Direction-Aware User Recommendation Based on Asymmetric Network Embedding

SHENG ZHOU, Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, China
XIN WANG, Tsinghua University, China
MARTIN ESTER, Simon Fraser University, Canada
BOLANG LI, CHEN YE, ZHEN ZHANG, and CAN WANG, Zhejiang University, China
JIAJUN BU, Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, China

User recommendation aims at recommending users with potential interests in the social network. Previous works have mainly focused on the undirected social networks with symmetric relationship such as friendship, whereas recent advances have been made on the asymmetric relationship such as the following and followed by relationship. Among the few existing direction-aware user recommendation methods, the random walk strategy has been widely adopted to extract the asymmetric proximity between users. However, according to our analysis on real-world directed social networks, we argue that the asymmetric proximity captured by existing random walk based methods are insufficient due to the inbalance in-degree and out-degree of nodes.

To tackle this challenge, we propose InfoWalk, a novel informative walk strategy to efficiently capture the asymmetric proximity solely based on random walks. By transferring the direction information into the weights of each step, InfoWalk is able to overcome the limitation of edges while simultaneously maintain both the direction and proximity. Based on the asymmetric proximity captured by InfoWalk, we further propose the qualitative (DNE-L) and quantitative (DNE-T) directed network embedding methods, capable of preserving the two properties in the embedding space. Extensive experiments conducted on six real-world benchmark datasets demonstrate the superiority of the proposed DNE model over several state-of-the-art approaches in various tasks.

CCS Concepts: • **Information systems → Similarity measures**; • **Networks → Topology analysis and generation;**

Additional Key Words and Phrases: User recommendation, random walk, graph neural networks

This work was supported by the National Key Research and Development Program (Grant 2018YFC2002603), the National Natural Science Foundation of China (Grants 61972349, U1866602), NSERC Discovery Grant "Data mining in heterogeneous information networks with attributes," and the Alibaba-Zhejiang University Joint Institute of Frontier Technologies.

Authors' addresses: S. Zhou and J. Bu (corresponding author), Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, Hang Zhou, Zhejiang, China; emails: {zhousheng_zju, bjj}@zju.edu.cn; X. Wang (corresponding author), Tsinghua University, Beijing, Beijing, China; email: xin_wang@tsinghua.edu.cn; M. Ester, Simon Fraser University, Burnaby, British Columbia, Canada; email: ester@cs.sfu.ca; B. Li, C. Ye, Z. Zhang, and C. Wang, Zhejiang University, Hang Zhou, Zhejiang, China; emails: {21821210, ye.chen, zhen_zhang, wcan}@zju.edu.cn.

1046-8188/2021/11-ART29 $15.00
https://doi.org/10.1145/3466754

ACM Reference format:
Sheng Zhou, Xin Wang, Martin Ester, Bolang Li, Chen Ye, Zhen Zhang, Can Wang, and Jiajun Bu. 2021. Direction-Aware User Recommendation Based on Asymmetric Network Embedding. *ACM Trans. Inf. Syst.* 40, 2, Article 29 (November 2021), 23 pages.
https://doi.org/10.1145/3466754

1 INTRODUCTION

Recent years have witnessed the explosive growth of social networks, like Facebook,[1] Twitter,[2] and Flickr.[3] These social network platforms allow users to build connections with each other throughout the world (i.e., making online friends). One crucial challenge is how to help users discover their possible target users efficiently and accurately, which is also known as user recommendation [19, 49]. Traditional recommendation algorithms such as similarity or "the friends of a friend are likely to be friends" might not satisfy the users' demand since the rich network structure information is not fully explored. Hence, in this article, we investigate the social network structures' intrinsic properties and devise a novel network embedding method to facilitate the recommendation procedure. More specifically, we address the user recommendation task from the perspective of link prediction in the network data, which will benefit from network embedding techniques.

Network embedding aims at learning low-dimensional representations of nodes so that the proximity between nodes in the original graph can be well preserved in the embedding space. Tasks such as link prediction [51, 66] or recommendation [13, 60], node classification [20, 27], and community detection [4, 54] can all greatly benefit from the learned node representations. Although network embedding has been widely investigated in graph analysis literature, it is non-trivial to directly apply them in the recommendation scenario because most existing network embedding methods have primarily focused on undirected networks. However, there are still many directed networks in real-world applications, including social networks, gene-protein networks, and author-paper citation networks, among others. Obtaining a good embedding for directed networks is able to help in many research fields, including social recommendation [7], network evaluation [21], and knowledge base interpretation [14]. Thus, our goal is to design a general method for effective node representation learning in directed networks applicable in recommendation scenarios.

The primary characteristic of the directed social network is the asymmetric proximity between users, which is desired to be preserved in the latent embedding space. Given two users u, v in a directed social network, the probability of user u reaching user v is different from the probability of user v reaching user u due to the differences in node degree distributions and the number of directed paths between them. It is critical for user recommendation to consider the asymmetric proximity, especially when the relationships between users are with a single direction. For example, the user u may have followed the user v while the user v does not follow user u. Only preserve the proximity between users will make bi-direction recommendation, which is not satisfied in the real-world scenario.

Although some existing methods have made attempts to preserve the asymmetric proximity in the directed networks, we argue that the asymmetric proximity they captured is ill defined. Early works [37] directly utilize a deterministic metric such as the Katz [23] score defined on the directed network to capture the asymmetric proximity, which relies on matrix multiplication and cannot scale to large datasets. Sun et al. [45] remove cycles from the network and then infer hierarchy

[1] https://www.facebook.com/.
[2] https://twitter.com/.
[3] https://www.flickr.com/.

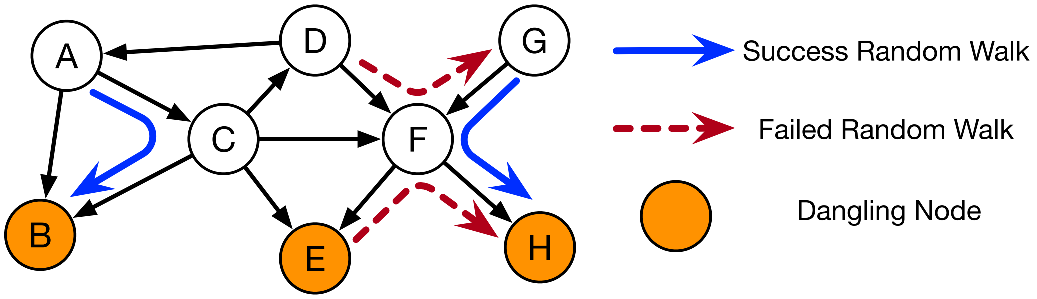


Fig. 1. An example of the random walk on a directed social network. The blue line denotes that random walk successfully follows the direction of edges. The red dotted line denotes that random walk failed to follow the direction of edges. Dangling nodes are nodes without out-edges. Best viewed on screen.

on the resulting incomplete network. Unfortunately, cycles widely exist in real-world networks and carry valuable relational information among nodes. Inferring the incomplete network without cycles will lose crucial relational information and result in suboptimal outcomes. Recent works [25, 64] extend the random walk strategy from undirected networks to directed networks by requiring the random walk to follow the direction of edges [64] or alternate between following and reversing the direction of edges [25]. However, according to our statistics on real-world datasets (Figure 1 presents a toy example, and Section 3.3 provides a detailed analysis), we argue that such random walk reachability suffers from the nodes (a.k.a. dangling nodes) without any outgoing edges and the absence of directed paths between nodes. Therefore, capturing the asymmetric proximity and effectively preserve the asymmetric proximity into embedding space demonstrate significant challenges for user recommendation in directed social networks while existing methods fail to do so.

To tackle the preceding challenges, in this article we first propose a novel informative walk strategy named *InfoWalk* to capture the asymmetric proximity in the directed social network. Intuitively, users who follow the same user will have similar interests (e.g., node *D* and node *G* in Figure 1). The followers of one user are usually interested in the users he follows. InfoWalk captures the preceding properties by enabling the reachability between users in the directed social network with allowing the walk on the network to visit nodes from all directions, which overcomes the limitations raised by the dangling nodes. During each step of the walk, the direction and proximity information are stored in a weight on the step. As a result, InfoWalk outputs a weighted node sequence where the asymmetric proximity can be easily inferred from it.

Given the asymmetric proximity between users captured by InfoWalk, we further propose a directed network embedding method (DNE) with two variants: qualitative directed network embedding (DNE-L) that preserves the discrete asymmetric proximity between nodes and quantitative directed network embedding (DNE-T) that preserves the continuous asymmetric proximity for embedding learning. Two independent embeddings are learned for each node by maximizing the likelihood of observing *directed graph context*, which will be defined in the following section. To evaluate the performance of our proposed directed network embedding method, we conduct extensive experiments on six real-world datasets and compare DNE with several state-of-the-art baseline methods. The experimental results of tasks, including node classification and link prediction, demonstrate the effectiveness of our proposed DNE against existing algorithms.

We summarize the contributions of our article as follows:

(1) We develop a novel informative random walk strategy, InfoWalk, to efficiently capture the asymmetric proximity between users in the directed social network for user recommendation.

(2) We propose our directed network embedding method (DNE) with two variants, qualitative and quantitative directed network embedding (DNE-L and DNE-T), to simultaneously preserve the asymmetric proximity in the latent embedding space.
(3) We conduct extensive experiments on real-world networks to illustrate the advantages of our DNE against state-of-the-art baselines.

