Embedding Temporal Network via Neighborhood Formation

Yuan Zuo
School of Economics and Management
Beihang University
Beijing 100191, China
zuoyuan@buaa.edu.cn

Guannan Liu*
School of Economics and Management
Beihang University
Beijing 100191, China
liugn@buaa.edu.cn

Hao Lin
School of Economics and Management
Beihang University
Beijing 100191, China
linhao2014@buaa.edu.cn

Jia Guo
School of Economics and Management
Beihang University
Beijing 100191, China
guojia1608@buaa.edu.cn

Xiaoqian Hu
School of Economics and Management
Beihang University
Beijing 100191, China
huxiaoqian@buaa.edu.cn

Junjie Wu†
School of Economics and Management
Beijing Advanced Innovation Center for Big Data and Brain Computing
Beihang University
Beijing 100191, China
wujj@buaa.edu.cn

ABSTRACT

Given the rich real-life applications of network mining as well as the surge of representation learning in recent years, network embedding has become the focal point of increasing research interests in both academic and industrial domains. Nevertheless, the complete temporal formation process of networks characterized by sequential interactive events between nodes has yet seldom been modeled in the existing studies, which calls for further research on the so-called *temporal network embedding* problem. In light of this, in this paper, we introduce the concept of *neighborhood formation sequence* to describe the evolution of a node, where temporal excitation effects exist between neighbors in the sequence, and thus we propose a Hawkes process based Temporal Network Embedding (HTNE) method. HTNE well integrates the Hawkes process into network embedding so as to capture the influence of historical neighbors on the current neighbors. In particular, the interactions of low-dimensional vectors are fed into the Hawkes process as base rate and temporal influence, respectively. In addition, attention mechanism is also integrated into HTNE to better determine the influence of historical neighbors on current neighbors of a node. Experiments on three large-scale real-life networks demonstrate that the embeddings learned from the proposed HTNE model achieve better performance than state-of-the-art methods in various tasks including node classification, link prediction, and embedding visualization. In particular, temporal recommendation based on arrival rate inferred from node embeddings shows excellent predictive power of the proposed model.

*Corresponding author
†Also with , Beijing Key Laboratory of Emergency Support Simulation Technologies for City Operations, Beihang University.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
KDD '18, August 19–23, 2018, London, United Kingdom
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-5552-0/18/08…$15.00
https://doi.org/10.1145/3219819.3220054

CCS CONCEPTS

• **Information systems → Data mining**; *Network data models*; • **Computing methodologies → Dimensionality reduction and manifold learning**;

KEYWORDS

Temporal Network; Network Embedding; Learning Representation; Hawkes Process

ACM Reference Format:
Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. 2018. Embedding Temporal Network via Neighborhood Formation. In *KDD '18: The 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 19–23, 2018, London, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3219819.3220054

1 INTRODUCTION

Network embedding has become a focal point of study in recent years, aiming at representing large-scale networks by mapping nodes to low-dimensional space [6, 9, 10, 20]. It provides an efficient way to uncover the network structure and perform various network mining tasks such as node classification [20], link prediction [9], community detection [5], *etc.* Recent work on network embedding methods [9, 20, 22] generally focuses on static network structure by considering various contextual information, *e.g.*, the neighbors of a node. One non-trivial but often-overlooked assumption underlying these methods is that the neighbors of a node are *unordered*; in other words, the link formation history is omitted.

In reality, however, a network is formed by adding nodes and edges sequentially, which indeed should be regarded as a dynamic process driven by interactive events between a node and its neighbors. As a result, the neighborhood of a node is not formed simultaneously and the observed snapshot network structure is the accumulation of neighborhood in certain time periods. For example,

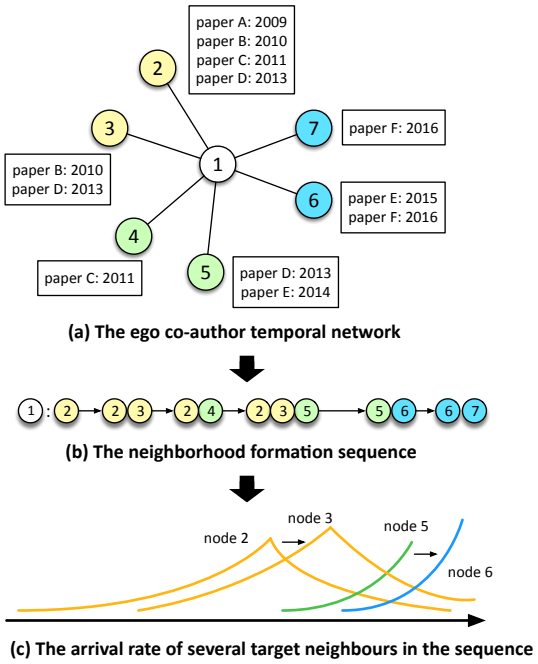


Figure 1: Toy example for temporal network and neighborhood formation sequence.

Figure 1a shows the ego network of one author: node 1, and his/her neighbors *i.e.*, nodes 2 to 6. Taking a snapshot perspective on the network structure, we only observe the up-to-date co-authorship, whereas how and when the nodes are connected remains unknown. As a matter of fact, in most real networks, edges between nodes are generally established by sequential events, which constitute the so-called *temporal network* [12]. For example, the co-author network is driven by co-authored papers with clear timestamps. As shown in Figure 1a, we see each edge is annotated with several papers co-authored between node 1 and its neighbors in chronological order. The ego temporal network can thus be unfolded into a node-specific neighbor sequence according to the timing of events, which is defined as *Neighborhood Formation Sequence* and shown in Figure 1b.

Neighborhood formation sequences indeed contain much richer information than the static network snapshot in representing nodes. We can see from Figure 1b that neighbors might appear repeatedly in the sequence due to the repeated co-authorship between the authors, which could provide more semantic meanings than one single edge. We can also observe the dynamic changes of neighbors more explicitly in the sequence that node 1 is more likely to co-author with nodes 2 and 3 in the earlier years while shifts to co-author with 5, 6, and 7 recently. Moreover, the events of the target neighbors in the sequence are correlated with each other, or in other words, historical events can influence the current neighborhood formation. For example, we assume node 1 is a Ph.D. student in the earlier years, and hence most of his/her papers are co-authored with his/her advisor, *e.g.*, node 2. Node 3 might be an academic friend of node 2, and therefore the co-authorship with node 2 can excite some other neighbor arrival events with node 3. Assume node 1 becomes a professor after graduation, then he/she can develop some new co-authorships, and the influence from the advisor might vanish and the newly connected co-authors (*e.g.*, node 5) might further excite other co-authors (*e.g.*, node 6), as shown in Figure 1c.

