A General View for Network Embedding as Matrix Factorization

Xin Liu
National Institute of Advanced Industrial Science and Technology
xin.liu@aist.go.jp

Tsuyoshi Murata
Dept. of Computer Science
Tokyo Institute of Technology
murata@c.titech.ac.jp

Kyoung-Sook Kim
National Institute of Advanced Industrial Science and Technology
ks.kim@aist.go.jp

Chatchawan Kotarasu
Faculty of ICT
Mahidol University
chatchawan.kot@gmail.com

Chenyi Zhuang
National Institute of Advanced Industrial Science and Technology
zhuang.chenyi@aist.go.jp

ABSTRACT

We propose a general view that demonstrates the relationship between network embedding approaches and matrix factorization. Unlike previous works that present the equivalence for the approaches from a skip-gram model perspective, we provide a more fundamental connection from an optimization (objective function) perspective. We demonstrate that matrix factorization is equivalent to optimizing two objectives: one is for bringing together the embeddings of similar nodes; the other is for separating the embeddings of distant nodes. The matrix to be factorized has a general form: $\mathbf{S}-\beta\cdot\mathbf{1}$. The elements of $\mathbf{S}$ indicate pairwise node similarities. They can be based on any user-defined similarity/distance measure or learned from random walks on networks. The shift number β is related to a parameter that balances the two objectives. More importantly, the resulting embeddings are sensitive to β and we can improve the embeddings by tuning β. Experiments show that matrix factorization based on a new proposed similarity measure and β-tuning strategy significantly outperforms existing matrix factorization approaches on a range of benchmark networks.

KEYWORDS

graph embedding, network representation learning, matrix factorization, node similarity, graph mining, social networks

ACM Reference Format:
Xin Liu, Tsuyoshi Murata, Kyoung-Sook Kim, Chatchawan Kotarasu, and Chenyi Zhuang. 2019. A General View for Network Embedding as Matrix Factorization. In *The Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19), February 11–15, 2019, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3289600.3291029

1 INTRODUCTION

Recently, there has been a surge of interest in network embedding, or learning representations of network nodes in a low dimensional vector space, such that the network structural information and properties are maximally preserved [3, 11, 15, 18, 60]. We can use the learned embeddings as feature inputs for downstream machine learning tasks. This technology is beneficial for many network analysis tasks, such as community detection [6, 55, 62], label classification [43, 44], link prediction [16], and visualization [41].

Many studies focus on preserving the proximity structure between nodes. That is, the (dis)similarity of embeddings in the low dimension space should, to some extent, approximate the (dis)similarity of nodes in the original network (the similarity depends on the user's definition). To address this problem, researchers have proposed different methods and many of them are based on matrix factorization. For example, SocDim [45] factorizes the modularity matrix [43] and the normalized Laplacian matrix; NEU [58] factorizes similarity matrices that encode higher order of the adjacency matrix; HOPE [35] factorizes several matrices based on different similarity measures; GraRep [4] factorizes a matrix that is related to the k-step transition probability matrix. Further, skip-gram [34] model based network embedding approaches, such as LINE [42], DeepWalk [36], PTE [41], and node2vec [16] are also equivalent to implicit matrix factorization [37, 49, 57].

In this paper, we propose a general view that demonstrates the relationship between network embedding approaches and matrix factorization. Unlike previous works [37, 49, 57] which present the equivalence for the approaches from a skip-gram perspective, we provides a more fundamental connection from an optimization (objective function) perspective. Our contribution is twofold. First, we propose an objective function with two sub-objectives: one is for bringing together the embeddings of similar nodes; the other is for separating the embeddings of distant nodes. This function is a framework that connects many network embedding methods. As a special case, maximizing this function is equivalent to factorizing a shifted similarity matrix $\mathbf{S}-\beta\cdot\mathbf{1}$. The elements of $\mathbf{S}$ indicate pairwise node similarities, which are optional. The shift number β is related to a parameter that balances the two sub-objectives. More importantly, we show that the resulting embeddings are sensitive to β and we can refine the embeddings by tuning β. For example, based on this tuning strategy we achieved a performance gain of up to 46.9%.

Our second contribution is the proposition a Global Resource Allocation (GRA) similarity measure. Experiments in ten commonly used datasets show that matrix factorization based on this new measure and β-tuning strategy significantly outperforms GraRep, LINE, node2vec, DeepWalk, and HOPE on the multilabel classification and link prediction tasks.

The rest of the paper is organized as follows. Section 2 proposes the objective function and demonstrates the relationship between network embedding and matrix factorization. Section 3 introduces GRA similarity. Section 4 presents our algorithm. Section 5 reports the experiment results. Section 6 surveys related work. Finally, Section 7 gives our conclusion.

2 OBJECTIVE FUNCTION

We begin with identifying the symbols that will be used. For simplicity and clarity, we limit our vision to an undirected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_i \mid i = 1, \cdots, n\}$ is the node set, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. $\mathbf{A}$ denotes the adjacency matrix, with the element $A_{ij} = A_{ji} \in \{0, 1\}$ indicating whether $(v_i, v_j) \in \mathcal{E}$. The goal is to map each node v_i to a vector or *embedding* $\boldsymbol{e_i} \in \mathbb{R}^d$, such that the proximities between nodes can be maximally preserved by the embeddings. $d \ll n$ is the embedding dimension.

2.1 Definition

We first define the following objective

$$O_1 = \sum_{i,j} f_\uparrow^v(s_{ij}^v) \cdot f_\uparrow^e(s_{ij}^e). \tag{1}$$

$s^v : (v_i, v_j) \rightarrow \mathbb{R}$ is a similarity measure for a pair of nodes (we will see the definition of this measure later): The greater s_{ij}^v is, the more similar v_i and v_j are. s^e: $(\boldsymbol{e_i}, \boldsymbol{e_j}) \mapsto \mathbb{R}$ is a similarity measure for a pair of embeddings: The greater s_{ij}^e is, the more similar $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$ are. $f_\uparrow^v, f_\uparrow^e$: $\mathbb{R} \mapsto \mathbb{R}_{\geq 0}$ are increasing functions. They serve for adjusting the relative magnitude of s_{ij}^v and s_{ij}^e, respectively. Intuitively, maximizing O_1 entails large s_{ij}^e for large s_{ij}^v. Therefore, the significance is that the similar nodes v_i and v_j will have similar embeddings $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$.

