Knowledge and Information Systems (2022) 64:2771–2795
https://doi.org/10.1007/s10115-022-01724-2

REGULAR PAPER

NEAWalk: Inferring missing social interactions via topological-temporal embeddings of social groups

Yinghan Shen[1,3] · Xuhui Jiang[1,3] · Zijian Li[1,3] · Yuanzhuo Wang[1,4] · Xiaolong Jin[2,3] · Shengjie Ma[5] · Xueqi Cheng[2,3]

Received: 23 February 2022 / Revised: 5 July 2022 / Accepted: 8 July 2022 /
Published online: 23 August 2022

Abstract
Real-world network data consisting of social interactions can be incomplete due to deliberately erased or unsuccessful data collection, which cause the misleading of social interaction analysis for many various time-aware applications. Naturally, the link prediction task has drawn much research interest to predict the missing edges in the incomplete social network. However, existing studies of link prediction cannot effectively capture the entangling topological and temporal dynamics already residing in the social network, thus cannot effectively reasoning the missing interactions in dynamic networks. In this paper, we propose the NEAWalk, a novel model to infer the missing social interaction based on topological-temporal features of patterns in the social group. NEAWalk samples the query-relevant walks containing both the historical and evolving information by focusing on the temporal constraint and designs a dual-view anonymization procedure for extracting both topological and temporal features from the collected walks to conduct the inference. Two-track experiments on several well-known network datasets demonstrate that the NEAWalk stably achieves superior performance against several state-of-the-art baseline methods.

Keywords Dynamic network completion · Dynamic graph representation learning · Social group · Anonymous walk

✉ Yuanzhuo Wang
wangyuanzhuo@ict.ac.cn

1 Data Intelligence System Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

2 Key Laboratory of Network data and Science & Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

3 University of Chinese Academy of Sciences, Beijing, China

4 Zhongke Big Data Academy, Zhengzhou, China

5 Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China

1 Introduction

Social interaction analysis through networks is widely used in various applications. For example, contact tracing by simultaneously modeling COVID-19 transmission is proved effective through the network of social interactions in physical approximation among infections and other individuals [1]. Most existing studies of social interaction analysis assume that the networks are complete, i.e., all interactions are available from data sources. However, the obtained interaction data are usually incomplete due to the reasons that part of data deliberately erased [2, 3] or unsuccessfully gathered by collectors [4, 5], which cause the misleading of social interaction analysis. The dynamic network completion task aims to predict the existence of queries in the form of (u, v, t) as missing interactions and can be applied for various time-aware applications. For instance, inferring the missing interactions with infections for teasing out the potential transmission chain of the virus can contribute to the control of outbreaks of COVID-19. Another practical application in fraud investigation [6, 7] is inferring the hidden trading behaviors of fraudsters to restore money transferring chains in financial transaction networks.

Link prediction methods have been researched for decades, and are widely used for predicting the existence of the edges in the social network. From the temporal perspective, link prediction methods fall into two categories [8–10]: the methods on static graphs [2, 4, 11, 12] and on dynamic graphs [13–15]. The former methods could not be directly applied to the dynamic network completion task because they only consider the graph structure and ignore the interlaced topological and temporal information contained in social networks. In this scenario, the links in static networks denote the "happened interactions", which could not reflect the frequency and the precise happening time of interactions.

To alleviate this problem, the latter methods take the temporal dynamics of interactions into account and capture the evolving laws from past to future in social networks. CTDNE [13] migrates the random walk method to dynamic graphs for learning the implicit temporal rules lying in the collected paths. TGN [14] designs a dynamic graph neural network to aggregate and encode the recent interactions to make the future prediction. Despite their success in accurately predicting future links, they only consider the history interactions and omit the evolving interactions which occur after the query. Moreover, they fail to capture the pattern information, which resides in the local neighborhoods called the social group in the individual sociology theories [16–18]. The patterns of the social group describe the form of interactions and give better interpretability over the model's inference process. An intuitive example is shown in Fig. 1. The upper triad forms after a introduces two friends u and v who do not know each other, followed by two interactions that (u, b, t_6) and (b, v, t_7) in the lower triad. Contrarily, once interaction (u, v, t_5) is missing, the unstable quadrangle pattern is formed rather than two stable triad forms. Making full use of the entangling topological and temporal patterns with a delicate model design becomes the key challenge of the dynamic network completion task.

To this end, we propose a novel model named neural network for encoding anonymous walks in behavioral context (abbreviated as NEAWalk), which can capture both topological and temporal features of patterns for the dynamic network completion task.

First, we introduce a new type of behavioral context walk (BCW), which comprehensively reflects the query-relevant information of the social group. In the BCW sampling procedure, we focus on collecting the sequencing order of the input query and each interaction from both historical and evolving timelines. Evidently, the patterns should describe general prediction laws, which can be applied to arbitrary queries and non-relevant to specific nodes. Then, we

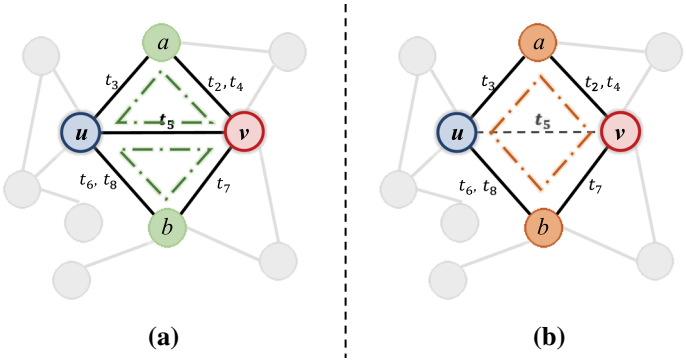


Fig. 1 Patterns of social group in social network. **a** shows stable pattern consisting of two triads, while **b** demonstrates the quadrangle unstable pattern of missing interaction

design the dual-view anonymization procedure for converting BCWs to anonymous walks of behavioral context (AWBC), which are not related to node features, and aim to extract the generalized features of patterns in the social group. Next, the topo-temp learning module encodes both topological and temporal pattern embeddings of AWBCs and aggregates them for predicting the probability of missing interaction. Specifically, to effectively capture the temporal displacement in walks, our proposed temporal-GRU unit encodes the sequence and the magnitude of the displacement. Finally, to validate the effectiveness of NEAWalk, we conducted both static and dynamic track experiments on five real-world social network datasets to prove that NEAWalk can be effectively generalized to different social networks. The experimental results in Table 2 demonstrate that the performance on AP/AUC metrics of NEAWalk is 720% superior to best baselines on the dynamic graph track and meanwhile achieve best results in three datasets on the static graph track.

It is worthwhile to highlight our contributions as follows:

(1) This paper provides a dynamic graph representation learning model NEAWalk, which could comprehensively leverage the abundant topological and temporal information in social groups to infer missing interactions. Specially, the behavioral context walk is introduced to describe the query-related information of social groups from both historical and evolving timelines.
(2) The dual-view anonymization procedure is designed to extract the high-quality topological and temporal features of patterns, which are independent of specific nodes in the query for universality.
(3) We conduct extensive experiments on real-world and publicly available social network datasets, for verifying the effectiveness of NEAWalk on the dynamic network completion task. Experimental results show that our NEAWalk can learn informative and high-quality representations for the query and achieves better performance over state-of-the-art baselines on the static and dynamic graph tracks.

2 Related work

2.1 Link prediction in social networks

The social network is one of the favorite means to perform social interactions and exchange information. Link prediction is a crucial task in social network analysis to infer the potential links between nodes, and can be applied in scenarios like IoT network analysis [19, 20], criminal network analysis [21], fraud detection [6, 7]. The objective of link prediction in the social network can be divided into two categories: find missing interactions in static graphs and predict future interactions in dynamic graphs [22].

Existing link prediction methods on the static graph [9, 10, 22, 23] aim to predict the probability of missing links. These methods model dynamic social interactions as temporal edges, and can be divided into similarity-based methods [24, 25], probabilistic-based methods [26], matrix factorization-based methods [12, 27] and GNN-based methods [28, 29]. Yet straightforward as it is to use atemporal links to model social interactions, however, static graphs fail to take temporal information of dynamic social networks into account. Due to lack of temporal information, the inferred interaction is remained to be unknown as that happened but missed, or is about to occur in the future.

Dynamic graphs can naturally model social networks and other dynamically evolving systems with temporal features. The link prediction task on dynamic graphs [35] aims to predict the likelihood of future interaction based on the historical interaction data. The dynamic graph representation learning methods [13–15, 21] learn evolving laws from history interactions and generate node embeddings for predicting future interactions. STGSN [21] transforms the whole dynamic graph into a sequence of static graphs by predefined time interval, and applies graph convolution layers and attention mechanism for learning the evolving laws of snapshots. CTDNE [13] generalizes the Skip-Gram architecture for learning time-preserving embeddings. TGN [14] integrates GNN-type aggregation rule with time-dependent node state vectors and time encoding information. CAW [15] utilizes causal anonymous walks to produce relative identity embeddings that have a tailor-made inductive bias. Although these models can infer dynamic social interactions, two problems have not been addressed: (1) Only the history data are considered in these methods, and the evolving information which occurs after the query is omitted. (2) These methods fail to capture pattern features in the local neighborhoods.