Therefore, how nodes connect to their neighbors sequentially can reveal the dynamic changes, and should be exploited to better represent the network. Though several recent dynamic network embedding methods [29, 30] have attempted to model the dynamics by segmenting timelines into fixed time windows, the learned embeddings are still representations in particular time periods without taking the dynamic process into account. To directly model neighborhood formation sequences, therefore, remains a great challenge.

To tackle the above challenge, in this paper, we propose a Hawkes process based Temporal Network Embedding (HTNE) method. Specifically, we firstly induce the neighborhood formation sequence from the network structure driven by sequential events. Since *Hawkes process* [11] well captures the exciting effects between sequential events, particularly the influence of history on the current events, we adapt it for modeling the *neighborhood formation process*. Then, in order to derive the node embeddings from the Hawkes process, low-dimensional vectors are fed into the Hawkes process by mapping the pairwise vectors to the base rate and the influence from the history, respectively. Moreover, the influence of historical neighbors on the current neighbor formation can vary with different nodes, and thus we further adopt *attention mechanism* to enhance the expressiveness of the influence from the neighborhood formation history on the current neighbor formation event.

In order to deal with large-scale networks, our HTNE model is solved by optimizing the likelihood of neighborhood formation sequences rather than the conditional intensity function. We conduct extensive experiments on three large-scale real-life temporal networks to train the node embeddings and apply them for several interesting tasks including node classification, link prediction and visualization. In particular, we design a temporal recommendation experiment by utilizing the conditional intensity function inferred from neighborhood formation sequence. The experimental results all show significant improvements over some state-of-the-art baseline methods.

2 PRELIMINARIES

2.1 Neighborhood Formation Sequence

Network formation can be viewed as a dynamic process of adding nodes and edges, which encodes the underlying mechanisms of how nodes connect with each other and evolve in the network. The static network snapshot is indeed accumulation of historical formation process and only represents the structure at one particular time period. Therefore, it is more desirable to consider the detail historical network formation process in order to recover the network structure and represent the network. However, most prior studies in network embedding often focus on network snapshot without resorting to how the network is formed, and most network embedding approaches are based solely on the static neighborhood in representing a node [9, 20, 22].

Traditionally, dynamic network attempts to capture the evolving network structures based on predefined time windows [29, 30],

which however, can only represent the snapshots in different time periods and cannot reveal the complete temporal process of network formation. Therefore in this paper, we trace back the network formation process by tracking the neighborhood formation of each node. As a matter of fact, edges are generally formed by sequential interactive events between pairwise nodes. For example, in co-author network, the relationship between authors are formed due to their co-authorship on a paper at certain time. In one network snapshot, the relationship between any two authors can only be represented by one single edge, with the times of papers ever co-authored as weight. Driven by the sequential interactive events on each edge, we can formally define the temporal network.

Definition 2.1. (**Temporal Network.**) Temporal network is a network with edges annotated by chronological interactive events between nodes, which can be denoted as $\mathcal{G} =< \mathcal{V}, \mathcal{E}; \mathcal{A} >$, where $\mathcal{V}$ denotes the set of nodes, $\mathcal{E}$ denotes the set of edges and $\mathcal{A}$ denotes the set of events. Each edge $(x, y) \in \mathcal{E}$ between nodes x and y is annotated by chronological events , *i.e.*, $\mathbf{a}_{x,y} = \{a_1 \rightarrow a_2 \rightarrow \cdots\} \subset \mathcal{A}$, where a_i denotes an event with timestamp t_i.

Given the temporal network, the evolution of co-authorship can be more explicitly depicted, which can also provide clues for predicting future co-authors of a node. Therefore, the adjacent neighbors of a node in the network can be organized as a sequence according to the ascending time of the interactive events with the neighbors, representing the neighborhood formation process. Then, we can formally define the neighborhood formation sequence in brief as follows.

Definition 2.2. (**Neighborhood Formation Sequence.**) Given a source node in temporal network $x \in \mathcal{V}$, the neighborhood of the node is $\mathcal{N}(x) = \{y_i | i = 1, 2, \cdots\}$, and the edge between the node and each neighbor is annotated with chronological interactive events $\mathbf{a}_{x,y_i}$. Mathematically, the neighborhood formation sequence can be represented as a series of target neighbor arrival events, *i.e.*, $\{x : (y_1, t_1) \rightarrow (y_2, t_2) \rightarrow \cdots \rightarrow (y_n, t_n)\}$, with each tuple representing an event that node y_i is formed as a neighbor of node x at time t_i.

It is worth mentioning that the neighborhood of a node usually indicates a non-repeated node set. While according to the definition of *Neighborhood Formation Sequence*, each neighbor can appear repeatedly in the sequence to represent multiple interactions with the source node. With the defined sequence, the changes of node connections over time can be manifested explicitly, such that the hidden structure of nodes in network can be inferred from the sequence. Take the co-author network as an example again, the neighborhood formation sequence of a node shows its changes of co-authors, and we can help infer the authors' research interests.

Moreover, the events in neighborhood formation sequence are not independent because the historical neighbor formation events can influence the current neighbor formation. For instance, a researcher may focus on one particular field such as data mining, and thus the co-authors are mainly data mining researchers. But when deep learning has become a focal point of study in recent years, we may gradually observe some AI researchers in his/her co-authors sequences, and can predict from recent neighborhood sequence that the author may shift the research interests to AI and the future co-authors may have more AI people. Therefore in this paper, we aim to take the dynamic neighborhood formation sequence into consideration, in order to learn the representations of the nodes.

2.2 Problem Definition

Given a large-scale temporal network $\mathcal{G} =< \mathcal{V}, \mathcal{E}; \mathcal{A} >$, the neighbors and the corresponding chronological events of each node $x \in \mathcal{V}$ can be induced into a neighborhood formation sequence $\mathcal{H}_x$ by tracking all the timestamped events in which x interacts with its neighbors. Then, temporal network embedding aims to learning a D-dimensional vector to represent each node, which is indeed learning a mapping function $\phi : \mathcal{V} \rightarrow \mathbb{R}^D$, where $D \ll |V|$.

Different from recent network embedding methods where only the static set of neighbors are considered, we tackle the embedding problem by firstly modeling the neighborhood formation sequence and the excitation effects between the neighbors.

