.894	3.95 ± 0.224	0.001	19.9 ± 0.308	3 ± 0.0
ants-2-1	0.100	28.05 ± 0.394	5 ± 0.0	0.001	24 ± 0.0	4 ± 0.0
ants-2-2	0.001	26.35 ± 0.671	3.6 ± 0.681	0.010	23.1 ± 0.641	1.35 ± 0.587
eu-email-dept4	0.001	24.25 ± 1.164	3.9 ± 0.308	0.100	10.25 ± 4.327	0.45 ± 0.759
eu-email-dept2	0.100	22.45 ± 1.099	4.85 ± 0.366	0.001	11 ± 0.0	2 ± 0.0
eu-email-dept3	0.010	27.65 ± 0.489	4 ± 0.0	0.001	19.85 ± 0.587	3 ± 0.0
sp-workplace	0.001	26.35 ± 0.813	2.9 ± 0.308	0.001	16.65 ± 0.489	3 ± 0.0
sp-hypertext	0.001	26.3 ± 0.923	4.45 ± 0.51	0.001	18 ± 0.0	3.2 ± 0.41
sp-hospital	0.001	24.25 ± 0.444	3.1 ± 0.553	0.001	24 ± 0.0	2 ± 0.0
haggle	0.100	28.9 ± 0.447	4.3 ± 0.47	0.001	28 ± 0.0	0.45 ± 0.759
manufacturing-email	0.100	26.25 ± 0.639	4.55 ± 0.51	0.001	19 ± 0.0	0.5 ± 0.513
sp-highschool-2013	0.001	22.45 ± 0.686	3.7 ± 0.47	0.001	9.95 ± 0.224	0 ± 0.0

Table 14: MAE scores for betweenness centralities

	DBGNN	GCN	EVO	TGN
ants-1-1	237.617 ± 0.512	246.328 ± 0.524	122.885 ± 14.865	61.646 ± 1.978
ants-1-2	23.272 ± 1.344	33.6 ± 2.015	145.597 ± 0.663	31.422 ± 2.085
ants-2-1	3.346 ± 0.229	5.395 ± 1.01	147.795 ± 23.11	5.243 ± 0.188
ants-2-2	34.481 ± 1.55	36.855 ± 0.217	76.492 ± 28.231	30.475 ± 6.468
eu-email-dept4	5.512 ± 0.372	9.345 ± 3.485	174.706 ± 2.694	3.044 ± 0.167
eu-email-dept2	3.048 ± 0.313	4.824 ± 2.114	132.947 ± 4.404	2.097 ± 0.08
eu-email-dept3	2.285 ± 0.108	3.019 ± 0.478	209.543 ± 1.928	2.248 ± 0.249
sp-workplace	70.803 ± 3.242	83.534 ± 0.18	89.494 ± 9.508	2.334 ± 0.236
sp-hypertext	66.583 ± 3.113	137.715 ± 0.332	257.435 ± 12.054	98.36 ± 24.553
sp-hospital	24.254 ± 1.505	43.885 ± 0.188	87.532 ± 8.734	6.864 ± 0.667
haggle	11.282 ± 0.313	14.85 ± 0.597	227.847 ± 4.947	12.542 ± 0.673
manufacturing-email	59.722 ± 4.245	74.79 ± 0.128	180.904 ± 54.201	38.472 ± 1.825
sp-highschool-2013	496.086 ± 15.04	813.573 ± 0.338	781.268 ± 11.873	2.754 ± 0.028

Table 15: MAE scores for closeness centralities

	DBGNN	GCN	EVO	TGN
ants-1-1	1164.711 ± 14.402	1597.068 ± 0.392	122.885 ± 14.865	372.877 ± 18.343
ants-1-2	183.19 ± 7.212	870.068 ± 0.33	145.597 ± 0.663	262.986 ± 16.023
ants-2-1	50.552 ± 2.196	406.514 ± 0.379	147.795 ± 23.11	173.593 ± 2.885
ants-2-2	120.622 ± 7.597	581.628 ± 0.596	76.492 ± 28.231	264.237 ± 17.492
eu-email-dept4	152.888 ± 10.107	603.253 ± 86.201	174.706 ± 2.694	254.053 ± 13.355
eu-email-dept2	277.821 ± 28.434	1285.315 ± 0.327	132.947 ± 4.404	400.356 ± 42.539
eu-email-dept3	70.476 ± 5.577	807.134 ± 0.285	209.543 ± 1.928	320.185 ± 4.476
sp-workplace	343.833 ± 12.627	1646.676 ± 0.377	89.494 ± 9.508	297.783 ± 30.86
sp-hypertext	1481.244 ± 48.473	5804.967 ± 0.312	257.435 ± 12.054	1559.56 ± 146.031
sp-hospital	600.69 ± 17.705	2286.325 ± 0.438	87.532 ± 8.734	244.128 ± 8.619
haggle	365.089 ± 17.982	2404.175 ± 0.412	227.847 ± 4.947	1507.635 ± 69.119
manufacturing-email	1244.012 ± 52.211	5652.538 ± 0.219	180.904 ± 54.201	892.742 ± 13.726
sp-highschool-2013	8000.875 ± 514.726	32138.809 ± 0.305	781.268 ± 11.873	417.952 ± 4.803