2.2 Mesoscopic-level information of social network

Mesoscopic-level information of social network is an intermediate level of the social world between individuals and entire social structures, denoting the group, cluster, or community in the social network. In the social sciences, a social group can be defined as two or more people who interact, describing the mesoscopic level information of social networks. The individual sociology theories [16–18] have proven that the behaviors are affected by the social group in which the individual belongs, i.e., its local neighborhoods. [30] explores mesoscopic features of social interactions from six real-world dynamic social networks and discovers mesoscopic level patterns such as “Star” and “Ordered-chain,” which involves entangled topological and temporal features.

Temporal motif [15, 31–33] concerns the coupling topological–temporal features of a small group of nodes and interactions within the hop range or time range, accurately reflecting the information of dynamic social networks at the mesoscopic level. A k-node, l-edge, δ-

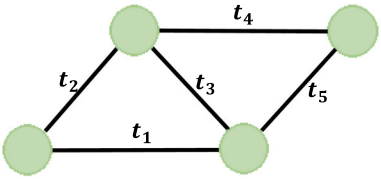


Fig. 2 A four-node, five-edge, δ-temporal motif, and $t_5 - t_1 \leq \delta$

temporal motif [34] comprises a sequence of l edges, which involves k nodes. The edges $M = [(u_1, v_1, t_1), (u_2, v_2, t_2)..., (u_l, v_l, t_l)]$ are time-ordered and temporal displacements are within a δ duration, i.e., $t_1 < t_2 < ... < t_l$ and $t_l - t_1 \leq \delta$. A example of temporal motif is shown in Fig. 2. User-specified temporal motifs can be helpful to better understand individuals' behavior. For example, triad motifs distinguish the formation of circles of friends in social networks, and loop motifs are associated with money laundering in transaction networks. However, when lacking precise ad hoc knowledge of the datasets, it is impractical to design appropriate temporal motifs [36], and it is hard to guarantee that the designed motifs are effective in specific scenarios. In addition, the temporal motif matching problem is proved to be NP-complete [33] with high time and space consumption. In the light of these issues, it is unrealistic to directly utilize temporal motifs as features to infer the missing interaction.

The random walk-based graph representation learning methods [37–39, 42] provide a flexible and unsupervised way to collect mesoscopic view information, which includes topological and temporal information. The random walk can be seen as a receptive path, and the length of the walk implies the radius of reception on mesoscopic level information. The anonymous walk [38, 39] on static graphs refines topological features from random walks and can reconstruct local neighborhoods [38]. The anonymous walk is defined as follows:

Definition 2.1 (Anonymous Walk, AW) Given a random walk $w = (n_1, n_2, ..., n_l)$, the anonymous walk for w is defined as:

$aw(w) = (\text{DIS}(w, n_1), \text{DIS}(w, n_2), ..., \text{DIS}(w, n_l))$,

where $\text{DIS}(w, n_i)$ denotes the the number of distinct nodes in w when n_i first appears in w: $\text{DIS}(w, n_i) = |\{n_1, n_2, ..., n_p\}|$, $p = \min_j\{n_j = n_i\}$.

In dynamic graphs, temporal walk [13] describes an interaction chain in temporal sequence. A l-length temporal walk starting from node n_1 to n_{l+1} is represented as a series of interactions $W = [(n_1, n_2, t_1), (n_2, n_3, t_2), ...(n_l, n_{l+1}, t_l)]$. Different from the random walk, the two adjacent interactions in temporal walk comply with the temporal restriction $t_i \leq t_{i+1}$, under the constraining of the unidirectionality of time. Our NEAWalk applies anonymous walk to extract both topological and temporal from collected temporal walks to replace temporal motifs to represent the mesoscopic features of social networks.

3 Problem statement

Given an incomplete dynamic social network G and a set of possible missing interactions $\hat{E}$, the dynamic network completion task aims to predict the existence of interactions in $\hat{E}$.

Specifically, the incomplete dynamic social network can be represented as $G \subseteq (V, E, T)$. Technically, G is a multi-graph where may exist multiple interactions between the same pair of nodes. V is the node set, and $E \subseteq V \times V \times R^+$ is the edge set containing the observed social interaction data. The social interaction $(u, v, t) \in E$ consists of two nodes u and v and timestamp t, denoting that u and v have an interaction at timestamp $t \in T$. The

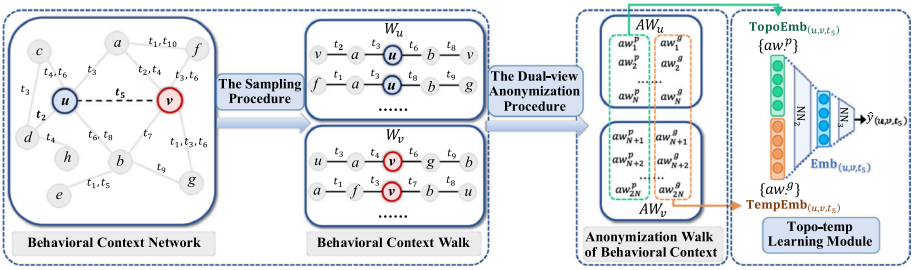


Fig. 3 A comprehensive flowchart of NEAWalk. Starting from the query (u, v, t_5), first, the sampling procedure captures the mesoscopic level information of the query in social network through exploring with behavioral context walks. Next, the dual-view anonymization procedure extracts topological and temporal features as forms of aw^p and aw^g from collected BCWs in $\{W_u, W_v\}$. Last, the topo-temp learning module encodes the topological and temporal pattern embeddings for assembling the query embedding for predicting

missing interactions $\hat{E}$ and the observed interactions E have no intersection. $T = [t_{\min}, t_{\max}]$ is the observed temporal field of G, where $t_{\min}$ and $t_{\max}$ are the minimum and maximum timestamp of interactions in E, respectively. In our work, we consider that all interactions are bidirectional. The types of social interaction include not only interactions among users in online social networks, but also real-world social behaviors such as physical contact.

4 The NEAWalk model

The proposed NEAWalk model consists of three parts, namely the sampling procedure, the anonymization procedure, and the topo-temp learning module. Figure 3 shows the comprehensive view of NEAWalk.

4.1 Behavioral context and sampling procedure

4.1.1 Behavioral context

The interactions of u and v near time t, called behavioral context, should be considered with the query (u, v, t). This section introduces two related concepts of behavioral context: behavioral context network and behavioral context walk. Furthermore, the sampling procedure of behavioral context walk is also described. We firstly introduce the concept of behavioral context network as follows:

Definition 4.1 (Behavioral Context Network, BCN) A behavioral context network of node pair (u, v) within h hops is defined as $BCN(< u, v >, h)$. The node set $V_{\text{BCN}} \subseteq V$ consists of the u, v and the h-hop neighborhoods of u and v. The edge set E_{BCN} contains the interactions between nodes in V_{BCN}.

$BCN(< u, v >, h)$ describes the social group information of node pairs (u, v) within range h. However, $BCN(< u, v >, h)$ is not related with t, and the entangled topological and temporal features in BCN are not easy to be directly extracted and encoded. The reception theory of random walks [42] indicates that random walks can capture local features in a static graph since a walk can be seen as a receptive path of the structure near the starting node. Based on this, we propose the behavioral context walk (BCW), which reflects the

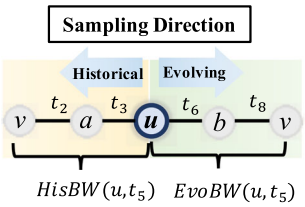


Fig. 4 BCW is sampled from bidirection, which includes historical and evolving directions

context information of t in BCN. A clear example of behavioral context walks sampled from $BCN(< u, v >, 2)$ is shown in the left part of Fig. 3. Theoretically, the information of a BCW with $2h$ length must be contained in a BCN with the size h. Therefore, we sample BCWs with $2h$ distance from the nodes u and v as BCW groups $\{W_u, W_v\}$ to describe the social group information of the query instead of BCN. Here, we give the formal definition of the behavioral context walk:

Definition 4.2 (Behavioral Context Walk, BCW) Behavioral Context Walk $BCW(u, t)$ is a sequence of interactions $[I_0, ..., I_{2h-1}]$, which starts sampling from node u at time t. h is the one-side sampling length of walk, and $I_i \in E$ denotes an interaction. I_i and I_{i+1} are two adjoining interactions in $BCW(u, t)$.