3 METHODOLOGY

3.1 Hawkes Process

Point process models the discrete sequential events by assuming that historical events before time t can influence the occurrence of the current event. Conditional intensity function characterizes the arrival rate of sequential events, which can be defined as the number of events occurring in a small time window $[t, t + \Delta t)$ given all the historical events $\mathcal{H}(t)$.

$$\lambda(t|\mathcal{H}(t)) = \lim_{\Delta t \rightarrow 0} \frac{\mathbb{E}[N(t + \Delta t)|\mathcal{H}_t]}{\Delta t}. \tag{1}$$

Hawkes process is a typical temporal point process, with the conditional intensity function defined as follows,

$$\lambda(t) = \mu(t) + \int_{-\infty}^{t} \kappa(t - s) \mathrm{d}N(s), \tag{2}$$

where $\mu(t)$ is the base intensity of a particular event, showing the spontaneous event arrival rate at time t; $\kappa(\cdot)$ is a kernel function that models the time decay effect of past history on the current event, which is usually in the form of an exponential function.

The conditional intensity function of Hawkes process shows that the occurrence of current event does not only depend on the event of last time step, but is also influenced by the historical events with time decay effect. Such property is desirable for modeling the neighborhood formation sequences, because the current neighbor formation can be influenced with higher intensity by the more recent events, while the events occurring in longer history would contribute less to the current occurrence of target neighbors.

In order to handle different types of arrival events, Hawkes process can be extended to multivariate case where the conditional intensity function is designed for each event type as one dimension [14]. The excitation effects are indeed a sum over all the historical events with different types, captured by an excitation rate $\alpha_{d,d'}$ between dimension d and d'. Next, we introduce how to model the neighborhood formation with multivariate Hawkes process.

3.2 Modeling Neighborhood Formation Sequence via Multivariate Hawkes Process

As discussed previously, when we regard each node as a source, a sequence of target neighbors driven by interactive events can be entailed. The neighborhood formation sequence of a node is indeed a counting process, with the current target node influenced by the historical events. Thus, it is naturally appealing to apply Hawkes process to model the neighborhood formation sequence of the source node x, and the conditional intensity function for the arrival event of target y in the sequence of x can be formulated as,

$$\tilde{\lambda}_{y|x}(t) = \mu_{x,y} + \sum_{t_h < t} \alpha_{h,y} \kappa(t - t_h), \tag{3}$$

where $\mu_{x,y}$ represents the base rate of the event to form an edge between x and y, while h is the historical target node in the neighborhood formation sequence of node x prior to time t. $\alpha_{h,y}$ represents the degree to which a historical neighbor h excites the current neighbor y, and the kernel function $\kappa(\cdot)$ denotes the time decay effect which can be written in the form of an exponential function,

$$\kappa(t - t_h) = \exp(-\delta_s(t - t_h)). \tag{4}$$

To note that the discount rate δ is a source dependent parameter, which illustrates the fact that for each source node, the historical neighbor can influence the current neighbor formation with different intensity.

By following the intensity function in Equation (3), the neighborhood formation sequence of each source node can be modeled. Next, in order to learn the D-dimensional representations for the nodes in the network, each node is assumed to be represented by a D-dimensional vector and fed into the intensity function. Specifically, assume that the node embedding of node i is $\mathbf{e}_i$, then the base rate for connecting source x to target y can be mapped from a function $f(\cdot) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$.

Intuitively, the base rate reveals the natural affinity of source node x with target node y. Thus, we use negative squared Euclidean distance as a similarity measure to capture the affinity between the embeddings of node x and y for brevity, *i.e.*, $\mu_{x,y} = f(\mathbf{e}_x, \mathbf{e}_y) = -||\mathbf{e}_x - \mathbf{e}_y||^2$. Similarly, in computing the historical influence on the current node $\alpha_{h,y}$, we use the same similarity measure $\alpha_{h,y} = f(\mathbf{e}_h, \mathbf{e}_y) = -||\mathbf{e}_h - \mathbf{e}_y||^2$.

As the similarity measure we introduced takes negative value, we apply an exponential function to transfer the conditional intensity rate to a positive real number, *i.e.*, $g : \mathbb{R} \to \mathbb{R}_+$, since $\lambda_{y|x}(t)$ should take positive value when regarded as a rate per unit time. Then, we can define the conditional intensity function for the neighbor as:

$$\lambda_{y|x}(t) = \exp(\tilde{\lambda}_{y|x}(t)). \tag{5}$$

As will be described later, using the $\exp(\cdot)$ as transfer function brings us convenience to define and optimize the likelihood.

3.3 Attention for Sequence Formation

Considering the conditional intensity function, the influence from historical events is decomposed as the affinity between the historical nodes with the current target node. Intuitively, the affinity between the history and the target node should depend on the source node. For example, some researchers may have relatively fixed co-authors through time, such that the neighborhood formation sequence remains stable and is more predictable, and the historical events have larger impacts on the current target node in this scenario. While some other researchers may change their co-authors from time to time, as a result they have varied intensity of affinity with different historical co-authors. Therefore, it is necessary to incorporate such characters of the source nodes in modeling the distinct excitation effects α, which is not addressed in previously proposed conditional intensity function.

Following the recent attention based models for neural machine translation [2], we define the weights between the source node and its historical nodes using a Softmax unit as follows:

$$w_{h,x} = \frac{\exp(-||\mathbf{e}_x - \mathbf{e}_h||^2)}{\sum_{h'} \exp(-||\mathbf{e}_x - \mathbf{e}_{h'}||^2)}. \tag{6}$$

For consistence, we choose negative Euclidean distance function to score the affinity between the source and history node. Therefore, the influence from the historical neighbors on the current target can be re-formulated as,

$$\alpha_{h,y} = w_{h,x} f(\mathbf{e}_h, \mathbf{e}_y) \tag{7}$$

3.4 Model Optimization

By modeling the neighborhood formation sequences with multivariate Hawkes process, we can infer the current neighbor formation events from the conditional intensity. Then, given the neighborhood formation sequence of node x before time t, denoted by $\mathcal{H}_s(t)$, the probability of forming connection between x and the target neighbor y at t can be inferred through the conditional intensity as,

$$p(y|x, \mathcal{H}_x(t)) = \frac{\lambda_{y|x}(t)}{\sum_{y'} \lambda_{y'|x}(t)}. \tag{8}$$

Then, the log likelihood of neighborhood formation sequences for all the nodes in the network can be written as,

$$\log \mathcal{L} = \sum_{x \in \mathcal{V}} \sum_{y \in \mathcal{H}_x} \log p(y|x, \mathcal{H}_x(t)). \tag{9}$$

Due to the $\exp(\cdot)$ transfer function introduced in Equation (5), $p(y|x, \mathcal{H}_x(t))$ is actually a Softmax unit applied to $\tilde{\lambda}_{y|x}(t)$, which can be optimized approximately via negative sampling [18]. Negative sampling helps us to avoid the summation over the entire set of nodes in calculating Equation (8), which costs huge computations. According to the degree distribution $P_n(v) \propto {d_v}^{3/4}$, where d_v is the degree for node v, we sample negative nodes which have not occurred in the neighborhood formation sequence. Then the objective function of the edge between a source x and a historical target node y at time t can be computed as follows,

$$\log \sigma(\tilde{\lambda}_{y|x}(t)) + \sum_{k=1}^{K} \mathbb{E}_{v^k \sim P_n(v)}[-\log \sigma(\tilde{\lambda}_{v^k|x}(t))], \tag{10}$$

where K is the number of negative nodes sampled according to $P_n(v)$, $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function.

