

Higher-Order Graph Convolutional Embedding for Temporal Networks

Xian Mo¹, Jun Pang², and Zhiming Liu¹(✉)

¹ College of Computer & Information Science
Southwest University, 400715 Chongqing, China
zhimingliu88@swu.edu.cn

² Faculty of Science, Technology and Medicine
Interdisciplinary Centre for Security, Reliability and Trust
University of Luxembourg, L-4364 Esch-sur-Alzette, Luxembourg

Abstract. Temporal networks are networks that edges evolve over time. Network embedding is an important approach that aims at learning low-dimension latent representations of nodes while preserving the spatial-temporal features for temporal network analysis. In this paper, we propose a spatial-temporal higher-order graph convolutional network framework (ST-HN) for temporal network embedding. To capture spatial-temporal features, we develop a truncated hierarchical random walk sampling algorithm (THRW), which randomly samples the nodes from the current snapshot to the previous one. To capture hierarchical attributes, we improve upon the state-of-the-art approach, higher-order graph convolutional architectures, to be able to aggregate spatial features of different hops and temporal features of different timestamps with weight, which can learn mixed spatial-temporal feature representations of neighbors at various hops and snapshots and can well preserve the evolving behavior hierarchically. Extensive experiments on link prediction demonstrate the effectiveness of our model.

Keywords: Temporal networks · representation learning · higher-order graph convolutional network · spatial-temporal features · link prediction

1 Introduction

In recent years, network science has become very popular for modeling complex systems and is applied to many disciplines, such as biological networks [4], traffic networks [14], and social networks [17]. Networks can be represented graphically: $G = \langle V, E \rangle$, where $V = \{v_1, \dots, v_n\}$ represents a set of nodes, and n is the number of nodes in the network, and $E \subseteq \{V \times V\}$ represents a set of links (edges). However, in the real world, most networks are not static but evolve with time, and such networks are called temporal networks [2]. A temporal network can be defined as $G_t = \langle V, E_t \rangle$, which represents a network $G = \langle V, E \rangle$ evolving over time and generates a sequence of snapshots $\{G_1, \dots, G_T\}$, where $t \in \{1, \dots, T\}$ represents the timestamps. The key point for temporal network analysis is how

to learn useful temporal and spatial features from network snapshots at different time points [5]. One of the most effective temporal network analysis approaches is temporal network embedding, which aims to map each nodes of each snapshot of the network into a low-dimensional space. Such a temporal network embedding method is proved to be very effective in link prediction, classification and network visualisation [12]. However, one of the biggest challenges in temporal network analysis is to reveal the spatial structure at each timestamp and the temporal property over time [5].

Many network embedding methods have been proposed in the past few years [4]. DeepWalk [10] adopt neural network to learn the representation of nodes. GraphSAGE [7] leverages node feature information to efficiently generate node embeddings for previously unseen data. But both methods focus on static networks. In order to obtain temporal network embedding, the following methods have been proposed in the literature. BCGD [17] only captures the spatial features, and LIST [16] and STEP [2] capture the spatial-temporal features by matrix decomposition. However, they cannot represent the highly nonlinear features [13] due to the fact that they are based on matrix decomposition. At present, the emergence of deep learning techniques brings new insights into this field. tNodeEmbed [12] learns the evolution of a temporal network’s nodes and edges over time, while DCRNN [8] proposes a diffusion convolutional recurrent neural network to captures the spatio-temporal dependencies. To achieve effective traffic prediction, STGCN [14] replaces regular convolutional and recurrent units that integrating graph convolution and gated temporal convolution. The flexible deep embedding approach (NetWalk) [15] utilises an improved random walk to extract the spatial and temporal features of the network. More recently, DySAT [11] computes node representations through joint self-attention along with the two dimensions of the structural neighborhood and temporal dynamics, and dyngraph2vec [6] learns the temporal transitions in the network using a deep architecture composed of dense and recurrent layers. However, these methods are not considered learning mixed spatial-temporal feature representations of neighbours at various hops and snapshots. Therefore, the representation ability of temporal networks is still insufficient.