Similarly, we can define another objective

$$O_2 = \sum_{i,j} f_\downarrow^v(s_{ij}^v) \cdot f_\downarrow^e(s_{ij}^e), \tag{2}$$

where $f_\downarrow^v, f_\downarrow^e$: $\mathbb{R} \mapsto \mathbb{R}_{\geq 0}$ are decreasing functions. Intuitively, maximizing O_2 entails small s_{ij}^e for small s_{ij}^v. Therefore, the result is that the distant nodes v_i and v_j will have distant embeddings $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$. Combining (1) and (2), our final objective function is

$$O = O_1 + \rho O_2, \tag{3}$$

where $\rho > 0$ is a parameter for balancing the two objectives.

2.2 As A Framework

Our objective function is a framework that allows freely defining s^v, s^e, $f_\uparrow^v$, $f_\uparrow^e$, $f_\downarrow^v$, and $f_\downarrow^e$. By appropriate selection of them, many network embedding methods such as DeepWalk [36], node2vec [16], Cauchy Graph Embedding [30], GraRep [4] and Laplacian Eigenmap [2] can be explained and unified.

As an example, we demonstrate that the objective function used in Cauchy Graph Embedding [30] is a a special case of our definition. To see it, we first give an alternative definition of Eq. (1) as

$$O_1 = \sum_{i,j} f_\uparrow^v(s_{ij}^v) \cdot f_\downarrow^e(d_{ij}^e), \tag{4}$$

where d^e: $(\boldsymbol{e_i}, \boldsymbol{e_j}) \mapsto \mathbb{R}_{\geq 0}$ is a distance measure for a pair of embeddings: The greater d_{ij}^e is, the more distant (less similar) $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$ are. Then, if we specify

$$s_{ij}^v = A_{ij}, \tag{5}$$

$$d_{ij}^e = \|\boldsymbol{e_i} - \boldsymbol{e_j}\|^2, \tag{6}$$

$$f_\uparrow^v(x) = x, \tag{7}$$

$$f_\downarrow^e(x) = 1/(x + \tau^2), \tag{8}$$

we can obtain

$$O_1 = \sum_{i,j} \frac{A_{ij}}{\|\boldsymbol{e_i} - \boldsymbol{e_j}\|^2 + \tau^2}, \tag{9}$$

which is just the objective used in Cauchy Graph Embedding [30]. More examples about our objective function as a unified framework are omitted due to space limitation.

2.3 Relation to Matrix Factorization

Suppose we specify

$$s_{ij}^e = \boldsymbol{e_i}^\top \boldsymbol{e_j}, \tag{10}$$

$$f_\uparrow^v(x) = \exp(\rho_1 x), \tag{11}$$

$$f_\uparrow^e(x) = \log \sigma_{\rho_3}(x), \tag{12}$$

$$f_\downarrow^v(x) = \exp(-\rho_2 x), \tag{13}$$

$$f_\downarrow^e(x) = \log \sigma_{\rho_3}(-x), \tag{14}$$

where $\sigma_{\rho_3}(x) = 1/(1 + \exp(-\rho_3 x))$ denotes the logistic function and $\rho_1, \rho_2, \rho_3 > 0$ are parameters, obtaining

$$\begin{aligned} O &= \sum_{i,j} \exp(\rho_1 s_{ij}^v) \big[\log \sigma_{\rho_3}(\boldsymbol{e_i}^\top \boldsymbol{e_j}) \big] \\ &\quad + \rho \sum_{i,j} \exp(-\rho_2 s_{ij}^v) \big[\log \sigma_{\rho_3}(-\boldsymbol{e_i}^\top \boldsymbol{e_j}) \big]. \end{aligned} \tag{15}$$

To maximize (15), we set its partial derivative with respect to $\boldsymbol{e_i}^\top \boldsymbol{e_j}$ to zero as in [24], and after simplification arrive at

$$\boldsymbol{e_i}^\top \boldsymbol{e_j} = \frac{(\rho_1 + \rho_2) s_{ij}^v - \log \rho}{\rho_3}. \tag{16}$$

Let $\mathbf{E} = (\boldsymbol{e_1}, \boldsymbol{e_2}, \cdots, \boldsymbol{e_n})^\top$ denote the embedding matrix, where the i-th row represents the embedding of v_i. The equivalent matrix form of (16) can be expressed as

$$\mathbf{E}\mathbf{E}^\top = \mathbf{S}, \tag{17}$$

where $\mathbf{S}$ has elements

$$S_{ij} = \frac{\rho_1 + \rho_2}{\rho_3}(s_{ij}^v - \frac{\log \rho}{\rho_1 + \rho_2}). \tag{18}$$

This implies that we can learn the embeddings by performing truncated Singular Value Decomposition (SVD) on $\mathbf{S}$. Due to the properties of SVD, the factor $(\rho_1 + \rho_2)/\rho_3$ merely contributes an overall multiplicative factor to the resulting embeddings, in which we are not interested, so we will henceforth drop it. Thus, the elements of $\mathbf{S}$ can be reduced to

$$S_{ij} = s_{ij}^v - \beta, \tag{19}$$

where $\beta = \log \rho/(\rho_1 + \rho_2)$ is a parameter.

The above analysis provides a general view for network embedding as matrix factorization. First, it demonstrates the relationship between matrix factorization and maximizing the objective function, in the sense that similar nodes have similar embeddings and distant nodes have distant embeddings. Secondly, the matrix $\mathbf{S}$ is optional, since s_{ij}^v can be based on any user-defined similarity/distance measure or can be learned from random walks. Thus, our view is not only limited to skip-gram based methods (such as DeepWalk [36], LINE [42], PTE [41], and node2vec [16]) as have already been addressed [37, 49, 57], but also applicable to a broad range of approaches that factorize various matrices (such as the adjacency matrix [1], the high-order adjacency matrix [58], modularity matrix [43], graph Laplacians [45], Rooted PageRank, Adamic-Adar, and Katz similarity matrices [35]). Thirdly, previous works [24, 37, 49, 57] for presenting the equivalence between matrix factorization and skip-gram based approaches arrive at a shifted matrix where the shift number indicates the number of negative sampling. This number is just used to reduce noise and replace negative elements of the matrix with zero. However, from our development, we come to see that the shift number β is actually related to the balance parameter of the two sub-objectives. As we will see, the resulting embeddings are sensitive to β and we can refine the embeddings by tuning β.

3 GRA SIMILARITY

A problem is how to define a similarity measure s_{ij}^v that will produce good embeddings. In recent years, researchers have proposed different measures, which can be classified into two categories based on whether they use a local or global structure of the network [29]. The local measures have the advantage of ease of computing. However, local measures seem insufficient because many node pairs are assigned the same similarity scores. Therefore, we are more interested in global measures. In the following, we review some local and global measures. Then, we propose a new global measure called GRA similarity.