In addition, the length of the neighborhood formulation sequence influences the computation complexity of $\lambda_{y|x}(t)$, where nodes have long historical lengths. Thus, in the model optimization, we fix the maximum length of history h and only retain the target nodes in the recent sequence.

Table 1: Data statistics.

	# nodes	# static edges	# temporal edges	# classes
DBLP	28,085	162,451	236,894	10
Yelp	424,450	2,610,143	2,610,143	5
Tmall	577,314	2,992,964	4,807,545	5

We adopt Stochastic Gradient Descent (SGD) to optimize the objective function in Equation (10). In each iteration, we sample a mini-batch of edges with timestamps and fixed length of recently formed neighbors of the source node to update the parameters.

4 EXPERIMENTAL SETUP

We validate the effectiveness of the proposed methods on three large scale real-world networks. Hereinafter, we use "HTNE-a" to denote the HTNE with attention. Four state-of-the-art baseline methods are included for a thorough comparative study.

4.1 Data Sets

We first briefly introduce the three real-world networks used in our experiments, with data statistics listed in Table 1.

DBLP: We derive a co-author network from DBLP[1] of ten research areas (see Table 2). We treat the research areas as labels, and assume that a researcher belongs to a particular area if over half of his or her most recent ten papers were published in corresponding conferences.

Yelp: This dataset is extracted from the Yelp[2] Challenge Dataset. Users and businesses are regarded as nodes, and commenting behaviors are taken as edges. Each business is assigned with one or more categories. We only retain the top five categories during the experiments, and the businesses with more than one categories are labeled by the top one category.

Tmall: This dataset is extracted from the sales data of the "Double 11" shopping event in 2014 at Tmall.com [3]. We take users and items as nodes, purchases as edges. Each item is assigned with one category. We only retain the five most frequently purchased categories during the experiments.

4.2 Baseline Methods

Following are four network embedding methods applied as baselines in our experiments.

LINE [22]: This method optimizes node representations by preserving first-order or second-order proximities for a network. In the comparative study, we employ the second-order proximity to learn representations.

DeepWalk [20]: This method first applies random walks to generate sequences of nodes from the network, and then uses it as input to the Skip-gram model to learn representations.

node2vec [9]: This method extends DeepWalk by developing a biased random walk procedure to explore neighborhood of a node, which can strike a balance between local and global properties of a network.

[1]http://dblp.uni-trier.de
[2]https://www.yelp.com
[3]https://tianchi.aliyun.com/datalab/dataSet.htm?id=5

Table 2: The ten research areas selected from DBLP.

Research Area	Conference
Database	ICDE, VLDB, SIGMOD
Data Mining	KDD, ICDM, SDM, CIKM
Information Retrieval	SIGIR
Artificial Intelligence	IJCAI, AAAI, ICML, NIPS
Computer Vision	CVPR, ICCV
Theory	STOC, SODA, COLT
Computational Linguistics	ACL, EMNLP, COLING
Computer Networks	SIGCOMM, INFOCOM
Operating Systems	SOSP, OSDI
Programming Languages	POPL

ComE [5]: This method models community embedding, which can be utilized to optimize the node embeddings by introducing a community-aware high-order proximity.

4.3 Parameter Settings

For our method, we set the mini-batch size, the learning rate of the SGD, and the number of negative samples to be 1000, 0.01, 5 respectively. We set the history length as 5, 2 and 2 for DBLP, Yelp and Tmall respectively. For LINE, we set the number of total edge samples to be 10 billion, and other parameters are set by default. For other baseline methods, we apply default parameters except for the embedding size, which is fixed to be 128 for all the methods.

4.4 Tasks and Evaluation Measures

We first validate the quality of the learned node embeddings from each model by treating them as features for tasks such as *node classification* and *link prediction*. Then, by performing a customized temporal recommendation task, we evaluate the conditional intensity function $\lambda_{y|x}(t)$ (see Equation (5)) inferred from the node embeddings in our method. We also visualize the node embeddings by arranging the network layout on a two-dimensional space. Finally, we perform a parameter sensitivity study. The evaluation tasks and corresponding measures are described as follows.

Node classification: Given the inferred node embeddings as node features, we train a classifier and predict the node labels. We use both Macro-F1 and Micro-F1 as measures.

Link prediction: We aim to determine if there is an edge between two given nodes based on the absolute difference in positions between their corresponding embedding vectors. We apply Macro-F1 as the measure.

Temporal Recommendation: Given a test time point t, we first train the node embeddings on the data in time interval $[t', t)$, then recommend possible new connections of a source node x at time t. We apply Precision@k and Recall@k as measures.

5 EXPERIMENTAL RESULTS

5.1 Evaluation of Node Embeddings

As described above, we evaluate the quality of learned representations by feeding the representations into the tasks including *node classification* and *link prediction*.

Node classification results. We first apply all the methods on each network to learn its node embeddings, and then train a *Logistic*

Table 3: Link prediction results.

	DBLP	Yelp	Tmall
DeepWalk	0.8126	0.7678	0.7745
LINE	0.6350	0.8529	**0.8265**
node2vec	0.8049	0.7712	0.5901
ComE	0.7921	0.8120	0.6917
HTNE	0.8521	**0.8944**	0.7834
HTNE-a	**0.8608**	0.8861	0.7928

Regression classifier with node embeddings as features. We vary the size of the training set from 10% to 90% and the remaining nodes as testing. We repeat each classification experiment for ten times and report the average performance in terms of both Macro-F1 and Micro-F1 scores. Results on *DBLP*, *Yelp* and *Tmall* are presented in Table 4, 5 and 6 respectively.

As the classification results show, our methods perform the best on all the three datasets. Specifically, HTNE-a performs the best on *DBLP* and *Yelp* consistently with all varying sizes of training data, as measured by both Macro-F1 and Micro-F1. HTNE performs the best on *Tmall* with all varying sizes of training data according to Micro-F1, and performs the best on *Tmall* when the training size is larger than 20% as measured by Macro-F1. The stable performances of our methods against different training sizes indicate the robustness of our learned node embeddings when served as features for node classification.