The rest of the article is organized as follows. We first briefly review the most related user recommendation and network embedding works in Section 2. Then we use a dataset analysis to give the problem definition and background in Section 3. The proposed direction-aware random walk strategy (InfoWalk) and user recommendation methods (DNE-L and DNE-T) are introduced in Section 4.4. The experimental results and discussions are presented in Section 5.7. Finally, we conclude the article and present some directions for future work in Section 6.

2 RELATED WORK

In this section, we will briefly review the related works of our proposed method, namely the user recommendation and the network embedding.

2.1 User Recommendation

Recommendation techniques have been extensively studied in the past decades. Here, we will give a simple review of different social recommendation methods.

Incorporating social relations has recently drawn massive attention in both academic [56, 57] and industrial communities. Some traditional methods [17, 30] utilize content similarity (e.g., text similarity or visual similarity) or popularity to perform follower/followee recommendation. Ma et al. [33] present a factorization method that shares a common latent space by ratings and social relations. Yang et al. [58] factorize the social trust network and map users into truster and trustee space for the recommendation. Fan et al. [12] unify probabilistic matrix factorization with a neural network for social relation recommendation. More details of this category algorithms could be found in the referred survey of Tang et al. [46]. Another branch of approachs model the recommendation task as a ranking problem. For example, Ding et al. [11] employ a Bayesian personalized ranking deep neural network to make user recommendations. Rafailidis and Crestani [40] investigate location-based social recommendation via deep pairwise learning. Wang et al. [55] design a neural social collaborative ranking recommender system. More recently, graph-based recommendation has attracted researchers' interest [6, 8, 50], and lots of models have been proposed. Among them, Silva et al. [44] use a genetic algorithm to design a graph-based user recommendation system. Lo and Lin [31] propose a weighted minimum-message ratio algorithm for personalized user recommendation. Fan et al. [13] utilize graph neural networks [16] to jointly model the interaction of a user-item graph. Wang et al. [56] divide the social relations into strong ties and weak ties to facilitate the recommendation. Jamali and Ester [22] propose a random walk model for combining trust-based and item-based recommendation. Chen et al. [7] model users' exposure to social knowledge and consumption influence for the recommendation. In another work, Chen et al. [10] conduct social recommendation with an informative sampling strategy. In addition, Chen et al. [9] perform social recommendations based on users' attention and preference. Van den Berg et al. [1] utilize a graph autoencoder invariant to extract embeddings from the user-item interaction graph. Ying et al. [60] propose an efficient graph convolutional neural network to learn node representations for the web-scale recommendation. Monti et al. [36] employ GNNs to learn representations for users and items, and then a diffusion process is conducted with recurrent neural networks [18]. Unlike the methods mentioned previously, our proposed model focuses on investigating the directed network's inherent properties to promote the recommendation procedure, which is rarely studied in the literature.

2.2 Network Embedding

Network embedding methods focus on embedding the nodes in an existing network into a low-dimensional vector space to understand semantic relationships between nodes better.

The proximity preserved in existing network embeddings comes from one of the two buckets: deterministic metric and random walk results. LINE [47] is proposed for the large-scale network, which preserves both first-order and second-order proximities to learn network representations. GraRep[2] can be regarded as an extension of LINE, which considers higher-order proximity. DNGR [3] utilizes denoising stacked autoencoder to learn nonlinear network representations with high-order proximities preservation. SDNE [51] and DGE [65] incorporate graph structure into a deep auto-encoder to preserve the highly nonlinear first-order and second-order proximities. The proximity preserved in the preceding methods relies on the matrix multiplication of the adjacent matrix, which is not scalable in large real-world datasets. To effectively calculate the proximity between nodes, random walks on graphs have been widely used to apply on network data. Among them, DeepWalk [38] and Node2Vec [15] employ a truncated random walk to generate node sequences, which is treated as sentences in language models and fed to the skip-gram model to learn the embeddings. In CARE [24], a custome community-aware random walk is proposed to consider both first- and higher-order proximities as well as community membership information for each node. The random walk results are also fed into the skip-gram model to learn node embedding.

All the approaches mentioned previously, however, are limited to dealing with undirected networks. To embed directed networks, one straightforward solution is ignoring the direction of edges and applying the preceding undirected network embedding methods on the transformed network, which may cause information loss, and the learned embedding method is faulty. Directed network embedding is then put forward since edges in real networks are often associated with directions. Random walk based network embedding methods, including Node2Vec [15] and DeepWalk [38], can be applied to the directed network by guiding the walk with the directed edges. However, the asymmetric proximity between nodes cannot be preserved by the skip-gram model. APP [64] is then proposed by implicitly preserving the Rooted PageRank (RPR), another higher-order proximity feature, in the embedding space. Each node is assigned with source embedding and target embedding to preserve the observed random walk based graph context. HOPE [37] is proposed to approximate asymmetric transitivity based on high-order proximity features (e.g., Adamic Adar (AA), Katz Index (KI), Common Neighbors (CN)) with source and target embedding. However, factorizing the asymmetric proximity matrix is unscalable. The preceding methods are also undermined by the cycles in the directed network, and ATP [45] is then proposed to incorporate both graph hierarchy and reachability information by constructing a novel asymmetric matrix. In NERD [25], an alternating random walk strategy is proposed to walk alternately along and reverse the direction of edges. Although such a strategy can walk along the inverse direction, the visited nodes are limited, and the proximity captured is incomplete. Unlike the preceding methods that preserve the high-order proximities, inspired by Newton's theory of universal gravitation, Salha et al. [42] recently proposed to learn node embedding by reconstructing asymmetric relationships. Some other network embedding methods also incorporate side information like node attributes [16, 27, 48, 66], signs of edgesc [26, 53, 61], and heterogeneous relationships [5], which also motivates the development of embedding complex networks. Another branch of research that is closely related to directed network embedding is the signed network embedding [52, 61]. Although both types of networks have special type of edges, the difference lies in the information contained in the edge. More specifically, the directed edge denotes the asymmetric proximity between nodes in the network, and the signed edges denote the edge type between nodes that is not necessary to be asymmetric. As a result, such asymmetric proximity is the key characteristic of the directed network, which should be preserved by the embedding method.

3 PRELIMINARIES

In this section, we first introduce some background of random walk based embedding methods, then we analyze the drawback of vanilla random walk on real-world networks that motivated our proposed method. Finally, we define the problem we studied in this article.

3.1 Background

Random walk is a popular method of deriving the relationship between nodes on network. Given a start node, it first selects a neighbor of a node at random, moves to this neighbor, then keeps selecting the neighborhood of the node and moves to it until it visits a predefined number of nodes. The sequence of nodes selected this way is a random walk on the graph. The *skip-gram model* originates from the language model and recently extended to network data for embedding learning. Given a sequence of nodes $v_1, v_2, \ldots, v_T$, the objective of the skip-gram model is to maximize the average log probability:

$$\sum_{u=1}^{|\mathcal{V}|} \sum_{-c \leq j \leq c, j \neq 0} \log p(v_{u+j}|v_u), \tag{1}$$

where c is the predefined size of the training context that is the distance on the node sequence generated by random walk. The probability of observing the context node depends on their latent embedding:

$$p(v_{u+j}|v_u) = \frac{\exp(h_u \cdot h_{u+j})}{\sum_{w \in \mathcal{V}} \exp(h_u \cdot h_w)}, \tag{2}$$

where h_u is the embedding of node u. w represents nodes outside the window that are randomly sampled from the node set. Above all, it is easy to find that the sequence generated by random walk plays a central role in embedding learning, as the target is predicting the co-occurrence of nodes on the sequence.

3.2 Definitions

Definition 1 (Directed Network). A directed network is defined as $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$, where $\mathbf{V} = \{v_1, v_2, \ldots, v_N\}$ denotes a set of nodes and N is the number of nodes. $\mathbf{E}$ is a set of direct edges between nodes, $\mathbf{E}_{ij} = 1$ if there exists a direct edge from node v_i to node v_j; otherwise, $\mathbf{E}_{ij} = 0$ and $M = |\mathbf{E}|$ is the number of edges. The neighbor of node v_i can be grouped into two sets named in-neighbor $\mathcal{N}_i^{in}$ and out-neighbor $\mathcal{N}_i^{out}$ where $\forall v_j \in \mathcal{N}_i^{in}, \mathbf{E}_{ji} = 1$ and $\forall v_j \in \mathcal{N}_i^{out}, \mathbf{E}_{ij} = 1$. The in-degree of node v_i is defined as $d_i^{in} = |\mathcal{N}_i^{in}|$ and the out-degree of node v_i is defined as $d_i^{out} = |\mathcal{N}_i^{out}|$.

Definition 2 (Directed Network Embedding). Given a direct network $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$, we aim at learning two independent lower-dimensional embeddings named source embedding $\mathbf{h}_i^s \in R^L$ and target embedding $\mathbf{h}_i^t \in R^L$ for each node $v_i \in \mathbf{V}$ to preserve the asymmetric proximity and hierarchy in the embedding space. The source embedding and target embedding represent the preference of sending and receiving edges for the node. L is the embedding dimension that satisfies $L \ll N$.