3.1 Local Measure

The most simple local measure is Common Neighbor (CN) similarity, which is based on the assumption that two nodes are similar if they have many common neighbors. Thus, this measure simply counts the number of common neighbors, as

$$s_{ij}^{\mathrm{cn}} = |\Gamma_i \cap \Gamma_j|, \tag{20}$$

where $\Gamma_i = \{v_j \mid A_{ij} > 0\}$ is the set of immediate neighbors of v_i. CN similarity can be regarded as a rudimentary measure between v_i and v_j. It is, however, not entirely satisfactory. It can take large values for nodes with high degree even if only a small fraction of their neighbors are the same, and in many cases this runs contrary to our intuition about what constitutes similarity [23]. Commonly therefore people normalize in some way and proposed variants, such as Cosine (COS), Jaccard (JAC), Hub-Promoted (HUB) and Resource Allocation (RA) similarities [64]. These definitions are shown in the following.

$$s_{ij}^{\mathrm{jac}} = \sum_{v_t \in \Gamma_i \cap \Gamma_j} \frac{1}{|\Gamma_i \cup \Gamma_j|}, \tag{21}$$

$$s_{ij}^{\mathrm{cos}} = \sum_{v_t \in \Gamma_i \cap \Gamma_j} \frac{1}{\sqrt{|\Gamma_i|\,|\Gamma_j|}}, \tag{22}$$

$$s_{ij}^{\mathrm{hub}} = \sum_{v_t \in \Gamma_i \cap \Gamma_j} \frac{1}{\min\{k_i, k_j\}}, \tag{23}$$

$$s_{ij}^{\mathrm{ra}} = \sum_{v_t \in \Gamma_i \cap \Gamma_j} \frac{1}{k_t}, \tag{24}$$

where $k_i = \sum_{j=1}^{n} A_{ij}$ denotes the degree of v_i.

The differences in (21)-(24) are just in the normalization parts (the denominators). Liben-Nowell et al. [26] and Zhou et al. [64] systematically compared these local measures and found that RA similarity has the best performance. Note that RA similarity appends a normalization to depress the contribution of high-degree common neighbors. This normalization method can be justified by a simple example. Consider in a social network v_i and v_j are two friends of v_t. If v_t is an ordinary person, it is much possible that v_i and v_j are friends with each other. But if v_t is a celebrity, it is even less likely that v_i and v_j know each other. This is because that a celebrity has thousands of friends, and thus it is natural to depress its contribution as a "binding agent" of v_i and v_j.

3.2 Global Measure

One of the most famous global measures is Katz similarity. It is based on a weighted summation over the number of paths between two nodes. The weight decays exponentially with the path length to assign higher weights on shorter paths. The mathematical expression of Katz similarity is

$$s_{ij}^{\mathrm{katz}} = \sum_{l=1}^{\infty} \alpha^l [\mathbf{A}^l]_{ij}, \tag{25}$$

where l denotes the path length, α is a parameter controlling how fast the weight decays with the path length, and $[\mathbf{A}^l]_{ij}$ is exactly equal to the number of length-l paths between v_i and v_j.

3.3 Definition of GRA Similarity

Now we propose GRA similarity, as an extension of RA and Katz similarities. Let us first point out that Katz similarity can be viewed as an extension of CN similarity. This is because that the local version of Katz similarity that is based on $l = 2$ paths is just CN similarity, as can be seen from the following equation

$$s_{ij}^{\text{katz}} \mid_{l=2} = [\mathbf{A}^2]_{ij} = \sum_{v_t \in \Gamma_i \cap \Gamma_j} 1 = s_{ij}^{\text{cn}}, \tag{26}$$

where we have omitted the overall multiplicative factor α^2.

Our idea is to extend RA similarity to longer paths, in the similar way that Katz similarity extends CN similarity. The specific definition of GRA similarity follows. Suppose $p_{ij}^l = (v_{i_0}, v_{i_1}, \ldots, v_{i_l})$ where $i_0 = i, i_l = j$ is a length-l path between v_i and v_j. First, we assume the contribution of p_{ij}^l to v_i and v_j's similarity is equal to the reciprocal of the product of the degrees of the intermediate nodes of the path. That is,

$$c(p_{ij}^l) = \frac{k_{i_0} k_{i_l}}{k_{i_0} k_{i_1} \cdots k_{i_l}}, \tag{27}$$

Secondly, for each contribution $c(p_{ij}^l)$, we associate a weight α^l, where again α is a parameter controlling the decay rate. Finally, the similarity is defined as a summation of the weighted contributions over all possible paths

$$s_{ij}^{\text{gra}} = \sum_{l=1}^{\infty} \alpha^l \sum_{p_{ij}^l} c(p_{ij}^l). \tag{28}$$

We can derive that the local version of GRA similarity based on $l = 2$ paths is RA similarity, as

$$s_{ij}^{\text{gra}} \mid_{l=2} = \sum_{p_{ij}^{l=2}} c(p_{ij}^{l=2}) = \sum_{v_t \in \Gamma_i \cap \Gamma_j} \frac{1}{k_t} = s_{ij}^{\text{ra}}, \tag{29}$$

where again we have omitted the overall multiplicative factor α^2.

Katz similarity can be defined in the same manner as GRA similarity by considering the contribution of each path as one, as

$$s_{ij}^{\text{katz}} = \sum_{l=1}^{\infty} \alpha^l [\mathbf{A}^l]_{ij} = \sum_{l=1}^{\infty} \alpha^l \sum_{p_{ij}^l} 1 = \sum_{l=1}^{\infty} \alpha^l \sum_{p_{ij}^l} c'(p_{ij}^l), \tag{30}$$

Therefore, both Katz and GRA similarities render a high score if there are a large number of paths between two nodes. A difference is that the former accepts equal contributions of the paths, whereas the latter depresses the contribution of the paths that contain high-degree intermediate nodes. Note that a high-degree intermediate node would result in many paths, and thus it is reasonable to depress the contributions of these paths. This comes down in a continuous line with the normalization method of RA similarity.