HTNE-a performs better than HTNE on *DBLP* and *Yelp* as measured by Macro-F1 and Micro-F1, which indicates that attention to history nodes based on the source node could help to learn better node embeddings. It is notable that HTNE-a performs slightly worse than HTNE on *Tmall*, which we believe is due to the fact that purchase behaviors of a user in short term may have less significant temporal patterns as compared to that in long term. More results in Figure 4c can serve as evidence for the above discussions, since when history length is larger than 2, HTNE-a can outperform HTNE on *Tmall*.

Link prediction results. Given an edge and its two ends x and y, we define the edge's representation as $|\mathbf{e}_x - \mathbf{e}_y|$, where $\mathbf{e}_x$ and $\mathbf{e}_y$ are embeddings of x and y respectively. The above definition works for any pair of nodes, no matter an edge exists or not between the nodes, which can be utilized as features for link prediction. On each dataset, we randomly hold out 10,000 edges as positive ones, and also choose 10,000 false ones (*i.e.*, two nodes share no link). We train a *Logistic Regression* classifier on the constructed datasets, and list the Macro-F1 results in Table 3.

From the results, we can find our methods perform the best on *DBLP* and *Yelp*, and HTNE-a performs the second best on *Tmall*. The above promising results suggest that the node embeddings learned by our methods can also serve as favorable features for link prediction. We also notice that LINE performs the best on *Tmall*, which might be due to the characteristics of the dataset. Moreover, LINE performs the best among the baseline methods on *Yelp* and *Tmall* but performs the worst on *DBLP*, while in contrast, our methods achieve satisfactory results in all the dataset, showing that our methods are more robust.

5.2 Evaluation of the Conditional Intensity Function

The conditional intensity function $\lambda_{y|x}(t)$ (see Equation(5)) indicates the arrival rate of target neighbor y given its source node x, time t and history $\mathcal{H}_x(t)$. Loosely speaking, under the temporal network scenario, $\lambda_{y|x}(t)$ can be viewed as the possibility of y being connected to x at t, which can be exploited to recommend the future neighbors of the node. Therefore, we design a temporal recommendation task on *DBLP* co-author network.

We first extract a co-author network from *DBLP* data in the time interval $[t', t)$, and fit each model to that network. Then, given an author, we can apply the fitted model to predict his or her top-k possible co-authors at time t. Since baseline methods purely learn node embeddings, we take the inner product of two researchers' embeddings as the ranking score. While in our method, we directly apply the $\lambda_{y|x}(t)$ as the ranking score for researcher x and y. Specifically, we set t to be the year of 2017, and set t' to vary from the year of 2012 to 2016, *i.e.*, the time span of the training set varies from 1 to 5. We apply the Precision@k and Recall@k as evaluation measures. Besides, to make the results under different time spans more comparable, we only recommend co-authors to the researchers that occur in every single year from 2012 to 2017, and the recommendation results are displayed in Figure 2, where $k = 5, 10$.

From the results we can see HTNE-a consistently outperforms all the baseline methods. As shown in Figure 2, the performance of DeepWalk decreases rapidly with the increasing of time span, and becomes the worst when the time span is larger than 3. Though the performance of node2vec decreases slowly, the performances remain at a worse level. In most cases, LINE performs the second best. However, the Precision@10 and Recall@10 of LINE decrease more rapidly than HTNE-a, which indicates that the modeling of neighborhood formation helps HTNE-a less influenced by out-of-date temporal patterns. The trend of ComE is very similar to that of HTNE-a, which indicates that the higher order network structure, *i.e.*, community structure, can prevent ComE from being severely influenced by out-of-date co-author patterns. Nevertheless, the performance of ComE is less competitive against our proposed HTNE-a. As witnessed in Figure 2, we can see that when the time span increases, the performances of all the methods decrease, which is indeed counterfactual at first glance. Because in most cases, a larger time span with more training data should generally provide a better performance; while the experimental results prove to be on the contrary, but this result exactly confirms the toy example described in Section 1 that the co-authorship of a researcher may evolve with time, such that recent data is more meaningful for recommendation.

5.3 Network Visualization

Network visualization is an effective approach to qualitatively evaluate node embeddings learned by different methods. Here, we employ the t-SNE method [24] to project embeddings of researchers to a 2-dimensional space on the *DBLP* data. Those researchers are sampled from three different research areas namely data mining, computer vision and computer networks. For each area, we randomly choose 500 researchers.

Table 4: Node classification results on *DBLP*.