Layer	Input dimensions	Output dimensions	Activation Function
GCNConv	$\|V\|$	16	Sigmoid
GCNConv	16	8	ELU
Linear layer	8	1	ELU

Table 16: Overview of proposed model architecture for simple GCN

https://doi.org/10.52202/079017-0949

Layer	Input dimensions	Output dimensions	Activation Function
GCNConv first order	$\|V\|$	16	Sigmoid
GCNConv second order	$\|E\|$	16	Sigmoid
Bipartite layer	16	8	ELU
Linear layer	8	1	ELU

Table 17: Overview of proposed model architecture for DBGNN

Layer	Input dimensions	Output dimensions	Activation Function
Linear Layer	16	8	ReLU
Linear Layer	8	1	ReLU

Table 18: Overview of proposed model architecture for EVO

shortest paths that pass through the node. The closeness centrality on the other hand is defined as

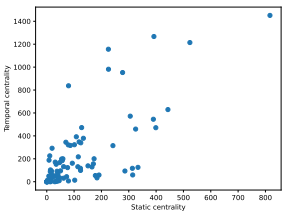
$$c_C(v) = \frac{1}{\sum_{u \in V} d(u, v)}$$

where $d(u, v)$ describes the distance (length of the shortest path) of node u to node v. Thus, in terms of closeness a node is considered more central if the overall distance to all other nodes in the graph is relatively small.

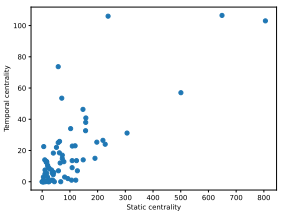
To contrast the temporal path-based centralities defined in section 2 with the corresponding static centralities defined above, in fig. 2 and fig. 3 we plot the temporal vs. static betweenness and closeness centralities of all nodes for all 13 empirical temporal graphs considered in our work (cf. table 4).

I Visualization of Node embeddings

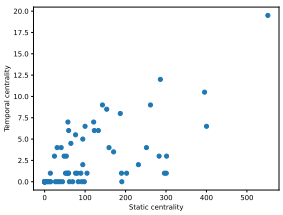
The plots in fig. 4 show the node emebddings obtained for a GCN and DBGNN model trained to predict temporal closeness centrality (a, b) as well as temporal betweenness centrality (c, d) for the `eu-email-4` data set.