We could equivalently express (28) in matrix form and define a similarity matrix

$$\mathbf{S}^{\text{gra}} = \alpha\mathbf{A} + \alpha^2\mathbf{A}\mathbf{D}^{-1}\mathbf{A} + \alpha^3\mathbf{A}\mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1}\mathbf{A} + \cdots, \tag{31}$$

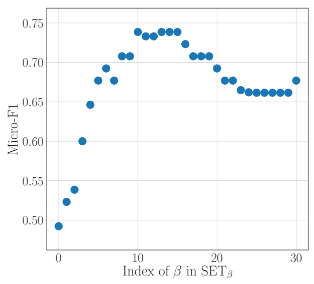

Figure 1: The embeddings are sensitive to β.

where $\mathbf{D} = \text{diag}(k_1, k_2, \cdots, k_n)$ is the degree diagonal matrix. Multiplying $\mathbf{D}^{-1}$ at both sides of (31), we get

$$\begin{aligned} \mathbf{S}^{\text{gra}}\mathbf{D}^{-1} &= -\mathbf{I} + \mathbf{I} + \alpha\mathbf{A}\mathbf{D}^{-1} + \alpha^2(\mathbf{A}\mathbf{D}^{-1})^2 + \cdots & (32)\\ &= (\mathbf{I} - \alpha\mathbf{A}\mathbf{D}^{-1})^{-1} - \mathbf{I} & (33)\\ &= (\mathbf{I} - \alpha\mathbf{A}\mathbf{D}^{-1})^{-1}\alpha\mathbf{A}\mathbf{D}^{-1}, & (34) \end{aligned}$$

where $\mathbf{I}$ is the identity matrix. Finally, we have

$$\mathbf{S}^{\text{gra}} = (\mathbf{I} - \alpha\mathbf{A}\mathbf{D}^{-1})^{-1}\alpha\mathbf{A}. \tag{35}$$

4 THE PROPOSED METHOD

As a special proposal, we can obtain embeddings by factorizing the shifted GRA similarity matrix $\mathbf{S}^{\text{gra}} - \beta \cdot \mathbf{1}$, where $\mathbf{1}$ denotes a matrix with all elements one.

4.1 Addressing the Parameters

Let us examine the parameters. One parameter is α, which indicates the decay rate in the definition of GRA similarity. To ensure convergence of the series in (32), α should lie in the range $0 < \alpha < 1$ (the spectral radius of $\mathbf{A}\mathbf{D}^{-1}$ is one). We empirically found that greater values of α produce good results. Furthermore, for values of α in the range $0.9 < \alpha < 0.98$, the precise value is not critical. Hence, we simply set $\alpha = 0.95$.

Another parameter is $\beta = \log \rho/(\rho_1 + \rho_2)$. Note that $\rho, \rho_1, \rho_2 > 0$, hence, theoretically, β can be any real number. It is not easy to select the best β automatically. However, the possible β that can produce sensible embeddings is highly related to $\mathbf{S}^{\text{gra}}$, since the elements of the shifted matrix should fall within a significant range.

In practice, we limit the search space of β to a small set SET$_\beta$. Specifically, suppose $S_{0\%}^{\text{gra}}$, $S_{60\%}^{\text{gra}}$, $S_{80\%}^{\text{gra}}$, and $S_{100\%}^{\text{gra}}$ denote the 0th, 60th, 80th, and 100th percentile of the elements in $\mathbf{S}^{\text{gra}}$, respectively; SET$_\beta$ is composed of ten evenly spaced values over $(S_{0\%}^{\text{gra}}, S_{60\%}^{\text{gra}}]$ and $(S_{60\%}^{\text{gra}}, S_{80\%}^{\text{gra}}]$, and ten logarithmically spaced values over $(S_{80\%}^{\text{gra}}, S_{100\%}^{\text{gra}}]$.

Figure 1 displays an example of the multilabel classification results generated by different β in the air-traffic network of Brazil [39]. The x-axis represents the index of β in SET$_\beta$. That is, index=0 represents $\beta = 0$; index=1,2,$\cdots$,10 represent the ten evenly spaced values over $(S_{0\%}^{\text{gra}}, S_{60\%}^{\text{gra}}]$; index=11,12,$\cdots$,20 represent the ten evenly spaced values over $(S_{60\%}^{\text{gra}}, S_{80\%}^{\text{gra}}]$; index=21,22,$\cdots$,30 represent the ten logarithmically spaced values over $(S_{80\%}^{\text{gra}}, S_{100\%}^{\text{gra}}]$. The y-axis

Table 1: Statistics of the datasets: number of nodes $|\mathcal{V}|$; number of edges $|\mathcal{E}|$; number of labels $|\mathcal{L}|$.

Dataset	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$	$\|\mathcal{L}\|$
BrazilAir	131	1,003	4
EuropeAir	399	5,993	4
USAir	1,190	13,599	4
Cora	2,708	5,278	7
Citeseer	3,264	4,551	6
DBLP	13,184	47,937	5
WikiPage	2,363	11,596	17
WikiWord	4,777	92,295	40
PPI	3,860	37,845	50
Flickr	80,513	5,899,882	195

represents the Micro-F1 scores as an quantitative evaluation of the embeddings. We can find that the embeddings are sensitive to β and setting $\beta = 0$ does not produce the best embeddings. Importantly, the quality of the embeddings change with β regularly. Therefore, we can search the best β from SET_β based on some validation data in a semi-supervised fashion.

4.2 Computational Issues

Note that $\mathbf{S}^{\text{gra}}$ has the form $\mathbf{S}^{\text{gra}} = \mathbf{M}_{\text{g}}^{-1}\mathbf{M}_{\text{l}}$ where $\mathbf{M}_{\text{g}} = \mathbf{I} - \alpha\mathbf{A}\mathbf{D}^{-1}$ and $\mathbf{M}_{\text{l}} = \alpha\mathbf{A}$, as indicated in (35). This implies that when $\beta = 0$ we can obtain the embeddings by following the fast generalized SVD algorithm of Ou et al. [35], which avoids calculating the similarity matrix $\mathbf{S}^{\text{gra}}$ and only requires a time complexity of near $\text{O}(|\mathcal{E}|)$.

When $\beta \neq 0$, we cannot use computation-saving shortcut of the generalized SVD algorithm, and thus require more computing power. In this case, the most time-consuming part is the calculation of $\mathbf{S}^{\text{gra}}$, which involves a matrix inversion operation. In practice, the calculation is most simply achieved by direct multiplication. We can rewrite (35) as

$$\mathbf{S}^{\text{gra}}\mathbf{D}^{-1} = \alpha\mathbf{A}\mathbf{D}^{-1}(\mathbf{I} + \mathbf{S}^{\text{gra}}\mathbf{D}^{-1}). \tag{36}$$

Making any guess for an initial value of $\mathbf{S}^{\text{gra}}\mathbf{D}^{-1}$, such as $\mathbf{S}^{\text{gra}}\mathbf{D}^{-1} = \mathbf{0}$, we iterate this equation repeatedly until it converges. We have empirically found good convergence after 100 iterations or less.

