Towards Fine-Grained Temporal Network Representation via Time-Reinforced Random Walk

Zhining Liu,[1]* **Dawei Zhou,**[2]* **Yada Zhu,**[3] **Jinjie Gu,**[4] **Jingrui He**[2]
[1]University of Electronic Science and Technology of China, [2]University of Illinois at Urbana-Champaign,
[3]IBM T. J. Watson Research Center, [4]Ant Financial Services Group
*Equal contribution
liuzhininguestc@gmail.com, dzhou21@illinois.edu,
yzhu@us.ibm.com, jinjie.gujj@antfin.com, jingrui@illinois.edu

Abstract

Encoding a large-scale network into a low-dimensional space is a fundamental step for various network analytic problems, such as node classification, link prediction, community detection, etc. Existing methods focus on learning the network representation from either the static graphs or time-aggregated graphs (e.g., time-evolving graphs). However, many real systems are not static or time-aggregated as the nodes and edges are timestamped and dynamically changing over time. For examples, in anti-money laundering analysis, cycles formed with time-ordered transactions might be red flags in online transaction networks; in novelty detection, a star-shaped structure appearing in a short burst might be an underlying hot topic in social networks. Existing embedding models might not be able to well preserve such fine-grained network dynamics due to the incapability of dealing with continuous-time and the negligence of fine-grained interactions. To bridge this gap, in this paper, we propose a fine-grained temporal network embedding framework named *FiGTNE*, which aims to learn a comprehensive network representation that preserves the rich and complex network context in the temporal network. In particular, we start from the notion of fine-grained temporal networks, where the temporal network can be represented as a series of timestamped nodes and edges. Then, we propose the time-reinforced random walk (TRRW) with a bi-level context sampling strategy to explore the essential structures and temporal contexts in temporal networks. Extensive experimental results on real graphs demonstrate the efficacy of our *FiGTNE* framework.

Introduction

Representation learning over graph-structured data, which aims to encode the meaningful patterns and rich network context into a low-dimensional space, has received great success in various domains, ranging from social networks (Kipf and Welling 2016) to collaborative networks (Dong, Chawla, and Swami 2017), from knowledge graphs (Wang et al. 2014) to protein-protein networks (Cannistraci, Alanis-Lobato, and Ravasi 2013). By learning the network embedding, conventional vector-based machine learning algorithms can be naturally incorporated to solve various high-impact tasks in graph analysis, such as node classification (Perozzi, Al-Rfou, and Skiena 2014), link prediction (Grover and Leskovec 2016), community detection (Wang et al. 2017), etc.

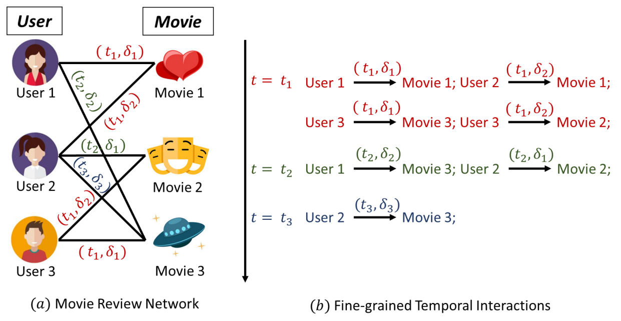

User
Movie
User 1
User 2
User 3
Movie 1
Movie 2
Movie 3
(t_1, δ_1)
(t_2, δ_2)
(t_1, δ_2)
(t_2, δ_1)
(t_3, δ_3)
(t_1, δ_2)
(t_1, δ_1)
(a) Movie Review Network
$t = t_1$ User 1 $\xrightarrow{(t_1, \delta_1)}$ Movie 1; User 2 $\xrightarrow{(t_1, \delta_2)}$ Movie 1;
User 3 $\xrightarrow{(t_1, \delta_1)}$ Movie 3; User 3 $\xrightarrow{(t_1, \delta_2)}$ Movie 2;
$t = t_2$ User 1 $\xrightarrow{(t_2, \delta_2)}$ Movie 3; User 2 $\xrightarrow{(t_2, \delta_1)}$ Movie 2;
$t = t_3$ User 2 $\xrightarrow{(t_3, \delta_3)}$ Movie 3;
(b) Fine-grained Temporal Interactions

Figure 1: An example of *Fine-Grained Temporal Networks*. (a) A movie review network with three users and three movies. (b) The corresponding streaming system logs, which is presented in the form of timestamped (i.e., $t = \{t_1, t_2, t_3\}$) interactions between users and movies within a certain duration $\delta = \{\delta_1, \delta_2, \delta_3\}$. For each movie review, the timestamp t indicates when a user comments a movie, while δ records the active time of the review on the website.

Despite the success of network embedding on static graphs, many real systems are not static as the nodes and edges are evolving over time, i.e., the vertices and edges may appear, vanish, or even reappear. In Fig. 1, we present a illustrative example of movie review network. Existing temporal network embedding approaches either model the networks as discrete-time dynamic graphs (Yu, Yin, and Zhu 2017) or continuous-time dynamic graphs (Nguyen et al. 2018). The former aims to capture the graph-level dynamics, which aggregates temporal information into a sequence of snapshots. While the latter one aims to preserve the node-level dynamics, which is often presented as a collection of timestamped edges. However, most of them neglect the existing duration of temporal nodes and edges in the real systems. For example, the length of bank account history (i.e., duration of nodes) is a key indicator for identifying synthetic identities (Zhou et al. 2018a); the average communication delay between sensors (i.e., duration of edges) is often used for studying the reliability of sensor networks (Chen et al. 2015).