3.3 Dataset Analysis

In this section, we conduct a thorough analysis on five real-world directed networks to better understand the drawback of vanilla random walk on directed networks. For each node in the network, we perform the random walk start from this node and the random walk stops when 40 nodes (if

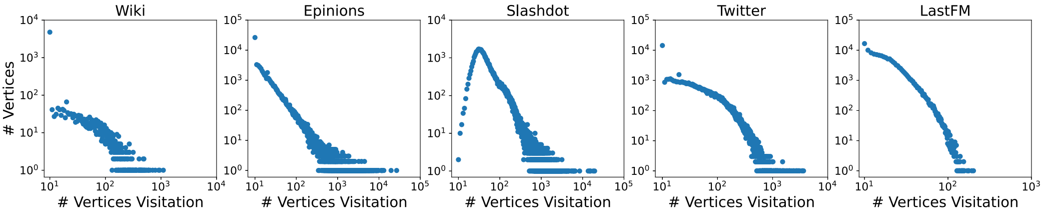


Fig. 2. Number of nodes visited by random walk on five real-world directed networks.

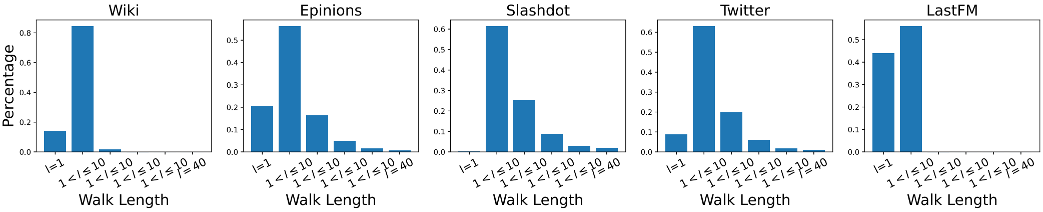


Fig. 3. Statistics of the length of vanilla random walk performed on five real-world directed networks.

possible) are visited by it. We repeat this 10 times and conduct statistic analysis on the visited nodes and random walk length.

Figure 2 illustrates the number of nodes visited by random walk on five real-world directed networks.

Although it satisfies the power-law distribution, we can observe that a considerable number of nodes (marked as "Failed Random Walk") are visited 10 times in the random walk, which means that they are visited only once in the random walk starts from them and they terminate immediately. This refers to the dangling nodes without any out-neighbors; random walk fails to explore the neighborhood information of these nodes and further affects the embedding of other nodes. Figure 4 illustrates the length of random walk from all nodes. We can observe that many random walks cannot walk to predefined walk length 40, and only 37.8% of nodes can walk to 40 nodes. In other words, many random walks cannot well explore the local topology structure due to the absence of a directed path between nodes.

Above all, we observe that many dangling nodes without out-degree exist in the directed network. These nodes and the absence of a directed path between nodes limits the ability of visiting nodes by random walk. It is necessary to overcome the limitation to capture the proximity between nodes without directed paths and further improve the quality of embedding.

4 PROPOSED METHOD

In this section, we first develop an informative random walk strategy (InfoWalk) to capture the asymmetric proximity between nodes in the directed network. We propose two unified methods named *qualitative directed network embedding* (DNE-L) and *quantitative directed network embedding* (DNE-T) to embed the asymmetric proximity into the embedding space. The notations used in this section and the explanations are denoted in Table 1.

4.1 InfoWalk Strategy

As discussed in Section 1, the vanilla random walk on the directed network suffers from the absence of the directed path between nodes and the limitation of dangling nodes. To overcome the limitation, we propose our informative random walk strategy (InfoWalk) in this section. The basic

Table 1. Notations Used in This Article

Notations	Explainations
$\mathcal{R}_{v_i}$	Random walk start from node v_i
$\mathbf{h}_i^s$	Source embedding of node v_i
$\mathbf{h}_i^t$	Target embedding of node v_i
$\mathcal{N}_i^{out}$	Number of out-neighbor nodes
$\mathcal{N}_i^{in}$	Number of in-neighbor nodes
$r_{i,i+1}$	Direction-aware step weight in step i
s_{ij}	Direction-aware score of between the i-th and j-th node in the random walk
$\phi_{u,v}$	Direction-aware weight between node u and node v
DC_u	Direction-aware context of node u

idea of InfoWalk is first to ignore the direction of edges and allow the random walk to visit nodes from all directions. During each step of the random walk, the direction and asymmetric proximity are stored in a carefully designed weight on the step. After the random walk reaches the specified length, we get a step weighted node sequence that expresses asymmetric proximity between nodes, which can be used for directed embedding learning.

Given a directed network $\mathbf{G}$, we denote a random walk started from node v_i as $\mathcal{R}_{v_i}$: $v_i \longrightarrow v_j \cdots \longrightarrow v_k$, which is a sequence of visited nodes, and $\mathcal{R}_{v_i}^k$ denotes the node visited in the k-th step in random walk $\mathcal{R}_{v_i}$. Suppose in the k-th step that the random walk arrives at node v_a: $\mathcal{R}_{v_i}^k = a$, and in the $(k+1)$-th step, the random walk will uniformly walk to in-neighbor $\mathcal{N}_a^{in}$ or out-neighbor $\mathcal{N}_a^{out}$ of node v_a:

$$P\left(\mathcal{R}_{v_i}^{k+1} = b | \mathcal{R}_{v_i}^k = a\right) = \begin{cases} \frac{1}{d_a^{out}+d_a^{in}} & \mathbf{E}_{ab} = 1 \text{ or } \mathbf{E}_{ba} = 1 \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

Such a random walk can be viewed as walking on an undirected network that ignores the direction of edges in $\mathbf{G}$. By compromising the direction, the walk can reach nodes without a path in the directed network and capture the asymmetric proximity. To capture the mixture of direction and proximity between nodes, we further introduce a direction-aware step weight $r_{i,i+1}$ on each step v_i, v_{i+1} with the following rules:

$$r_{i,i+1} = \begin{cases} 1 & \text{if } \mathbf{E}_{i,i+1} = 1 \text{ and } \mathbf{E}_{i+1,i} = 0 \\ -1 & \text{if } \mathbf{E}_{i,i+1} = 0 \text{ and } \mathbf{E}_{i+1,i} = 1 \\ 0 & \text{if } \mathbf{E}_{i,i+1} = 1 \text{ and } \mathbf{E}_{i+1,i} = 1 \end{cases}, \tag{4}$$

where $r_{i,i+1} = 1$ denotes that the random walk follows the direction of the edge, $r_{i,i+1} = -1$ denotes that the random walk step reverses the direction of the edge, and $r_{i,i+1} = 0$ denotes that there exist directed edges in both directions between nodes v_i and v_{i+1}. The motivation behind this is that the indicator $r_{i,i+1}$ stores the direction transformation caused by each random walk step on the directed network, which can be further used for inferring the direction of unobserved edges. For a weighted directed network, $r_{i,i+1}$ can be set by further multiplying the observed weight on the edge, and we leave this as future work.

Given the weight $r_{i,i+1}$ on each step, the result of InfoWalk can be represented as an edge weighted node sequence: $\mathcal{R}_{v_i} : v_i \xrightarrow{r_{i,j}} v_j \xrightarrow{r_{j,j+1}} \cdots \xrightarrow{r_{k-1,k}} v_k$. Based on the step weighted node sequence, we define a score $s_{i,i+k}$ of nodes v_i and v_{i+k} on the sequence as the sum of indicators r

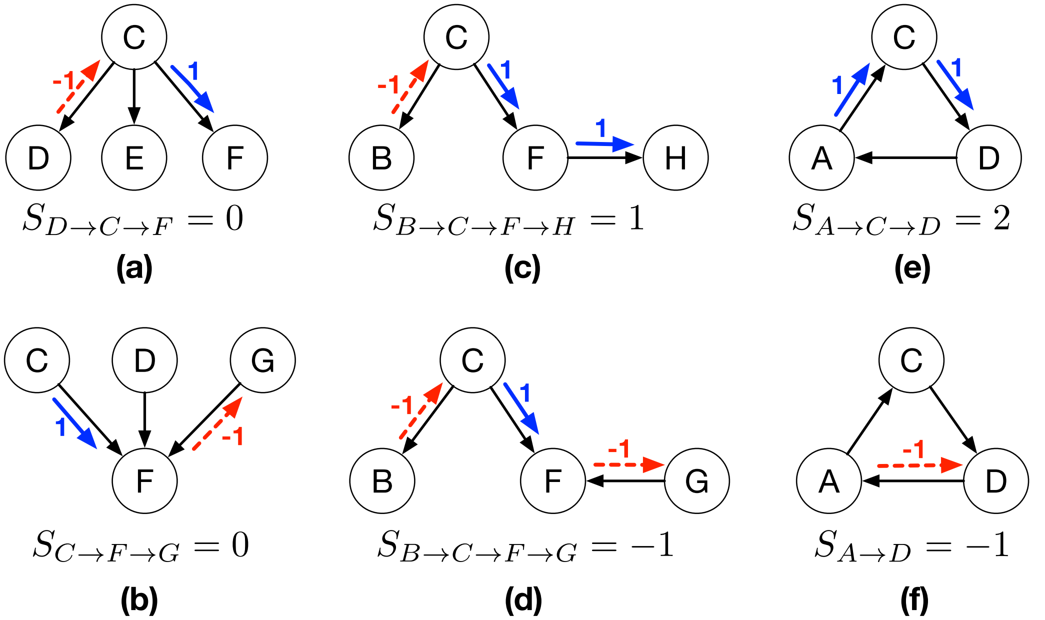


Fig. 4. Example of InfoWalk on a directed network. Red arrows denote steps that reverse the direction of edges. Blue arrows denote steps that follow the direction of edges.

of each step between them:

$$s_{i,\,i+k} = \frac{1}{k} \sum_{j=i}^{i+k-1} r_{j,j+1}, \tag{5}$$

where $r_{j,j+1}$ is step weight j, and $1/k$ is used to normalize the impact from number of steps. Since nodes with long distance from current node cannot provide useful information for embedding learning and calculating scores for these nodes is time consuming, we follow the vanilla random walk strategy and only calculate $s_{i,\,i+k}$ with a small k. From the results of InfoWalk, the following desired properties of a directed network for embedding learning can be inferred:

(1) *Direction transition*: Since each step weight r stores the random walk step follows or reverses the edge's direction, each step's direction transition is also stored. As a result, the sign of $s_{i,\,i+k}$ denotes the direction between nodes: $s_{i,\,i+k} > 0$ denotes that observing node v_i tends to form a directed edge to node v_{i+k}, $s_{i,\,i+k} < 0$ denotes that observing node v_{i+k} tends to form a directed edge to node v_i, and $s_{i,\,i+k} = 0$ denotes that observing node v_i tends to form bi-direction edge to node v_{i+k}. Figure 4 illustrates some typical examples of asymmetric proximity captured by InfoWalk.