To tackle the aforementioned problems, we propose in the current paper a spatial-temporal Higher-Order Graph Convolutional Network framework (ST-HN) for temporal network embedding hierarchically (the overview of ST-HN is described in Figure 1). Some work [10] shows that extracting the spatial relation of each node can be used as a valid feature representation for each node. Moreover, the current snapshot topological structure of temporal networks is derived from the previous snapshot topology, it is necessary to combine the previous snapshot to extract the spatial-temporal features for the current snapshot. Inspired by these ideas, we develop a truncated hierarchical random walk sampling algorithm (THRW) to extract both spatial and temporal features of the network, which randomly samples the nodes from the current snapshot to the previous one and it can well extract networks’ spatial-temporal features. Because snapshots closer to the current snapshot contributes more to the current snapshot for tem-

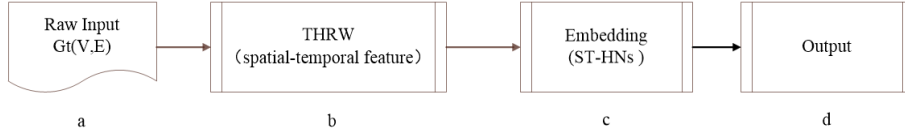


Fig. 1. Overview of the framework ST-HN. (a) Model input which is a temporal network G_t . (b) A spatial-temporal feature extraction layer which extracts $\Gamma(v, t_{Sta}, t_{End})$ of each node v of each network snapshot. (c) An embedding layer (ST-HNs) which maps each node of each snapshot to its D-dimensional representation. (d) Model output: Y_t with $t \in \{1, \dots, T\}$, where each $Y_t \in R^{N \times k}$ (N is the number of nodes and K is the dimension) is the representation of G_t ;

poral features. The THRW also incorporates a decaying exponential to assign longer walk length to more recent snapshots, which can better preserve the evolving behavior of temporal networks. Inspired by social networks, a user’s friends contain some information, and friends of their friends have different information, all of which are useful, so we should consider these features together. Then, we improve upon the state-of-the-art approach, higher-order graph convolutional architectures, to embed the nodes, which can aggregate spatial-temporal features hierarchically and further reinforces the time-dependence for each snapshot. Besides, it can learn mixed spatial-temporal feature representations of neighbors at various hops and snapshots. Finally, we test the embedded vector’s performance on link prediction task to verify its performance.

Our major contributions in this work can be summarised as follows. (1) We propose a model ST-HN to perform temporal network embedding. The model improves upon a Higher-Order Graph Convolutional Architecture (MixHop) [1] to hierarchically aggregate temporal and spatial features, which can better learn mixed spatial-temporal feature representations of neighbours at various hops and snapshots and can further reinforces the time-dependence for each network snapshot. (2) We also propose the THRW method for both spatial and temporal feature extraction. It adopts a random walk to sample neighbours for the current node v from the current snapshot to the previous snapshots, which can well extract networks’ spatial-temporal features. It also incorporates a decaying exponential to assign longer walk length to more recent snapshots, which can better preserve the evolving behavior of temporal networks. (3) Extensive experiments on link prediction demonstrate that our ST-HN consistently outperforms a few state-of-the-art baseline models.

2 Related Work

In this section, we briefly summarise related work for temporal network embedding. Network embedding for static networks has been extensively studied (e.g., see a recent literature survey [4]). Since networks are continually evolving with time in real-life, hence it is necessary to study network embedding