4.1.2 The BCW sampling procedure

The time-constraint sampling method [13] can effectively represent a feasible route for the temporal dependency information of interactions. Inspired by this, we further propose a BCW sampling method, which focuses on collecting the sequencing order of the input query and each interaction, and comprehensively reflects query-relevant information of the BCN from bidirectional timelines of t. Taking the timestamp t as the boundary, we distinguish historical interactions (happened before t) and evolving interactions (happening after t) in the sampling procedure. Therefore, the $BCW(u, t)$ consists of the historical behavior walk $HisBW(u, t) = [I_0, ..., I_{h-1}]$ and the evolving behavior walk $EvoBW(u, t) = [I_h, ...I_{2h-1}]$. As shown in Fig. 4, the timestamps of two adjoining interactions in $HisBW(u, t)$ are monotonically decreasing while increasing in $EvoBW(u, t)$. We propose a BCW sampling procedure in Algorithm 4.1 to collect the BCW group. The time complexity of the BCW sampling procedure is $O(Nh|E|)$, where the sampling of a single $HisBW(u, t)$ or $EvoBW(u, t)$ mainly depends on the number of observed interactions $|E|$.

In Algorithm 4.1, $\mathrm{T}(I)$ denotes the timestamp of interaction I. The softmax function $\mathrm{SM}(\cdot)$ assigns higher sampled probability to closer interaction:

$$\mathrm{SM}\,(I, TNS, Prev\_t) = \frac{\exp[(Prev\_t - \mathrm{T}(I))]}{\sum_{I' \in TNS} \exp\left[\mathrm{abs}(Prev\_t - \mathrm{T}(I'))\right]}, \tag{1}$$

where $\mathrm{abs}(\cdot)$ denotes the absolute value. The $\mathrm{Former\_Node}(\cdot)$ denotes the first node in the interaction, and the $\mathrm{Latter\_Node}(\cdot)$ is the last node in the interaction.

4.2 The dual-view anonymization procedure

This section will introduce the concept of the anonymous walk of behavioral context, and how the dual-view anonymization procedure refines topological and temporal features from BCW groups. Figure 5 demonstrates the dual-view anonymization procedure.

Algorithm 4.1: The BCW Sampling Procedure.

Input: (1) Target node u, (2) Time spot t, (3) One-side sampling length h, (4) BCW group size N, (5) Observed interaction set E.
Output: (1) The BCW group W_u.

Initialize $W_u = [\,]$;
For $iter = 1 : N$ **do**
  Initialize $HisBW = [\,]$, $EvoBW = [\,]$;
  $Prev\_t_h = t$, $Prev\_n_h = u$, $Prev\_t_e = t$, $Prev\_n_e = u$;
  For $hop = 1 : h$ **do**
    Construct historical temporal neighborhood set TNS_h from E.
    The $I' \in TNS_h$ conforms $\mathrm{T}(I') \leq Prev\_t_h$ and $\mathrm{Former\_Node}(I') = Prev\_n_h$;
    Sample $I \in TNS_h$ with $Pr(I) = \mathrm{SM}(I, TNS_h, Prev\_t_h)$;
    $HisBW = \mathrm{Concat}([I], HisBW)$;
    $Prev\_t_h = \mathrm{T}(I)$, $Prev\_n_h = \mathrm{Latter\_Node}(I)$;
    Construct evolving temporal neighborhood set TNS_e from E.
    The $I' \in TNS_e$ conforms $\mathrm{T}(I') \geq Prev\_t_e$ and $\mathrm{Former\_Node}(I') = Prev\_n_e$;
    Sample $I \in TNS_e$ with $Pr(I) = \mathrm{SM}(I, TNS_e, Prev\_t_e)$;
    $EvoBW = \mathrm{Concat}(EvoBW, [I])$;
    $Prev\_t_e = \mathrm{T}(I)$, $Prev\_n_e = \mathrm{Latter\_Node}(I)$;
  End For
  $BCW = \mathrm{Concat}(HisBW, EvoBW)$;
  $W_u = W_u \cup BCW$;
End For
return W_u;

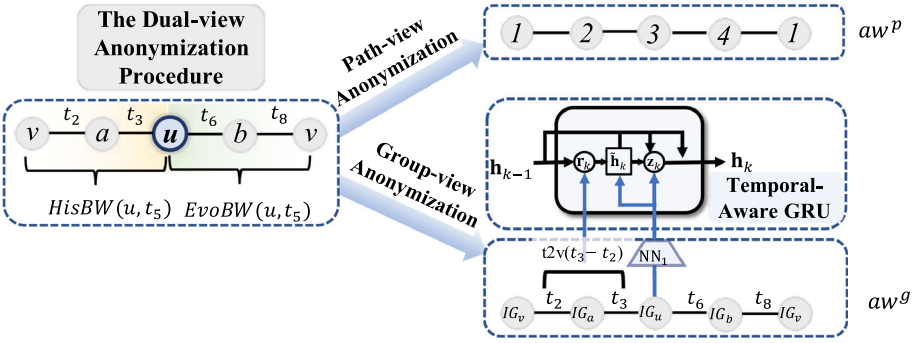


Fig. 5 Dual-view anonymization procedure of anonymous walk of behavioral context

The BCW groups $\{W_u, W_v\}$ reflect the social group information of the query (u, v, t). However, the pattern features should describe the generalized laws and not be associated with specific social groups. The node features are replaced with anonymous identifiers, which concern capturing local pattern features in a highly general manner. Considering this, we propose the concept of anonymous walk of behavioral context (AWBC), and design a dual-view anonymization procedure to extract topological and temporal features of patterns from BCWs, respectively. After the dual-view anonymization procedures, we obtain the AWBC groups $\{AW_u, AW_v\}$ from $\{W_u, W_v\}$. Here, we define AWBC as follows:

Definition 4.3 (Anonymous Walk of Behavioral Context, AWBC) The AWBC aw consists the path-view anonymous form and the group-view anonymous form: $aw = [aw^p, aw^g]$.

4.2.1 Path-view anonymization procedure

The path-view anonymization form provides the topological information of social groups. We adapt the anonymous walk concept in Definition 1 to generate the path-view anonymization form of BCW. In this procedure, nodes are replaced by their path-view position identifiers, which are the minimum indexes of the nodes appearing in the BCW. For the example shown in Fig. 4, given a BCW $w = [v, a, u, b, v]$, the path-view identifier of node a is 2. Hence, the path-view anonymous form of w is [1, 2, 3, 4, 1].

4.2.2 Group-view anonymization procedure

The social role is the relative distance of the individual from the others in the social group and reflects the individual's unique features. Therefore, we propose a group-view anonymization procedure to assign unique group-view identifiers for nodes. The group-view identifier is defined as an empirical distribution of normalized positional frequency in the BCW groups. We define the social group node set V_{SG}, which contains the nodes appearing in $\{W_{\mathrm{u}}, W_{\mathrm{v}}\}$. Given the node $n \in V_{\mathrm{SG}}$, the group-view node identifier IG_{n} with respect to $\{W_{\mathrm{u}}, W_{\mathrm{v}}\}$ is composed of two sub-identifiers IG_n^u and IG_n^v: $IG_{\mathrm{n}} = [IG_n^u, IG_n^v]$, where IG_n^u and IG_n^v are the group-view identifiers of W_{u} and W_{v}, respectively. For example, given the BCW groups $\{W_{\mathrm{u}}, W_{\mathrm{v}}\}$:

$$W_{\mathrm{u}} = \{[v, a, u, b, v], [f, a, u, b, g], [d, c, u, b, v], [v, a, u, b, g]\},$$
$$W_{\mathrm{v}} = \{[u, a, v, g, b], [a, f, v, b, u], [u, a, v, b, u], [a, f, v, g, b]\}.$$

The group-view identifier IG_a^u of node a is [0, 1, 0, 0, 0], for a appears 3 times in W_{u}, all appearing at the second positions. Similarly, IG_a^v is [0.5, 0.5, 0, 0, 0], for a appears 4 times in W_{v} at the first, second positions both 2 times, etc. The group-view identifier IG_{a} is the concatenation of IG_a^u and IG_a^v: [0, 1, 0, 0, 0, 0.5, 0.5, 0, 0, 0].

Finally, the aw^g is the sequence of the group-view identifier of nodes in the BCW. For example, the aw^g of $[v, a, u, b, v]$ is $[IG_v, IG_a, IG_u, IG_b, IG_v]$. The group-view anonymization is an inductive procedure because this procedure generates group-view identifiers for nodes through heuristic statistical methods, rather than relying on original node features.

4.3 Topo-temp learning module

For learning the topological and temporal features from the AWBC groups to infer missing interaction, we introduce the topo-temp learning module, which encodes the topological and temporal pattern embedding of the dual-view anonymous walks in $\{AW_{\mathrm{u}}, AW_{\mathrm{v}}\}$, and learns for inference.

4.3.1 Topological pattern embedding

The path-view anonymous form represents unique topological information of the social group. For example, the path-view anonymous walk [1, 2, 3, 1, 2] denotes an underlying triad dissimilar to the two unclosed triads in [1, 2, 3, 4, 1]. If the frequency of the latter walk is higher than the former walk, it denotes that the pattern is unstable, and more likely to be missing interactions.