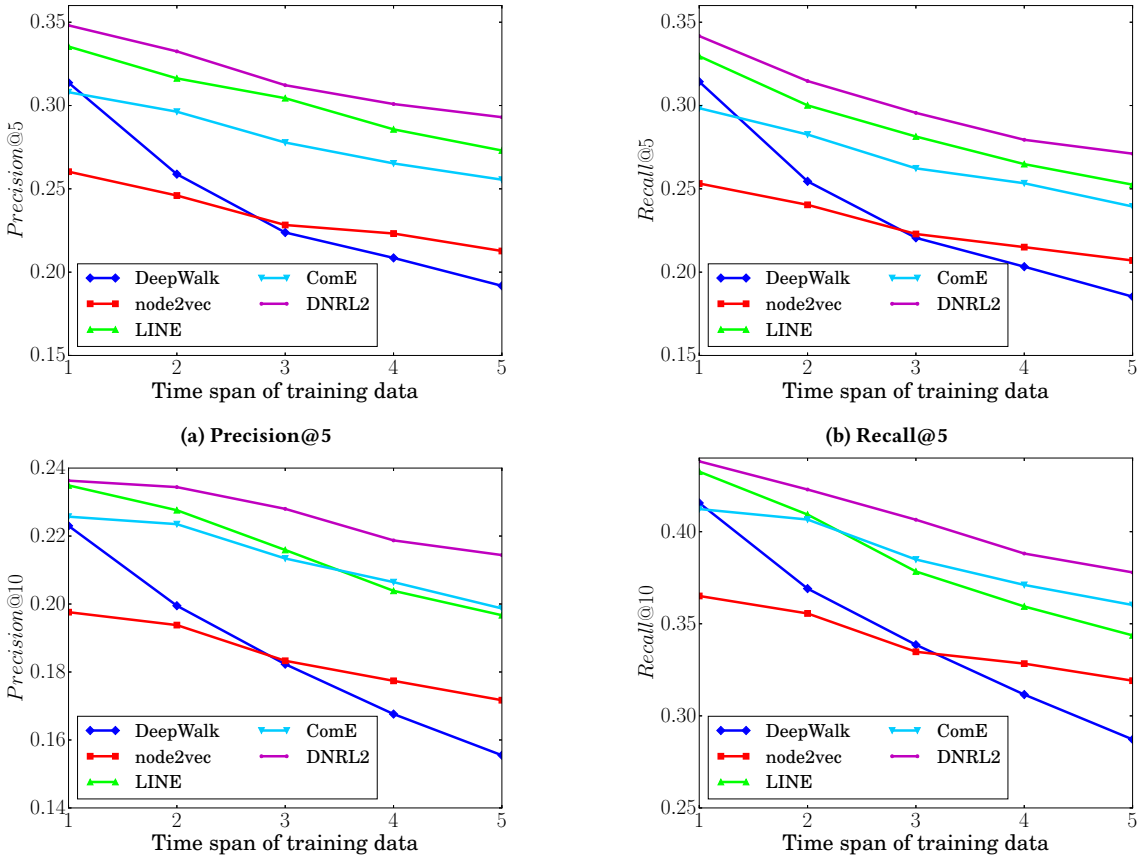
Metric	Method	10%	20%	30%	40%	50%	60%	70%	80%	90%
Macro-F1	DeepWalk	0.6345	0.6553	0.6635	0.6681	0.6698	0.6721	0.6734	0.6725	0.6745
	node2vec	0.6332	0.6511	0.6589	0.6631	0.6655	0.6667	0.6670	0.6639	0.6660
	LINE	0.6163	0.6350	0.6415	0.6455	0.6474	0.6489	0.6498	0.6466	0.6490
	ComE	0.6508	0.6632	0.6680	0.6718	0.6753	0.6764	0.6794	0.6769	0.6791
	HTNE	0.6235	0.6409	0.6490	0.6526	0.6564	0.6592	0.6596	0.6570	0.6608
	HTNE-a	**0.6528**	**0.6656**	**0.6729**	**0.6768**	**0.6799**	**0.6824**	**0.6854**	**0.6836**	**0.6844**
Micro-F1	DeepWalk	0.6435	0.6604	0.6662	0.6690	0.6700	0.6711	0.6711	0.6709	0.6719
	node2vec	0.6492	0.6626	0.6688	0.6717	0.6736	0.6742	0.6742	0.6730	0.6735
	LINE	0.6229	0.6371	0.6433	0.6455	0.6476	0.6484	0.6487	0.6470	0.6463
	ComE	0.6608	0.6723	0.6758	0.6782	0.6806	0.6810	0.6816	0.6801	0.6816
	HTNE	0.6620	0.6693	0.6737	0.6752	0.6778	0.6797	0.6793	0.6777	0.6784
	HTNE-a	**0.6706**	**0.6789**	**0.6834**	**0.6853**	**0.6869**	**0.6881**	**0.6883**	**0.6879**	**0.6866**

Table 5: Node classification results on *Yelp*.

Metric	Method	10%	20%	30%	40%	50%	60%	70%	80%	90%
Macro-F1	DeepWalk	0.3739	0.3776	0.3786	0.3800	0.3791	0.3809	0.3811	0.3807	0.3796
	node2vec	0.4086	0.4193	0.4248	0.4271	0.4269	0.4287	0.4282	0.4278	0.4299
	LINE	0.4097	0.4161	0.4191	0.4188	0.4188	0.4185	0.4188	0.4188	0.4186
	ComE	0.4187	0.4284	0.4358	0.4372	0.4373	0.4375	0.4387	0.4384	0.4401
	HTNE	0.3627	0.3803	0.3892	0.3943	0.3967	0.3997	0.4006	0.3990	0.3993
	HTNE-a	**0.4211**	**0.4348**	**0.4421**	**0.4460**	**0.4485**	**0.4508**	**0.4511**	**0.4507**	**0.4487**
Micro-F1	DeepWalk	0.5361	0.5461	0.5488	0.5505	0.5510	0.5513	0.5516	0.5520	0.5515
	node2vec	0.5488	0.5600	0.5641	0.5662	0.5663	0.5670	0.5672	0.5677	0.5704
	LINE	0.5456	0.5573	0.5611	0.5628	0.5628	0.5632	0.5641	0.5643	0.5635
	ComE	0.5589	0.5672	0.5717	0.5727	0.5730	0.5733	0.5745	0.5749	0.5762
	HTNE	0.5569	0.5644	0.5684	0.5708	0.5716	0.5733	0.5741	0.5734	0.5728
	HTNE-a	**0.5834**	**0.5901**	**0.5941**	**0.5961**	**0.5974**	**0.5988**	**0.5989**	**0.5983**	**0.5971**

Table 6: Node classification results on *Tmall*.

Metric	Method	10%	20%	30%	40%	50%	60%	70%	80%	90%
Macro-F1	DeepWalk	0.4862	0.4892	0.4913	0.4922	0.4923	0.4927	0.4939	0.4941	0.4940
	node2vec	0.5298	0.5348	0.5363	0.5377	0.5368	0.5376	0.5386	0.5391	0.5391
	LINE	0.4311	0.4350	0.4364	0.4370	0.4370	0.4369	0.4382	0.4387	0.4384
	ComE	**0.5373**	0.5416	0.5435	0.5442	0.5442	0.5451	0.5465	0.5455	0.5428
	HTNE	0.5292	0.5413	**0.5476**	**0.5511**	**0.5524**	**0.5539**	**0.5559**	**0.5563**	**0.5563**
	HTNE-a	**0.5373**	**0.5433**	0.5468	0.5479	0.5485	0.5491	0.5496	0.5493	0.5507
Micro-F1	DeepWalk	0.5652	0.5704	0.5721	0.5732	0.5736	0.5742	0.5749	0.5759	0.5758
	node2vec	0.5971	0.6025	0.6037	0.6049	0.6046	0.6052	0.6059	0.6068	0.6067
	LINE	0.5285	0.5339	0.5358	0.5365	0.5369	0.5370	0.5377	0.5388	0.5388
	ComE	0.6059	0.6100	0.6110	0.6117	0.6119	0.6126	0.6136	0.6131	0.6113
	HTNE	**0.6219**	**0.6286**	**0.6314**	**0.6328**	**0.6332**	**0.6339**	**0.6345**	**0.6352**	**0.6343**
	HTNE-a	0.6194	0.6231	0.6248	0.6251	0.6253	0.6259	0.6259	0.6262	0.6266



(a) Precision@5 **(b) Recall@5**

(c) Precision@10 **(d) Recall@10**

Figure 2: Temporal recommendation results.