for temporal networks. One common approach is based on matrix decomposition to explore the spatial topology of the networks [17]. The main idea is that the closer two nodes are in the current timestamp, the more likely they are to form a link in the future timestamp. However, real-life networks are often evolving, methods considering only spatial information may have poor performance. There exist a few other methods focusing on both spatial and temporal evolution features such as STEP [2] and LIST [16]. STEP constructs a sequence of higher-order proximity matrices to capture the implicit relationships among nodes, while LIST defines the network dynamics as a function of time, which integrates the spatial topology and the temporal evolution. However, they have a limited ability to extract the correlation of high dimensional features since they are based on matrix decomposition. In recent years, neural network-based embedding methods have gained great achievements in link prediction and node classification [11]. tNodeEmbed [12] presents a joint loss function to learn the evolution of a temporal network’s nodes and edges over time. DCRNN [8] adopts the encoder-decoder architecture, which uses a bidirectional graph random walk to model spatial dependency and recurrent neural network to capture the temporal dependencies. STGCN [14] integrates graph convolution and gated temporal convolution through Spatio-temporal convolutional blocks to capture spatio-temporal features. The DySAT model [11] stacks temporal attention layers to learn node representations, which computes node representations through joint self-attention along with the two dimensions of the structural neighborhood and temporal dynamics, while dyngraph2vec [6] learns the temporal transitions in the network using a deep architecture composed of dense and recurrent layers, which learns the structure of evolution in dynamic graphs and can predict unseen links. NetWalk [15] is a flexible deep embedding approach, and it uses an improved random walk to extract the spatial and temporal features. However, these methods are not considered to learn mixed spatial-temporal feature representations of neighbours at various hops and snapshots. Therefore, the representation ability of these methods for temporal networks is still insufficient.

3 Problem Formulation

We introduce some definitions and formally describe our research problem.

Definition 1 (Node W -walks neighbours). Let $G = \langle V, E \rangle$ be a network. For a given node v , its W -walks neighbours are defined as the multi-set $N(v, W)$ containing all the nodes with W steps using a random walk algorithm on the network G .

Definition 2. Let $G_t = \langle V, E_t \rangle$ be a temporal network. For a node v , its all W -walks neighbours from time t_{Sta} up to time t_{End} (i.e., $t_{Sta} \leq t_{End}$) are defined as $\Gamma(v, t_{Sta}, t_{End}) = \bigcup_{t=t_{Sta}}^{t_{End}} \{N^t(v, W^t)\}$, where W^t represents the number of steps of random walk at timestamp t and $W^{t-1} = a * W^t$, where a represents the decaying exponential between 0 and 1, and $N^t(v, W^t)$ is a multi-set of W -walks neighbours of v in a network snapshot G_t where $t \in \{t_{Sta}, \dots, t_{End}\}$.

Algorithm 1: Truncated hierarchical random walk sampling (THRW)

Input : $G_t(V, E_t)$: a temporal network;
 W^t : the number of steps at snapshot t ;
 N : sampling temporal window size;
 a : decaying exponential;

Output: $X[i]$ with $i \in \{1, \dots, T\}$, where each $X[i]$ consists of the neighbour sets within the network snapshots of window size N for every node v ;

```

1 for  $i \in \{1, \dots, T\}$  do
2   if  $i - N \leq 0$  then
3     for  $v \in V$  do
4        $X[i].add(\Gamma(v, 1, i))$ 
5     end
6   else
7     for  $v \in V$  do
8        $X[i].add(\Gamma(v, i - N + 1, i))$ 
9     end
10  end
11 end

```

Temporal network embedding: For a temporal network $G_t = \langle V, E_t \rangle$, we can divide it evenly into a sequence of snapshots $\{G_1, \dots, G_t\}$ by timestamp t . For each snapshot G_t , we aim to learn a mapping function $f^t : v_i \rightarrow R^k$, where $v_i \in V$ and k represents dimensions and $k \ll |V|$. The purpose of the function f^t is to preserve the similarity between v_i and v_j on the network structure and evolution patterns of a given network at timestamp t .

4 Our Method

In this section, we introduce our model ST-HN. We firstly propose a truncated hierarchical random walk sampling algorithm (THRW) to extract both spatial and temporal features for each node (Section 4.1). Then, we use the proposed ST-HNs to embed the node for each snapshot (Section 4.2).