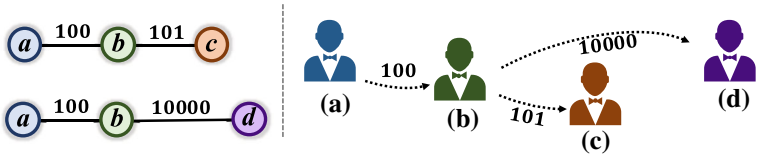


Fig. 6 Two group-view anonymous walks with different temporal displacements

To this end, we adapt the feature-based anonymous embedding method [39] to generate the topological pattern embedding via the empirical distribution of aw^p in $\{AW_{\mathrm{u}}, AW_{\mathrm{v}}\}$. First, we define path-view anonymous walk forms $awf_1, awf_2, \ldots$ according to their lexicographical order. For example, when $h = 2$, $awf_1 = (1, 2, 1, 2, 1)$, $awf_2 = (1, 2, 1, 2, 3)$, $awf_3 = (1, 2, 3, 1, 2)$, etc. We calculate the empirical distribution of awf_{k} as follows:

$$\hat{p}(awf_{\mathrm{k}}) = \frac{\sum_{aw_i^p \in \{AW_{\mathrm{u}}, AW_{\mathrm{v}}\}} \mathbb{I}(\mathrm{aw}(aw_i^p) = awf_k)}{2N}, \tag{2}$$

where $2N$ is the total number of aw^p in $\{AW_{\mathrm{u}}, AW_{\mathrm{v}}\}$. aw $\cdot$) is the function to project the path-view anonymous walk to corresponding form. $\mathbb{I}(\cdot)$ is the indicator function. The topological pattern embedding of $\{AW_{\mathrm{u}}, AW_{\mathrm{v}}\}$ is a η-dimentional vector:

$$\mathbf{TopoEmb}_{(u,v,t)} = [\hat{p}(awf_1), \hat{p}(awf_2), ..., \hat{p}(awf_\eta)], \tag{3}$$

where η is the total number of the possible path-view anonymous walk forms, which can be calculated by the length of aw^{p}.

4.3.2 Temporal pattern embedding

The aw_i^g represents the sequence of interactions acted by different social roles on the temporal dependency. Recurrent neural networks (RNN) suit for capturing the dependency of sequence data and can be applied to model aw_i^g. However, one major problem of RNN is that it records the orders of the interactions without considering the temporal displacement, which limits capturing the influences of different displacements on representing temporal patterns. Given two group-view anonymous walks in Fig. 6, the interaction $(a, b, 100)$ conveys more influential information from b to c than to node d, for the reason that the temporal displacement between c and b is smaller than d and b. In the former case, the dependency between the two interactions is stronger than the two interactions in the latter case. To this end, we propose the temporal-aware gated recurrent unit (GRU) architecture to encode the temporal pattern representation of the aw_i^g. Specifically, we modify the original recurrent unit in the GRU to capture the effects concerning the different temporal displacements and pass to the next recurrent unit.

First, for introducing the recurrent unit with the temporal displacement, we adapt the time2vec [41] method to project the real-value temporal displacement Δt to a d-dimensional embedding t2v(Δt) by Fourier Transform:

$$\mathrm{t2v}(\Delta t) = [\cos(\boldsymbol{w}_1 \Delta t + \boldsymbol{\phi}_1), ..., \cos(\boldsymbol{w}_d \Delta t + \boldsymbol{\phi}_d)], \tag{4}$$

where series of $\boldsymbol{w}$s and $\boldsymbol{\phi}$s are all weight parameters of the Fourier transform function.

At each time step, except for receiving the previous hidden state and the current input vectors, we also consider the temporal displacement from the last time step. In the temporal-aware GRU unit, we use a temporal displacement vector $\mathrm{t2v}(\Delta t)$ to encode the reset gate $\mathbf{r}_k$. In this regard, the temporal displacement plays a role in determining whether the previous state should be stored or not. For aw^g with L length as the form $[IG_1, IG_2, ..., IG_{\mathrm{L}}]$, the temporal pattern embedding of aw^g is calculated as follows:

$$\mathbf{z}_k = \sigma(\mathbf{W}_z \mathrm{NN}_1(IG_k) + \mathbf{U}_z \mathbf{h}_{k-1}) \tag{5}$$

$$\mathbf{r}_k = \sigma(\mathbf{W}_r \mathrm{t2v}(\Delta t_k) + \mathbf{U}_r \mathbf{h}_{k-1}) \tag{6}$$

$$\tilde{\mathbf{h}}_k = \tanh(\mathbf{W}_h IG_k + \mathbf{U}_h \odot \mathbf{h}_{k-1}) \tag{7}$$

$$\mathbf{h}_k = (1 - \mathbf{z}_k) \odot \mathbf{h}_{k-1} + \mathbf{z}_k \odot \tilde{\mathbf{h}}_k, \tag{8}$$

where IG_{k} denotes the k-th group-view node identifier in aw_i^g, and $\mathbf{W}_z$, $\mathbf{U}_z$, $\mathbf{W}_r$, $\mathbf{U}_r$, $\mathbf{W}_h$, $\mathbf{U}_h$ are all weights and bias parameters of the temporal GRU. The NN_1 is a one-layer neural network, which projects the group-view node identifier to a vector. σ is the sigmoid function and $\odot$ denotes the element-wise product. The $\mathrm{t2v}(\Delta t_k)$ is the temporal displacement of the k-th interaction and the $(k-1)$-th interaction. In the beginning, Δt_0 is 0. The temporal pattern embedding $\mathbf{TempEmb}(aw^g)$ is the final state of the temporal-aware GRU $h_{\mathrm{L}-1}$. The temporal pattern embedding of the query can be calculated in the mean-pooling method as follows:

$$\mathbf{TempEmb}_{(u,v,t)} = \frac{1}{2N} \sum_{i=1}^{2N} \mathbf{TempEmb}(aw_i^g), \tag{9}$$

where $aw_i^g \in \{AW_u, AW_v\}$.

4.3.3 The inference module

The inference module concatenates the temporal pattern embedding and the topological pattern embedding to predict the existence of query. We obtain the query embedding $\mathbf{Emb}_{(u,v,t)}$ as follows: the topological and temporal pattern embeddings are concatenated and afterward projected by a one-layer neural network NN_2:

$$\mathbf{Emb}_{(u,v,t)} = \mathrm{NN}_2\left(\mathrm{Concat}\left[\mathbf{TopoEmb}_{(u,v,t)}, \mathbf{TempEmb}_{(u,v,t)}\right]\right). \tag{10}$$

After that, we use a two-layer neural network NN_3 and a sigmoid function σ to predict the existence of the query (u, v, t):

$$\hat{y}_{(u,v,t)} = \sigma\left(\mathrm{NN}_3(\mathbf{Emb}_{(u,v,t)})\right), \tag{11}$$

where $\hat{y}_{(u,v,t)}$ is the predicted label of (u, v, t).

In the training procedure, given the interactions $\mathcal{E}$ chosen for the training set, we first generate negative samples $\mathcal{E}^-$ in the uniform way [37], in which each non-existent interaction in the form of (u, v', t), and both u and v' are drawn from the nodes involved in $\mathcal{E}$. The training set is composed of $\mathcal{E}$ and $\mathcal{E}^-$, which are of equal size. We use the cross-entropy loss to train NEAWalk as follows:

$$\mathcal{L} = \frac{1}{|\mathcal{E} \cup \mathcal{E}^-|} \sum_{(u,v,t) \in \mathcal{E} \cup \mathcal{E}^-} -[y_{(u,v,t)} \cdot \ln(\hat{y}_{(u,v,t)}) + (1 - y_{(u,v,t)}) \cdot \ln(\hat{y}_{(u,v,t)})], \tag{12}$$

Table 1 Statistics of five social network datasets

Dataset	#Node	#Int.	Avg. NBHD.	Max DEG.	DIAM.	CENT.
Enron	184	125,235	23.0	21,512	4	0.48
Social evolution	66	66,898	36.9	4758	3	0.32
CollegeMsg	1899	20,296	14.6	1546	8	0.13
Bitcoin OTC	5881	35,592	7.3	1298	9	0.13
Math overflow	24,818	506,550	15.2	11,309	9	0.09

where $y_{(u,v,t)}$ is the label for the interaction (u, v, t), with 0 for the non-existent interaction and 1 for the existent interaction.

5 Experiments

In this section, for evaluating the performance of NEAWalk, we first design the two-track experiments for the dynamic network completion task, including the datasets, baseline methods, experiment settings, and following with experimental results and discussions. We then conduct ablation studies of the NEAWalk to evaluate the effectiveness of each component in the NEAWalk. Next, we compare the model performance in terms of BCW length, the ratio of observed data, and query embedding dimension. Finally, we analyze the complexity of the sampling and training procedure in NEAWalk.

