Asymmetric Transitivity Preserving Graph Embedding

Mingdong Ou
Tsinghua University
oumingdong@gmail.com

Peng Cui
Tsinghua University
cuip@tsinghua.edu.cn

Jian Pei
Simon Fraser University
jpei@cs.sfu.ca

Ziwei Zhang
Tsinghua University
zhangziwei1993@126.com

Wenwu Zhu
Tsinghua University
wwzhu@tsinghua.edu.cn

ABSTRACT

Graph embedding algorithms embed a graph into a vector space where the structure and the inherent properties of the graph are preserved. The existing graph embedding methods cannot preserve the asymmetric transitivity well, which is a critical property of directed graphs. Asymmetric transitivity depicts the correlation among directed edges, that is, if there is a directed path from u to v, then there is likely a directed edge from u to v. Asymmetric transitivity can help in capturing structures of graphs and recovering from partially observed graphs. To tackle this challenge, we propose the idea of preserving asymmetric transitivity by approximating high-order proximity which are based on asymmetric transitivity. In particular, we develop a novel graph embedding algorithm, *High-Order Proximity preserved Embedding* (*HOPE* for short), which is scalable to preserve high-order proximities of large scale graphs and capable of capturing the asymmetric transitivity. More specifically, we first derive a general formulation that cover multiple popular high-order proximity measurements, then propose a scalable embedding algorithm to approximate the high-order proximity measurements based on their general formulation. Moreover, we provide a theoretical upper bound on the RMSE (Root Mean Squared Error) of the approximation. Our empirical experiments on a synthetic dataset and three real-world datasets demonstrate that *HOPE* can approximate the high-order proximities significantly better than the state-of-art algorithms and outperform the state-of-art algorithms in tasks of reconstruction, link prediction and vertex recommendation.

Keywords

asymmetric transitivity, directed graph, high-order proximity, graph embedding

1. INTRODUCTION

Nowadays, more and more applications are based on larger and larger networks and graphs. It is well recognized that graph data is sophisticated and challenging. To process graph data effectively, the first critical challenge is graph data representation, that is, how to represent graphs properly so that advanced analytic tasks, such as pattern discovery, analysis, and prediction, can be conducted efficiently in both time and space. However, graph representation in general remains an open problem.

Recently, the graph embedding paradigm [35] is proposed to represent vertices of a graph in a low-dimensional vector space while the structures (i.e. edges and other high-order structures) of the graph can be reconstructed in the vector space. With proper graph embedding, we can easily apply classic vector-based machine learning techniques to process graph data. Meanwhile, it effectively facilitates the parallelization of graph analysis by decoupling highly correlated vertices into independent vectors. However, most of the existing graph embedding methods target on undirected

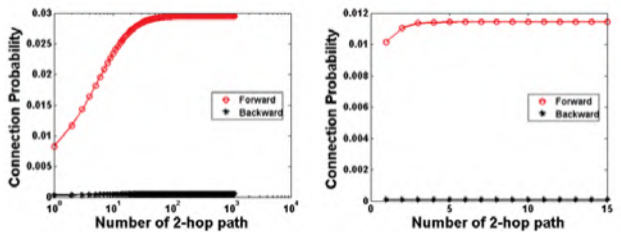

Figure 1: Asymmetric Transitivity in Real Data. Given a pair of vertexes (u, v), the horizontal axis is the Number of 2-hop paths from u to v. For the red (or black) curve with circle (or star) marker, the vertical axis represents the cumulative forward (or backward) transiting probability, that is the connection probability from u (or v) to v (or u), when the number of 2-hop paths is less than corresponding horizontal axis value. The left figure is the statistics from the social network of Tencent Weibo, and the right figure is the statistics from the social network of Twitter. As the two red curves increases monotonically, we can claim that the more paths from u to v there are, the more probable it is that there exists an edge from u to v, which reflects the transitivity assumption. The forward connection probability is much larger than the backward connection probability, which reflects the asymmetry assumption. In summary, asymmetric transitivity can be widely observed in real datasets.

KDD '16, August 13-17, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN 978-1-4503-4232-2/16/08...$15.00
DOI: http://dx.doi.org/10.1145/2939672.2939751

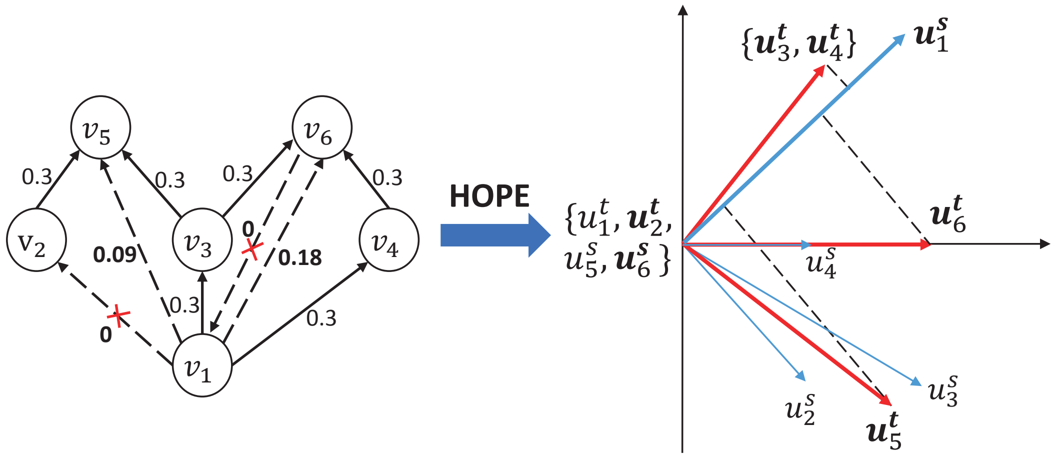

Figure 2: Framework of Asymmetric Transitivity Preserving Graph Embedding. The left is a input directed graph, and the right is the embedding vector space of the left graph, which is learned by our algorithm. In the left directed graph, the solid arrows represent observed directed edges and the numbers along with the solid arrows are the edge weights. The numbers along with the dashed arrows is the Katz proximity, which is highly correlated with asymmetric transitivity. For example, according to asymmetric transitivity, the two paths, $v_1 \to v_3 \to v_6$ and $v_1 \to v_4 \to v_6$, suggest that there may exist $v_1 \to v_6$. Then, the Katz proximity from v_1 to v_6 is relatively large, i.e. 0.18. On the other hand, because $v_6 \to v_1$ is in the opposite direction, $v_1 \to v_3 \to v_6$ and $v_1 \to v_4 \to v_6$ do not suggest $v_6 \to v_1$. Then, the Katz proximity from v_6 to v_1 is small, i.e. 0. In the embedding space, the arrows represent the embedding vectors of vertices, where row vectors $\mathbf{u}_i^s$ and $\mathbf{u}_i^t$ represent the source vector and target vector of v_i respectively. We use the inner product between $\mathbf{u}_i^s$ and $\mathbf{u}_j^t$ (i.e. $\mathbf{u}_i^s \cdot {\mathbf{u}_j^t}^\top$) as the approximated proximity from v_i to v_j. Note that $\mathbf{u}_1^t$, $\mathbf{u}_2^t$, $\mathbf{u}_5^s$ and $\mathbf{u}_6^s$ are all zero vectors. We can find that the approximated proximity perfectly preserve the Katz proximity. For example, with respect to source vector $\mathbf{u}_1^s$, the inner product with target vector $\mathbf{u}_6^t$ is larger than with $\mathbf{u}_5^t$, which preserves the rank order of corresponding Katz proximities.

graphs. Directed graphs, a natural and popularly used representation of data in many applications, such as social networks and webpage networks, are largely untouched.