4.1 Spatial-temporal feature extraction

We propose the THRW algorithm (Algorithm 1) to sampling $\Gamma(v, t_{Sta}, t_{End})$ for each node v . Algorithm 1 has three parameters: W^t the number of steps in random walks of a given node v at snapshot t , N a sampling window size defining how many previous network snapshot are taken into account when sampling v 's nodes, and a is a decaying exponential defining the current snapshot to have more steps than the previous snapshot (see Definitions 1 and 2 for more details of these parameters). In Algorithm 1, each $X[i]$ represents the sets of nodes for all nodes in the network at time i , i.e. $\Gamma(v, i - N + 1, i)$, thus X contains all such sets, i.e., $X[i]$ with $i \in \{1, \dots, T\}$, for the complete network snapshot

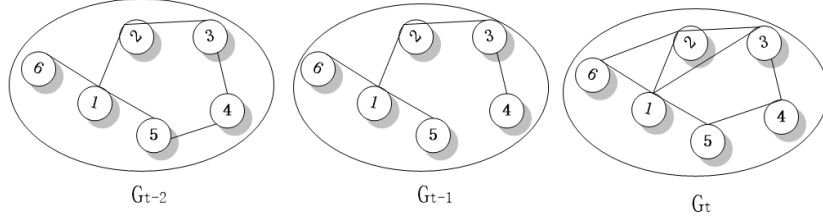


Fig. 2. The THRW algorithm: an illustrative example

sequence. The first layer loop is used to select snapshot at time i from the temporal network $G_t(V, E)$, and the purpose is to traverse all the snapshots. The algorithm extracts spatial and temporal features from N snapshots to better simulate the evolutionary behavior of the temporal network. If the number of the previous snapshots is greater than N , sampling neighbour nodes is performed between $i - N + 1$ and i snapshots. Otherwise, it is only sampled from the very first snapshot to the current snapshot i . The second loop is to sample the sets of neighbours for every node in the network. As shown in Figure 2, there are 6 nodes in the temporal network, and we extract features with the previous 2 snapshots for the node 1 in the current snapshot G_t . If we set $W^t=8$ and $a=0.5$, then for the snapshot G_t the walk length is 8 and the multi-set of sampled nodes is $\{1,2,3,1,5,4,3,1\}$. For the snapshot G_{t-1} , the walk length is 4 and the multi-set of sampled nodes is $\{1,2,3,2\}$, and for the snapshot G_{t-2} the walk length is 2 and the multi-set of sampled nodes is $\{1,2\}$. Then we combine the previous 2 snapshots sampled nodes as the final features for the node 1 of the snapshot G_t : $\{1,2,3,1,5,4,3,1,1,2,3,2,1,2\}$.

4.2 ST-HNs

Our proposed model refers to Higher-Order Graph Convolutional Architectures via Sparsified Neighborhood Mixing (MixHop) [1], which mixes feature representations of neighbors at various distances to learn neighborhood mixing relationships. More precisely, it combine 1-hop, 2-hop, \dots , neighbours in distinct feature spaces such that it can effectively aggregate features of different hops in the network. Each layer of MixHop is formally defined as follows:

$$H^{(i+1)} = \|\|_{j=0}^P \sigma(L^j H^{(i)} W_j^{(i)}) \quad (1)$$

where $H^{(i)} \in R^{N \times d_i}$ and $H^{(i+1)} \in R^{N \times d_{i+1}}$ are input and output of layer i . N represents the number of network nodes. $W_j^{(i)} \in R^{d_i \times d_{i+1}}$ is a weight matrix. σ is a nonlinear activation function. L^j is a symmetrically normalized Laplace matrix and can be constructed by $L^j = D^{-\frac{j}{2}} A D^{\frac{j}{2}}$. A is an adjacency matrix with self-connections: $A = A + I_N$, where I_N is a self-connections matrix. D is a diagonal degree matrix with $D_{mm} = \sum_n A_{mn}$, where m represents a row and n represents a column, and j is the powers of L , ranging from 0 to P . L^j

denotes the matrix L multiplied by itself j times, and \parallel denotes column-wise concatenation. For example, L^2 represents 2-hop neighbours in feature spaces. The MixHop model can learn features in difference feature space between immediate and further neighbours, hence it has a better representational capability. However, for temporal networks, MixHop cannot capture their temporal features. Our model, ST-HNs, improves upon MixHop and utilizes an aggregator to learn mixed spatial-temporal feature representations of neighbours at various hops and snapshots.