5.1 Datasets description

We select five open-source dynamic social network datasets with different interaction types (including email/online message/physical proximity/online transaction/Q&A forum) in experiments to show the potential to apply our model in various real-world scenarios. The granularity of time information in all datasets is accurate to second. The five dynamic social network datasets are processed in the format of (u, v, t) per line, which represents an interaction in the dataset. The detailed statistics of five datasets (including the number of nodes, number of interactions, the average number of neighborhoods, max degree, network diameter, and network centralization) are shown in Table 1 from left to right.
Enron dataset[1] is an email communication network of 184 core employees in a company over several of years. One email record denotes an interaction between a person pair at a specific time. We leave out the content of emails, and keep the sender, recipient and the post time.

Social evolution dataset[2] is collected from the daily trajectory data from 66 undergraduates with wireless signals of mobile phones to record the physical proximity between people. We select interaction data with a time of 2 weeks from the original data.

CollegeMsg dataset[3] comprises user private message type interactions on an online social network at the University of California, Irvine. Users could search the network for others

[1] https://www.cs.cmu.edu/~./enron/.

[2] http://realitycommons.media.mit.edu/socialevolution.html.

[3] http://snap.stanford.edu/data/CollegeMsg.html.

and then initiate a conversation based on profile information. We leave out the content of conversations, and keep the sender, recipient and the start time of conversation.

Bitcoin OTC dataset [4] is a user rating network dataset collected from a bitcoin transaction platform called OTC[5]. Users tend to trade with other users they trust or have a higher rating score to prevent transactions with fraudulent and risky users. We leave out the rating of transactions, and keep the source and target user of transaction and the time of transaction.

Math overflow dataset [6] records the interactions among users on the stack exchange website Math Overflow. There are three different types of interactions represented by a directed edge (u, v, t), including answer to question, comments to question and comments to answer. We leave out the content of answer and comment, and keep the source and target user and the time of interaction.

5.2 Baseline methods

We choose two types of baseline methods, including link prediction methods on static and dynamic graphs. The methods on static graphs include Deepwalk, GAE, VGAE, GCMC, DeepNC, and NEAWalk(w/o dynamic). The methods on dynamic graphs include AIM, CTDNE, TGN, TGAT, APAN and our method. DeepWalk [37] proposes a random walk method on static graphs to collect local information of nodes and uses the Skip-gram model to learn node embeddings.

GAE and VGAE [29] use the encoder–decoder architecture to predict links via a graph generation way on the static graph. The main difference between the two models is that the GAE uses the autoencoder, while the VGAE uses the variational autoencoder to represent the hidden state of the graph.

GCMC(Graph Convolutional Matrix Completion) [27] uses the graph auto-encoder framework based on differentiable message passing procedure to complete the missing edges on the static social network.

DeepNC [12] uses the deep generative model to learn the feature of fully observed networks and infers the missing edges and nodes of the partial graph. In our experiments, we concern with the performance of the missing edges.

NEAWalk (SG) randomly samples walks on the static graph. The walks cannot discriminate the historical and the evolving interaction information due to the loss of temporal information. The temporal displacement vectors are replaced by random vectors of the same size in the temporal-aware GRU.

AIM (Agent Interaction Model) [3] applies the Hawkes process to infer the interaction based on the observed part by maximizing the energy over the observed interactions of all node pairs.

CTDNE [13] collects unidirectional temporal walks dynamic graphs and then feeds them into the Skip-Gram model for learning node embeddings. For a fair comparison, we modify the temporal walk sampling method in CTDNE to the BCW sampling method, i.e., both collecting historical and evolving behavior walks.

TGN [14] proposes an encoder–decoder framework to predict future interactions on dynamic social network. For a fair comparison, we add another memory module in TGN

[4] https://snap.stanford.edu/data/soc-sign-bitcoin-otc.html.

[5] http://www.bitcoin-otc.com/.

[6] http://snap.stanford.edu/data/sx-mathoverflow.html.

to store the evolving interactions and use both the historical and evolving information to infer the missing interactions.

TGAT [43] applies the self-attention mechanism as building block of temporal graph neural network, and develop a functional time to learn the evolving features of nodes.

APAN [44] employs an asynchronous mail propagator in the temporal graph neural network, for spreading the evolving information in the interaction to the neighborhoods of related nodes.

5.3 Experimental settings and evaluation metrics

5.3.1 Experimental settings

In experiments, we design two tracks for comparing the static/dynamic network completion methods: static graph track and dynamic graph track.

Static graph track We ignore the temporal information of interactions, and treat interactions as atemporal links. Multiple interactions between two nodes are merged as one link. Due to the lack of temporal information, the prediction of links is equivalent to whether interactions have occurred. We randomly divide links as positive links of the training/validation/test set according to the ratio of 7:2:1, and sample the equal number of non-existent links as negative links of the training/validation/test set.

Dynamic graph track AIM optimizes parameters of the Hawkes process function through predicting the probability of interactions as events. To this end, we randomly choose 100 node pairs that have at least ten interactions for all datasets, and randomly split the 90% interactions for optimizing parameters and 10% interactions for the test. For the other baseline models on the dynamic graph track and NEAWalk, we first split all interactions into the observed interaction set E and the unobserved interaction set $\hat{E}$ as the ratio of 4:1. $\hat{E}$ contains the positive interactions that are represented as missing interactions. Then, we generate the same number of non-existent interactions in $\hat{E}$. These interactions are partitioned into training/validation/test interaction set as the ratio of 7:2:1. We only sample interactions in E in the BCW sampling procedure of training/validation/test phase.

For implementation details, we use the Pytorch[7] and the Deep Graph Library[8] to implement all models, and we optimize all models through Adam optimizer. For NEAWalk, we have tuned the hyperparameters manually, and setup the optimal hyperparameters in the experiments as following: in the sampling procedure, we use the optimal hyperparameters as follows: we set the walk length h of BCW to 2, the sampled walk number N of the BCW group to 64; in the topo-temp learning procedure, we set the size of topological and temporal pattern embeddings as 64, and the query embedding size is 128. In the training phase, the learning rate is 0.0001, the batch size is set to 64 and the dropout rate is set to 0.1. We discuss the experiment results of hyperparameter analysis in Subsect. 5.6. In the experiment, we repeat all experiments 10 times and calculate the results with standard deviation. The experiments are performed on a CentOS Machine with sixteen 2.1 GHz Intel cores and four 24 GB TITAN RTX GPUs.

[7] https://pytorch.org/.

[8] https://github.com/dmlc/dgl.

5.3.2 Evaluation metrics

To evaluate the performance of the models, we adopt two widely used metrics in the link prediction task: AUC (Area Under the ROC Curve) and the AP (Average Precision). Higher AP and AUC values represent better quality of the social network completion task. In the static graph track, the AUC and AP metrics are calculated on the prediction results in the test set of atemporal links. In the dynamic graph track, the AUC and AP metrics are calculated on the prediction results in the test set of interactions. The analysis of the experimental results will be discussed in the following subsection.

5.4 Experimental results and discussion

We show the two-track experiment results (mean AP std) under the evaluation of AP and AUC in Table 2. Although NEAWalk is not applied to static social network link prediction tasks, we notice that NEAWalk (SG) still achieves the best performance on Enron, Bitcoin OTC, and Math Overflow. This phenomenon shows that even in static graphs, the social group features without temporal information can effectively help to infer the missing interactions between nodes. The methods based on graph neural networks (GAE and GCMC) achieve the best results in two other datasets that share conducive macroscopic features for inferring the missing links, and both two methods focus on reconstructing the whole graph via the auto-encoder mechanism.

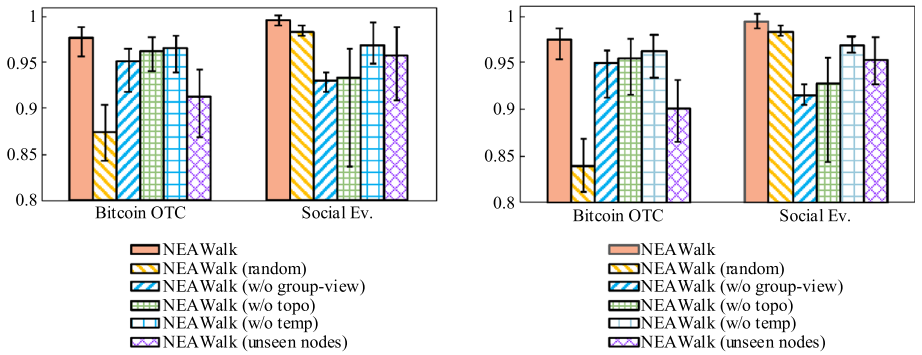
In the dynamic graph track, NEAWalk outperforms all competitors both in AP and AUC metrics with a significant advantage that the effectiveness of NEAWalk can be hereby verified. As for other competitors, the performance of AIM is unsatisfactory. An apparent reason is that AIM treats interactions within node pairs independently but ignores social group features, which results in weak performance. CTDNE achieves the second-best performance on Enron and Social Evolution, which both are two graph datasets with high average neighborhoods, but cannot compete with NEAWalk in all datasets, and three neural network-based methods (TGAT, TGN and APAN) in three other datasets. These experimental results attest to the importance of the anonymization procedure on temporal walks. The results of TGN and APAN are satisfactory and more stable than other baselines, while NEAWalk still performs better than the two high-performing baselines. It reveals the importance of leveraging the pattern features of social group information within multi-hop in NEAWalk, while TGN only considers one-hop related interactions of nodes, and the mailbox mechanism of APAN only covers the k-hop information of social group, but fails to extract the pattern information.