Needless to say, it is crucial to encode such fine-grained dynamic patterns into a salient representation. In this paper, we want to answer the following open answers: First (*C1. Model*), how to define a complementary temporal network model, which preserves the fine-grained graph dynamics? Second (*C2. Context Sampling*) how to identify and sample the key structural and temporal patterns that exist in the real systems? Third (*C3. Algorithm*), how to learn a salient network representation that captures the rich and complex network contexts of the network structures and network dynamics?

To address the aforementioned challenges, we formally define the fine-grained temporal network embedding problem and present a salient temporal network embedding framework named *FiGTNE*. The objective of *FiGTNE* is to learn a salient network embedding that preserves the structural and temporal contexts in temporal networks. In particular, we propose the time-reinforced random walk (TRRW) to sample the structural and temporal contexts over network evolution. Moreover, to guide the context sampling of TRRW, we develop a bi-level sampling strategy that carefully selects the key nodes with certain timestamps as initial points of random walks to extract the temporal network context.

To summarize, our work makes the following contributions:

- A novel notion of temporal networks and the formal problem definition of fine-grained temporal network embedding;
- A fine-grained temporal network embedding (*FiGTNE*) framework, which is able to admit a variety of fine-grained structural-temporal patterns and encode rich temporal network context into a salient representation;
- Extensive experiments on several real datasets, showing *FiGTNE* achieves consistent prediction performance improvement over baseline methods.

Our paper is organized as follows. Related works are reviewed in Section 2, followed by the notation and problem definition in Section 3. In Section 4, we present our proposed framework *FiGTNE*. Experimental results are reported in Section 5 before we conclude the paper in Section 6.

Related Work

Network embedding (Zhang et al. 2019) has been increasingly employed to learn low-dimensional representations of networks, such that the vector-based machine learning models can be easily performed on various graph analytic tasks, such as classification (Perozzi, Al-Rfou, and Skiena 2014), recommendation (Tang et al. 2015; Liu et al. 2019), truth discovery (Zhou and He 2017), rare category characterization (Zhou et al. 2018c; 2018b). A number of methods are proposed for the static graphs, which can be typically categorized into the matrix factorization based methods (Li et al. 2018), the random walks based methods (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016; Nguyen et al. 2018) and the graph neural network based methods (Kipf and Welling 2016; Wu, He, and Xu 2019).

More recently, there is a growing interest in learning network embedding in a dynamic setting. In general, the existing models can be categorized into *discrete* dynamic network embedding and *continuous* dynamic network embedding, where the former one assumes the graph aggregates temporal information into a series of graph snapshots and the latter one targets the graph with continuously timestamped nodes and edges. In the context of *discrete* dynamic network embedding, Zhu et al. generates node embeddings by employing non-negative matrix factorization with the temporal smoothness constraint on a series of time-evolving adjacency matrices; Another line of works (Yu, Yin, and Zhu 2017) is developed based on graph convolution networks (Kipf and Welling 2016) for dealing with attributed temporal networks; Point process is a commonly-used method for random events in time, which has been introduced to model the neighborhood formation sequence (Zuo et al. 2018) and interleaved dynamics (Trivedi et al. 2018); Zhou et al. proposes DynamicTriad to model how to derive a closed triad from an open triad. Unlike *discrete* dynamic networks, the *continuous* dynamic networks (Paranjape, Benson, and Leskovec 2017) are often presented as a collection of timestamped interactions between entities. Nguyen et al. is one of the first to learn the time-respecting representation of continuous-time dynamic networks by simulating temporal random walks. However, most of the existing methods neglect the fact that duration information of nodes and edges in real systems. Moreover, noisy temporal contexts and non-uniform distributions over timestamped graph elements are also overlooked. All aforementioned issues would lead to the incomplete delineation of fine-grained structures in temporal networks. Therefore, in this paper, we propose a generic framework for learning the representation of the fine-grained temporal network by introducing the novel notion of temporal networks and two techniques for preserving fine-grained network dynamics.

Problem Definition

A conventional way to model temporal networks is to build time-aggregated graphs (e.g., strictly growing graphs, time-evolving graphs), which typically collect the changes over time and aggregate the temporal information into a sequence of snapshots. However, an open question here is how to select the temporal granularity, i.e., the sliding window size, for generating time-aggregated graphs. When the granularity is coarse, some fine-grained patterns (e.g., a dense cluster showing in a short burst) may be ignored; when the granularity is fine, it may result in a huge number of snapshots and prohibitive computational costs (Zhou et al. 2015). To address this issue, various continuous-time dynamic network models (Rossi et al. 2013; Paranjape, Benson, and Leskovec 2017; Nguyen et al. 2018) are proposed to address the temporal granularity issues. While, most, if not all, existing works either model the network as increasing graphs or ignore the duration of graph elements (e.g., nodes, edges). In this paper, we introduce the definitions of the temporal node, temporal edge and the induced temporal network as follows.