5 EXPERIMENTS

In this section, we conduct experiments to answer the following questions:

Q1. Is GRA a good measure to produce embeddings?
Q2. Does tuning β help improve the embeddings?
Q3. Is the proposed method better than existent matrix factorization related approaches?

Experiment Setup. To answer these questions, we do experiments on two embedding-enabled tasks: multilabel classification [36] and link prediction [16]. We uniformly set the embedding dimension to 120 for all the experiments.

Multilabel classification aims to predict the correct node labels. To be specific, we randomly sample a portion of the labeled nodes for training, with the rest for testing. Then, we use the learned embeddings and the corresponding labels of the training nodes to train a one-vs-all logistic regression (LR) classifier. Feeding the embeddings of the testing nodes to the classifier we predict their labels, which will be compared to the true labels for evaluation. We repeat this procedure 10 times and evaluate the performance in terms of the Macro-F1 and Micro-F1 scores. Due to space limitation, we report only Micro-F1, since we experience similar behaviors with Macro-F1.

In the link prediction task, we are given a network $\mathcal{G}'$ with 50% of edges removed from the original network $\mathcal{G}$ and we aim to predict the missing edges (*i.e.* the 50% removed edges). Specifically, based on the node embeddings learned from $\mathcal{G}'$, we generate edge embeddings for pairs of nodes using the Hadamard operator[1]:

$$[\boldsymbol{e_{ij}}]_t = [\boldsymbol{e_i}]_t \cdot [\boldsymbol{e_j}]_t, \tag{37}$$

where $t \in 1, \cdots, d$ denotes the subscript of the t-th element of an embedding. We label an edge embedding as positive if the corresponding edge exists in $\mathcal{G}'$ and negative otherwise. Then, we train a binary LR classifier using all of the edge embeddings that have positive labels and the same amount of randomly sampled edge embeddings that have negative labels. After that, feeding an edge embedding to the LR classifier we can calculate the existence probability of the corresponding edge and do link prediction. We evaluate the performance based on the probabilities of the missing edges and non-existent edges (*i.e.* the edges that do not exist in $\mathcal{G}$) in terms of the Area Under the Curve (AUC).

Datasets. We use ten real-world network datasets, which come from various domains and are commonly used by other researchers. A brief description of these datasets follows.

- BrazilAir [39], EuropeAir [39], USAir [39]: The air-traffic networks of Brazil, Europe, and the USA, respectively. The nodes indicate airports and the edges denote the existence of commercial flights. The labels represent the capacity levels of the airports.
- Cora [58], Citeseer [22], DBLP [41]: Paper citation networks. The labels represent the topics of the papers.
- WikiPage [49]: A network of webpages in Wikipedia, with edges indicating hyperlinks. The labels represent the topic categories of the webpages.
- WikiWord [16]: A co-occurrence network of the words appearing in Wikipedia. The labels represent the part-of-speech tags inferred using the Stanford POS-Tagger.
- PPI [16]: A part of the protein-protein interactions network for Homo Sapiens. The labels represent the biological states.

[1]We omit several element-wise operators and only use the Hadamard operator to generate the edge embeddings. One reason is that all of the methods considered here express the node similarity in a (varied) inner product form that conforms to the Hadamard operator. Another reason is that [16] shows that the Hadamard operator is superior to the others when used with node2vec and DeepWalk.

Table 2: Multilabel classification results for different similarities in terms of Micro-F1 scores.

Method	EuropeAir					USAir				
	10%	30%	50%	70%	90%	10%	30%	50%	70%	90%
GRA	0.3666	0.4480	0.4668	0.4925	0.5550	0.5139	0.5847	0.6071	0.6336	0.6597
Katz	0.3844	0.4599	0.4985	0.5042	0.5625	0.5020	0.5805	0.5876	0.5949	0.5966
SimRank	0.3014	0.3670	0.3895	0.4125	0.4875	0.5262	0.5808	0.6019	0.6347	0.6366
LHN	0.3819	0.3867	0.3944	0.4120	0.4325	0.4691	0.4917	0.4944	0.4983	0.5118
GRA-β	0.5384 (46.9%)	0.5695 (27.1%)	0.5945 (27.3%)	0.6000 (21.8%)	0.6725 (21.2%)	0.6360 (23.8%)	0.6793 (16.2%)	0.7003 (15.4%)	0.7151 (12.9%)	0.7387 (12.0%)
Katz-β	0.5209 (35.5%)	0.5495 (19.5%)	0.5874 (17.8%)	0.5780 (15.0%)	0.6700 (19.1%)	0.6044 (20.4%)	0.6483 (11.7%)	0.6659 (13.3%)	0.6748 (13.4%)	0.7193 (20.6%)
Method	**Cora**					**CiteSeer**				
	10%	30%	50%	70%	90%	10%	30%	50%	70%	90%
GRA	0.7788	0.8197	0.8352	0.8442	0.8638	0.5622	0.6040	0.6184	0.6272	0.6245
Katz	0.7225	0.7859	0.8036	0.8121	0.8284	0.5018	0.5458	0.5585	0.5706	0.5743
SimRank	0.7672	0.8036	0.8169	0.8241	0.8376	0.5538	0.5826	0.5905	0.6014	0.6046
LHN	0.6012	0.6611	0.6749	0.6819	0.6996	0.3758	0.4207	0.4380	0.4439	0.4617
GRA-β	0.7846 (0.7%)	0.8236 (0.6%)	0.8418 (0.8%)	0.8525 (1.0%)	0.8731 (1.1%)	0.5688 (1.2%)	0.6204 (2.7%)	0.6362 (2.9%)	0.6508 (3.8%)	0.6610 (5.8%)
Katz-β	0.7321 (1.3%)	0.7964 (1.3%)	0.8096 (0.8%)	0.8221 (1.2%)	0.8424 (1.7%)	0.5231 (4.2%)	0.5676 (4.0%)	0.5796 (3.8%)	0.5909 (3.6%)	0.6092 (6.1%)

- Flickr [43]: A network for the contacts between users in Flickr. The labels represent the interest groups of the users.

We remove self-loop edges and transform bi-directional edges to undirected edges for each network. The statistics of the datasets after pre-processing are summarized in Table 1.

Comparing with Different Similarities. We compare the embeddings generated by different similarities (setting $\beta = 0$). In addition to Katz, we consider SimRank [21] and LHN [23] similarities as baselines. Both SimRank and LHN are based on the concept that two nodes are similar if they are connected to similar nodes. All of the similarities here are global measures and widely used. Table 2 reports the multilabel classification results in four of the networks[2]. GRA achieves the best overall performance. We attribute this to the proper definition of GRA similarity, which depresses the contribution of the paths that contain high-degree intermediate nodes.