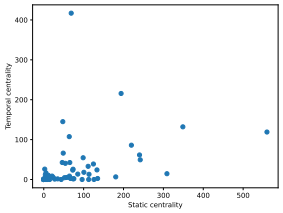
(a) ants-1-1



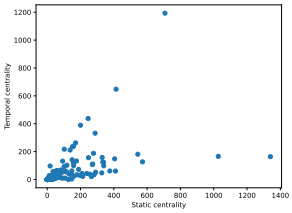
(b) ants-1-2



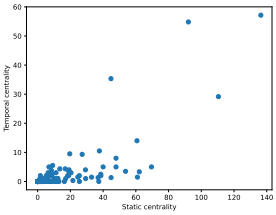
(c) ants-2-1



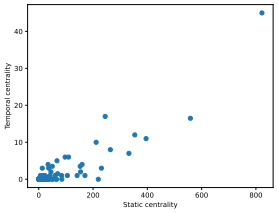
(d) ants-2-2



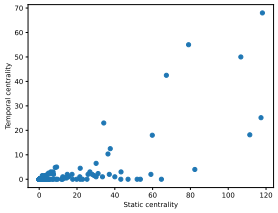
(e) company-emails



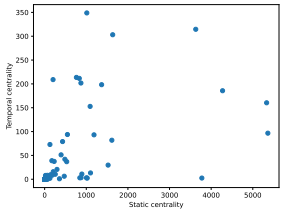
(f) eu-email-dept2



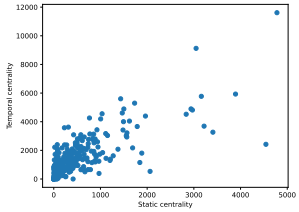
(g) eu-email-dept3



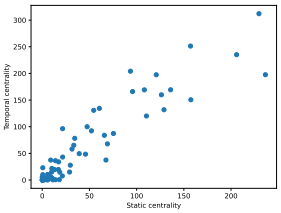
(h) eu-email-dept4



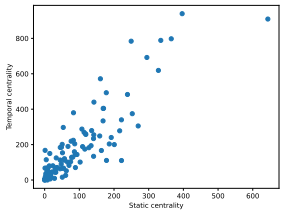
(i) haggle



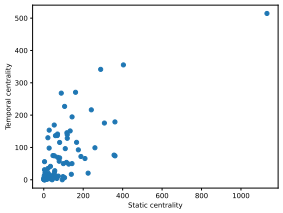
(j) sp-highschool



(k) sp-hospital



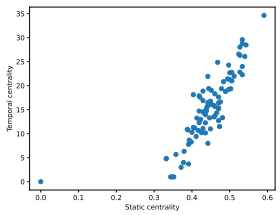
(l) sp-hypertext



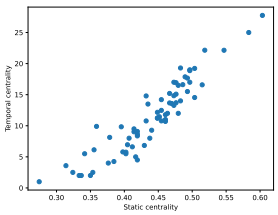
(m) sp-workplace

Figure 2: Static vs temporal betweenness centralities of all nodes in 13 empirical dynamic graphs

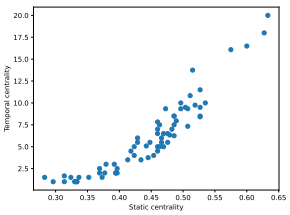
https://doi.org/10.52202/079017-0949



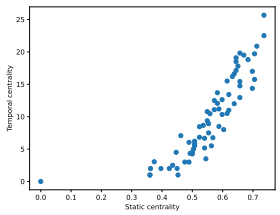
(a) ants-1-1



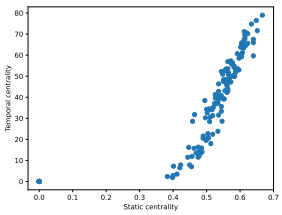
(b) ants-1-2



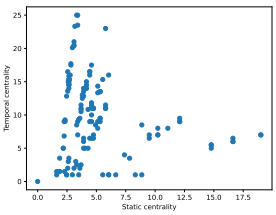
(c) ants-2-1



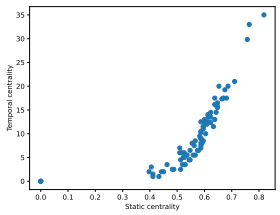
(d) ants-2-2



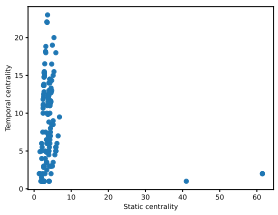
(e) company-emails



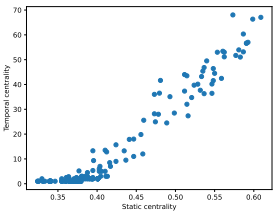
(f) eu-email-dept2



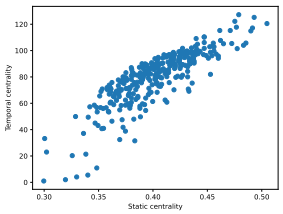
(g) eu-email-dept3



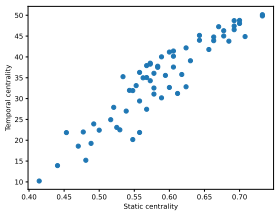
(h) eu-email-dept4



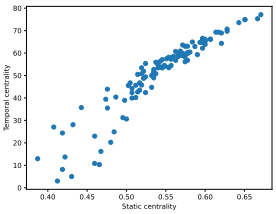
(i) haggle



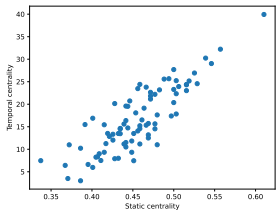
(j) sp-highschool



(k) sp-hospital

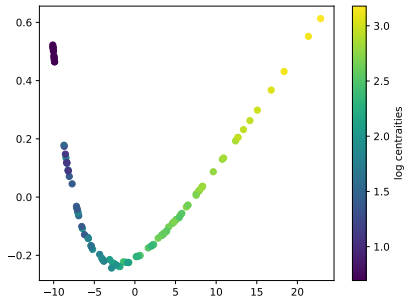


(l) sp-hypertext

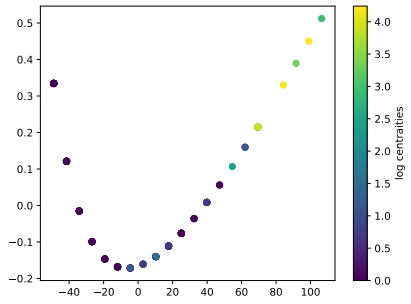


(m) sp-workplace

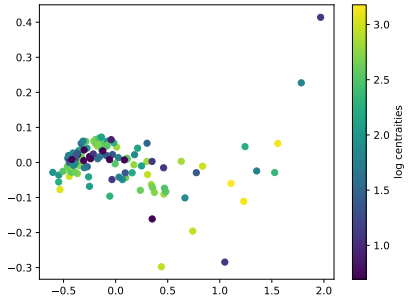
Figure 3: Static vs temporal closeness centralities of all nodes in 13 empirical dynamic graphs