5.5 Ablation studies

In this section, we conduct experiments by ablating the proposed method to analyze the effects of each component in NEAWalk. We choose two representative datasets: the Bitcoin OTC and Social Evolution dataset. The commonality of Bitcoin OTC, CollegeMsg, and Math Overflow is that the nodes are large in scale, and the interactions in these datasets are sparse. Both Social Evolution and Enron record frequent interactions in small organizations, and individuals are closely connected. In all ablated methods, we keep other hyperparameters the same as the original NEAWalk. We repeat all experiments 10 times and calculate the mean results with error bars (the best and the worst records). The experiment results are shown in Fig. 7.

Table 2 Experiment results of social network completion

Dataset	Metric	Static graph track						Dynamic graph track					
		DeepWalk	GAE	VGAE	GCMC	DeepNC	NEAWalk(SG)	AIM	TGN	CTDNE	TGAT	APAN	NEAWalk
Enron	AP	0.9132	0.7217	0.6129	0.8912	0.5335	**0.9376**	0.5382	0.8235	0.8609	0.8003	0.8065	**0.9991**
	±	0.0022	0.0168	0.0202	0.0031	0.0059	0.0048	0.0047	0.0144	0.0018	0.009	0.0029	0.001
	AUC	0.934	0.7153	0.6025	0.8971	0.5419	**0.9554**	0.5105	0.856	0.8724	0.7795	0.8232	**0.9988**
	±	0.0022	0.0174	0.0214	0.0027	0.0064	0.0105	0.004	0.0092	0.0021	0.0104	0.0022	0.0013
Social evolution	AP	0.5691	0.6776	0.5052	**0.7988**	0.553	0.5959	0.5609	0.8198	0.8809	0.7457	0.7449	**0.9958**
	±	0.024	0.0412	0.0278	0.0062	0.0038	0.0022	0.0024	0.0072	0.0031	0.009	0.0199	0.0032
	AUC	0.6005	0.6975	0.5315	**0.8178**	0.573	0.608	0.5555	0.8623	0.8739	0.7367	0.7919	**0.9942**
	±	0.0395	0.0289	0.0331	0.0056	0.0035	0.0017	0.0039	0.0061	0.0032	0.008	0.0175	0.0046
CollegeMsg	AP	0.8383	**0.8848**	0.862	0.8283	0.5331	0.7305	0.634	0.8761	0.7483	0.7347	0.8575	**0.9559**
	±	0.0016	0.0086	0.0087	0.0045	0.0089	0.0139	0.0061	0.0126	0.0035	0.0078	0.0031	0.0048
	AUC	0.8124	**0.8769**	0.849	0.8367	0.5432	0.7423	0.5529	0.867	0.7495	0.7516	0.8604	**0.9546**
	±	0.0021	0.0093	0.0075	0.0024	0.0072	0.0102	0.0091	0.01	0.0029	0.009	0.0042	0.0065
Bitcoin OTC	AP	0.6465	0.9278	0.9151	0.7064	0.5564	**0.9539**	0.6379	0.8134	0.6764	0.7762	0.7784	**0.9771**
	±	0.0019	0.0044	0.0035	0.0065	0.0043	0.0076	0.001	0.0129	0.0026	0.0054	0.0036	0.0012
	AUC	0.6076	0.9158	0.904	0.6928	0.5651	**0.9546**	0.4954	0.7574	0.7063	0.7743	0.7579	**0.9745**
	±	0.0022	0.0059	0.0035	0.0099	0.0055	0.0024	0.0042	0.0143	0.0042	0.0052	0.0045	0.0013
Math overflow	AP	0.9069	0.962	0.9532	0.8587	0.5591	**0.9668**	0.6412	0.9229	0.8054	0.8911	0.9167	**0.9697**
	±	0.0002	0.0032	0.0025	0.0032	0.0021	0.0039	0.0039	0.0035	0.0015	0.0006	0.0022	0.0002
	AUC	0.8729	0.9648	0.9568	0.8431	0.5789	**0.9659**	0.5264	0.9209	0.8305	0.8762	0.9059	**0.9696**
	±	0.0002	0.0052	0.0035	0.0039	0.003	0.0035	0.0041	0.005	0.0026	0.0015	0.0039	0.0002



(a) Ablation Study Results (AP) **(b)** Ablation Study Results (AUC)

Fig. 7 Ablation experiments results

5.5.1 The time constraint in BCW sampling procedure

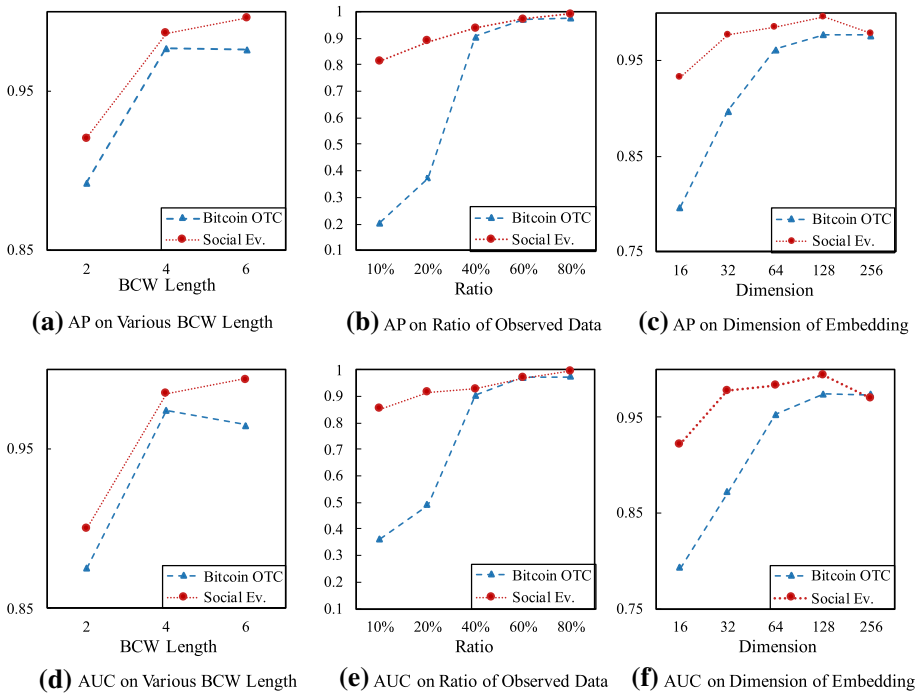
To verify the claim that the time constraint in the BCW sampling procedure can effectively represent temporal dependency information, we replace time constraint with random walk in the NEAWalk (random). The BCWs sampled in NEAWalk(random) collect temporally disordered interaction sequences rather than unidirectional increasing interaction sequences. We see that the impact of time constraint ablation is more prominent in Bitcoin OTC (average dropping 10.3%, 10.4% in AP and AUC, and longer error bars), while slightly dropping in Social Evolution (average dropping 1.2%, 0.9% in AP and AUC). These experimental results prove that the disordered temporal information in BCW weakens the performance.

5.5.2 The effect of the group-view identifier

The motivation of this experiment is to explore the effect of the group-view identifiers, which represent the social role features of nodes. We resort to the NEAWalk (w/o group-view), where the nodes in the BCW groups are assigned the exact dimensional random embeddings rather than the group-view identifiers. The performance of NEAWalk (w/o group-view) degrades with varying degrees, and the decline in Bitcoin OTC is more slightly because the Bitcoin OTC describes an online anonymized transaction network and the social role of nodes is not prominent.