(2) *Asymmetric proximity*: InfoWalk can easily capture the asymmetric proximity since InfoWalk walks on the network by ignoring the direction of edges, and nodes with higher in-degree and out-degree will be visited more frequently. As a result, such nodes have a higher chance of occurring in the window of other nodes.

4.2 Directed Network Embedding

In this section, we first define the directed graph context to clarify the target of embedding learning. We propose both qualitative directed network embedding (DNE-L) and quantitative directed

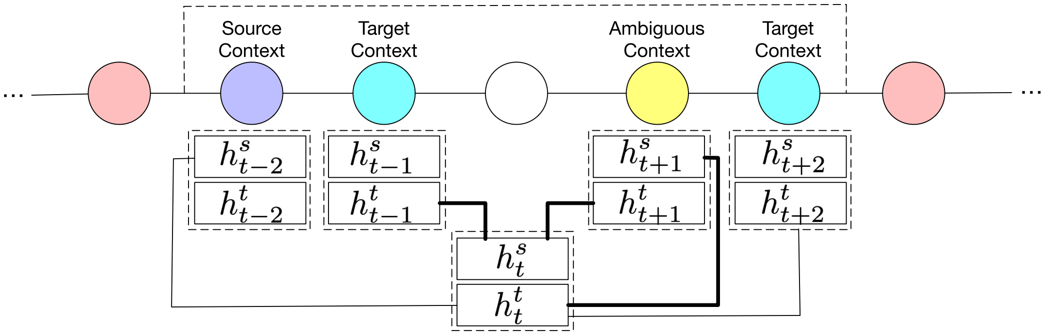


Fig. 5. Overall framework of the DNE method. Given the direction-aware random walk on the directed network, sequences of nodes are generated. The directed graph context is then defined based on the score s_{ij}. The directed relationship between nodes is preserved by the source embedding and target embedding of each node.

network embedding (DNE-T). For each variant, two independent embeddings named *source embedding h^s* and *target embedding h^t* are learned to preserve the asymmetric proximity. The difference between variants lies in how to preserve the asymmetric proximity. Figure 5 illustrates the basic structure of DNE-L and DNE-T.

Definition 3 (Directed Graph Context). Given informative random walk results $\mathcal{R}$ on directed network $\mathbf{G}$, we define the directed graph context as follows: *source context*, target context, and ambiguous context. The source context refers to nodes reached by the DNE method and has a potential direct link to it. The target context refers to nodes reached by the DNE method and has a potential direct link from it. The ambiguous context refers to nodes reached by the DNE method, but the direction between them is ambiguous.

4.2.1 Qualitative Directed Network Embedding. The qualitative directed network embedding methods preserve the asymmetric proximity by maximizing the likelihood of observing the directed graph context node:

$$\max_{\mathbf{H}^s,\mathbf{H}^t} \sum_{u\in\mathbf{V}} \sum_{v\in DC_u} \log P(v|u, s_{u,v}), \tag{6}$$

where DC_u is the directed context of node u, and $s_{u,v}$ is calculated by the DNE method. $P(v|u, s_{u,v})$ is the probability of observing node v in the directed context of node u with score $s_{u,v}$, which can be formulated as follows:

$$P\left(v|u, s_{u,v} > 0\right) = \frac{\exp\left(h^s_u \cdot h^t_v\right)}{\sum_{k\in V} \exp\left(h^s_u \cdot h^t_v\right)}, \tag{7}$$

$$P\left(v|u, s_{u,v} < 0\right) = \frac{\exp\left(h^s_v \cdot h^t_u\right)}{\sum_{k\in V} \exp\left(h^s_v \cdot h^t_u\right)}, \tag{8}$$

$$P\left(v|u, s_{u,v} = 0\right) = \frac{\exp\left(h^s_v \cdot h^t_u + h^s_u \cdot h^t_v\right)}{\sum_{k\in V} \exp\left(h^s_v \cdot h^t_u + h^s_u \cdot h^t_v\right)}, \tag{9}$$

where h^s is the source embedding and h^t is the target embedding. The probability of observing the score is the dot product between source embedding of the node u and target embedding of the node

v. When the score $s_{u,v} = 0$, node u and node v tend to form directed edges from both directions between them. As a result, the probability is the sum of producing embedding from both directions.

4.2.2 Quantitative Directed Network Embedding. Intuitively, the directed graph context nodes have different probability to be visited by InfoWalk from the centering node. Thus, it is reasonable to weight the importance of contextual nodes based on their relative score $s_{u,v}$ to the current node. However, directly apply the score $s_{u,v}$ to weight the importance is suboptimal for the following reasons:

(1) The weight of context nodes with score $s_{u,v} = 0$ should have a positive weight instead of zero.
(2) The weight of context nodes with score $s_{u,v} = 0$ but different random walk length should have different weights.

To overcome the preceding limitations, we have to first reformulate the score score $s_{u,v}$ for weighted training. The weighting function should obey the following properties:

(1) $\pi_0 > 0$,
(2) $\forall m > n, \pi_m > \pi_n$
(3) $\forall i > j, \pi_m^i < \pi_m^j$

where π_m^i denotes the transformed weight of score m with length i. In this work, we use the following transformation from score to weight:

$$\pi_{u,v} = \log\left(\frac{s_{u,v} + 1}{v - u} + b\right), \tag{10}$$

where $s_{u,v}$ is the score calculated in Equation (5), and $b > 0$ is a bias to ensure that the weight is positive. The transformation ensures the following properties of the score:

(1) Nodes with larger score $s_{u,v}$ will have larger weight $\pi_{u,v}$,
(2) Nodes with longer distance on the random walk will have smaller weight $\pi_{u,v}$.

The source and target embedding can be learned by a weighted skip-gram optimization:

$$\begin{aligned} \max_{\mathbf{H}^s,\mathbf{H}^t} \sum_{u \in \mathbf{V}} \sum_{v \in DC_u} & \log P\left(v|u, \pi_{u,v}\right) \\ &= \log \frac{\pi_{u,v} \cdot \exp\left(h_u^s \cdot h_v^t\right)}{\sum_{k \in V} \exp\left(h_u^s \cdot h_v^t\right)}, \end{aligned} \tag{11}$$

4.2.3 Model Optimization. To improve the training efficiency, negative sampling and stochastic gradient descent are used, and the objective can be formulated as follows:

$$\mathcal{L}_{DNE-L} = \log\sigma\left(h_u^s \cdot h_v^t\right) + \sum_{i=1}^{k} \mathbb{E}_{w \sim P_n(v)}\left[\log\sigma\left(-h_u^s \cdot h_w^t\right)\right], \tag{12}$$

$$\mathcal{L}_{DNE-T} = \pi_{u,v}\log\sigma\left(h_u^s \cdot h_v^t\right) + \sum_{i=1}^{k} \mathbb{E}_{w \sim P_n(v)}\left[\log\sigma\left(-h_u^s \cdot h_w^t\right)\right]. \tag{13}$$

4.3 Theoretical Analysis

In this section, we give a theoretical analysis of the asymmetric proximity captured by the InfoWalk method. Given directed network **G**, we use $\hat{\mathbf{G}}$ to represent the undirected network that ignores the direction of edges in directed network **G**.

Let $\mathbf{A}$ be the adjacent matrix of directed network $\mathbf{G}$, and let $\hat{\mathbf{A}}$ be the adjacent matrix of $\hat{\mathbf{G}}$, which can be formulated by

$$\hat{\mathbf{A}} = \mathbf{A} + \mathbf{A}^T - \mathbf{A} \circ \mathbf{A}^T, \tag{14}$$

where $\circ$ is the Hadamard product. The transition probability matrix $\mathbf{P}$ can be formulated as $\mathbf{P} = \hat{\mathbf{D}}^{-1}\hat{\mathbf{A}}$, where $\hat{\mathbf{D}}$ is the diagonal degree matrix of undirected network $\hat{G}$. The weight function of each random walk step can be written as $\mathbf{W} = \mathbf{A} - \mathbf{A}^T$. Since the score $s_{u,v}$ is sum of the edge weights of all the steps taken by the random walk, we first write the score in the iterative matrix form as

$$\mathbf{S}^1 = \mathbf{P} \circ \mathbf{W}, \tag{15}$$

$$\mathbf{S}^k = \hat{\mathbf{D}}^{-1}\mathbf{S}^{k-1} + \hat{\mathbf{D}}^{-1}\mathbf{P}^{k-1}\mathbf{W}. \tag{16}$$

The formulation can be understood as adding weights from each neighborhood visited by the last step (denoted as $\hat{\mathbf{D}}^{-1}\mathbf{S}^{k-1}$) with the weights by the next step (denoted as $\hat{\mathbf{D}}^{-1}\mathbf{P}^{k-1}\mathbf{W}$) . The expectation of the score between nodes that reach in after K steps can be written as

$$\mathbf{S}^k = \sum_{i=1}^{k} \mathbf{P}^{i-1}(\hat{\mathbf{D}}^{-1}\mathbf{W})\mathbf{P}^{k-i}, \tag{17}$$

where $\mathbf{S}$ is the score matrix, $\mathbf{A}$ is the transmission matrix, and $\circ$ is the Hadamard product. As the proximity between nodes decreases as the random walk goes deeper, we introduce the attenuation coefficient $\frac{1}{k}$ with respect to the random walk length k. The overall asymmetric proximity between nodes in the directed network can be written as

$$\mathbf{S} = \sum_{k=1}^{T} \frac{1}{k}\mathbf{S}^k. \tag{18}$$

The preceding equation shows the matrix form of the asymmetric proximity captured by InfoWalk, which can be used to analyze the relationship with existing random walk based methods, and we leave this as future work.