Definition 1 (δ-Temporal Node) *In the temporal network, a δ-temporal node $v^{(t,\delta)}$ represents the vertex v that appears at timestamp t and exists for δ duration.*

Definition 2 (δ-Temporal Edge) *In the temporal network, a δ-temporal edge $(u,v)^{(t,\delta)}$ represents the connection between node u and node v that appears at timestamp t and exists for δ duration.*

Definition 3 (Temporal Network) *A temporal network $\widetilde{G} = (V,E)$ is formed by a collection of n temporal nodes $V = \{v_1^{(t_{v_1},\delta_{v_1})}, v_2^{(t_{v_2},\delta_{v_2})}, \ldots, v_n^{(t_{v_n},\delta_{v_n})}\}$ and a sequence of m temporal edges $E = \{e_1, e_2, ..., e_m\}$, where $e_i := (u_{e_i}, v_{e_i})^{(t_{e_i},\delta_{e_i})}$.*

Definition 3 provides a generic definition of the aforementioned networks. In particular,

- when $t_{v_1} = t_{v_2} = ... = t_{v_n} = t_{e_1} = t_{e_2} = ... = t_{e_m}$ and $\delta_{v_1} = \delta_{v_2} = ... = \delta_{v_n} = \delta_{e_1} = \delta_{e_2} = ... = \delta_{e_m} = \infty$, $\widetilde{G}$ represents a static graph, where all nodes and edges exist once the graph is created.
- when $\delta_{v_1} = \delta_{v_2} = ... = \delta_{v_n} = \infty$ and $\delta_{e_1} = \delta_{e_2} = ... = \delta_{e_m} = \infty$, $\widetilde{G}$ represents a strictly growing graph, where the nodes and edges can be only added but not removed.
- when $\{t_{v_i}\}_{i=1}^n$ and $\{t_{e_i}\}_{i=1}^m$ only has p unique values $t_1 < t_2 < ... < t_p$ ($p << n$ and $p << m$), $\delta_{v_i} = t_{k+1} - t_k, \forall i \in [1,n]$ and $\delta_{e_i} = t_{k+1} - t_k, \forall i \in [1,m]$ and $k \in [1,p]$, $\widetilde{G}$ represents a time-evolving graph that all nodes and edges exist within a single snapshot.

In (Nguyen et al. 2018), the authors define the temporal walk as a sequence of edges with non-decreasing timestamps. However, the temporal non-decreasing constraint would easily discard a number of informative edges, which leads to too short random walks. Following the similar mechanism but without the temporally non-decreasing constraint, we introduce the notion of k-Length Temporal Walk as a sequence of temporal edges.

Definition 4 (k-Length Temporal Walk) *In the temporal network, a k-length temporal walk $\widetilde{W} = \{\widetilde{w}_1, \ldots, \widetilde{w}_k\}$ is defined as a sequence of incident temporal edges traversed one after another, i.e., $\{(u_{\widetilde{w}_1}, v_{\widetilde{w}_1})^{(t_{\widetilde{w}_1},\delta_{\widetilde{w}_1})}, \ldots, (u_{\widetilde{w}_k}, v_{\widetilde{w}_k})^{(t_{\widetilde{w}_k},\delta_{\widetilde{w}_k})}\}$.*

With the above notations, our problem can be formally defined as follows:

Problem 1 *Fine-Grained Temporal Network Embedding*

Input: *(i) a temporal network $\widetilde{G} = (V,E)$, (ii) the user-defined dimension of embedding space d.*

Output: *a d-dimensional network embedding D.*

Proposed Algorithm

In this section, we introduce the details of our proposed *FiGTNE* framework. We begin with reviewing the basics of static network embedding and then generalize it to the dynamic setting, by (1) introducing the mechanism of TRRW, and (2) developing a bi-level sampling strategy to jointly guide random walks to focus on the key regions of the temporal network.

Static Network Embedding

We briefly introduce the basics of static network embedding approaches (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016) that based on word2vec model (Mikolov et al. 2013a). In general, these methods operate in two steps: (1) they first sample and extract pair-wise graph context information, mostly node-context pairs (v,c), via random walks; then (2) they leverage the node proximity manifested to learn a low-dimensional vector-wise representation for each node in the observed graph. Usually, given a static graph $G = (V,E)$, the network embedding method aims to maximize the overall likelihood of G in terms of the local structures by the following objective function

$$\mathcal{L} = \prod_{v \in V} \prod_{c \in \mathcal{N}(v)} p(c|v;\theta) \tag{1}$$

where $\mathcal{N}(v)$ is the neighborhood of v, $p(c|v;\theta)$ defines the conditional probability of having a context c given a node v and θ denotes the trainable embedding representation.

Fine-Grained Temporal Network Embedding

Here, we generalize the Skip-Gram based framework to learn the node representation that incorporates fine-grained context information (e.g., timestamp) in the temporal network. The critical challenge is how to extract such temporal context information from the observed graph. To achieve this, we propose TRRW, which is designed for extracting both topological structure and temporal context information from temporal networks. Also, we design a bi-level context sampling strategy to guide the evolutionary random walk to select the network context from the key regions and timestamps. The overall framework is given in Fig. 2.