Can we straightforwardly apply undirected graph embedding methods on directed graphs? The answer is no due to a fundamentally different characteristic of directed graphs: asymmetric transitivity.

Transitivity is a common characteristic of undirected and directed graphs [29] (see Figure 1). In undirected graphs, if there is an edge between vertices u and w, and one between w and v, then it is likely that u and v are connected by an edge. Transitivity plays a key role in graph inference and analysis tasks, such as calculating similarities between nodes [16, 14] and measuring the importance of nodes. Transitivity is symmetric in undirected graphs. However, transitivity is asymmetric in directed graphs. If there is a directed link from u to w and a directed link from w to v, there is likely a directed link from u to v, but not from v to u. More generally, If there is a directed path from u to v, there is likely a directed link from u to v, but not from v to u.

Preserving the symmetric transitivity in undirected graphs in an embedding space is natural and straightforward, as most of the distance metrics defined in the vector space possess the property of symmetric transitivity. However, how to preserve the asymmetric transitivity of directed graphs in a vector space is much more challenging.

Recently, a paucity of studies [37, 20, 3, 2, 24] focus on directed graph embedding. In order to reflect the asymmetric property in vector space, these methods design asymmetric metrics on the embedding vectors. Unfortunately, those asymmetric metrics are not fully transitive, which severely limit the capability of the learned embedding space in reflecting the structure of graphs as well as supporting graph inference and analysis.

In this paper, we tackle the challenging problem of asymmetric transitivity preserving graph embedding. Our major idea is that we learn two embedding vectors, source vector and target vector, for each node to capture asymmetric edges, as illustrated in Figure 2. Then, for a directed link from v_1 to v_2 without the reverse link from v_2 to v_1, we can assign similar values to v_1's source vector and v_2's target vector, and assign very different values to v_1's target vector and v_2's source vector. In this way, the feasibility of asymmetry transitivity is preserved. The key challenge is how well the asymmetric transitivity of directed graphs can be preserved in the embedding learning.

From the learning perspective, a good way is to let the learned embedding vectors directly approximate a target measure reflecting asymmetric transitivity of graphs. From the graph embedding pserspective, the property of asymmetric transitivity leads to the assumption that the more and the shorter paths from v_i to v_j, the more similar should be v_i's source vector and v_j's target vector. This assumption coincides with high-order proximities of nodes in graphs, such as Katz [16] and rooted pagerank [30]. What is more, these high-order proximities are defined to be asymmetric in directed graphs. Thus, in its place, we propose to use high-order proximities of nodes as the target measure, resulting in a novel directed graph embedding algorithm, High Order Proximity preserved Embedding (HOPE).

In this model, we theoretically derive a general form covering multiple high-order proximities. Interestingly, the general form is consistent with the formulation of generalized SVD. Based on this, we propose a scalable embedding algorithm for large-scale graphs by avoiding the time-consuming computation of high-order proximities. Moreover, we derive a theoretical upper bound on the approximation error of HOPE, which is used to estimate the embedding quality in theory and determine embedding dimensions automatically.

To verify the advantages of our algorithm, we conduct experiments on both synthetic data and 3 real datasets. The experiments show that our method consistently and significantly outperforms the state-of-the-art baselines in several tasks, including high-order proximity approximation, graph reconstruction, link prediction and vertex recommendation.

The main contributions of this paper are as follows:

- We propose a high-order proximity preserved embedding (HOPE) method to solve the challenging problem of asymmetric transitivity in directed graph embedding.
- We derive a general form covering multiple commonly used high-order proximities, enabling the scalable solution of HOPE with generalized SVD.
- We provide an upper bound on the approximation error of HOPE.
- Extensive experiments are conducted to verify the usefulness and generality of the learned embedding in various applications.

The rest of the paper is organized as follows. In Section 2, we review the related work. We develop our method in Section 3 and report the experimental results in Section 4. We conclude the paper in Section 5.

2. RELATED WORK

2.1 Graph Embedding

Graph embedding technology has been widely studied in the fields of dimensionality reduction [15, 27, 33, 1, 8], natural language processing [18], network analysis [11] and so on.

For dimensionality reduction, adjacency matrices of graphs are constructed from the feature similarity (distance) between samples [35]. And the graph embedding algorithms aim to preserve the feature similarity in the embedded latent space. For example, Laplacian Eigenmaps (LE) [1] aims to learn the low-dimensional representation to expand the manifold where data lie. Locality Preserving Projections (LPP) [8] is a linearization variant of LE which learns a linear projection from feature space to embedding space. Besides, there are many other graph embedding algorithms for dimensionality reduction, including non-linear [33, 27], linear [15, 6], kernlized [28] and tensorized [36] algorithms. All of these algorithms are based on undirected graphs derived from symmetric similarities. Thus, they cannot preserve asymmetric transitivity.

In the field of natural language processing, the graph of words is often used to learn the representation of words [19, 18, 23]. Mikolov et. al. [19] propose to ultilize the context of words to learn representation, which has been proved equivalent to factorizing word-context matrix [18]. Pennington et. al. [23] exploit a word-word co-occurrance matrix.

In network analysis, Hoff et. al. [11] first propose to learn latent space representation of vertexes in graph and the probability of a relation depends on the distance between vertexes in the latent space, and they apply it to link prediction problem [10]. Handcock et. al. [7] propose to apply the latent space approach to clustering in graph. And Zhu et. al. [37] propose to address the classification problem in graph with graph embedding model. While early graph embedding works focus on modeling the observed first-order relationship (i.e. edges in graph) between vertexes, some recent works try to model the directed higher order relationships between vertexes in sparse graphs [24, 2]. GraRep [2] is related to our work. But, it cannot fully capture transitivity. Moreover, it is not scalable for large-scale graph.