4.4 Complexity and Scalability

Given a directed network $G = \{\mathbf{V}, \mathbf{E}\}$, we only need $O(|V|d)$ space since we employ the stochastic gradient update on the directed graph contexts generated by directed random walk. The time complexity of DNE is $O(|V|drlk)$ where $|V|$ is the number of nodes, d is the dimension of embedding, r is the number of walks per node, l is the walk length, and k is the number of iterations. Our proposed DNE is efficient in both space and time, which can be applied on large-scale datasets.

5 EXPERIMENTAL EVALUATION

In this section, we conduct extensive experiments on several real-world network datasets to evaluate the performance of our proposed DNE. We particularly consider the motivation and impact of directed edges in social networks and the design direction aware user recommendation experiment through empirical evaluation. We aim to answer the following research questions:

RQ1: How does DNE perform compared with state-of-the-art methods on user recommendation tasks?

RQ2: Is it beneficial to overcome the limitation of a non-existing path and dangling nodes by InfoWalk?

RQ3: How do the hyperparameters affect the performance of DNE?

ALGORITHM 1: InfoWalk Strategy and the DNE Algorithm

```
Input: Directed network $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$, embedding dimension $d$, walks per node $r$, walk length $l$,
    window size $k$.
Output: Source embedding $\mathbf{H}^s$ and target embedding $\mathbf{H}^t$
    Initialize $\mathbf{H}^s, \mathbf{H}^t$. Walks={}
    for k=1 to r do
        for $v_i \in \mathcal{V}$ do
            Perform informative random walk of length $l$ start from node $v_i$;
            Modify the step weight $r$ on each step and append weighted sequence $\mathcal{R}_{v_i} : v_i \xrightarrow{r_{i,i+1}} v_{i+1} \longrightarrow \cdots \xrightarrow{r_{l-1,l}} v_l$ to Walks;
        end for
    end for
    for walk ∈ walks do
        for node pair (i,j) within window size $k$ in walk do
            Calculate the score $s_{ij}$;
            Randomly sample negative pairs $(i, k)$;
            Update $\mathbf{H}^s, \mathbf{H}^t$ with Equations (7), (8), (9) and (11);
        end for
    end for
    Return $\mathbf{H}^s, \mathbf{H}^t$.
```

5.1 Experimental Settings

5.1.1 Dataset. We conduct experiments on several real-world social network datasets and bibliographic networks with labels for each node. The social networks with directed edges are used for evaluating user recommendations, whereas the bibliographic networks are used for user profiling. It is worth noting that since collecting large-scale social networks with ground truth labels is hard, we take the bibliographic network with directed edges instead. The statistics of datasets used in our experiments are summarized in Table 2. We have the following:

- *Slashdot networks*: Slashdot is a technology-related news website in which the users can tag each other as friends or foes. There are 77,360 users and 905,468 "friend/foe" relationships between users in the dataset [29]. This dataset has been widely used for social network analysis and user recommendation.
- *Epinions network*: Epinions is a who-trusts-whom online social network of the general consumer review site Epinions.com. This dataset contains the "trust" relationship between users. There are 75,879 users and 508,837 "trust" relationships in the dataset [41]. This dataset has been widely used for trust user recommendation and social recommendation.
- *Twitter network*: Twitter is one of the most popular social network platforms globally. This dataset contains the "following" relationship among users crawled from the network. There are 90,908 users in the network and 443,399 "following" relationships in the dataset [32]. This dataset has been widely used for network analysis and social recommendation.
- *LastFM network*: Last.FM is a streaming radio service provider where users can search for music and get a personalized recommendation. There are 136,420 users and 1,685,524 "following" links among the users in the dataset [62]. This dataset has been widely used for music recommendations.
- *Wiki-Vote network*: Wikipedia is a free encyclopedia written collaboratively by volunteers around the world. The users can vote for another to promote adminship, and this dataset

Table 2. Statistics of Network Datasets Used in the Experiments

Dataset	# Nodes	# Edges	# Labels	% Dangling Node	% Bi-directional Edges
Wiki	7,115	103,689	—	0.141	0.0565
Epinions	75,879	508,837	—	0.204	0.4052
Slashdot	77,360	905,468	—	0.271	0.8783
Twitter	90,908	443,399	—	0.087	0.6066
LastFM	136,409	1,685,524	—	0.439	0.0009
PubMed	19,717	44,338	3	0.803	0.0001
CoCit	44,034	195,361	15	0.451	0.0001

contains the vote data among users. There are 7,115 users and 103,689 "voting" relationships from one user to another. This dataset [28] has been widely used for analyzing the "trust" relationship in the online community.

- *CoCit and PubMed networks*: CoCit and PubMed [43] are two public bibliographic datasets. Nodes represent the published paper, and edges represent the citation relationship between them. Labels indicate the research categories that each paper belongs to. We conduct a node classification experiment on these two directed networks to simulate the user profiling experiments in social networks.

5.1.2 Baseline Methods. We compare our proposed method with several state-of-the-art directed network embedding methods and user recommendation methods to evaluate our proposed DNE. It is worth noting that we do not compare with the social network based user-item recommendation method, as we focus on evaluating the performance of learned user/node embedding in the directed graph:

- *DeepWalk* [38] and *Node2Vec* [15] are two popular random walk based network embedding methods that can be used for modeling the relationship among users. However, these methods only preserve the proximity between nodes while ignore the direction of edges. We compare these methods to demonstrate the importance of considering the direction of edges.
- *APP* [64] and *NERD* [25] are two random walk based methods designed for directed networks. In APP, the random walk follows the direction of edges to capture the direction of edges. However, such a strategy cannot deal with the dangling nodes and only preserve the ill-defined asymmetric proximity. In NERD, the random walk alternates the direction between steps. This strategy can somehow deal with dangling nodes, but the transitivity of direction is ignored. We compare these two random walk based methods to show the advantage of the strategy used in DNE.
- *LINE* [47], *HOPE* [37], and *GraRep* [2] are matrix factorization based graph embedding methods. These methods first generate the proximity matrix in different ways, then utilize matrix factorization to get the low-dimensional representation. More specifically, LINE combines the first- and second-order proximity, and HOPE utilize Katz distance [23] as the proximity metric. GraRep employs the PPMI matrix between nodes as the proximity matrix and uses SVD to learn node embeddings.
- *ATP* [45] is a three-step graph embedding framework that includes removing cycles in the network, inferring the incomplete hierarchy on the reduced network, embedding learning with SVD. Previous work [39] has proved that the skip-gram based method can be treated as a variant of the matrix factorization method. We compare with the preceding matrix factorization based methods to show the advantage of capturing the asymmetric proximity.

Table 3. Parameter Setting of Baseline Methods

Method	Parameter Setting

Node2Vec	walk_length=10,number_of_walks=10,window_size=4
	p=0.25,q=2

DeepWalk	walk_length=80,number_of_walks=10,window_size=4
LINE	negative-ratio=5,order=first+second
GraRep	K-step=4
Hope	Similarity=Katz

APP	walk_length=80,number_of_walks=10,window_size=4
	Negative=5, jump factor=0.15,alpha=0.0025

NERD	walk_length=80,number_of_walks=10,
	Negative=5, rho=0.025,joint=1

ATP	Rank=64, strategy=linear
Gravity	epsilon=0.01(cora,citeseer2)/10(pubmed)
DNE	num_walks=10,walk_length=10,window_size=10

- *GraphSAGE* [16] and *GAT* [48] are two popular graph neural network methods that are widely used for graph embedding. These methods learn node embedding by aggregating information from neighbored nodes. In directed networks, information can be aggregated from in-neighbors.
- *Gravity* [42] is another directed network embedding method inspired by Newton's theory of universal gravitation. It learns an additional parameter of mass for each node, and directed edges are formed from both mass and distance. However, during the aggregation of these methods, asymmetric proximity is missed. We compare with the preceding graph neural networks to demonstrate the effectiveness of our proposed method.
- *GREED* [34] and *ShortWalk* [63] are two random walk based directed network embedding methods. Although they have tried to capture the asymmetric proximity between nodes in the network, they fail to consider the dangling nodes, which results in incomplete proximity preserved in the embedding.