To solve the problem above, we propose Spatial-Temporal Higher-Order Graph Convolutional Temporal Network (ST-HNs), where each node of each snapshot aggregate spatial and temporal features from their 1-hop neighbours to P -hop neighbours through the previous A snapshot. In this way, it can learn mixed spatial-temporal feature representations of neighbours at various hops and snapshots. Our model is formally defined as follows:

$$H^i = \begin{cases} \parallel \{\oplus(Y_{t-A}, \dots, Y_{t-1}), X_t\} & \text{if } i = 0 \\ \parallel_{j=0}^P \sigma(L^j H^{(i-1)} W_j^{(i-1)}) & \text{if } i \in [1, \dots, M] \end{cases} \quad (2)$$

where P is the number of powers defining aggregation features of different hops in the network, M is the number of layers, \parallel represents concatenation operate, A is the temporal window size defining the number of the snapshots used to aggregate spatial-temporal features, and the operator \oplus represents an aggregator. We adopt the GRU aggregator to aggregate the temporal features of the previous A snapshots, which can further reinforce the time-dependence for each snapshot of temporal networks. The GRU aggregator is based on an GRU architecture [3]. The input is a number of network snapshots $\{Y_{t-A}, \dots, Y_{t-2}\}$ and the ground truth is the most recent snapshot Y_{t-1} , where the $Y_{t-1} \in R^{N \times k}$ is the representation of G_{t-1} , where N represents the number of nodes, k represents dimensions and $k \ll N$. Y_1 can be obtained from the MixHop (Equation 1) with M layers. If the number of the previous snapshots is greater than A , aggregation is performed between $t - A$ and $t - 1$ snapshots. Otherwise, it is only aggregate from the very first snapshot to the snapshot $t - 1$. We use the input and ground truth to train the GRU model and update parameter. After training, we shift the window one step forwards to obtain the representation of temporal features Y . The Y and X_t are concatenate to get the aggregated spatial-temporal feature H^0 , where $X_t \in R^{N \times d_0}$ is a feature matrix of each node of the snapshot at time t and it is obtained by the THRW algorithm (see Algorithm 1). The model input is the aggregated spatial-temporal feature H^0 . Through M times ST-HNs layers, the model output is $H^M = Z \in R^{N \times d_M}$, where Z is a learned feature matrix. The Z contains mixed spatial-temporal feature representations of neighbours at various hops and snapshots and it can better preserve the temporal and spatial features of temporal networks. For each snapshot, to learn representations, $z_u, u \in \mathbf{V}$, in a fully unsupervised setting, we apply a graph-based loss function to tune the weight matrices $w_j^i, j \in (1, \dots, P), i \in (1, \dots, M)$. The loss function encourages nearby nodes to have similar representations and enforces that the representations of disparate nodes are highly distinct:

Table 1. The statistics of four temporal networks.

Network	#Nodes	#Links	Clustering coefficient	Format
Hep-Ph	28,093	4,596,803	28.0%	undirected
Digg	30,398	87,627	0.56%	directed
Facebook wall posts	46,952	876,993	8.51%	directed
Enron	87,273	1,148,072	7.16%	directed

$$Loss = \frac{1}{N} \sum_{u=1}^N (z_u - Mean(Adj(z_u)))^2 \quad (3)$$

where N is the number of nodes, $Adj(z_u)$ obtains neighborhood node representation of u . $Mean$ means average processing.

5 Experiments

We describe the datasets and baseline models, and present the experimental results to show ST-HN’s effectiveness for link prediction in temporal networks.