Improve Embeddings by Tuning β. Table 2 also compares the multilabel classification results with and without tuning β. GRA-β and Katz-β indicate the results by tuning β based on 10% ground truth validation data in a semi-supervised fashion for each network and each task (This is in the same way as node2vec [16] for tuning the in-out and return hyperparameters). The figures in the parentheses show the relative improvement after tuning β. We can see that by tuning β we achieve a significant performance gain. For example, the gain for GRA-β is as high as 46.9% in the EuropeAir network when the prediction is made based on 10% labeled nodes. The tuning strategy does not only work for GRA similarity but also other similarities such as Katz. This is because β is closely related to the parameter ρ that balances the two sub-objectives for bringing together the embeddings of similar nodes and separating the embeddings of distant nodes.

Comparing with Existent Approaches. Next, we compare GRA-β with DeepWalk [36], node2vec [16], HOPE [35], GraRep [4], and LINE [42], which are representatives of matrix factorization related approaches.

The parameter settings for these approaches are the same as the original literature. Specifically, for DeepWalk and node2vec, we set the window size to 10, the walk length to 80, and the number of walks per node to 10. For HOPE, we set the decay rate to 0.95 divided by the spectral radius of $\mathbf{A}$. For LINE, we set the number of negative samples to 5. For GraRep, we set the maximum transition step to 6. Lastly, for node2vec, we obtain the best in-out and return hyperparameters based on a grid search over $\{0.25, 0.50, 1, 2, 4\}$.

Figure 2 displays the results for multilabel classification. Table 3 lists the results for link prediction[3]. GRA-β demonstrate the best performance over the baselines. For the multilabel classification task, it is markedly superior to the others in eight out of the ten networks. In particular, it outperforms the baselines by a considerable margin in networks such as BrazilAir, EuropeAir, USAir, Cora, Citeseer, WikiPage, and PPI. For the link prediction task, it obtains the best

[2]Because SimRank and LHN are computationally expensive, the results are limited to small and medium networks.

[3]The results for GraRep in the Flickr network are not reported due to the scalability issue.

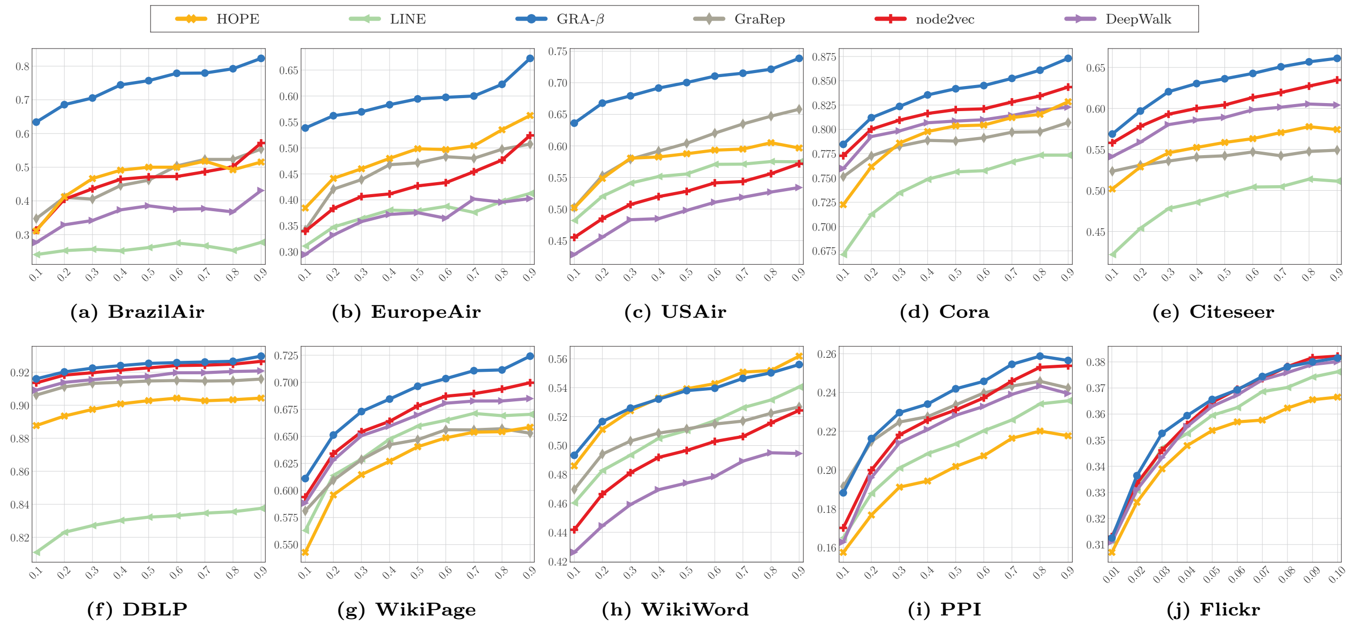

Figure 2: Multilabel classification results for different methods. The x-axis represents the ratio of nodes with known labels. The y-axis represents the Micro-F1 scores.

scores in all of the networks. In particular, it outperforms the opponents by up to 13.8%, 25.9%, 47.1%, and 37.7% in Cora, Citeseer, WikiWord, and PPI networks, respectively.

6 RELATED WORK

The study of network embedding dates back to the early 2000s. In this period, researchers attempted to keep connected nodes closer to each other in the embedding space and developed methods such as LLE [40], IsoMap[46], and Laplacian Eigenmap [2]. In the past decade, thanks to a few pioneer works such as SocDim [43, 45], DeepWalk [36], and LINE [42], this study has attracted a great deal of interest. Researchers have proposed various methods such as matrix factorization [4, 35, 58] and deep neural networks [5, 7, 12, 17, 22, 51, 52, 56]. The topics are also varied, including semi-supervised network embedding [22, 49, 51, 59], community preserving embedding [6, 10, 55], network reconstruction based on embedding [28, 35, 51], and embedding in different types of networks, such as the heterogeneous networks [7, 13, 20, 41, 53, 61], the multi-relational networks [33, 38], the signed networks [9, 53, 54], the dynamic networks [25, 32, 63], the scale-free networks [14], the hyper-networks [50], and the attributed networks [19, 25, 48, 54, 57].

Our work is closely related to similarity matrix factorization. Previous studies have shown the effectiveness of factorizing different matrices such as the adjacency matrix [1], the high-order adjacency matrix [58], modularity matrix [43], graph Laplacians [37, 45], and Katz similarity matrix [35]. These matrices can more or less be viewed as similarity matrices. However, none of the authors studied why their method works, except Hamilton *et al.*, who recently explained the effectiveness of similarity matrix factorization based on a unified encoder-decoder framework [18].