2.2 Directed Graph

Theoretically, any type (undirected and directed) of graph can be represented as directed graph. So, modeling directed graph is a critical problem for graph analysis. Holland et. al. [12] propose the p_1 distribution model to capture the structural properties in directed graph, including the attractiveness and expansiveness of vertexes and the reciprocation of edges. Besides these properties, Wang et. al. [34] take the group information of vertexes into consideration. Recently, some works adopt graph embedding [4, 26, 25, 21] to model directed graphs. Chen et. al. [4] learn the embedding vectors in Euclidean space with locality property preserved. Perrault-Joncas et. al. [26, 25] and Mousazadeh et. al. [21] learn the embedding vectors based on Laplacian type operators and preserve the asymmetry property of edges in a vector field. However, all of these methods cannot preserve asymmetry property in embedding vector space.

3. HIGH-ORDER PROXIMITY PRESERVED EMBEDDING

In this section, we will derive how to preserve high-order proximities in the embedding space. Before introducing the detailed derivation, we will clarify the symbols and definition that will be used.

3.1 Notations

We define a directed graph as $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$. $\mathbf{V}$ is the vertex set. Let $\mathbf{V} = \{v_1, \cdots, v_i, \cdots, v_N\}$ where N is the number of vertexes. $\mathbf{E}$ is the directed edge set. $\mathbf{e}_{ij} = (\mathbf{v}_i, \mathbf{v}_j) \in \mathbf{E}$ represents a directed edge from $\mathbf{v}_i$ to $\mathbf{v}_j$. The adjacency matrix is denoted as $\mathbf{A}$. We define a high-order proximity matrix as $\mathbf{S}$, where S_{ij} is the proximity between $\mathbf{v}_i$ and $\mathbf{v}_j$. And $\mathbf{U} = [\mathbf{U}^s, \mathbf{U}^t]$ is the embedding matrix, where the i-th row, $\mathbf{u}_i$, is the embedding vector of v_i. $\mathbf{U}^s, \mathbf{U}^t \in \mathcal{R}^{N \times K}$ are the source embedding vectors and target embedding vectors respectively, where K is the embedding dimensions. For any matrix $\mathbf{B}$, the lowercase symbol $\mathbf{b}_i$ represents the i-th row of $\mathbf{B}$, and B_{ij} represents the element at i-th row and j-th column.

Table 1: General Formulation for High-order Proximity Measurements

Proximity Measurement	$\mathbf{M}_g$	$\mathbf{M}_l$
Katz	$\mathbf{I} - \beta \cdot \mathbf{A}$	$\beta \cdot \mathbf{A}$
Personalized Pagerank	$\mathbf{I} - \alpha \mathbf{P}$	$(1-\alpha) \cdot \mathbf{I}$
Common neighbors	$\mathbf{I}$	$\mathbf{A}^2$
Adamic-Adar	$\mathbf{I}$	$\mathbf{A} \cdot \mathbf{D} \cdot \mathbf{A}$

3.2 Problem Definition

In this paper, we focus on the directed graph embedding problem. It aims to represent the vertexes in the numerical vector space, where asymmetric transitivity is preserved. As high-order proximities are derived from asymmetric transitivity, we propose to preserve the asymmetric transitivity by approximating high-order proximity. Formally, we adopt the L2-norm below as the loss function which need to be minimized:

$$\min \|\mathbf{S} - \mathbf{U}^s \cdot \mathbf{U}^{t^\top}\|_F^2 \tag{1}$$

.

As Figure 2 shows, the embedding vectors can well preserve the asymmetric transitivity.

3.3 High order proximities

Many high-order proximity measurements in graph can reflect the asymmetric transitivity. Moreover, we found that many of them share a general formulation which will facilitate the approximation of these proximities, that is:

$$\mathbf{S} = \mathbf{M}_g^{-1} \cdot \mathbf{M}_l \tag{2}$$

, where $\mathbf{M}_g$ and $\mathbf{M}_l$ are both polynomial of matrices. Below, we will introduce some popular proximity measurements and transform them into this formulation.

Katz Index [16]. This index is an ensemble of all paths, which is a weighted summation over the path set between two vertexes. The weight of a path is a exponential function of its length. Actually, the formula of Katz index can also be written as a recurrent formula:

$$\mathbf{S}^{Katz} = \sum_{l=1}^{\infty} \beta \cdot \mathbf{A}^l = \beta \cdot \mathbf{A} \cdot \mathbf{S}^{Katz} + \beta \cdot \mathbf{A} \tag{3}$$

, where β is a decay parameter. It determines how fast the weight of a path decay when the length of path grows. β should be properly set to preserve the series converging. Actually, β must be smaller than the spectral radius of adjacency matrix.

Then, we can get that:

$$\mathbf{S}^{Katz} = (\mathbf{I} - \beta \cdot \mathbf{A})^{-1} \cdot \beta \cdot \mathbf{A} \tag{4}$$

, where $\mathbf{I}$ is a identity matrix.

For Katz index,

$$\mathbf{M}_g = \mathbf{I} - \beta \cdot \mathbf{A} \tag{5}$$

$$\mathbf{M}_l = \beta \cdot \mathbf{A} \tag{6}$$

Rooted PageRank (RPR). $\mathbf{S}_{ij}^{RPR}$ is the probability that a random walk from node v_i will locate at v_j in the steady state. Consider one step of a random walk from v_i, the random walker randomly jumps to one of the neighbor of current node with probability α, and jumps back to v_i with probability α. The fomula is:

$$\mathbf{S}^{RPR} = \alpha \cdot \mathbf{S}_{ij}^{RPR} \cdot \mathbf{P} + (1-\alpha) \cdot \mathbf{I} \tag{7}$$

, where $\alpha \in [0, 1)$ is the probability to randomly walk to a neighbor, and $\mathbf{P}$ is the probability transition matrix satisfying that $\sum_{i=1}^{N} P_i j = 1$.

Then, we can get:

$$\mathbf{S}^{RPR} = (1-\alpha) \cdot (\mathbf{I} - \alpha \mathbf{P})^{-1} \tag{8}$$

So, for rooted PageRank,

$$\mathbf{M}_g = \mathbf{I} - \alpha \mathbf{P} \tag{9}$$

$$\mathbf{M}_l = (1-\alpha) \cdot \mathbf{I} \tag{10}$$

.

Common Neighbors (CN). $\mathbf{S}_{ij}^{CN}$ counts the number of vertexes connecting to both v_i and v_j. For directed graph, $\mathbf{S}_{ij}^{CN}$ is the number of vertexes which is the target of an edge from v_i and the source of an edge to v_j. Formally,

$$\mathbf{S}^{CN} = \mathbf{A}^2 \tag{11}$$

So, for common neighbors,

$$\mathbf{M}_g = \mathbf{I} \tag{12}$$

$$\mathbf{M}_l = \mathbf{A}^2 \tag{13}$$

Adamic-Adar (AA). Adamic-Adar is a variant of common neighbors. Unlike common neighbors, Adamic-Adar assigns each neighbor a weight, that is the reciprocal of the degree of the neighbor. This means that the more vertexes one vertex connected to, the less important it is on evaluating the proximity between a pair of vertex. Formally,

$$\mathbf{S}^{AA} = \mathbf{A} \cdot \mathbf{D} \cdot \mathbf{A} \tag{14}$$

, where $\mathbf{D}$ is a diagonal matrix,

$$\mathbf{D}_{ii} = 1 / \sum_j (A_{ij} + A_{ji}) \tag{15}$$

.