Time-Reinforced Random Walk. One key challenge is how to perform random walks on the temporal networks, where each node and edge are associated with a specific timestamp t. The conventional static random walk can be interpreted as a Markov Chain process where each node represents a single state. However, there are possibly multiple edges between the same two nodes, which makes it impossible to differentiate which edge is traversed by recording the node sequence. Therefore, same with (Nguyen et al. 2018), we define the transition probability matrix $\boldsymbol{P} \in \mathbf{R}^{m \times m}$ based on edge streams, which is

$$P(e_{out}|e_{in}) = \frac{w(e_{out}|e_{in})}{\sum\limits_{e \in \mathcal{I}(v_{e_{in}})} w(e|e_{in})} \tag{2}$$

where $w(e_{out}|e_{in})$ is the weight of observing e_{out} given the previous temporal edge e_{in} in the given temporal network $\widetilde{G}$; $\mathcal{I}(v_{e_{in}})$ denotes the set of edges that are incident (only out-edges are considered if the graph is directed) to the node $v_{e_{in}}$. Comparing with the static random walks, the process of transition defined by Eq. 2 can be described as a Markov Chain process, where each temporal edge represents a single state.

Based on Eq. 2, we can easily simulate random walks on temporal networks by initializing $w(e_{out}|e_{in})$ with equal weights or temporally biased distributions (e.g., (Nguyen et

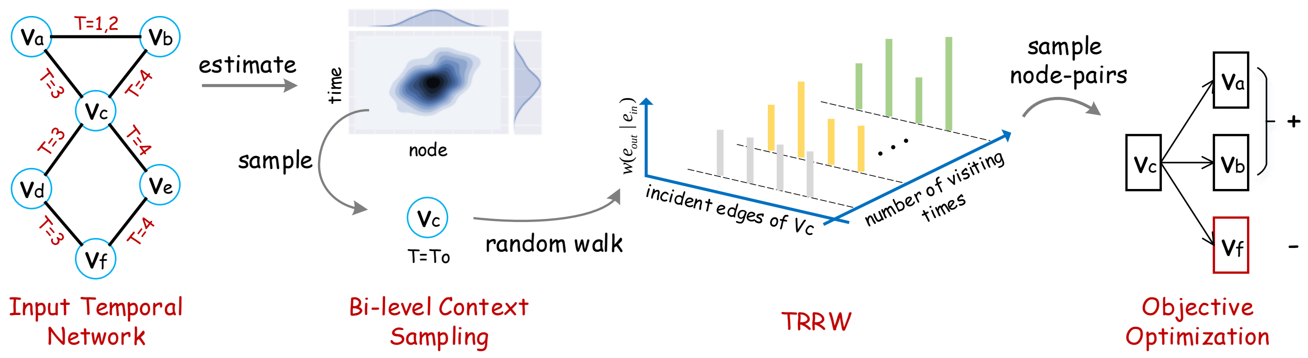


Figure 2: An illustration of the proposed framework.

al. 2018)). However, in this way, we may be still not able to fully capture the evolution process of temporal networks. Inspired by Vertex Reinforced Random Walk (Pemantle 1992), which dynamically modifies the transition probability between each pair of nodes based on the visiting times of random walks step by step, we propose a novel random walk model named TRRW to extend this idea to model the evolution process of temporal networks. In particular, we define the cross-edge weight $w(e_{out}|e_{in})$ as

$$w(e_{out}|e_{in}, s) = min[n(e_{out}|e_{in}, s) + \epsilon, w_T(e_{out}|e_{in})] \tag{3}$$

where $n(e_{out}|e_{in}, s)$ indicates the number of visiting times for the edge pairs (e_{in}, e_{out}) at step s. ϵ is a small positive constant to ensure the random walk can be conducted properly from the beginning when $n(e_{out}|e_{in}, s) = 0$. Here we set ϵ to 1 if $e_{out} \in \mathcal{I}(v_{e_{in}})$ otherwise 0. The temporal context function $w_T(e_{out}|e_{in})$ measures the relationship between e_{in} and e_{out} based on the timestamp of temporal edges. From the definition of Eq. 3, we can conclude that starting with temporally unbiased strategy of random walk, $w(e_{out}|e_{in}, s)$ would gradually transit into the temporal context defined by $w_T(e_{out}|e_{in})$. In this process, the generated random walks encode both temporally unbiased and temporally biased information at the same time. As for the temporal context function $w_T(e_{out}|e_{in})$, a number of temporal contexts can be selected here, and two different strategies are discussed in the following.

Unbiased: In this naive case, the cross-edge weight does not depends on the timestamp of e_{in} and e_{out}, and all edges in $\mathcal{I}(v_{e_{in}})$ share the same weight, which is

$$w_T(e_{out}|e_{in}) = 1 \tag{4}$$

Rank: In this case, the cross-edge weight depends on time closeness between e_{in} and e_{out}, i.e. the difference value of t_{out} and t_{in}, which is

$$w_T(e_{out}|e_{in}) = rank(|t_{out} - t_{in}|, \mathbf{\Delta t}) \tag{5}$$

where $\mathbf{\Delta t} = \{|t_e - t_{in}| : e \in \mathcal{I}(v_{e_{in}})\}$; the function $rank(x, \mathbf{s})$ outputs the rank of x in the sequence $\mathbf{s}$, which is sorted in a descending order.

As a summary, the details of TRRW are presented in Algorithm 1.