5.1 Datasets and baseline models

We select four temporal networks from different domains in the KONECT project.³ All networks have different sizes and attributes. Their statistic properties present in Table 1. Hep-Ph dataset is the collaboration network of authors of scientific papers. Nodes represent authors and edges represent common publications. For our experiment, we select 5 years (1995 - 1999) and denote them as H_1 to H_5 . Digg dataset is the reply network of the social news website. Nodes are users of the website, and edges denote that a user replied to another user. We evenly merged it into five snapshots by day and denote it as D_1 to D_5 . Facebook wall posts dataset is the directed network of posts to other user’s wall on Facebook. The nodes represent Facebook users, and each directed edge represents one post. For our experiment, we combined 2004 and 2005 data into one network snapshot and defined it as W_1 . The rest of the data is defined as W_2 to W_5 by year and each snapshot contains a one-year network structure. Enron dataset is an email network that sent between employees of Enron between 1999 and 2003. Nodes represent employees and edges are individual emails. We select five snapshots in every half year during 2000-01 to 2002-06 and denote them as E_1 to E_5 . For our experiments, the last snapshot was used as ground-truth of network inference and the other snapshots was used to train the model.

Baselines: We compare ST-HNs with the following baseline models: LIST [16] describes the network dynamics as a function of time, which integrates the spatial and temporal consistency in temporal networks; BCGD [17] proposes a temporal

³ <http://konect.uni-koblenz.de/>

Table 2. Prediction results (AUC values).

Model	Hep-Ph	Digg	Facebook wall posts	Enron
BCGD	0.60	0.68	0.74	0.64
LIST	0.63	0.73	0.72	0.67
STEP	0.61	0.74	0.76	0.71
NetWalk	0.69	0.71	0.74	0.72
ST-HN	0.74	0.81	0.83	0.79

latent space model for link prediction, which assumes two nodes are more likely to form a link if they are close to each other in their latent space; STEP [2] utilises a joint matrix factorisation algorithm to simultaneously learn the spatial and temporal constraints to model network evolution; NetWalk [15] model updates the network representation dynamically as the network evolves by clique embedding, and it focuses on anomaly detection, and we adopt its representation vector to predict the link.

Parameter settings: In our experiments, we randomly generate the number of non-linked edges smaller than twice linked edges to ensure data balance [17]. For the Hep-Ph and Digg, we set the embedding dimensions as 256. For the Facebook wall posts dataset and Enron, we set the embedding dimensions as 512. For different datasets, the parameters for baselines are tuned to be optimal. Other settings include: the learning rate of the model is set as 0.0001; the number of powers P is set as 3; the number of steps W is set as 200; the temporal window A is set as 3; the sampling temporal window N is set as 3; the M for layers of the ST-HNs is set as 5; the decaying exponential a is set as 0.6. For the result of our experiments, we adopt the area under the receiver operating curve (AUC) to evaluate predictive power for future network behaviors of different methods and we carry out five times independently and reported the average AUC values for each dataset.

5.2 Experimental results

We compare the performance of our proposed model on the four datasets with four baselines for link prediction. We use ST-HNs to embed each node into a vector at each snapshot. Then we use the GRU to predict the vector representation of each node for the very last snapshot. In the end, we use the obtained representations to predict the network structure similar to [11]. Table 2 summarises the AUCs of applying different embedding methods for link prediction over the four datasets. Compared with other models, our model, ST-HN, achieves the best performance. Essentially, we use the THRW algorithm to sampling $\Gamma(v, t_{Sta}, t_{End})$ for each node v , which can better capture both spatial and temporal features for each node. It also incorporates a decaying exponential to assign longer walk length to more recent snapshots, which can better preserve the evolving behavior of temporal networks. Then we apply ST-HNs to embed spatial-temporal features of nodes, which can aggregate spatial-temporal features hierarchically and