5.1.3 Parameter Setting of Baseline Methods. Among baseline methods, Node2Vec, DeepWalk, APP, and NERD are random walk based methods. To make a fair comparison, we set the random walk parameters in these methods the same as our proposed DNE. More specifically, we set the length of random walk as $l = 10$, window size as $k = 4$, and the number of walkers per node as $r = 10$. For the Node2Vec method, the probability of Breadth-First Sampling (BFS) is set as 0.25, and the probability of Depth-First Sampling (DFS) is set as 0.5. We use the inner product of the embedded vectors to estimate the proximity between nodes. The APP, ATP, NERD, and HOPE methods preserve the asymmetric proximity by learning the two independent source and target embeddings. For tasks like node classification, we test the performance with both embeddings and report the best results. LINE learns two embeddings for each node, namely context embedding and node embedding. We also test both of them and report the best result. We use the open source code from the authors and fine tune them with gradient search for all the baseline methods. We implement the proposed DNE with PyTorch and TensorFlow. The model parameters are randomly initialized with a Xavier initializer, and an Adam optimizer is employed for optimization. We set the learning rate to 0.0005 and the batch size to 512. The vector dimension of all the methods is 128. The detailed parameter setting of baseline methods is listed in Table 3. All the experiments are conducted on a Linux server with one NVIDIA Titan XP GPU and a 24-core Intel Xeon E5-2690 CPU. We have provided the PyTorch and TensorFlow implementation of DNE in GitHub.[4]

5.1.4 Detailed Evaluation Metric. In this section, we provide the details of the evaluation metric used in our experiments. For classification task, Micro-F1 and Macro-F1 are used, which can be defined as follows:

$$\text{Precision} = \frac{\sum_{A\in C} TP(A)}{\sum_{A\in C}(TP(A) + FP(A))}, \tag{19}$$

$$\text{Recall} = \frac{\sum_{A\in C} TP(A)}{\sum_{A\in C}(TP(A) + FN(A))}, \tag{20}$$

$$\text{Micro-F1} = \frac{2 * Precision * Recall}{Precision + Recall}, \tag{21}$$

$$\text{Macro-F1} = \frac{\sum_{A\in C} Micro - F1(A)}{|C|}. \tag{22}$$

In the formulas mentioned previously, TP(A), FP(A), and FN(A) represent the number of true positives, false positives, and false negatives in the instances that are predicted as A, respectively. Suppose that C is the overall label set. Micro-f1(A) is the Micro-f1 measure for label A.

5.2 Vanilla User Recommendation (RQ1)

In this section, we conduct experiments on real-world social network datasets concerning vanilla user recommendation tasks to evaluate the proposed DNE. As we discussed in Section 1, most of the existing methods only preserve the proximity among nodes while failing to preserve the direction of edges. However, our proposed method (DNE) preserves both the proximity and direction between nodes in a unified framework. To evaluate the performance of preserving the proximity between nodes, we first conduct vanilla user recommendations that only predict edges between nodes and ignore the direction of edges. We will test the performance of predicting edge direction in the next section.

5.2.1 Experiment Setup. Following the same experimental procedure in many existing works [66], we randomly hold out 30% of the existing links as positive instances in the test set and randomly sample the same amount of non-existing links as negative instances. The residual network is used to train the network embedding methods. We evaluate the user recommendation task in the edge labeled dataset after learning the node embedding for each node/user in the network. Specifically, we rank both positive and negative instances according to node/user embeddings' cosine similarity. To judge the ranking quality, we employ the AUC score [35] and **mean average precision (MAP)** score to evaluate the ranking list, and a higher value indicates better performance. The train/test split is conducted independently five times, and we report the mean of results as the final output.

5.2.2 Experimental Results and Analysis. Table 4 shows the vanilla user recommendation results in five real-world social network datasets. We use "NA" to denote the situation that cannot run on our hardware setup due to memory limitation or runtime over 1 week.

To summarize, we have the following observations from the experimental results:

(1) The basic observation is that our proposed DNE and two variants DNE-L achieve better performance than the existing methods in most network datasets, which demonstrates the effectiveness of capturing the asymmetric proximity in directed social networks.

[4] https://github.com/zhoushengisnoob/DNE.

Table 4. Vanilla User Recommendation on a Real-World Dataset with Respect to the AUC Score and MAP

Dataset	Wiki		Epinions		Slashdot		Twitter		LastFM	
Metric	AUC	MAP	AUC	MAP	AUC	MAP	AUC	MAP	AUC	MAP
Node2Vec	0.855	0.805	0.853	0.84	0.738	0.740	0.874	0.910	0.923	0.933
DeepWalk	0.69	0.638	0.585	0.584	0.390	0.4155	0.852	0.892	0.825	0.838
GraRep	0.905	0.893	NA	NA	NA	NA	NA	NA	NA	NA
LINE	0.913	0.917	0.857	0.894	0.764	0.7909	0.791	0.8375	0.898	0.923
HOPE	0.93	0.948	0.889	0.924	0.777	0.8524	0.801	0.8417	NA	NA
APP	0.919	0.907	0.898	0.928	0.868	0.8877	0.873	0.918	0.926	0.935
ATP	0.85	0.779	NA	NA	NA	NA	NA	NA	NA	NA
Gravity	0.955	0.927	NA	NA	NA	NA	NA	NA	NA	NA
NERD	0.517	0.565	0.818	0.872	0.832	0.8767	0.694	0.742	0.744	0.773
GraphSAGE	0.938	0.917	**0.930**	**0.942**	**0.886**	0.895	0.849	0.8875	0.948	0.950
GAT	0.839	0.785	0.786	0.776	0.631	0.569	0.821	0.862	0.909	0.914
GREED	0.793	0.725	0.633	0.543	0.720	0.712	0.650	0.666	0.826	0.828
ShortWalk	0.708	0.673	0.787	0.805	0.638	0.660	0.889	0.920	0.899	0.913
DNE-L	0.960	0.955	0.926	0.939	0.863	**0.899**	**0.899**	**0.928**	**0.951**	**0.956**
DNE-T	**0.968**	**0.963**	0.929	0.941	0.857	0.896	0.889	0.921	0.946	0.951
Impv%	†0.8%	†0.8%	—	—	—	†0.4%	†2.9%	†1.0%	†0.3%	†0.6%

Negative links contains the reverse direction of positive edges. NA denotes the methods that cannot run on our hardware setup. † indicates that the result of a paired difference test is significant at $p < 0.05$.

(2) Among the baseline methods, some matrix factorization based methods cannot run on our experimental settings. This is explainable since the matrix factorization is both time consuming and memory consuming, which cannot scale to large-scale datasets.

(3) Another interesting observation is that graph neural network based methods achieve better performance than random walk based baseline methods in preserving proximity. This is explainable since the neighbored nodes play different roles in embedding learning for graph neural network based methods, whereas the random walk based methods fail to do so.

5.3 Direction- Aware User Recommendation (RQ1)

We further evaluate the direction-aware user recommendation task to simulate the real-world scenario where the recommendation direction should be considered. Given the social networks with directed edges, recommending users to "follow/trust" is one of the critical applications in the real world. The vanilla user recommendation task only predicts the existence of edges, which cannot guarantee that the direction is also well predicted. For example, there exists a directed edge from v_i to v_j but no edge from v_j to v_i, and methods that predict edges from both directions can muddle through the metric, as the positive link E_{ij} is already corrected predicted. However, the reverse direction edge E_{ji} may not be sampled as a negative link to penalize the reverse direction. Following the experimental setting of existing methods, we also test the performance of the *direction-aware user recommendation* task. A total of 30% of links are randomly sampled from the original network as the positive links. The negative links contain two parts: randomly sampled from non-existence edges in the original network and the non-existing reverse edges (if they exist) of positive edges. Following the evaluation strategy of existing work [59], we use the AUC score[5] and MAP to evaluate the performance. The train/test split is conducted independently five times, and we report the mean of results as the final output. Table 5 illustrates the performance of direction-aware user recommendation and classic user recommendation on six real-world datasets.

[5] https://en.wikipedia.org/wiki/Receiver_operating_characteristic.

Table 5. Direction-Aware Recommendation on a Real-World Dataset with Respect to the AUC Score and MAP

Dataset	Wiki		Epinions		Slashdot		Twitter		LastFM	
Metric	AUC	MAP	AUC	MAP	AUC	MAP	AUC	MAP	AUC	MAP
Node2Vec	0.692	0.470	0.759	0.646	0.714	0.690	0.807	0.749	0.712	0.481
DeepWalk	0.603	0.403	0.574	0.478	0.400	0.401	0.788	0.74	0.662	0.452
GraRep	0.727	0.522	NA	NA	NA	NA	NA	NA	NA	NA
LINE	0.722	0.512	0.761	0.672	0.744	0.740	0.739	0.690	0.698	0.477
HOPE	0.746	0.546	0.772	0.662	0.7546	0.789	0.807	0.740	NA	NA
GraphSAGE	0.724	0.4763	0.806	0.687	**0.854**	0.829	0.789	0.739	0.696	0.444
GAT	0.677	0.4610	0.713	0.591	0.703	0.638	0.783	0.737	0.713	0.471
APP	0.698	0.449	0.803	0.711	0.833	0.813	0.807	0.762	0.614	0.369
ATP	0.863	0.698	NA	NA	NA	NA	NA	NA	NA	NA
Gravity	0.812	0.603	NA	NA	NA	NA	NA	NA	NA	NA
NERD	0.430	0.304	0.709	0.597	0.795	0.796	0.640	0.592	0.525	0.322
GREED	0.675	0.474	0.663	0.479	0.683	0.640	0.676	0.612	0.771	0.742
ShortWalk	0.628	0.430	0.711	0.615	0.624	0.617	0.814	0.753	0.699	0.475
DNE-L	0.849	0.678	0.816	0.694	0.837	0.837	**0.842**	0.812	0.864	0.732
DNE-T	**0.887**	**0.751**	**0.826**	**0.714**	0.832	**0.837**	0.839	**0.816**	**0.872**	**0.732**
Impv%	†2.7%	†7.5%	†2.4%	†0.4%	—	†0.9%	†4.3%	†8.9%	†22.4%	†52.1%

Negative links contains the reverse direction of positive edges. NA denotes that the methods cannot run on our hardware setup. † indicates that the result of a paired difference test is significant at $p < 0.05$.