Algorithm 1 Time-Reinforced Random Walk

```
Require:
    Temporal network $\widetilde{G} = (V, E)$, temporal context $w_T(e_{out}|e_{in})$, walking length $l$, number of random walks $n_w$.
Ensure:
    A sequence of random walks $\widetilde{W}$.
1: Initialize the weight of each pair of connected temporal edges $e_{in}$ and $e_{out}$ to be 1, i.e., $w(e_{out}|e_{in}) = 1$, and compute $\boldsymbol{P}$ based on Eq. 2.
2: for $i = 1 : n_w$ do
3:     for $j = 1 : l$ do
4:         Conduct TRRW for one step based on the current transition probability matrix $P$.
5:         Add the traversed temporal edge $(u, v)^{(t,\delta)}$ to one random walk in $\widetilde{W}$.
6:         Update $w$ using Eq. 3 and recalculate $P$ using Eq. 2.
7:     end for
8: end for
9: Return $\widetilde{W}$.
```

Bi-Level Context Sampling. Different from the static graph, the nodes and edges may appear multiple times at different timestamps in the temporal network. As temporal nodes and edges are mostly not distributed uniformly over the structural domain and temporal domains, that is to say, the importance of network structures showing at different regions of the graph and at different timestamps may vary a lot. Sampling the network context by uniformly traversing each node at each timestamp is not a good option to capture the key temporal network context in terms of computational efficiency. In this paper, we propose a bi-level context sampling strategy to guide our embedding framework to select the initial temporal node $v^{(t,\delta)}$ for conducting TRRW.

Here, we focus on estimating the sampling probability distribution, i.e., $Pr(y = 1|v^{(t,\delta)}) = Pr(y = 1|v_s, v_t)$, of each temporal nodes $v^{(t,\delta)}$, where $y = 1$ indicates $v^{(t,\delta)}$ will be sampled, v_s and v_t denote the structural location (the distribution over nodes) and temporal location (the distribution over time) of $v^{(t,\delta)}$. To estimate the overall sampling probability $Pr(y=1|v_s,v_t)$, we assume the structural location and the temporal location of $v^{(t,\delta)}$ follow a weak dependence (Abney 2002) given $y=1$. Then, we have

$$\begin{aligned}
&Pr(y=1|v_s,v_t)\\
&=\frac{Pr(y=1)Pr(v_s,v_t|y=1)}{Pr(v_s,v_t)}\\
&\geq\frac{\alpha Pr(y=1)Pr(v_t|y=1)Pr(v_s|y=1)}{Pr(v_s,v_t)}\\
&=\alpha\frac{Pr(y=1)\frac{Pr(y=1|v_t)Pr(v_t)}{Pr(y=1)}\frac{Pr(y=1|v_s)Pr(v_s)}{Pr(y=1)}}{Pr(v_s,v_t)}\\
&=\frac{\alpha}{Pr(y=1)}Pr(y=1|v_t)Pr(y=1|v_s)\frac{Pr(v_t)Pr(v_s)}{Pr(v_t,v_s)}\\
&=\gamma Pr(y=1|v_t)Pr(y=1|v_s)\frac{Pr(v_t)Pr(v_s)}{Pr(v_t,v_s)} \qquad (6)
\end{aligned}$$

where $\gamma=\frac{\alpha}{Pr(y=1)}$ is a constant. In practical, we let $Pr(y=1|v_t)$ and $Pr(y=1|v_s)$ follow a uniform distribution over structural and temporal domains, which could also be any other user-defined distributions; $Pr(v_s)$, $Pr(v_t)$ and $Pr(v_t,v_s)$ can be estimated using empirical distribution or kernel density estimation approaches (Scott 2015). As a special case, given a temporal node $v^{(t,\delta)}$, when the structural location and temporal location are conditional independent, i.e., $Pr(v_s,v_t|y=1)=Pr(v_s|y=1)Pr(v_t|y=1)$, then $\alpha=1$ and *Inequality* 6 becomes equality. By estimating the joint distribution $Pr(y=1|v_s,v_t)$, our method can guide the TRRW to extract the network context from the key regions of the temporal network.

Algorithm 2 Fine-Grained Temporal Network Embedding

```
Require:
    Temporal network $\widetilde{G}=(V,E)$, negative sampling size $k$, embedding size $d$.
Ensure:
    A temporal network embedding $D$.
 1: Compute the sampling distribution $Pr(y=1|v^{(t,\delta)})$ based on Eq. 6.
 2: while not converged do
 3:     Sample a batch of temporal nodes $S$ following the sampling distribution $Pr(y=1|v^{(t,\delta)})$.
 4:     for $s \in S$ do
 5:         Sample positive pairs $(i,c)$ using TRRW in Alg. 1.
 6:         Sample $k$ negative pairs for each $(i,c)$.
 7:     end for
 8:     Update $D$ via optimizing Eq. 1.
 9: end while
10: Return $D$.
```

Optimization Algorithm

Now, we are ready to present our proposed *FiGTNE* algorithm, which is summarized in Algorithm 2. The algorithm is optimized via negative sampling (Mikolov et al. 2013b). The given inputs are the temporal network $\widetilde{G}$, negative sampling size k, and embedding size d. Step 1 computes the sampling distribution $Pr(y=1|v^{(t,\delta)})$ for selecting initial temporal nodes. Within each batch iteration, we first sample a batch of initial temporal nodes S in Step 3; then, for each initial node, our algorithm samples positive pairs from generated random walks and k negative pairs for each positive pair; in Step 8, we adopt stochastic gradient descent (SGD) to train our model using the sampled positive and negative node-context pairs. The algorithm finally returns the learned temporal network embedding D after convergence.

Experiments

In this section, we demonstrate the performance of our proposed *FiGTNE* algorithm on quantitative evaluations and parameter sensitivity analysis.