So, for Adamic-Adar,

$$\mathbf{M}_g = \mathbf{I} \tag{16}$$

$$\mathbf{M}_l = \mathbf{A} \cdot \mathbf{D} \cdot \mathbf{A} \tag{17}$$

We list the corresponding formula of $\mathbf{M}_g$ and $\mathbf{M}_l$ of each proximity measurement in Table 1. Note that $\mathbf{M}_g$ and $\mathbf{M}_l$ are both the polynomial of adjacency matrix or its variants. The above proximity measurements can be classified into two types, i.e. global proximity and local proximity. Global proximities, i.e. Katz index and rooted PageRank, are derived from a recurrent formula, which make the proximity can preserve global asymmetric transitivity. And local proximities, i.e. common neighbors and Adamic-Adar, have no recurrent structure and just preserve the asymmetric transitivity in the local structure, which we call it local asymmetric transitivity. Intuitively, $\mathbf{M}_g$ is highly related to global asymmetric transitivity. $\mathbf{M}_g$ has a formulation like $\mathbf{I} - \alpha \cdot \mathbf{B}$, where α is a parameter and $\mathbf{B}$ is a transition matrix. The larger the alpha is, the easier the observed relationship (the edges of the graph) is to be transited in the graph. When $\alpha = 0$, the observed relationship can just transit in a subgraph whose range is limited by the order of $\mathbf{M}_l$.

3.4 Approximation of High-Order Proximity

The objective in Equation (1) aims to find an optimal rank-K approximation of the proximity matrix $\mathbf{S}$. According to [13], the solution is to perform SVD (Singular Value Decomposition) on $\mathbf{S}$ and use the largest K singular value and corresponding singular vectors to construct the optimal embedding vectors. Formally, if

$$\mathbf{S} = \sum_{i=1}^{N} \sigma_i \mathbf{v}_i^s {\mathbf{v}_i^t}^\top \tag{18}$$

, where $\{\sigma_1, \sigma_2, \cdots, \sigma_N\}$ is the singular values sorted in decreasing order, $\mathbf{v}_i^s$ and $\mathbf{v}_i^t$ are corresponding singular vectors of σ_i,

then, we can get the optimal embedding vectors as:

$$\mathbf{U}^s = [\sqrt{\sigma_1} \cdot \mathbf{v}_1^s, \cdots, \sqrt{\sigma_K} \cdot \mathbf{v}_K^s] \tag{19}$$

$$\mathbf{U}^t = [\sqrt{\sigma_1} \cdot \mathbf{v}_1^t, \cdots, \sqrt{\sigma_K} \cdot \mathbf{v}_K^t] \tag{20}$$

.

However, this solution need to calculate the proximity matrix $\mathbf{S}$. Even for sparse adjacency matrix, the time complexity of matrix inversion is up to $O(N^3)$. And the matrix polynomial operation on adjacency matrix will make $\mathbf{S}$ much denser than adjacency matrix $\mathbf{A}$ which will also make the SVD on $\mathbf{S}$ very expensive. Thus, the solution is not feasible for large scale graphs.

As the calculation of $\mathbf{S}$ is the efficiency bottleneck and $\mathbf{S}$ is just the intermediate product in our problem, we propose a novel algorithm to avoid the calculation of $\mathbf{S}$ and learn the embedding vectors directly. As many proximity measurements have the general formulation in Equation (2), we transform the original SVD problem into a generalized SVD problem [22] for proximity measurements with the general formulation.

According to [22], it is easy to derive the following theorem:

Theorem 1. *If we have the singular value decomposition of the general formulation*

$$\mathbf{M}_g^{-1} \cdot \mathbf{M}_l = \mathbf{V}^s \Sigma {\mathbf{V}^t}^\top$$

, where $\mathbf{V}^t$ and $\mathbf{V}^s$ are two orthogonal matrices,

$$\Sigma = diag(\sigma_1, \sigma_2, \cdots, \sigma_N)$$

.

Then, there exists a nonsingular matrix $\mathbf{X}$ and two diagonal matrices, i.e. Σ^l and Σ^g, satisfying that

$${\mathbf{V}^t}^\top \mathbf{M}_l^\top \mathbf{X} = \Sigma^l$$

$${\mathbf{V}^s}^\top \mathbf{M}_g^\top \mathbf{X} = \Sigma^g$$

, where

$$\Sigma^l = diag(\sigma_1^l, \sigma_2^l, \cdots, \sigma_N^l)$$

$$\Sigma^g = diag(\sigma_1^g, \sigma_2^g, \cdots, \sigma_N^g)$$

$$\sigma_1^l \geq \sigma_2^l \geq \cdots \geq \sigma_K^l \geq 0$$

$$0 \leq \sigma_1^g \leq \sigma_2^g \leq \cdots \leq \sigma_K^g$$

$$\forall i \qquad {\sigma_i^l}^2 + {\sigma_i^g}^2 = 1$$

,

and

$$\sigma_i = \frac{\sigma_i^l}{\sigma_i^g} \tag{21}$$

Algorithm 1 High-order Proximity preserved Embedding

```
Require: adjacency matrix $\mathbf{A}$, embedding dimension $K$,
    parameters of high-order proximity measurement$\theta$.
Ensure: embedding source vectors $\mathbf{U}^s$ and target vectors
    $\mathbf{U}^t$.
1: calculate $\mathbf{M}_g$ and $\mathbf{M}_l$.
2: perform JDGSVD with $\mathbf{M}_g$ and $\mathbf{M}_l$, and obtain the generalized singular values $\{\sigma_1^l, \cdots, \sigma_K^l\}$ and $\{\sigma_1^g, \cdots, \sigma_K^g\}$,
    and the corresponding singular vectors, $\{\mathbf{v}_1^s, \cdots, \mathbf{v}_K^s\}$
    and $\{\mathbf{v}_1^t, \cdots, \mathbf{v}_K^t\}$.
3: calculate singular values $\{\sigma_1, \cdots, \sigma_K\}$ according to
    Equation (21).
4: calculate embedding matrices $\mathbf{U}^s$ and $\mathbf{U}^t$ according to
    Equation (19) and (20).
```

As the generalized SVD can also derive the results of SVD, we can still use Equation (19) (20) to get the embedding vectors.

Complete generalized SVD will also achieve $O(N^3)$ time complexity, which is not feasible for large scale graphs. As we only need the largest K singular values and corresponding singular vectors, we adopt a partial generalized SVD algorithm [9], which we call it JDGSVD. This is an iterative Jacobi-Davidson type algorithm which is very scalable when $K \ll N$. Algorithm 1 lists the steps of our algorithm.