We illustrate the scatter plots of the 1500 researchers in Figure 3, using the color and shape of a node to indicate its research area. Specifically, we use purple triangle to represent "data Mining", blue dot to represent "computer network" and green star to represent "computer vision". It's not hard to find that both LINE, DeepWalk and node2vec failed to separate all the three areas apart clearly. For example, as shown in Figure 3a, LINE mixtures the data mining and computer networks areas. Besides, three areas are mixed together in the middle of Figure 3a. By modeling the community embedding, ComE achieves satisfactory visualizing result among baseline methods, as the three areas are roughly separated apart from each other. However, there is no clear margin between the areas. Both of our methods can clearly separate three areas apart, while the one with attention achieves a larger margin. Above results indicate that by modeling nodes' neighborhood formation sequence, our method has potential to be applied to community-level applications such as community detection with good performances.

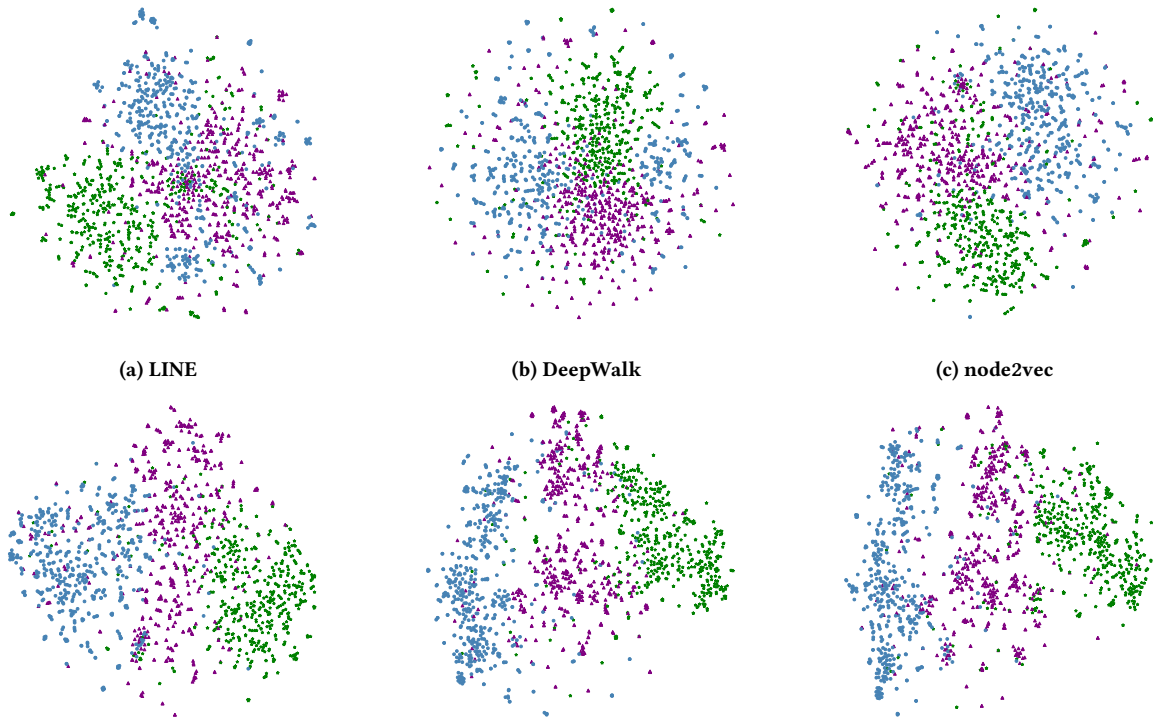
5.4 Parameter Sensitivity

In this subsection, we study an important parameter named history length h, which is designed to truncate the whole history of a source node at a specific time into a recent sequence with fixed length, for reducing computation costs. Specifically, as illustrated in Figure 4, we report the Macro-F1 of HTNE and HTNE-a on *DBLP*, *Yelp* and *Tmall*, with h varying from 1 to 5.

From the results of HTNE, we can see h affects the Macro-F1 differently on three datasets. For example, in *DBLP* and *Tmall*, the Macro-F1 of HTNE first increases along with h, and then begins to drop when $h > 2$. In contrast, the Macro-F1 of HTNE starts to drop at the beginning. Moreover, we can also find that the attention mechanism introduced into HTNE helps our method to be more robust against different settings of h, as the Macro-F1 of HTNE-a is stable on *Yelp* and *Tmall*, even increases along with h on *DBLP*.

6 RELATED WORK

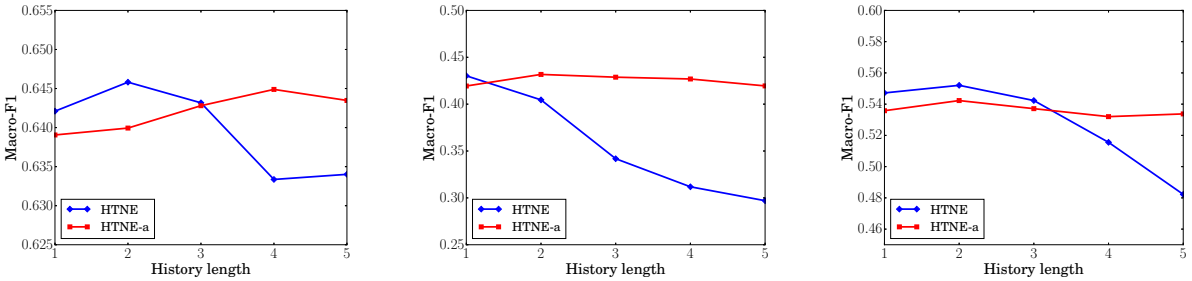
Network embedding, also known as graph embedding or graph representation learning, aims to find a low-dimensional vector space that can maximumly preserve the original network structural information and network properties [4]. Conventional network embedding works have been developed with general dimension reduction techniques, *e.g.*, by constructing and embedding the affinity graph into a low dimensional space [3, 13, 21, 23], or by applying matrix factorization to find the low-dimensional embedding [1].



(a) LINE (b) DeepWalk (c) node2vec

(d) ComE (e) HTNE (f) HTNE-a

Figure 3: Network visualizations. Color of a node indicates the community of the author. Purple: "data mining", blue: "computer network", green: "computer vision".



(a) DBLP (b) Yelp (c) Tmall

Figure 4: Impacts of history length on *DBLP*, *Yelp* and *Tmall*.

However, works along this line usually suffer from heavy computational cost or statistical performance drawbacks, making them neither practical nor effective in large-scale networks.