Experiment Setup

Table 1: Statistics of datasets

Name	#Nodes	#Edges	#Labels	Timespan (days)
DBLP	1909	8237	4	9855
Taobao	10924	97479	5	229
Epinion	40331	112929	3	4322
Criteo	22158	65503	7	90

Datasets: The statistics of all datasets used in our experiments are summarized in Table 1.

DBLP (Zhou et al. 2018b) is a citation network based on papers from several visualization conferences. Each citation would create an edge between two papers and the timestamps of the edges are in years. These papers are categorized into four different topics, which serve as the label of each paper.

Taobao[1] is a bipartite online shopping network, where both customers and items are considered as vertices, and the behavior of purchasing an item would create an undirected edge between the item and the customer. The category of the item is considered as the label of the node.

Epinion[2] is a consumer review network between users and items, and each rating corresponds to a temporal edge between the related user and item. We consider the category of the item as the label.

Criteo[3] is an action network. Each action is timestamped, which creates a temporal edge between the customer and product, and the category of the product is considered as the label.

Comparison Methods: Our proposed method is compared with two static network embedding approaches

[1]https://tianchi.aliyun.com/datalab/dataSet.html?spm=5176.100/073.0.0.4703ea7EOGscIdataId=649

[2]https://www.cse.msu.edu/ tangjili/trust.html

[3]http://ailab.criteo.com/criteo-sponsored-search-conversion-log-dataset/

(i.e., DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) and node2vec (Grover and Leskovec 2016)) and two dynamic network embedding approaches, i.e., TNE (Zhu et al. 2016) and CTDNE (Nguyen et al. 2018). The dimension of the node embedding is set to 64.

For the static network embedding baselines, we aggregate all our datasets into unweighted graphs. As for hyperparameters, we tune three hyperparameters in DeepWalk, which are the number of random walks n_w, the walk length l and the window size w, respectively. All combinations of parameters are tested given $n_w \in \{10, 30, 50\}$, $l \in \{10, 30, 50\}$ and $w \in \{3, 5, 7\}$, and the best metric is reported; node2vec extends DeepWalk by introducing a mechanism of generating biased random walks. Two hyperparameters p and q control the procedure of generating random walks. We test all combinations of $p \in \{0.5, 1, 1.5\}$ and $q \in \{0.5, 1, 1.5\}$.

For the dynamic network embedding baselines, we first aggregate all our datasets into ten snapshots with equal time intervals to accommodate the input of TNE; CTDNE is designed for continuous-time dynamic networks, where each node and edge are timestamped. Same with DeepWalk, we test CTDNE given different combinations of the number of random walks n_w, the walk length l and the window size w. In addition, we test CTDNE variants with uniform and linear distributions.

Three different variants of *FiGTNE* with different definitions of $w(e_{out}|e_{in})$ are compared here. $w(e_{out}|e_{in})$ in *FiGTNE*-Unbiased and *FiGTNE*-Rank are set to fixed temporal contexts defined in Eq. 4 and Eq. 5, respectively. $w(e_{out}|e_{in}, s)$ in *FiGTNE*-TRRW is set to the dynamic weight defined in Eq. 3 with $w_T(e_{out}|e_{in})$ setting to the one defined in Eq. 5.

Quantitative Evaluations

In each dataset, we apply the embedding methods on the following two tasks:

Node Classification: The task of node classification is to predict the label of the node based on its features. By considering the embedding representations generated by the network embedding methods as node features, we train a logistic classifier with a given number of training data and predict the labels of the rest nodes. We report the Micro-F1 with different sizes of training data to measure the performance of the methods.

Link prediction: The task of link prediction aims to predict the probability of two nodes that are connected by an edge. Here we use the dot product to compute the probability of two nodes being linked based on the learned embedding vectors. For each dataset, we randomly select a specific ratio (i.e., 0.1%, 0.5% and 1%) of edges from the most recent time as positive ones and remove them from the original dataset. Then the same number of node pairs that are not connected by an edge is also randomly sampled as negative ones. We report the AUC with different ratios of sampled edges to evaluate the comparison methods.

Qualitative results regarding node classification and link prediction are shown in Fig. 3 and Fig. 4, respectively. In summary, we have the following observations: (1) our proposed algorithm *FiGTNE*-TRRW outperforms the comparison methods across all the datasets in most cases of the two tasks; (2) there are two exceptions that node2vec performs slightly better than *FiGTNE* in the link prediction task. A possible guess here is that the temporal context would possibly introduce noise to the link prediction model, when some certain connections between some vertices that belong to the same category but are not closed in the time; (3) the performance of TNE is worse than the static network embedding methods (DeepWalk and node2vec). A potential explanation would be that naively aggregating the networks into several snapshots would break the temporal patterns to several snapshots, which deteriorates the performance.

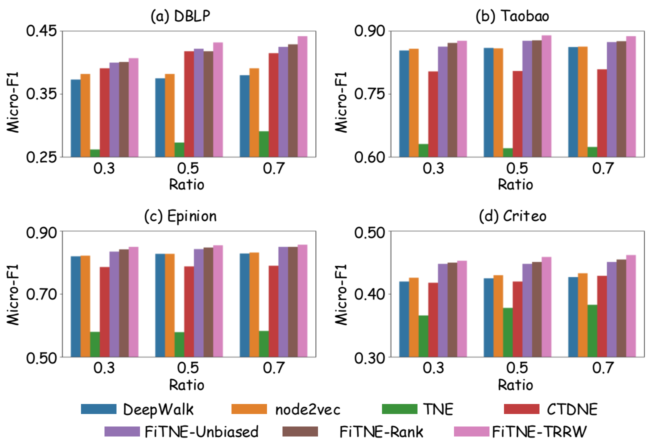


Figure 3: Node classification results (Micro-F1) with different ratios of training data.