3.4.1 Complexity Analysis

In this algorithm, we do not explicitly perform the polynomial operation on adjacency matrix in $\mathbf{M}_l$ and $\mathbf{M}_g$, whose time complexity is up to $O(N^3)$. Because, in JDGSVD, we only need to multiply $\mathbf{M}_g$ and $\mathbf{M}_l$ with some thin matrices whose size is $N \times K$. If we change the multiplication order, and first perform multiplication between adjacency matrix and thin matrix, the time complexity of this operation will reduce to $O(m \cdot K)$, where m is the number of non-zero elements in adjacency matrix (i.e. the number of edges in the graph) and much smaller than N^2 in sparse graph. And, the total time complexity of JDGSVD is $O(m \cdot K^2 \cdot L)$, where L is the iteration number. We can see that the time complexity is just linear with respect to the volumn of data (i.e. the number of edges), which means that it is scalable for large scale graphs.

3.4.2 Approximation Error

Finally, we give the error bound of our algorithm:

Theorem 2. *Given the proximity matrix,* $\mathbf{S}$, *of a directed graph, and the embedding vectors,* $\mathbf{U}^s$ *and* $\mathbf{U}^t$, *learned by* HOPPE. *Then the approximation error is*

$$\|\mathbf{S} - \mathbf{U}^s \cdot \mathbf{U}^t\|_F^2 = \sum_{i=K+1}^{N} \sigma_i^2$$

, and the relative approximation error is:

$$\frac{\|\mathbf{S} - \mathbf{U}^s \cdot \mathbf{U}^t\|_F^2}{\|\mathbf{S}\|_F^2} = \frac{\sum_{i=K+1}^{N} \sigma_i^2}{\sum_{i=1}^{N} \sigma_i^2} \tag{22}$$

where $\{\sigma_i\}$ are the singular values of $\mathbf{S}$ in descend order.

Prove:

$$\begin{aligned}\|\mathbf{S} - \mathbf{U}^s \cdot \mathbf{U}^t\|_F^2 &= \| \sum_{i=1}^{N} \sigma_i \cdot \mathbf{v}_i^s \mathbf{v}_i^{t\top} - \sum_{j=1}^{K} \sigma_j \cdot \mathbf{v}_j^s \mathbf{v}_j^{t\top} \|_F^2 \\ &= \| \sum_{i=K+1}^{N} \sigma_i \cdot \mathbf{v}_i^s \mathbf{v}_i^{t\top} \|_F^2 \\ &= \sum_{i=K+1}^{N} \sigma_i^2\end{aligned}$$

and we have:

$$\|\mathbf{S}\| = \| \sum_{i=1}^{N} \sigma_i \cdot \mathbf{v}_i^s \mathbf{v}_i^{t\top} \| = \sum_{i=1}^{N} \sigma_i^2$$

.

So, the relative approximation error is:

$$\frac{\|\mathbf{S} - \mathbf{U}^s \cdot \mathbf{U}^t\|_F^2}{\|\mathbf{S}\|_F^2} = \frac{\sum_{i=K+1}^{N} \sigma_i^2}{\sum_{i=1}^{N} \sigma_i^2}$$

If $\mathbf{S}$ is low-rank, then the singular values $\{\sigma_{K+1}, \cdots, \sigma_N\}$ will be close to zero and the error will be very small. That is, the lower the rank of $\mathbf{S}$ is, the smaller the error is.

4. EXPERIMENTS

We conduct experiments on a synthetic data and three real-world datasets. We will first introduce the experiment setting, then show the results and analysis on the two types of datasets respectively.

4.1 Experiment Setting

4.1.1 Datasets

We use four datasets, whose statistics are summarized in Table 2.

Table 2: Statistics of datasets. $|V|$ denotes the number of vertexes and $|E|$ denotes the number of edges.

	Syn	Cora	SN-Twitter	SN-TWeibo
$\|\mathbf{V}\|$	10,000	23166	465,017	1,944,589
$\|\mathbf{E}\|$	144,555	91500	834,797	50,655,143

- Synthetic Data (Syn): We generate the synthetic data by the forest fire model [17]. The model can generate powerlaw graphs. We can observe asymmetric transitivity in the generated graphs. As the time complexity of computation of some high-order proximities (e.g. Katz, RPR) is too high, we generate small-scaled synthetic data to allow for the computation of high-order proximities, so that we can evaluate the accuracy of high-order proximity approximation. We randomly generate ten synthetic datasets.

- Cora[1] [31]: This is a citation network of academic papers. The vertexes are academic papers and the directed edges are the citation relationship between papers.

[1]http://konect.uni-koblenz.de/networks/subelj_cora

- Twitter Social Network[2] (SN-Twitter) [5]: This dataset is a subnetwork of Twitter. The vertexes are users of Twitter, and the directed edges are following relationships between users.

- Tencent Weibo Social Network[3] (SN-TWeibo): This dataset contains a subnetwork of the social network in Tencent Weibo[4], a Twitter-style social platform in China. The vertexes are users and the directed edges are following relationship between users.

Due to high time complexity of calculating high-order proximity measurements, we first evaluate the high-order proximity approximation error on two small datasets, i.e. Synthetic Data and Cora, where it is feasible to calculate high-order proximity. Furthermore, we conduct other application experiments on two large datasets, i.e. SN-Twitter and SN-TWeibo, to evaluate the performance of our algorithm in real large-scale graphs. For Synthetic Data and Cora, the graph in each dataset is randomly separated into training set and testing set, and training set contains 80% edges. All the reported performances are the average value on these ten datasets. For SN-Twitter and SN-TWeibo, we generate three datasets by randomly separated the graph into training graph and testing graph, where training graph contains 80% edges and testing graph contains the rest edges.

4.1.2 Baseline Methods

- LINE [32]: This algorithm preserves the first-order and second-order proximity between vertexes. but it only can preserve symmetric second-order proximity when applied to directed graph. We use vertex vectors as source vectors and context vectors as target vectors. We use LINE1 to represent LINE preserving first-order proximity and LINE2 to represent LINE preserving second-order proximity.

- DeepWalk [24]: this algorithm first randomly walks on the graph, and assumes a pair of vertexes similar if they are close in the random path. Then, the embedding is learned to preserve these pairwise similarities in the embedding.

- PPE (Partial Proximity Embedding) [30]: This algorithm first selects a small subset of vertexes as landmarks, and learns the embedding by approximating the proximity between vertexes and landmarks.

- Common Neighbors: We rank the links by the number of common neighbors. We use it for link prediction and vertex recommendation.

- Adamic-Adar: We rank the links by the Adamic-Adar values. We use it for link prediction and vertex recommendation.

In graph reconstruction, link prediction and vertex recommendation experiments, we adopt Katz as the target high-order proximity for HOPE and PPE methods, as Katz has shown superior performance in related tasks in previous works, to approximate for HOPE and PPE. For PPE, we sample 1000 landmarks. For LINE and DeepWalk, we search the parameters grid search.