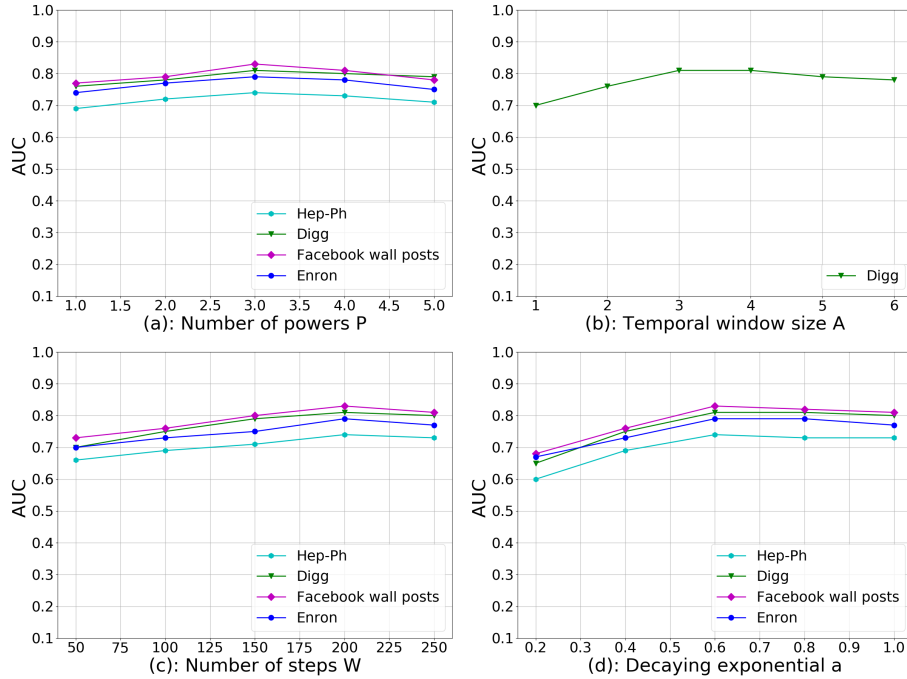


Fig. 3. Results of parameter sensitivity analysis.

learn mixed spatial-temporal feature representations of neighbours at various hops and snapshots and can further reinforcing the time-dependence for each snapshot. In this way, our embedding method can well preserve the evolving behaviour of the networks.

5.3 Parameter sensitivity analysis

We further perform parameter sensitivity analysis in this section, and the results are summarised in Figure 3. Specifically, we estimate how different the number of powers P and the temporal window A and the number of steps W and the the decaying exponential a can affect the link prediction results.

The number of powers P : We vary the power from 1 to 5, to prove the effect of varying this parameter. As P increases from 1 to 3, performance continues to increase due to the mixing of neighbour information with more hops in the current node. The best result is at $P = 3$ (the detail is described in Figure 3a). After which the performance decreases slightly or remains unchanged, while P continues to increase. The reason might be that the further away from the current node, the less information there is about the current node.

Temporal window size A : Due to the fact that the Digg dataset contains sixteen-days records, we split it by day and it will generates 16 snapshots. Since

the Digg dataset has more snapshots than others, we select 7 snapshots to analyze the parameter A . We vary the window size from 1 to 6 to check the effect of varying this parameter. The results show that the best results are obtained when $A = 3$. The reason might be that the closer snapshot is to the current snapshot, the more information can be captured about the current snapshot. But the accuracy no longer increases, when A continuously increases (see Figure 3b).

The number of steps W and the decaying exponential a : Since W and a jointly determine the sampling size of the current node, we analyse these two parameters together. When analysing W , a is set to 0.6. The experimental results show that the performance is the best when $W = 200$. As W increases from 50 to 200, performance continues to increase. The reason might be that the more nodes sampled, the more features of the current node are included. But the accuracy no longer increases, when W continuously increases. When analyzing a , W is set to 200. The experimental results show that the performance is the best when $a = 0.6$. As a increases from 0.2 to 0.6, performance continues to increase. The reason might be that the previous snapshot contains useful features for the current snapshot, as the number of sampled features increases. After which the performance decreases slightly or remains unchanged, while a continue to increase(the detail is described in Figure 3c and 3d).