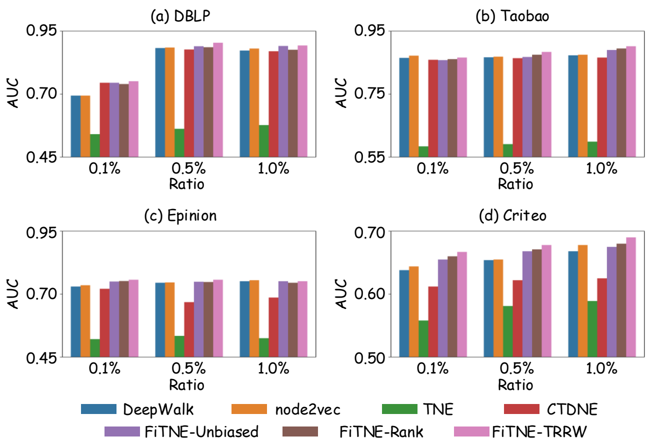


Figure 4: Link prediction results (AUC) with different ratios of sampled edges.

To test the effectiveness of bi-context sampling, we compare it with the case that $Pr(y = 1|v_s, v_t)$ is set to uniform distribution based on *FiGTNE*-TRRW, and Table 2 gives the relative performance gain. We can obverse that bi-context sampling performs better than the uniform distribution.

Table 2: Relative performance gain of bi-context sampling on the two tasks of node classification and link prediction with different ratios of training data.

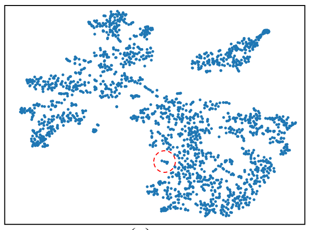
	Node Classification			**Link Prediction**		
Dataset	30%	50%	70%	0.1%	0.5%	1.0%
DBLP	0.5%	1.2%	1.1%	1.2%	1.4%	0.9%
Taobao	2.3%	2.5%	2.7%	1.6%	2.1%	2.3%
Epinion	3.3%	3.9%	2.8%	4.0%	3.1%	3.0%
Criteo	2.7%	3.6%	3.6%	2.5%	2.9%	2.1%

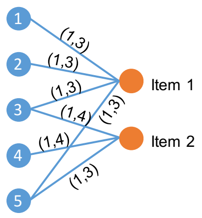
Case Study

Here we conduct further analysis on the Criteo dataset to demonstrate the effectiveness of the temporal context that is defined based on the similarity of time duration. In the Criteo dataset, each edge also has an attribute called time duration that represents the time between the action and conversion (the action leads to one consumption). Therefore, we generate the embedding representations of users using *FiGTNE*-TRRW based on the temporal context defined as

$$\begin{aligned} & w_T(e_{out}|e_{in}) = \\ & rank(|t_{out} - t_{in}|, \mathbf{\Delta t}) + rank(|\delta_{out} - \delta_{in}|, \mathbf{\Delta d}) \end{aligned} \tag{7}$$

where $\mathbf{\Delta t} = \{|t_e - t_{in}| : e \in \mathcal{I}(v_{e_{in}})\}$ and $\mathbf{\Delta d} = \{|\delta_e - \delta_{in}| : e \in \mathcal{I}(v_{e_{in}})\}$. Then we visualize part of them (three thousand users) using t-SNE, which is shown in Fig. 5a. Similarly, we give a mininature that forms a small cluster in Fig. 5b. We can see that the five users show similar interest on the products and behaviors, which suggests that there may be a brushing for making fake sales.


(a)



(b)

Figure 5: (a) Visualization of embedding representations generated from Criteo using t-SNE; (b) Visualization of the mininature consists of nodes marked in Fig. 5(a). The timestamp and time duration of the edge are labeled on the edge.

Parameter Analysis

In this section, we study the two additional hyperparameters in TRRW, which are the interval of updating edge weights K and the increment per update Δ. The introduction of K is mainly for parallel implementation. If the transition matrix $\boldsymbol{P}$ is fixed, we can easily have multiple walkers and generate the random walks starting from different nodes in a parallel way. But if we update $\boldsymbol{P}$ once an edge is traversed, the random walks can only be generated in a sequential way, which is quite inefficient. In addition, other than the simplest case given in Eq. 3, we can increase the speed of evolution of $w(e_{out}|e_{in}, s)$ by multiplying $n(e_{out}|e_{in}, s)$ with a positive constant Δ that is larger than 1. The two hyperparameters control how fast $\boldsymbol{P}$ would fully encode temporal information. Here we test all combinations on the dataset Taobao given $K \in \{n_r, 2n_r, 3n_r, 4n_r, 5n_r, 6n_r, 7n_r, 8n_r, 9n_r, 10n_r\}$ (n_r represents the number of updates of $\boldsymbol{P}$ when dealing with one batch of nodes in Algorithm 1) and $\Delta \in \{1, 2, 4, 8, 16, 32, 64, 128\}$. The results are shown in Figure 6. We can see that a relatively fast speed (i.e., large Δ and small K) of changing $\boldsymbol{P}$ leads to high Micro-F1 score. There are also some spikes. We presume that they would relate to the randomness of generating random walks.