Researchers have shown that DeepWalk [36] is related to matrix factorization and derived the closed form of the matrix [49, 57]. Further, Qiu et al. proved that some other skip-gram based methods (DeepWalk [36], LINE [42], PTE [41], and node2vec [16]) are also equivalent to matrix factorization [37]. However, there are significant differences between their work and ours. Firstly, their work only involves the skip-gram based methods, while our objective function provides a more general framework that unifies several other methods such as HOPE, Caughy Graph Embedding, GraRep, and Eigenmap. Secondly, the proofs are totally different. They separately derive the matrices that are being factorized, while we do proofs from another angle that is the relation between the skip-gram models and the similarity matrices. Thirdly, the aims are different. They focus on what is the exact mathematical form of the matrices that DeepWalk, LINE, PTE, and node2vec are factorizing, while we center on how to improve the similarity matrix factorization method.

Besides, Chen et al. introduced a GEM-D [8] framework and Hamilton et al. put forward an encoder-decoder [18] framework for network embedding. The two frameworks are based on a high-level point of view, by decomposing a method into several components. With appropriate choice of each component, many methods including similarity matrix factorization can be unified. On the other hand, the framework proposed in our paper is based on a more specific point of view, by designing an objective that involves node similarities. With appropriate choice of the similarity function

Table 3: Link prediction results for different methods in terms of AUC scores.

Dataset	HOPE	GRA	GRA-β	GraRep	node2vec	DeepWalk	LINE
BrazilAir	0.8472	0.8977	**0.9117**	0.8555	0.7505	0.7025	0.5215
EuropeAir	0.8895	0.9001	**0.9157**	0.9083	0.7387	0.7004	0.6905
USAir	0.9366	0.9527	**0.9621**	0.9434	0.8295	0.8045	0.8372
Cora	0.7018	0.7294	**0.7707**	0.6775	0.7372	0.7306	0.6779
Citeseer	0.6601	0.6255	**0.7047**	0.5597	0.6315	0.6130	0.5712
DBLP	0.9077	0.9237	**0.9311**	0.9211	0.9227	0.9176	0.8680
WikiPage	0.8839	0.9066	**0.9180**	0.8836	0.8568	0.8555	0.8495
WikiWord	0.8839	0.8036	**0.9127**	0.8910	0.6690	0.6205	0.6433
PPI	0.8635	0.8704	**0.9024**	0.8546	0.6796	0.6554	0.6980
Flickr	0.9286	0.9337	**0.9563**	——	0.8686	0.8620	0.8455

and adjusting function, our objective also connects several methods.

Similarity information has been utilized for network embedding. struc2vec [39] measures node similarities at different scales and obtains embeddings from structural identity. SNS [31] uses both neighbor information and local subgraphs similarity to learn embeddings. AA$^+$Emb [27] employs node similarities instead of random walk simulation. VERSE [47] derive embeddings by sampling similarity information. However, these methods are not related to matrix factorization.

Another relevant work is [24], which is the first to demonstrate that the word embedding method based on the skip-gram model [34] implicitly factorizes a shifted word-context mutual information matrix. However, the shift number, which indicates the number of negative samples, is a global constant, and thus, it is not tuned for improving the embeddings.

To the best of our knowledge, we are the first to 1) show the general form of factorizing a shifted similarity matrix for network embedding; 2) prove that this general form is equivalent to maximizing two objectives, one for bringing together the embeddings of similar nodes and the other for separating the embeddings of distant nodes; 3) improve the embeddings by tuning the shift number, which is related to a parameter that balances the two objectives.

7 CONCLUSIONS

We proposed a general view that demonstrates the relationship between network embedding and matrix factorization. It is general because the scope is not only limited to skip-gram based approaches but also applicable to a broad range of methods that factorize various matrices. We also proposed a network embedding method based on factorizing GRA similarity matrix and a parameter tuning strategy. Experiments showed that our method significantly outperforms the state-of-the-art matrix factorization approaches.

The proposed method can be extended to different types of networks, such as the directed networks, the heterogeneous networks, and the uncertain networks. The key point is to define proper similarity measures. Take the directed network as an example. If we choose a directed similarity measure $s^v_{i\to j}$, our definitions (1)-(4) still hold, and thus we can factorize a shifted asymmetric matrix for embedding. We will extend our method to these networks in the future.

ACKNOWLEDGMENTS

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO). Tsuyoshi Murata would like to thank JSPS Grant-in-Aid for Scientific Research(B) (Grant Number 17H01785) and JST CREST (Grant Number JPMJCR1687).

REFERENCES

[1] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. 2013. Distributed large-scale natural graph factorization. In *Proceedings of WWW*. 37–48.
[2] Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*. Vol. 14. 585–591.
[3] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: problems, techniques and applications. *IEEE Transactions on Knowledge and Data Engineering* 30, 9 (2018), 1616–1637.
[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning graph representations with global structural information. In *Proceedings of CIKM*. 891–900.
[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *Proceedings of AAAI*. 1145–1152.
[6] Sandro Cavallari, Vincent W. Zheng, Hongyun Cai, Kevin ChenChuan Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *Proceedings of CIKM*. 377–386.
[7] 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 KDD*. 119–128.
[8] Siheng Chen, Sufeng Niu, Leman Akoglu, Jelena Kovačević, and Christos Faloutsos. 2017. Fast, warped graph embedding: Unifying framework and one-click algorithm. *arXiv preprint arXiv:1702.05764* (2017).
[9] Kewei Cheng, Jundong Li, and Huan Liu. 2017. Unsupervised feature selection in signed social networks. In *Proceedings of KDD*. 777–786.
[10] Jun Jin Choong, Xin Liu, and Tsuyoshi Murata. 2018. Learning community structure with variational autoencoder. In *Proceedings of ICDM*. 69–78.
[11] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2018. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering* (2018).
[12] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang. 2018. Adversarial Network Embedding. In *Proceedingss of AAAI*. 2167–2174.
[13] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of KDD*. 135–144.

[14] Rui Feng, Yang Yang, Wenjie Hu, Fei Wu, and Yueting Zhuang. 2018. Representation learning for scale-free networks. In *Proceedingss of AAAI*. 282–289.

[15] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems* 151 (2018), 78–94.

[16] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of KDD*. 855–864.

[17] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of NIPS*. 1025–1035.

[18] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation learning on graphs: methods and applications. *IEEE Data Engineering Bulletin* 40 (2017), 52–74.