5.5.3 Topological and temporal pattern embedding

To explore the effects of topological pattern embedding and temporal pattern embedding, we design NEAWalk (w/o topo) and NEAWalk (w/o temp). The topological pattern embedding in NEAWalk (w/o topo) and the temporal displacement vector in NEAWalk (w/o temp) replaced the same dimension random embeddings. The performances of NEAWalk (w/o topo) drop acutely in Social Evolution (6.2% and 6.6%). Besides, the error bar of NEAWalk (w/o topo) on Social Evolution is longer than the original NEAWalk, reflecting the fact that the performance of NEAWalk (w/o topo) is unstable. NEAWalk (w/o temp) only retains the sequencing order information but ignores the temporal displacement between interactions. Therefore. AP and AUC metrics of NEAWalk (w/o temp) also drop compared to original NEAWalk, but still better than the NEAWalk (w/o topo). This phenomenon shows that the



(a) AP on Various BCW Length **(b)** AP on Ratio of Observed Data **(c)** AP on Dimension of Embedding

(d) AUC on Various BCW Length **(e)** AUC on Ratio of Observed Data **(f)** AUC on Dimension of Embedding

Fig. 8 Hyperparameter analysis experiment results

topological pattern embedding has more significant effects than temporal pattern embedding, and the joint combination of both two embeddings can contribute to better results.

5.5.4 The generalizing ability of NEAWalk

The anonymous characteristic of NEAWalk guarantees that it is naturally inductive and thus can be generalized to social group patterns rather than specific nodes. We design NEAWalk (unseen nodes) to analyze the generalizing ability of NEAWalk. Specifically, we randomly choose 10% nodes as unseen nodes and mask their related interactions, which are removed from E and the training/validation interaction set. We evaluate the performance of NEAWalk on these interactions of unseen nodes. We can find that the experimental results of NEAWalk (unseen nodes) are still considerable (both AP and AUC are above 90% on Bitcoin OTC, above 95% on social evolution), demonstrating the effectiveness of NEAWalk in generalizing the patterns of interactions containing nodes out of observed data.

5.6 The hyperparameter analysis

In this section, we adjust the hyperparameters in NEAWalk to explore the effects on prediction results with different hyperparameter values and conduct experiments on the Bitcoin OTC dataset and Social Evolution dataset. Figure 8 shows the hyperparameter analysis results.

5.6.1 The sampling length of BCWs

BCWs with different sampling lengths represent different sizes of reception fields on social groups. Concretely, BCWs with too short length leads to insufficient information of social group, and excessively long BCWs may introduce noise information, thereby affecting the performance. We set different BCW sampling lengths at [2,4,6] respectively, with the other hyperparameters unchanged. The experimental results are shown in Fig. 8a, d. As for the results of the Bitcoin OTC dataset, both AP and AUC metrics have been increasing and reaching saturation at the length of 4 and slightly dropping at the length of 6. Although 6-hop BCWs can slightly improve the model performance on the Social Evolution dataset, it needs to be at the cost of a longer training time. The results demonstrate that the 4-hop BCW is sufficient to express social group information.

5.6.2 The portion of observed interactions

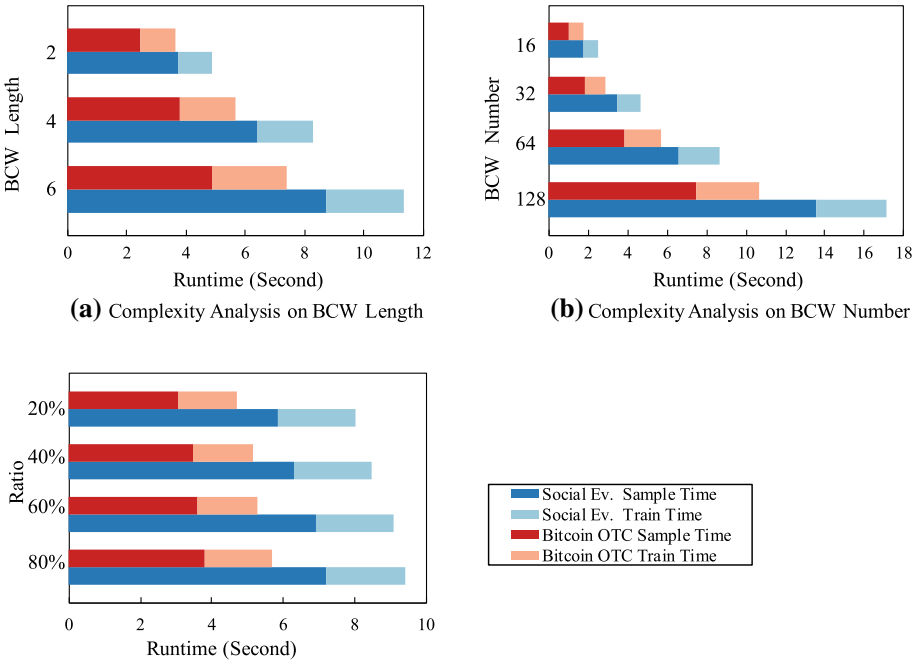
Theoretically, the smaller proportion of observed data would result in inferior performance. To show the robustness of NEAWalk on incomplete interaction data, we set different ratios of observed interaction data in {10%, 20%, 40%, 60%} compare the performance with the original ratio 80%. The experimental results are shown in Fig. 8b, e. We can observe that AP and AUC metrics do not decrease markedly in two datasets until 40% observed data. This phenomenon indicates that NEAWalk still has excellent performance with severely incomplete interaction data (under the situation that 60% loss of interactions). The results in Bitcoin OTC drop significantly from 40% to 20%, indicating that the NEAWalk on sparse social networks can be more affected by the proportion of observed data.

5.6.3 The query embedding dimension

We conduct experiments for different query embedding dimension sizes to explore the model performance with different embedding dimension sizes. The experimental results are shown in Fig. 8c, f. The predictions do not achieve the desired results when the query embedding dimension size is relatively small because the topological and temporal features are not fully expressed. As the size increases, the prediction results improve and saturate at 128 dimensions.

5.7 The complexity analysis of runtime

In this section, we analyze how the sampling and training runtime of NEAWalk depends on the length of BCW L, the number of BCWs in BCW group N, and the ratio of observed interaction data. Concretely, we conduct the complexity analysis experiments on Social Evolution and Bitcoin OTC dataset by recording the sampling and training runtime of NEAWalk per batch. The NEAWalk models with different settings will be run 10 times, and the average runtime will be reported. The experimental results in Fig. 9a, b clearly show that the sampling time and the training time both increase linearly with L and N. Compared with different datasets, the sampling time is longer in the Social Evolution, which contains more interactions. Furthermore, we conduct a complexity analysis experiment for studying the sampling/training time with the observed interaction data number. Specifically, we change the ratio of observed data with L and N fixed and record the sampling and training runtime.



(a) Complexity Analysis on BCW Length

(b) Complexity Analysis on BCW Number

(c) Complexity Analysis on Ratio of Observed Data

Fig. 9 Complexity analysis results on various BCW length/BCW number/ratio of observed data

The experimental results in Fig. 9c prove that the sampling time of NEAWalk increases linearly with the observed data number, which is consistent with the time complexity analysis of the sampling method. Since the training time is related to a specific query, the training runtime keep stable in this experiment.

6 Conclusion

In this paper, we propose neural network for encoding anonymous walks in behavioral context (NEAWalk), a novel inductive and effective method for inferring the missing interactions. By incorporating the behavioral context walk sampling algorithm and a dual-view anonymization procedure with a novel topo-temp embedding approach, NEAWalk comprehensively explores the historical and evolving pattern features residing in the social group of queries and achieves the best performances on the dynamic network completion task on the learned query embeddings. Extensive experiments on five real-world social network datasets have proved the superiority of NEAWalk over other methods for inferring missing interactions both on the static graph track and dynamic graph track.

Real-world applications are often oriented to systems with multiple types of entities and complex types of relationships. In the future work, we seek to extend the sampling and anonymization procedures of NEAWalk to describe rich and heterogeneous information in the dynamic social network, which contains multiple types of entities and interactions.

Acknowledgements This work is supported by the National Natural Science Foundation of China (Nos. U1836206, 91646120, U21B2046, 62172393), the National Key Research and Development Program of China under grants (No. 2018YFB1402601) , the Zhongyuanyingcai program-funded to central plains science and technology innovation leading talent program (No. 204200510002) and Major Public Welfare Project of Henan Province (No. 201300311200).