6 Conclusion

We have proposed a new and effective framework, ST-HN, for temporal network embedding. Through intensive experiments we demonstrated its effectiveness in link prediction. In particular, we proposed the THRW algorithm to extract spatial-temporal features in each snapshot to model network evolution. Moreover, we proposed the ST-HNs framework to embed nodes in the network, which can learn mixed spatial-temporal feature representations of neighbours at various hops and snapshots and can well preserve the evolving behavior of the network hierarchically and can further reinforcing the time-dependence for each snapshot. Our future work will study performance improvements from different aggregation methods and aggregate other information [9] in temporal networks.

Acknowledgements. This work has been supported by the Chongqing Graduate Research and Innovation Project (CYB19096), the Fundamental Research Funds for the Central Universities (XDJK2020D021), the Capacity Development Grant of Southwest University (SWU116007), the China Scholarship Council (202006990041), and the National Natural Science Foundation of China (61672435, 61732019, 61811530327).

References

1. Abu-El-Haija, S., Perozzi, B., Kapoor, A., Harutyunyan, H., Alipourfard, N., Lerman, K., Steeg, G.V., Galstyan, A.: Mixhop: Higher-order graph convolution architectures via sparsified neighborhood mixing. In: Proceedings of the 36th International Conference on Machine Learning. pp. 21–29 (2019)

2. Chen, H., Li, J.: Exploiting structural and temporal evolution in dynamic link prediction. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 427–436. ACM (2018)
3. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. In: Proceedings of the 8th Workshop on Syntax, Semantics and Structure in Statistical Translation. p. 103–111 (2014)
4. Cui, P., Wang, X., Pei, J., Zhu, W.: A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* **31**(5), 833–852 (2018)
5. Cui, W., Wang, X., Liu, S., Riche, N.H., Madhyastha, T.M., Ma, K.L., Guo, B.: Let it flow: a static method for exploring dynamic graphs. In: Proceedings of the 2014 IEEE Pacific Visualization Symposium. pp. 121–128. IEEE (2014)
6. Goyal, P., Chhetri, S.R., Canedo, A.: dyngraph2vec: Capturing network dynamics using dynamic graph representation learning. *Knowledge-Based Systems* **187**, 104816 (2020)
7. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the Advances in Neural Information Processing Systems. pp. 1024–1034 (2017)
8. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: Proceedings of the 2018 International Conference on Learning Representations. pp. 1–16. IEEE (2018)
9. Pang, J., Zhang, Y.: DeepCity: A feature learning framework for mining location check-ins. In: Proceedings of the 11th International AAAI Conference on Web and Social Media. pp. 652–655 (2017)
10. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 701–710. ACM (2014)
11. Sankar, A., Wu, Y., Gou, L., Zhang, W., Yang, H.: Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In: Proceedings of the 13th International Conference on Web Search and Data Mining. pp. 519–527. ACM (2020)
12. Singer, U., Guy, I., Radinsky, K.: Node embedding over temporal graphs. In: Proceedings of the 2019 International Joint Conference on Artificial Intelligence. pp. 4605–4612. Morgan Kaufmann (2019)
13. Tenenbaum, J.B., De Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. *Science* **290**(5500), 2319–2323 (2000)
14. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. pp. 3634–3640. Morgan Kaufmann (2018)
15. Yu, W., Cheng, W., Aggarwal, C.C., Zhang, K., Chen, H., Wang, W.: Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 2672–2681. ACM (2018)
16. Yu, W., Wei, C., Aggarwal, C.C., Chen, H., Wei, W.: Link prediction with spatial and temporal consistency in dynamic networks. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp. 3343–3349 (2017)
17. Zhu, L., Dong, G., Yin, J., Steeg, G.V., Galstyan, A.: Scalable temporal latent space inference for link prediction in dynamic social networks. *IEEE Transactions on Knowledge and Data Engineering* **28**(10), 2765–2777 (2016)