[2]http://konect.uni-koblenz.de/networks/munmun_twitter_social
[3]http://www.kddcup2012.org/c/kddcup2012-track1/data
[4]http://t.qq.com/

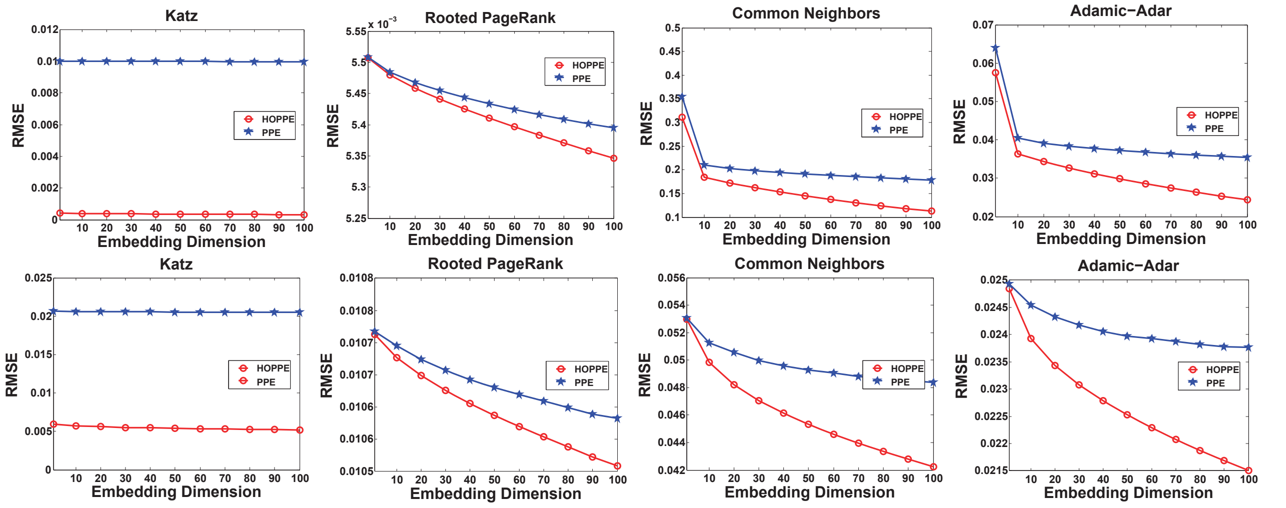

Figure 3: Error of proximity approximation. We evaluate the errors of HOPE and PPE in approximating four proximity measurements, including Katz, RPR, Common Neighbors and Adamic-Adar. First row is the results on Synthetic Data, and second row is the results on Cora. For Katz, $\beta = 0.1$; for RPR, $\alpha = 0.5$.

4.1.3 *Evaluation Metrics*

In the experiments, we adopt RMSE, Precision@k and MAP (Mean Average Precision) as the evaluation metrics.

RMSE is used to evaluate the approximation error of the proximity approximation algorithms, including HOPE and PPE. The formula of RMSE in our problem is:

$$RMSE = \sqrt{\frac{\|\mathbf{S} - \mathbf{U}^s\mathbf{U}^{t\top}\|_F^2}{N^2}}$$

NRMSE (Normalized RMSE) is used to evaluate the relative error of the proximity approximation algorithms. The formula of NRMSE in our problem is:

$$NRMSE = \frac{\sqrt{\frac{\|\mathbf{S} - \mathbf{U}^s\mathbf{U}^{t\top}\|_F^2}{N^2}}}{\sqrt{\frac{\|\mathbf{S}\|_F^2}{N^2}}} = \sqrt{\frac{\|\mathbf{S} - \mathbf{U}^s\mathbf{U}^{t\top}\|_F^2}{\|\mathbf{S}\|_F^2}}$$

Precision@k is used to evaluate the performance of link prediction, which measures the prediction precision of top k links. The formula of Precision@k is:

$$Precision@k = \frac{|\{(i,j)|(i,j) \in \mathbf{E}_p \cap \mathbf{E}_o\}|}{|\mathbf{E}_p|}$$

where $\mathbf{E}_p$ is the set of top k predicted links, $\mathbf{E}_o$ is the set of observed links and $|\cdot|$ represents the size of set.

MAP is used to evaluate the performance of vertex recommendation, which measures the rank accuracy of recommended vertex list. The formula of MAP@k is:

$$AP@k(i) = \frac{\sum_{j=1}^{k} P_i(j) \cdot \delta_i(j)}{\sum_{j=1}^{k} \delta_i(j)}$$

$$MAP@k = \frac{\sum_{v_i \in \mathbf{V}_t} AP(i)}{|\mathbf{V}_t|}$$

4.2 High-order Proximity Approximation

As we preserve asymmetric transitivity by approximating high-order proximity, the error of approximation can evaluate how well we preserve asymmetric transitivity.

We evaluate the approximation error on Synthetic Data and Cora. Besides our algorithms, PPE can also approximate high-order proximity. Here, we compare HOPE with PPE on four proximity measurements, including Katz, RPR, Common Neighbors and Adamic-Adar. Figure 3 shows the RMSE with different number of embedding dimensions. When the number of embedding dimensions grows, the approximation error, RMSE, will decrease. We can see that HOPE achieves much lower RMSE than PPE on all the proximity measurements, which is stable in the ten randomly selected datasets. Especially on Katz, the error of HOPE is one order of magnitude smaller than the error of PPE. With the number of embedding dimensions growing, the error of both algorithms decreases but the margin between HOPE and PPE becomes larger. Although more embedding dimensions can make these two methods be able to incorporate more proximity information, PPE can only approximate a sub-block of the proximity matrix, while HOPE can approximate the whole proximity matrix. Thus HOPE can take more advantage of the increased embedding dimension.

Furthermore, we evaluate how the rank (more precisely, the condition number) of proximity matrix will influence the approximation error. In section 3.4.2, we have theoretically proven that the approximation error of our method is related to the rank of the promixity matrix. Here, we take Rooted PageRank as an example to empirically demonstrate the claim. The parameter α of RPR is highly related to the rank of RPR matrix (i.e. $\mathbf{S}^{RPR}$). When $\alpha = 0$, $\mathbf{S}^{RPR} = \mathbf{I}$, which is full rank. When $\alpha = 1$, all the rows of $\mathbf{S}^{RPR}$ are the stationary distribution of transition matrix $\mathbf{P}$, and the rank of $\mathbf{S}^{RPR}$ is 1. Intuitively, the larger α is, the lower the rank of $\mathbf{S}^{RPR}$ tends to be. We use NRMSE to evaluate the relative error of RPR approximation. Figure 4 shows the NRMSE of HOPE with different α. The larger α is,

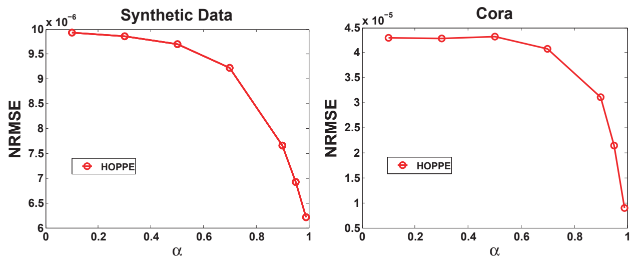

Figure 4: Correlation between relative approximation error and the rank of proximity matrix. The parameter α of Rooted PageRank is highly related to the rank of proximity matrix. Here, we use α to simulate the rank of proximity matrix. The embedding dimension is 100.

the smaller the NRMSE is. This suggests that HOPE can achieve better performance on lower-rank proximity matrix.