To summarize, we have the following observations:

(1) Among all the evaluated methods, our proposed DNE-L and DNE-T achieve the best performance on all datasets with respect to two evaluation metrics, and we observe a significant improvement over existing methods.
(2) Comparing the same method in Tables 4 and 5, we can observe that all methods have decreased performance on direction-aware user recommendation. Further, methods that learn single embedding perform worse than those capturing the asymmetric proximity. This demonstrates the necessity of considering the direction of edges and asymmetric proximity.
(3) Comparing DNE-L with DNE-T, we can observe improvement in both of the two tasks. Interestingly, in direction-aware user recommendations, the improvement is more significant than in classic user recommendations. This further indicates the importance of considering the impact of direction in predicting the directed links between nodes.

5.4 User Profiling (RQ1)

User profiling is another important task of user modeling, especially in directed social networks. The target of user profiling is to find the group to which users belong, which is the same as the classic node classification task. Following the same experimental procedure in other works [2, 15], we randomly sample a portion of labeled nodes (30%) for training and use the rest of the nodes for testing. The learned embeddings are fed into the same SVM classifier, and we use Micro-F1 and Macro-F1 scores to evaluate the performance. For methods that learn two independent embeddings for each node, we concatenate the embedding for evaluation.

Figure 6 illustrates the results on real-world datasets. To summarize, we have the following observations:

(1) The basic observation is similar to the user recommendation task that our proposed DNE achieves better performance than existing methods with respect to two evaluation metrics.
(2) We found that the undirected network embedding methods gain considerable performance in classification tasks compared with directed network embedding methods.

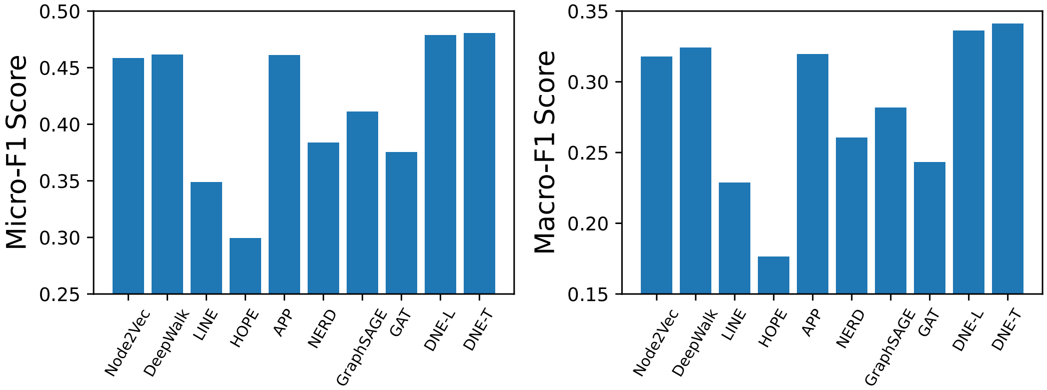
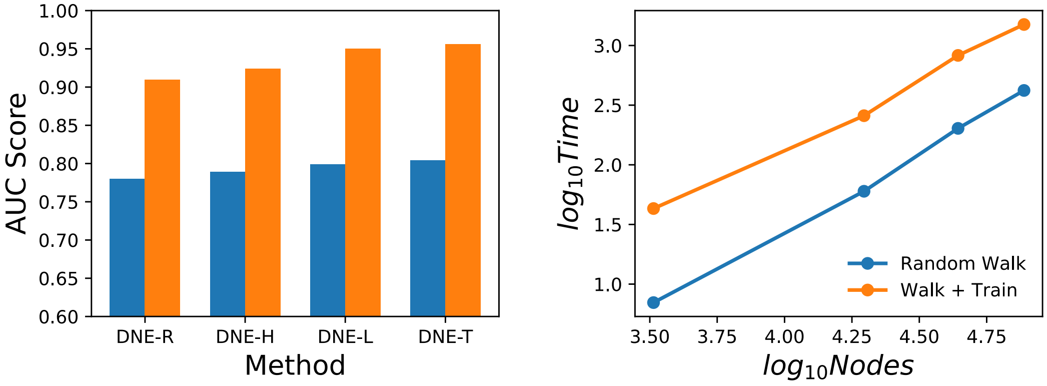


Fig. 6. User profiling experiment on the CoCit dataset with respect to micro-F1 score and macro-F1 score.



(a) Ablation Study (b) Scalability

Fig. 7. Ablation study and scalability analysis of the proposed method. DR-DNE denotes DNE with a directed random walk. NH-DNE denotes DNE without direction.

(3) DNE-T does not gain too much improvement over DNE-L. This is explainable since the classification task is not very sensitive to the direction of edges.

5.5 Ablation Study (RQ2)

We further design a detailed ablation study to answer question RQ2. In other words, we remove different components at a time and compare DNE with its special cases: DNE-R and DNE-H. Here, DNE-R denotes that we force the random walk to follow the direction of edges, and we try to prove the importance of visiting nodes from all directions. DNE-H denotes that the score of all direct context nodes are the same, and we try to prove the importance of the direction. DNE-T and DNE-L are two variants of DNE. Figure 7(a) illustrates the results of the ablation study. We observe that the method with integrated asymmetric proximity outperforms DNE-R and DNE-H, proving the benefits of capturing the asymmetric proximity.

5.6 Scalability (RQ3)

According to our theory, in Section 4.4, DNE scales linearly with the number of nodes. To verify the scalability of DNE, we report the time of node representation learning on a different scale of

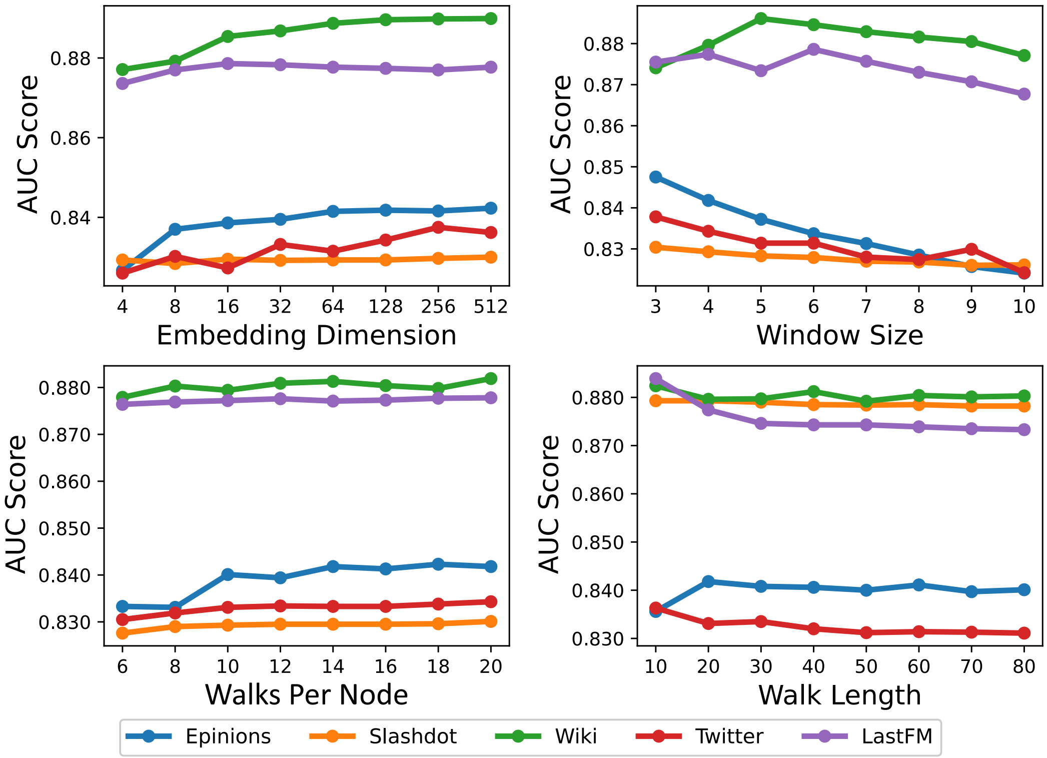


Fig. 8. Hyperparameter tuning of DNE.

real-world networks. Figure 7(b) illustrates the results on the dataset. We empirically observe that DNE scales linearly with an increase in the number of nodes.

5.7 Parameter Sensitivity (RQ4)

In this section, we examine how different choices of parameters affect the performance of DNE. For network embedding methods, the fundamental parameter to tune is the dimension of learned embedding. We examine three hyperparameters for our random walk strategy: windows size, number of walks per node, and walk length. Figure 8 illustrates the parameter tuning results of the AUC score of the direction-aware user recommendation task on six datasets. We observe that the performance has minor changes on different windows size and walks the directed random walk length, which shows that DNE is not very sensitive to these parameters. We observe that performance tends to saturate once the representations' dimension reaches around 64, which shows that DNE is not very sensitive to the dimension of source/target embedding.