References

1. Firth JA, Hellewell J, Klepac P, Kissler S, Kucharski AJ, Spurgin LG (2020) Using a real-world network to model localized COVID-19 control strategies. Nat Med 26(10):1616–1622
2. Guimerá R, Sales-Pardo M (2009) Missing and spurious interactions and the reconstruction of complex networks. Proc Natl Acad Sci 106(52):22073–22078
3. Stomakhin A, Short MB, Bertozzi AL (2011) Reconstruction of missing data in social networks based on temporal patterns of interactions. Inverse Probl 27(11):115013
4. Kim M, Leskovec J (2011) The network completion problem: inferring missing nodes and edges in networks. In: Proceedings of the 2011 SIAM international conference on data mining. Society for Industrial and Applied Mathematics, pp. 47-58
5. Zhang SK, Li CT, Lin SD (2020) A joint optimization framework for better community detection based on link prediction in social networks. Knowl Inf Syst 62(11):4277–4296
6. Liu G, Guo J, Zuo Y, Wu J, Guo RY (2020) Fraud detection via behavioral sequence embedding. Knowl Inf Syst 62(7):2685–2708
7. Cheng X, Liu S, Sun X, Wang Z, Zhou H, Shao Y, Shen H (2021) Combating emerging financial risks in the big data era: a perspective review. Fundam Res 1(5):595–606
8. Yang Y, Lichtenwalter RN, Chawla NV (2015) Evaluating link prediction methods. Knowl Inf Syst 45(3):751–782
9. Haghani S, Keyvanpour MR (2019) A systemic analysis of link prediction in social network. Artif Intell Rev 52(3):1961–1995
10. Daud NN, Ab Hamid SH, Saadoon M, Sahran F, Anuar NB (2020) Applications of link prediction in social networks: a review. J Netw Comput Appl 166:102716
11. Gupta AK, Sardana N (2018) Prediction of missing links in social networks: feature integration with node neighbour. Int J Web Based Commun 14(1):38–53
12. Tran C, Shin WY, Spitz A, Gertz M (2020) DeepNC: Deep generative network completion. In: IEEE transactions on pattern analysis and machine intelligence
13. Nguyen GH, Lee JB, Rossi RA, Ahmed NK, Koh E, Kim S (2018) Continuous-time dynamic network embeddings. In: Companion proceedings of the the web conference 2018. pp. 969–976
14. Rossi E, Chamberlain B, Frasca F, Eynard D, Monti F, Bronstein M (2020) Temporal graph networks for deep learning on dynamic graphs. arXiv preprint arXiv:2006.10637
15. Wang Y, Chang YY, Liu Y, Leskovec J, Li P (2020) Inductive representation learning in temporal networks via causal anonymous walks. In: International conference on learning representations
16. Turner JH (1988) A theory of social interaction. Stanford University Press, California
17. Giddens A, Duneier M, Appelbaum RP, Carr DS (1991) Introduction to sociology. Norton, New York, p 672
18. Anolli L, Duncan JrS, Magnússon MS, eds (2005) The hidden structure of interaction: from neurons to culture patterns (Vol. 7). Ios Press
19. Guo Z, Yu K, Li Y, Srivastava G, Lin JCW (2021) Deep learning-embedded social internet of things for ambiguity-aware social recommendations. In: IEEE transactions on network science and engineering
20. Djenouri Y, Srivastava G, Belhadi A, Lin JCW (2021) Intelligent blockchain management for distributed knowledge graphs in IoT 5G environments. In: Transactions on emerging telecommunications technologies, pp. e4332
21. Min S, Gao Z, Peng J, Wang L, Qin K, Fang B (2021) STGSN-A spatial-temporal graph neural network framework for time-evolving social networks. Knowl Based Syst 214:106746
22. Kumar A, Singh SS, Singh K, Biswas B (2020) Link prediction techniques, applications, and performance: a survey. Phys A Stat Mech Appl 553:124289
23. Wang P, Xu B, Wu Y, Zhou X (2015) Link prediction in social networks: the state-of-the-art. Sci China Inf Sci 58(1):1–38
24. Newman ME (2001) Clustering and preferential attachment in growing networks. Phys Rev E 64(2):025102

25. Barabâsi AL, Jeong H, Néda Z, Ravasz E, Schubert A, Vicsek T (2002) Evolution of the social network of scientific collaborations. Phys A Stat Mech Appl 311(3–4):590–614
26. Wang C, Satuluri V, Parthasarathy S (2007) Local probabilistic models for link prediction. In: Seventh IEEE international conference on data mining (ICDM 2007). IEEE, pp. 322-331
27. Berg RVD, Kipf TN, Welling M, (2017) Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263
28. Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907
29. Kipf TN, Welling M, (2016) Variational graph auto-encoders. arXiv preprint arXiv:1611.07308
30. Zhang YQ, Li X, Xu J, Vasilakos AV (2014) Human interactive patterns in temporal networks. IEEE Trans Syst Man Cybern Syst 45(2):214–222
31. Zhou L, Yang Y, Ren X, Wu F, Zhuang Y (2018) Dynamic network embedding by modeling triadic closure process. In: Proceedings of the AAAI conference on artificial intelligence. pp. 32(1)
32. Huang H, Fang Z, Wang X, Miao Y, Jin H (2020) Motif-preserving temporal network embedding. In: IJCAI. pp. 1237–1243
33. Fu D, Zhou D, He J, (2020) Local motif clustering on time-evolving graphs. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 390–400
34. Paranjape A, Benson AR, Leskovec J (2017) Motifs in temporal networks. In: Proceedings of the tenth ACM international conference on web search and data mining. pp. 601-610
35. Divakaran A, Mohan A (2020) Temporal link prediction: a survey. New Gener Comput 38(1):213–258
36. Fournier-Viger P, He G, Cheng C, Li J, Zhou M, Lin JCW, Yun U (2020) A survey of pattern mining in dynamic graphs. Wiley Interdiscip Rev Data Min Knowl Discov 10(6):e1372
37. Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 701–710
38. Micali S, Zhu ZA (2016) Reconstructing markov processes from independent and anonymous experiments. Discret Appl Math 200:108–122
39. Ivanov S, Burnaev E (2018) Anonymous walk embeddings. In: International conference on machine learning. PMLR, pp. 2186-2195
40. Barceló P, Geerts F, Reutter J, Ryschkov M (2021) Graph neural networks with local graph parameters. Adv Neural Inf Process Syst 34:25280–25293
41. Kazemi SM, Goel R, Eghbali S, Ramanan J, Sahota J, Thakur S, Wu S, Smyth C, Poupart P, Brubaker M (2019) Time2vec: learning a vector representation of time. arXiv preprint arXiv:1907.05321
42. Jin Y, Song G, Shi C (2020) GraLSP: graph neural networks with local structural patterns. Proc AAAI Conf Artif Intell 34(04):4361–4368
43. Xu D, Ruan C, Korpeoglu E, Kumar S, Achan K (2020) Inductive representation learning on temporal graphs. arXiv preprint arXiv:2002.07962
44. Wang X, Lyu D, Li M, Xia Y, Yang Q, Wang X, Wang X, Cui P, Yang Y, Sun B, Guo Z (2021) APAN: asynchronous propagation attention network for real-time temporal graph embedding. In: Proceedings of the 2021 international conference on management of data. pp. 2628-2638

Yinghan Shen is currently a PhD candidate in the Data Intelligence Research Center, Institute of Computing Technology, Chinese Academy of Sciences. His research interests lie in data mining, social knowledge graph, and temporal knowledge graph reasoning.

Xuhui Jiang is currently a PhD candidate in the Data Intellengence Research Center, Institute of Computing Technology, Chinese Academy of Sciences. His research interests lie in data mining, social knowledge graph, and graph neural network.

Zijian Li is currently a master in the Data Intelligence Research Center, Institute of Computing Technology, Chinese Academy of Sciences. His research interests lie in data mining, social knowledge graph, and graph neural network.

Yuanzhuo Wang is currently a professor in the Institute of Computing Technology, Chinese Academy of Sciences, and the director of Zhongke Big Data Academy. His main research interests include network science, data mining, and knowledge computing. He has published about more than 200 research papers in prestigious journals including ACM and IEEE Trans., and conference proceedings including ACM SIGIR, CIKM, WWW, AAAI, ACL, IJCAI, and so on.

Xiaolong Jin received the PhD degree in Computer Science from Hong Kong Baptist University in 2005. He is currently a professor in the Institute of Computing Technology, Chinese Academy of Sciences. His current research interests include knowledge graph, knowledge engineering, social computing etc. He has published four monographs and more than 200 papers in reputable journals and conferences including IEEE TKDE, IEEE TWC, WWW, ACL, SIGIR, EMNLP, COLING, etc. He has received the Best (Student/Academic) Paper Awards in IEEE ICBK (2017), IEEE CIT (2015), CCF Big Data (2015), IEEE AINA (2007), and ICAMT (2003).

Shengjie Ma received his B.S. degree in communication engineering from Southwest Jiaotong University and his MS degree in computer engineering from New York University. He is currently a PhD student in Gaoling School of Artificial Intelligence, Renmin University of China. His principal research interests focus on retrieval and ranking.

Xueqi Cheng is a professor in the Institute of Computing Technology, Chinese Academy of Sciences (ICT-CAS), and the director of the CAS Key Laboratory of Network Data Science and Technology. His main research interests include network science, web search and data mining, big data processing and distributed computing architecture. He has published more than 200 publications in prestigious journals and conferences, including IEEE Transactions on Information Theory, IEEE Transactions on Knowledge and Data Engineering, Journal of Statistical Mechanics, Physical Review E., ACM SIGIR, WWW, ACM CIKM, WSDM, AAAI, IJCAI, ICDM, and so on. He was awarded the NSFC Distinguished Youth Scientist (2014), the National Prize for Progress in Science and Technology (2012,2017), the China Youth Science and Technology Award (2011).