4.3 Graph Reconstruction

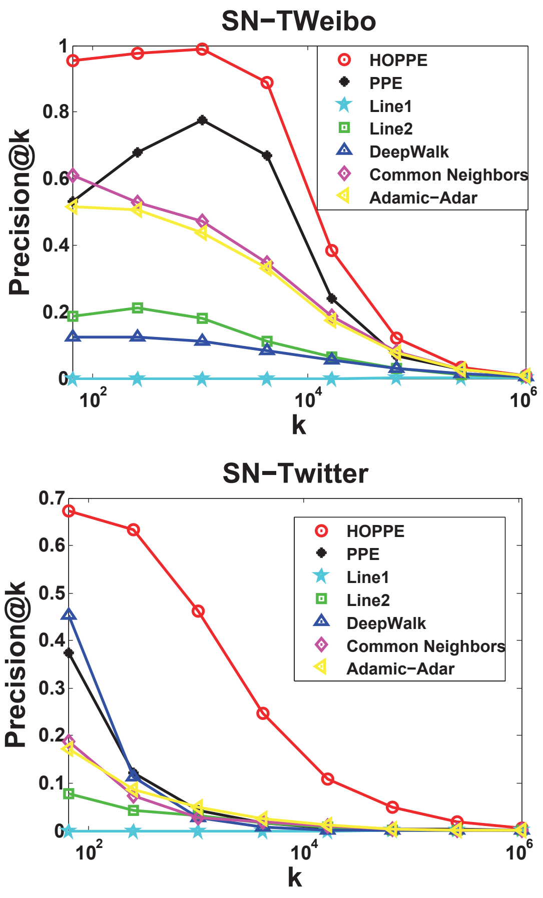

Figure 5: Precision@k of graph reconstruction on SN-TWeibo and SN-Twitter. We rank pairs of vertexes according to their reconstructed proximity and evaluate the reconstruction precision in top k pairs of vertexes.

As the representation of a graph, embedding vectors are expected to well reconstruct the graph. We evaluate the reconstruction ability on training sets of SN-TWeibo and SN-Twitter. We reconstruct the graph edges based on the reconstructed proximity between vertexes. For graph embedding algorithms (i.e. HOPE, PPE, Line and DeepWalk), we use the inner product between embedding vectors to reconstruct the proximity matrix. For Common Neighbors and Adamic-Adar, we directly calculate the proximity matrix. We rank the pairs of vertexes according to their corresponding reconstructed proximity. Then, we calculate the ratio of real links in top k pairs of vertexes as the reconstruction precision. As the number of possible pairs of vertexes $(N \cdot (N-1))$ is too large in SN-TWeibo and SN-Twitter, we randomly sample about 0.1% pairs of vertexes for evaluation.

Figure 5 shows the Precision@k with different k. HOPE significantly outperforms baselines. As mentioned above, PPE just approximates a sub-block of proximity matrix. So, it works poorer than HOPE. Comparing with Line2 which directly reconstructs the directed links, HOPE still achieves much better performance. It may because HOPE uses high-order proximity as weight of directed edges, but LINE2 assigns equivalent weight for each directed edges. The edges in densely connected vertex clusters will obtain higher weights. Thus, these edges will be preserved first. Moreover, the number of these edges is larger than that in sparsely connected vertex clusters. Thus, HOPE may reconstruct more edges than Line2.

4.4 Link Prediction

As asymmetric transitivity captures the correlation among edges, we can use it to predict missing edges in graphs. On the other hand, the performance of HOPE on link prediction can also reflect how well the asymmetric transitivity is preserved. This experiment is conducted on SN-TWeibo and SN-TWitter. We train the embedding vectors on training graphs, and evaluate the prediction performance on testing graphs. We randomly sample about 0.1% pairs of vertexes for evaluation. Then, we rank them according to the inner product between embedding vectors (for HOPE, LINE1, LINE2 and PPE) or the calculated proximity (for Common Neighbors and Adamic-Adar), and evaluate the prediction precision in top k pairs of vertexes.

Figure 6 shows the precision@k of link prediction with different k. Our algorithm, HOPE, outperforms the baselines significantly. Compared to Section 4.3, the performance of PPE significantly decreases in prediction task. This is mainly because PPE only trains on partial proximity matrix, which is easy to overfit. Besides, we can see that all the curves converges to a point when k is large. This is because almost all the real edges have been correctly predicted by all the algorithms.

4.5 Vertex Recommendation

The setting of training procedure in this experiment is the same as link prediction. But, we evaluate the performance of algorithms from the vertex view, i.e. vertex recommendation performance. We randomly sample 1000 vertexes. For each vertex v_i, we randomly hide 20% outgoing links as groundtruth. Then we derive the top 100 vertexes with the highest proximity with v_i as the candidates that v_i will possible point to. After that, we use MAP@10, MAP@50 and MAP@100 to evaluate the quality of recommendation. Table 3 shows the MAPs of different algorithms. HOPE outperforms the baseline algorithms.

Table 3: MAP of vertex recommendation on SN-TWeibo and SN-Twitter. For each vertex, the recommended vertex list is ranked according to the predicted proximity between vertexes. For embedding algorithms, we calculate the predicted proximity by performing inner product between embedding vectors.