With the advent of deep learning methods, significant efforts have been devoted to designing neural network-based representation learning models. Especially, Mikolov et al. proposed an efficient neural network framework to learn the distributed representations of words in natural language [16, 17]. Motivated by this work, Perozzi et al.[20] utilized random walks to generate sequences of nodes in large-scale network, and then considered the walking path, *i.e.*, sequence of nodes, as a sentence of words. Grover and Leskovec [9] extended the random walk procedure to a biased version, which can strike a balance between local and global properties of a network. Tang et al. preserved network structures by approximating first-order and second-order proximities in the embedding space [22]. Moreover, high-order proximities of nodes as well as community structures have also been taken into consideration in network embedding models [25, 28].

More recently, network embedding have been continuously studied and received arising attentions from different perspectives. For instance, rich auxiliary information has been leveraged to facilitate the embedding. Pan et al. proposed a tri-party deep neural

network model, which jointly models node structures, contents and labels [19]. To address the problem that previous transductive approaches do not naturally generalize to unseen nodes, Hamilton et al. developed an inductive framework that incorporates node feature information for generating node embeddings [10]. Besides, the strengths of generative adversarial networks [8] have also been exploited with network embedding [6, 26]. Nevertheless, most existing network embedding techniques mainly focused on the setting of static networks. To address this issue, Zhu et al. developed a dynamic network embedding algorithm based on matrix factorization [30]. Yang et al. presented a model with exploring the evolution patterns of triads, which can preserve structural information and get the latent representation vectors for vertices at different timesteps [29]. The dynamics of these embedding methods [27, 29, 30] only focus on segmenting the timelines into fixed time windows, such that the learned node embeddings are only a representation of the snapshot network. In contrast, our model takes the full historical neighborhood formation process into account, providing a more comprehensive representation in view of the history.

Moreover, our proposed network embedding methods are based on Hawkes process, which is a traditionally powerful temporal point process in modeling sequences [11] and has also been studied extensively to adapt for different scenarios. Particularly, recent studies on Hawkes process mainly focus on tackling the challenges of scalability by employing the memoryless property [15] or imposing the low-rank structure on the infectivity matrix [7, 14], etc.

7 CONCLUSIONS

In this paper, we propose a Hawkes process based Temporal Network embedding (HTNE) method. By formulating the neighborhood formation sequence of a temporal network as Hawkes process, HTNE achieves the learning of node embedding, and capturing the influence of the historical neighbors on the current neighbor formation simultaneously. By plugging an attention mechanism in the influence rate of Hawkes process, HTNE gains ability to decide which parts of the historical neighbor are more influential. Extensive experiments on three large scale real-world networks demonstrate the superiority of our methods to leading network embedding methods. Future work includes integrating the attributes of the temporal edges into our model, and seeking the potential of our formulation in solving the optimization problem of Hawkes process with large scale event types.

8 ACKNOWLEDGMENTS

Dr. Junjie Wu's work was partially supported by the National Natural Science Foundation of China (NSFC) (71531001, U1636210, 71725002). Dr. Guannan Liu's work was supported in part by NSFC (71701007).

REFERENCES

[1] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J. Smola. 2013. Distributed Large-scale Natural Graph Factorization. In *WWW*. ACM, New York, NY, USA, 37–48.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[3] Mikhail Belkin and Partha Niyogi. 2001. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. In *NIPS*. MIT Press, Cambridge, MA, USA, 585–591.

[4] HongYun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2017. A Comprehensive Survey of Graph Embedding: Problems, Techniques and Applications. *CoRR* abs/1709.07604 (2017).

[5] Sandro Cavallari, Vincent W. Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2017. Learning Community Embedding with Community Detection and Node Embedding on Graphs. In *CIKM*. 377–386.

[6] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang. 2017. Adversarial Network Embedding. *CoRR* abs/1711.07838 (2017).

[7] Nan Du, Yichen Wang, Niao He, and Le Song. 2015. Time-sensitive Recommendation from Recurrent User Activities. In *NIPS*. 3492–3500.

[8] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *NIPS*. MIT Press, Cambridge, MA, USA, 2672–2680.

[9] Aditya Grover and Jure Leskovec. 2016. Node2Vec: Scalable Feature Learning for Networks. In *SIGKDD*. ACM, New York, NY, USA, 855–864.

[10] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017).

[11] Alan G Hawkes. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58, 1 (1971), 83–90.

[12] Petter Holme and Jari SaramÃďki. 2012. Temporal networks. *Physics Reports* 519, 3 (2012), 97 – 125.

[13] Joseph B Kruskal and Myron Wish. 1978. *Multidimensional Scaling*. CRC press. 875–878 pages.

[14] RÃľmi Lemonnier, Kevin Scaman, and Argyris Kalogeratos. 2017. Multivariate Hawkes Processes for Large-Scale Inference. In *AAAI*.

[15] Remi Lemonnier and Nicolas Vayatis. 2014. Nonparametric Markovian Learning of Triggering Kernels for Mutually Exciting and Mutually Inhibiting Multivariate Hawkes Processes. In *Machine Learning and Knowledge Discovery in Databases*. 161–176.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *NIPS*. Curran Associates Inc., USA, 3111–3119.

[18] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. 3111–3119.

[19] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri-party Deep Network Representation. In *IJCAI*. AAAI Press, 1895–1901.

[20] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *SIGKDD*. ACM, New York, NY, USA, 701–710.

[21] Sam T. Roweis and Lawrence K. Saul. 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290, 5500 (2000), 2323–2326.

[22] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1067–1077.

[23] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290, 5500 (2000), 2319–2323.

[24] Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing High-Dimensional Data Using t-SNE. *JMLR* 9 (2008), 2579–2605.

[25] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *SIGKDD*. ACM, New York, NY, USA, 1225–1234.

[26] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2017. GraphGAN: Graph Representation Learning with Generative Adversarial Nets. *CoRR* abs/1711.08267 (2017).

[27] Jingyuan Wang, Fei Gao, Peng Cui, Chao Li, and Zhang Xiong. 2014. Discovering urban spatio-temporal structure from time-evolving traffic networks. In *Proceedings of the 16th Asia-Pacific Web Conference*. Springer International Publishing, 93–104.

[28] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community Preserving Network Embedding.

[29] Lekui Zhou, Yang Yang, Xiang Ren, Fei Wu, and Yueting Zhuang. 2018. Dynamic Network Embedding by Modeling Triadic Closure Process. In *The AAAI Conference on Artificial Intelligence*.

[30] L. Zhu, D. Guo, J. Yin, G. V. Steeg, and A. Galstyan. 2016. Scalable Temporal Latent Space Inference for Link Prediction in Dynamic Social Networks. *IEEE Transactions on Knowledge and Data Engineering* 28, 10 (Oct 2016), 2765–2777.