6 CONCLUSION

In this article, we explored utilization of the directed network structure information for user recommendation. Specifically, we transformed the user recommendation problem into the link prediction task and addressed it with network embedding techniques. We proposed a novel random walk strategy (InfoWalk) to efficiently capture the hierarchy and proximity between nodes in a directed network. Two directed network embedding methods (DNE-L and DNE-T) were proposed for embedding learning. Experiments on real-world social and citation networks showed that our proposed method is superior to the existing embedding methods in tasks including link prediction and node classification.

REFERENCES

[1] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph convolutional matrix completion. arXiv:1706.02263.

[2] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*.

[3] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*.

[4] 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 *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, New York, NY, 377–386.

[5] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C. Aggarwal, and Thomas S. Huang. 2015. Heterogeneous network embedding via deep architectures. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[6] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and debias in recommender system: A survey and future directions. arXiv:2010.03240.

[7] Jiawei Chen, Yan Feng, Martin Ester, Sheng Zhou, Chun Chen, and Can Wang. 2018. Modeling users' exposure with social knowledge influence and consumption influence for recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 953–962.

[8] Jiawei Chen, Chengquan Jiang, Can Wang, Sheng Zhou, Yan Feng, Chun Chen, Martin Ester, and Xiangnan He. 2020. CoSam: An efficient collaborative adaptive sampler for recommendation. arXiv:2011.07739.

[9] Jiawei Chen, Can Wang, Qihao Shi, Yan Feng, and Chun Chen. 2019. Social recommendation based on users' attention and preference. *Neurocomputing* 341 (2019), 1–9.

[10] Jiawei Chen, Can Wang, Sheng Zhou, Qihao Shi, Yan Feng, and Chun Chen. 2019. SamWalker: Social recommendation with informative sampling strategy. In *Proceedings of the World Wide Web Conference*. 228–239.

[11] Daizong Ding, Mi Zhang, Shao-Yuan Li, Jie Tang, Xiaotie Chen, and Zhi-Hua Zhou. 2017. BayDNN: Friend recommendation with Bayesian personalized ranking deep neural network. In *Proceedings of the 2017 ACM Conference on Information and Knowledge Management*. 1479–1488.

[12] Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep modeling of social relations for recommendation. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*. 8075–8076.

[13] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *Proceedings of the World Wide Web Conference*. 417–426.

[14] Jun Feng, Minlie Huang, Yang Yang, and Xiaoyan Zhu. 2016. GAKE: Graph aware knowledge embedding. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 641–651.

[15] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*.

[17] John Hannon, Mike Bennett, and Barry Smyth. 2010. Recommending Twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the 4th ACM Conference on Recommender Systems*. 199–206.

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[19] Shangrong Huang, Jian Zhang, Lei Wang, and Xian-Sheng Hua. 2015. Social friend recommendation based on multiple network correlation. *IEEE Transactions on Multimedia* 18, 2 (2015), 287–299.

[20] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. ACM, New York, NY, 731–739.

[21] Matthew O. Jackson and Alison Watts. 2002. The evolution of social and economic networks. *Journal of Economic Theory* 106, 2 (2002), 265–295.

[22] Mohsen Jamali and Martin Ester. 2009. TrustWalker: A random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 397–406.

[23] Leo Katz. 1953. A new status index derived from sociometric analysis. *Psychometrika* 18, 1 (1953), 39–43.

[24] Mohammad Mehdi Keikha, Maseud Rahgozar, and Masoud Asadpour. 2018. Community aware random walk for network embedding. *Knowledge-Based Systems* 148 (2018), 47–54.

[25] Megha Khosla, Jurek Leonhardt, Wolfgang Nejdl, and Avishek Anand. 2018. Node representation learning for directed graphs. arXiv:1810.09176.

[26] Junghwan Kim, Haekyu Park, Ji-Eun Lee, and U. Kang. 2018. Side: Representation learning in signed directed networks. In *Proceedings of the 2018 World Wide Web Conference*.

[27] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv:1609.02907.

[28] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Predicting positive and negative links in online social networks. In *Proceedings of the 19th International Conference on World Wide Web*. 641–650.

[29] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. 2009. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics* 6, 1 (2009), 29–123.

[30] Nan Li and Guanling Chen. 2009. Multi-layered friendship modeling for location-based mobile social networks. In *Proceedings of the 2009 6th Annual International Mobile and Ubiquitous Systems: Networking and Services, MobiQuitous*. IEEE, Los Alamitos, CA, 1–10.

[31] Shuchuan Lo and Chingching Lin. 2006. WMR—A graph-based algorithm for friend recommendation. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings) (WI'06)*. IEEE, Los Alamitos, CA, 121–128.

[32] Tiancheng Lou, Jie Tang, John Hopcroft, Zhanpeng Fang, and Xiaowen Ding. 2013. Learning to predict reciprocity and triadic closure in social networks. *ACM Transactions on Knowledge Discovery from Data* 7, 2 (2013), 1–25.

[33] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. 2008. SoRec: Social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. 931–940.

[34] Ramanujam Madhavan and Mohit Wadhwa. 2020. Directed graph representation through vector cross product. arXiv:2010.10737.

[35] Donna Katzman McClish. 1989. Analyzing a portion of the ROC curve. *Medical Decision Making* 9, 3 (1989), 190–195.

[36] Federico Monti, Michael Bronstein, and Xavier Bresson. 2017. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*. 3697–3707.

[37] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[38] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[39] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*.

[40] Dimitrios Rafailidis and Fabio Crestani. 2018. Friend recommendation in location-based social networks via deep pair-wise learning. In *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'18)*. IEEE, Los Alamitos, CA, 421–428.

[41] Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. 2003. Trust management for the Semantic Web. In *Proceedings of the International Semantic Web Conference*.

[42] Guillaume Salha, Stratis Limnios, Romain Hennequin, Viet-Anh Tran, and Michalis Vazirgiannis. 2019. Gravity-inspired graph autoencoders for directed link prediction. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 589–598.

[43] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Magazine* 29, 3 (2008), 93.

[44] Nitai B. Silva, Ren Tsang, George D. C. Cavalcanti, and Jyh Tsang. 2010. A graph-based friend recommendation system using genetic algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation*. IEEE, Los Alamitos, CA, 1–7.

[45] Jiankai Sun, Bortik Bandyopadhyay, Armin Bashizade, Jiongqian Liang, P. Sadayappan, and Srinivasan Parthasarathy. 2018. ATP: Directed graph embedding with asymmetric transitivity preservation. arXiv:1811.00839.

[46] Jiliang Tang, Xia Hu, and Huan Liu. 2013. Social recommendation: A review. *Social Network Analysis and Mining* 3, 4 (2013), 1113–1133.

[47] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*.

[48] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. arXiv:1710.10903.

[49] Shengxian Wan, Yanyan Lan, Jiafeng Guo, Chaosheng Fan, and Xueqi Cheng. 2013. Informational friend recommendation in social media. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1045–1048.

[50] Can Wang, Jiawei Chen, Sheng Zhou, Qihao Shi, Yan Feng, and Chun Chen. 2020. SamWalker++: Recommendation with informative sampling strategy. arXiv:2011.07734.

[51] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

[52] Suhang Wang, Charu Aggarwal, Jiliang Tang, and Huan Liu. 2017. Attributed signed network embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management.* 137–146.

[53] Suhang Wang, Jiliang Tang, Charu Aggarwal, Yi Chang, and Huan Liu. 2017. Signed network embedding in social media. In *Proceedings of the 2017 SIAM International Conference on Data Mining.*

[54] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence.*

[55] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval.* 185–194.

[56] Xin Wang, Wei Lu, Martin Ester, Can Wang, and Chun Chen. 2016. Social recommendation with strong and weak ties. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management.* 5–14.

[57] Xin Wang, Wenwu Zhu, and Chenghao Liu. 2019. Social recommendation with optimal limited attention. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 1518–1527.

[58] Bo Yang, Yu Lei, Jiming Liu, and Wenjie Li. 2016. Social collaborative filtering by trust. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, 8 (2016), 1633–1647.

[59] Yuan Yin and Zhewei Wei. 2019. Scalable graph embeddings via sparse transpose proximities. arXiv:1905.07245.

[60] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 974–983.

[61] Shuhan Yuan, Xintao Wu, and Yang Xiang. 2017. SNE: Signed network embedding. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining.* 183–195.

[62] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S. Yu. 2015. COSNET: Connecting heterogeneous social networks with local and global consistency. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 1485–1494.

[63] Fen Zhao, Yi Zhang, and Jianguo Lu. 2021. ShortWalk: An approach to network embedding on directed graphs. *Social Network Analysis and Mining* 11, 1 (2021), 1–12.

[64] Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, and Jun Gao. 2017. Scalable graph embedding for asymmetric proximity. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence.*

[65] Sheng Zhou, Xin Wang, Jiajun Bu, Martin Ester, Pinggang Yu, Jiawei Chen, Qihao Shi, and Can Wang. 2020. DGE: Deep generative network embedding based on commonality and individuality. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6949–6956.

[66] Sheng Zhou, Hongxia Yang, Xin Wang, Jiajun Bu, Martin Ester, Pinggang Yu, Jianwei Zhang, and Can Wang. 2018. PRRE: Personalized relation ranking embedding for attributed networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.*

Received November 2020; revised May 2021; accepted May 2021