[19] Xiao Huang, Jundong Li, and Xia Hu. 2017. Label informed attributed network embedding. In *Proceedings of WSDM*. 731–739.

[20] Yann Jacob, Ludovic Denoyer, and Patrick Gallinari. 2014. Learning latent representations of nodes for classifying in heterogeneous social networks. In *Proceedings of WSDM*. 373–382.

[21] Glen Jeh and Jennifer Widom. 2002. SimRank: a measure of structural-context similarity. In *Proceedings of KDD*. 538–543.

[22] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*.

[23] Elizabeth A Leicht, Petter Holme, and M. E. J. Newman. 2006. Vertex similarity in networks. *Phys. Rev. E* 73, 2 (2006), 026120.

[24] Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Proceedings of NIPS*. 2177–2185.

[25] Jundong Li, Harsh Dani, Xia Hu, Jiliang Tang, Yi Chang, and Huan Liu. 2017. Attributed network embedding for learning in a dynamic environment. In *Proceedings of CIKM*. 387–396.

[26] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology* 58, 7 (2007), 1019–1031.

[27] Xin Liu, Natthawut Kertkeidkachorn, Tsuyoshi Murata, Kyoung-Sook Kim, Julien Leblay, and Steven Lynden. 2018. Network embedding based on a quasi-local similarity measure. In *Proceedings of PRICAI*. 429–440.

[28] Xin Liu, Tsuyoshi Murata, and Kyoung-Sook Kim. 2018. Measuring graph reconstruction precisions—how well do embeddings preserve the graph proximity structure?. In *Proceedings of WIMS*. 25:1–4.

[29] L. Lü and T. Zhou. 2011. Link prediction in complex networks: a survey. *Physica A* 390 (2011), 1150–1170.

[30] Dijun Luo, Chris Ding, Feiping Nie, and Heng Huang. 2011. Cauchy graph embedding. In *Proceedings of ICML*. 553–560.

[31] Tianshu Lyu, Yuan Zhang, and Yan Zhang. 2017. Enhancing the network embedding quality with structural similarity. In *Proceedings of CIKM*. 147–156.

[32] Jianxin Ma, Peng Cui, and Wenwu Zhu. 2018. DepthLGP: learning embeddings of out-of-sample nodes in dynamic networks. In *Proceedings of AAAI*. 370–377.

[33] Yao Ma, Zhaochun Ren, Ziheng Jiang, Jiliang Tang, and Dawei Yin. 2018. Multi-dimensional network embedding with hierarchical structure. In *Proceedingss of WSDM*. 387–395.

[34] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*. 3111–3119.

[35] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of KDD*. 1105–1114.

[36] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of KDD*. 701–710.

[37] 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 WSDM*. 459–467.

[38] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. 2017. An attention-based collaboration framework for multi-view network representation learning. In *Proceedings of CIKM*. 1767–1776.

[39] Leonardo F. R. Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of KDD*. 385–394.

[40] Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 5500 (2000), 2323–2326.

[41] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of KDD*. 1165–1174.

[42] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of WWW*. 1067–1077.

[43] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *Proceedings of KDD*. 817–826.

[44] Lei Tang and Huan Liu. 2009. Scalable learning of collective behavior based on sparse social dimensions. In *Proceedings of CIKM*. 1107–1116.

[45] Lei Tang and Huan Liu. 2011. Leveraging social media networks for classification. *Data Min. Knowl. Discov.* 23, 3 (2011), 447–478.

[46] Joshua B Tenenbaum, Vin De Silva, and John C Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 5500 (2000), 2319–2323.

[47] Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. VERSE: versatile graph embeddings from similarity measures. In *Proceedings of WWW*. 539–548.

[48] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. CANE: Context-aware network embedding for relation modeling. In *Proceedings of ACL*. 1722–1731.

[49] Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. 2016. Max-margin DeepWalk: discriminative learning of network representation. In *Proceedings of IJCAI*. 3889–3895.

[50] Ke Tu, Peng Cui, Xiao Wang, Fei Wang, and Wenwu Zhu. 2018. Structural deep embedding for hyper-networks. In *Proceedingss of AAAI*. 426–433.

[51] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural deep network embedding. In *Proceedings of KDD*. 1225–1234.

[52] Hongwei Wang, Jia Wang, Jialin Wang, MIAO ZHAO, Weinan Zhang, Fuzheng Zhang, Xie Xing, and Minyi Guo. 2018. GraphGAN: graph representation learning with generative adversarial nets. In *Proceedings of AAAI*. 2508–2515.

[53] Hongwei Wang, Fuzheng Zhang, Min Hou, Xing Xie, Minyi Guo, and Qi Liu. 2018. SHINE: Signed Heterogeneous Information Network Embedding for Sentiment Link Prediction. In *Proceedings of WSDM*. 592–600.

[54] Suhang Wang, Charu Aggarwal, Jiliang Tang, and Huan Liu. 2017. Attributed signed network embedding. In *Proceedings of CIKM*. 137–146.

[55] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding. In *Proceedings of AAAI*. 203–209.

[56] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and Philip S Yu. 2018. On exploring semantic meanings of links for embedding social networks. In *Proceedings of WWW*. 479–488.

[57] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Chang. 2015. Network representation learning with rich text information. In *Proceedings of IJCAI*. 2111–2117.

[58] Cheng Yang, Maosong Sun, Zhiyuan Liu, and Cunchao Tu. 2017. Fast network embedding enhancement via high order proximity approximation. In *Proceedings of IJCAI*. 3894–3900.

[59] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-Supervised learning with graph embeddings. In *Proceedings of ICML*. 40–48.

[60] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2018. Network representation learning: a survey. *IEEE Transactions on Big Data* (2018).

[61] Jiawei Zhang, Congying Xia, Chenwei Zhang, Limeng Cui, Yanjie Fu, and Philip S Yu. 2017. BL-MNE: emerging heterogeneous social network embedding through broad learning with aligned autoencoder. In *Proceedingss of ICDM*. 605–614.

[62] Yuan Zhang, Tianshu Lyu, and Yan Zhang. 2018. COSINE: community-preserving social network embedding from information diffusion cascades. In *Proceedings of AAAI*. 2620–2627.

[63] Ziwei Zhang, Peng Cui, Jian Pei, Xiao Wang, and Wenwu Zhu. 2018. TIMERS: error-bounded SVD restart on dynamic networks. In *Proceedings of AAAI*. 224–231.

[64] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. 2009. Predicting missing links via local information. *Eur. Phys. J. B* 71, 4 (2009), 623–630.