Method	SN-TWebio			SN-Twitter		
	MAP@10	MAP@50	MAP@100	MAP@10	MAP@50	MAP@100
HOPE	**0.2295**	**0.1869**	**0.169**	**0.1000**	**0.0881**	**0.0766**
PPE	0.0928	0.0845	0.077	0.0061	0.0077	0.0081
LINE1	0	0	0.005	0.0209	0.0221	0.0221
LINE2	0.051	0.051	0.048	0.0044	0.0043	0.0035
DeepWalk	0.0635	0.0583	0.004	0.0006	0.0008	0.001
Common Neighbors	0.1217	0.1031	0.155	0.0394	0.0379	0.0369
Adamic-Adar	0.1173	0.0990	0.156	0.0455	0.0442	0.0423

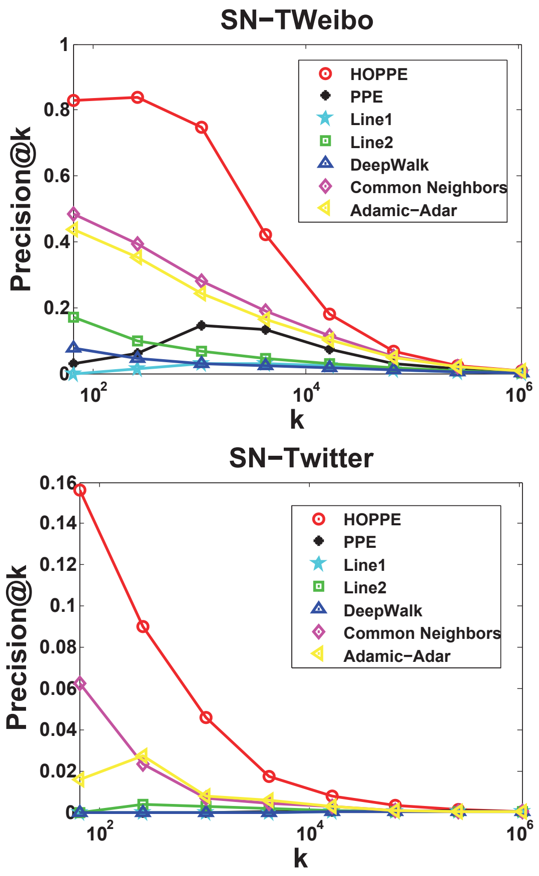

Figure 6: Precision@k of link prediction on SN-TWeibo and SN-Twitter. We rank pairs of vertexes according to their reconstructed proximity and evaluate the reconstruction precision in largest k pairs of vertexes.

Especially, the MAP@10 of HOPE achieves at least 88.5% improvement and the MAP@50 of HOPE achieves at least 81.2% improvement over all baseline algorithms. In most settings, the algorithms preserving directed high-order relationship, including HOPE, Common Neighbors, Adamic-Adar and PPE, outperforms LINE2 which only preserves directed first-order relationship. This proves that preserving directed high-order relationship is critical to capture the structure of directed graphs.

5. CONCLUSION

In this paper, we aim to preserve asymmetric transitivity in directed graphs, and propose to preserve asymmetric transitivity by approximating high-order proximities. We propose a scalable approximation algorithm , called High-Order Proximity preserved Embedding (HOPE). In this algorithm, we first derive a general formulation of a class of high-order proximity measurements, then apply generalized SVD to the general formulation, whose time complexity is linear with the size of graph. The empirical study demonstrates the superiority of asymmetric transitivity and our proposed algorithm, HOPE. Our future direction is to develop a nonlinear model to better capture the complex structure of directed graphs.

6. ACKNOWLEDGEMENTS

This work was supported by National Program on Key Basic Research Project, No. 2015CB352300; National Natural Science Foundation of China, No. 61370022, No. 61531006, No. 61472444 and No. 61210008. Thanks for the research fund of Tsinghua-Tencent Joint Laboratory for Internet Innovation Technology.

7. REFERENCES

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.
[2] S. Cao, W. Lu, and Q. Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 891–900. ACM, 2015.
[3] S. Chang, G.-J. Qi, C. C. Aggarwal, J. Zhou, M. Wang, and T. S. Huang. Factorized similarity learning in networks. In *Data Mining (ICDM), 2014 IEEE International Conference on*, pages 60–69. IEEE, 2014.
[4] M. Chen, Q. Yang, and X. Tang. Directed graph embedding. In *IJCAI*, pages 2707–2712, 2007.
[5] M. De Choudhury, Y.-R. Lin, H. Sundaram, K. S. Candan, L. Xie, A. Kelliher, et al. How does the data sampling strategy impact the discovery of information diffusion in social media? *ICWSM*, 10:34–41, 2010.
[6] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.
[7] M. S. Handcock, A. E. Raftery, and J. M. Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354, 2007.

[8] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using laplacianfaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(3):328–340, 2005.

[9] M. Hochstenbach. A jacobi–davidson type method for the generalized singular value problem. *Linear Algebra and its Applications*, 431(3):471–487, 2009.

[10] P. D. Hoff. Multiplicative latent factor models for description and prediction of social networks. *Computational and Mathematical Organization Theory*, 15(4):261–272, 2009.

[11] P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *Journal of the american Statistical association*, 97(460):1090–1098, 2002.

[12] P. W. Holland and S. Leinhardt. An exponential family of probability distributions for directed graphs. *Journal of the american Statistical association*, 76(373):33–50, 1981.

[13] J. Hopcroft and R. Kannan. Foundations of data science. 2014.

[14] G. Jeh and J. Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002.

[15] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

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

[17] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187. ACM, 2005.

[18] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185, 2014.

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

[20] K. Miller, M. I. Jordan, and T. L. Griffiths. Nonparametric latent feature models for link prediction. In *Advances in neural information processing systems*, pages 1276–1284, 2009.

[21] S. Mousazadeh and I. Cohen. Embedding and function extension on directed graph. *Signal Processing*, 111:137–149, 2015.

[22] C. C. Paige and M. A. Saunders. Towards a generalized singular value decomposition. *SIAM Journal on Numerical Analysis*, 18(3):398–405, 1981.

[23] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12:1532–1543, 2014.

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

[25] D. Perrault-Joncas and M. Meila. Estimating vector fields on manifolds and the embedding of directed graphs. *arXiv preprint arXiv:1406.0013*, 2014.

[26] D. C. Perrault-Joncas and M. Meila. Directed graph embedding: an algorithm based on continuous limits of laplacian-type operators. In *Advances in Neural Information Processing Systems*, pages 990–998, 2011.

[27] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[28] B. Scholkopft and K.-R. Mullert. Fisher discriminant analysis with kernels. *Neural networks for signal processing IX*, 1:1, 1999.

[29] T. A. Snijders, P. E. Pattison, G. L. Robins, and M. S. Handcock. New specifications for exponential random graph models. *Sociological methodology*, 36(1):99–153, 2006.

[30] H. H. Song, T. W. Cho, V. Dave, Y. Zhang, and L. Qiu. Scalable proximity estimation and link prediction in online social networks. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 322–335. ACM, 2009.

[31] L. Šubelj and M. Bajec. Model of complex networks based on citation dynamics. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 527–530. International World Wide Web Conferences Steering Committee, 2013.

[32] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[33] J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[34] Y. J. Wang and G. Y. Wong. Stochastic blockmodels for directed graphs. *Journal of the American Statistical Association*, 82(397):8–19, 1987.

[35] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin. Graph embedding and extensions: a general framework for dimensionality reduction. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(1):40–51, 2007.

[36] J. Ye, R. Janardan, and Q. Li. Two-dimensional linear discriminant analysis. In *Advances in neural information processing systems*, pages 1569–1576, 2004.

[37] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 487–494. ACM, 2007.