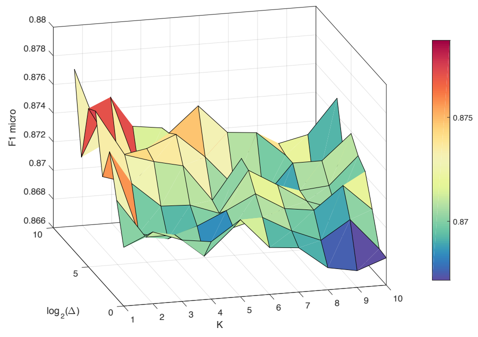


Figure 6: Hyperparameter analysis on Taobao.

Conclusion

In this paper, we propose a fine-grained temporal network embedding framework named *FiGTNE*. We start by introducing the novel notions of temporal networks, where each node and edge are timestamped and exist within a certain duration. On top of that, we propose two techniques that are designed for preserving the structural and temporal contexts in temporal networks: time-reinforced random walk and bi-level context sampling. Extensive experiments on several real-world networks demonstrate the performance of *FiGTNE* in terms of effectiveness and parameter sensitivity. Our future work would focus on how to design a learnable temporal context function since current temporal contexts are all hand-crafted, which limits the performance of our proposed model.

Acknowledgments

This work is supported by National Science Foundation under Grant No. IIS-1552654 and Grant No. IIS-1813464, the U.S. Department of Homeland Security under Grant Award Number 17STQAC00001-02-00 and Ordering Agreement Number HSHQDC-16-A-B0001, and an IBM Faculty Award. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

References

Abney, S. 2002. Bootstrapping. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

Cannistraci, C. V.; Alanis-Lobato, G.; and Ravasi, T. 2013. Minimum curvilinearity to enhance topological prediction of protein interactions by network embedding. *Bioinformatics*.

Chen, L.; Warner, J.; Yung, P. L.; Zhou, D.; Heinzelman, W.; Demirkol, I.; Muncuk, U.; Chowdhury, K.; and Basagni, S. 2015. Reach 2-mote: a range-extending passive wake-up wireless sensor node. *ACM Transactions on Sensor Networks (TOSN)* 11(4):64.

Dong, Y.; Chawla, N. V.; and Swami, A. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

Kipf, T. N., and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Li, J.; Cheng, K.; Wu, L.; and Liu, H. 2018. Streaming link prediction on dynamic attributed networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM.

Liu, X.; He, J.; Duddy, S.; and O'Sullivan, L. 2019. Convolution-consistent collective matrix completion. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2209–2212. ACM.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Nguyen, G. H.; Lee, J. B.; Rossi, R. A.; Ahmed, N. K.; Koh, E.; and Kim, S. 2018. Continuous-time dynamic network embeddings. In *3rd International Workshop on Learning Representations for Big Networks (WWW BigNet)*.

Paranjape, A.; Benson, A. R.; and Leskovec, J. 2017. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM.

Pemantle, R. 1992. Vertex-reinforced random walk. *Probability Theory and Related Fields*.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 701–710. ACM.

Rossi, R. A.; Gallagher, B.; Neville, J.; and Henderson, K. 2013. Modeling dynamic behavior in large evolving graphs. In *WSDM*. ACM.

Scott, D. W. 2015. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee.

Trivedi, R.; Farajtbar, M.; Biswal, P.; and Zha, H. 2018. Representation learning over dynamic graphs. *arXiv preprint arXiv:1803.04051*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Wang, X.; Cui, P.; Wang, J.; Pei, J.; Zhu, W.; and Yang, S. 2017. Community preserving network embedding. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Wu, J.; He, J.; and Xu, J. 2019. Net: Degree-specific graph neural networks for node and graph classification. *arXiv preprint arXiv:1906.02319*.

Yu, B.; Yin, H.; and Zhu, Z. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.

Zhang, S.; Tong, H.; Xu, J.; and Maciejewski, R. 2019. Graph convolutional networks: a comprehensive review. *Computational Social Networks* 6(1):11.

Zhou, Y., and He, J. 2017. A randomized approach for crowdsourcing in the presence of multiple views. In *2017 IEEE International Conference on Data Mining (ICDM)*, 685–694. IEEE.

Zhou, D.; Wang, K.; Cao, N.; and He, J. 2015. Rare category detection on time-evolving graphs. In *Data Mining (ICDM), 2015 IEEE International Conference on*. IEEE.

Zhou, D.; He, J.; Davulcu, H.; and Maciejewski, R. 2018a. Motif-preserving dynamic local graph cut. In *2018 IEEE International Conference on Big Data (Big Data)*, 1156–1161. IEEE.

Zhou, D.; He, J.; Yang, H.; and Fan, W. 2018b. Sparc: Self-paced network representation for few-shot rare category characterization. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

Zhou, L.; Yang, Y.; Ren, X.; Wu, F.; and Zhuang, Y. 2018c. Dynamic network embedding by modeling triadic closure process. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhu, L.; Guo, D.; Yin, J.; Ver Steeg, G.; and Galstyan, A. 2016. Scalable temporal latent space inference for link prediction in dynamic social networks. *IEEE Transactions on Knowledge and Data Engineering*.

Zuo, Y.; Liu, G.; Lin, H.; Guo, J.; Hu, X.; and Wu, J. 2018. Embedding temporal network via neighborhood formation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.