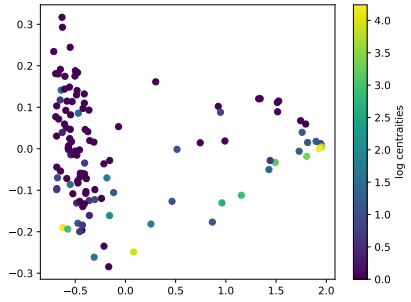
(a) Embedding of nodes based on DBGNN model trained for prediction of temporal closeness centrality in eu-email-dept4



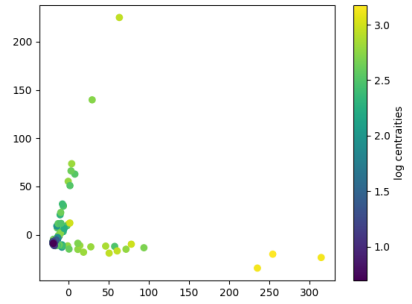
(b) Embedding of nodes based on DBGNN model trained for prediction of temporal betweenness centrality in eu-email-dept4



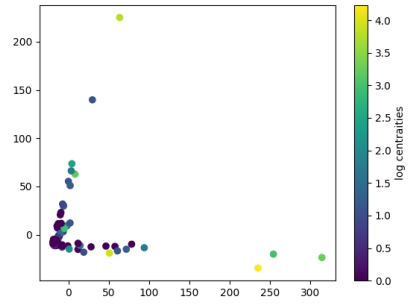
(c) Embedding of nodes based on GCN architecture trained for prediction of temporal closeness centrality in eu-email-dept4



(d) Embedding of nodes based on GCN architecture trained for prediction of temporal betweenness centrality in eu-email-dept4



(e) Embedding of nodes based on EVO trained for prediction of temporal closeness centrality in eu-email-dept4



(f) Embedding of nodes based on EVO trained for prediction of temporal betweenness centrality in eu-email-dept4

Figure 4: Comparison of node embeddings generated by DBGNN model (top), GCN (middle), and EVO (bottom) for the eu-email-dept4 data set. Nodes are colored according to their temporal closeness (left) and betweenness (right) centrality.

https://doi.org/10.52202/079017-0949

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: In the motivation, we provide a list of the contributions of our work. Moreover, in section 4 we list four research questions that are addressed by our experiments.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: The conclusion (section 5) of our manuscript includes a detailed paragraph that highlights open questions and issues left for future work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA] .

Justification: Our work does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes] .

Justification: In section 4 we include a detailed explanation of our experimental setup, a description of the (publicly available) data sets along with their sources (cf. table 4) as well as details on our training and evaluation procedure. We further include all details on the hyperparameters and how they have been chosen in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

https://doi.org/10.52202/079017-0949

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] .

   Justification: All data sets used in this paper are publicly available online. The code of the experiments is available at `https://doi.org/10.5281/zenodo.10202791`.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes] .

   Justification: In section 4 we include a detailed explanation of our experimental setup, the data splits, as well as details on our training and evaluation procedure. We further include all details on the hyperparameters and how they have been chosen in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes] .

   Justification: Our main results are arithmetic mean across 20 runs with a fixed manual seed (reported in the appendix) for the sake of reproducibility. We additionally report the standard deviation of results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes] .

   Justification: The computational resources used to obtain the experimental results are stated in the appendix of our work.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes] .

   Justification: Our works neither involves human subject nor the collection of data. All data sets used are freely available, not deprecated and have been collected in adherence with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA] .

https://doi.org/10.52202/079017-0949

Justification: This paper explores foundational research on the use of graph neural networks to predict node centralities in temporal graph data, which is neither tied to a specific application nor to a deployment.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA] .

Justification: Our work neither involves pretrained models nor scraped data sets that could pose safety risks and that thus would require special safeguards.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes] .

Justification: We properly cite the original collectors of all empirical data sets. We further properly cite the creators of the DBGNN architecture as well as the baseline models, whose code has been reused in our experiments.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA] .

    Justification: The paper does not provide new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: Our work does not make use of crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: Our work does not involve crowdsourcing nor research with human subjects.

https://doi.org/10.52202/079017-0949

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.