Big Data Mining and Analytics

Volume 2 | Issue 1 Article 4

2019

A Brief Review of Network Embedding

Yaojing Wang
the State Key Laboratory for Novel Software Technology Nanjing University, Nanjing 210023, China.

Yuan Yao
the State Key Laboratory for Novel Software Technology Nanjing University, Nanjing 210023, China.

Hanghang Tong
the School of Computing, Informatics and Decision Systems Engineering, Arizona State University, AZ 85281, USA.

Feng Xu
the State Key Laboratory for Novel Software Technology Nanjing University, Nanjing 210023, China.

Jian Lu
the State Key Laboratory for Novel Software Technology Nanjing University, Nanjing 210023, China.

BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 04/06 pp35–47
Volume 2, Number 1, March 2019
DOI: 10.26599/BDMA.2018.9020029

A Brief Review of Network Embedding

Yaojing Wang, Yuan Yao*, Hanghang Tong, Feng Xu, and Jian Lu

Abstract: Learning the representations of nodes in a network can benefit various analysis tasks such as node classification, link prediction, clustering, and anomaly detection. Such a representation learning problem is referred to as network embedding, and it has attracted significant attention in recent years. In this article, we briefly review the existing network embedding methods by two taxonomies. The technical taxonomy focuses on the specific techniques used and divides the existing network embedding methods into two stages, i.e., context construction and objective design. The non-technical taxonomy focuses on the problem setting aspect and categorizes existing work based on whether to preserve special network properties, to consider special network types, or to incorporate additional inputs. Finally, we summarize the main findings based on the two taxonomies, analyze their usefulness, and discuss future directions in this area.

Key words: network embedding; node representations; context construction

1 Introduction

Network embedding, which learns the node representations of a given network, is essential for various analysis tasks including node classification[1], link prediction[2], clustering[3], and anomaly detection[4]. The key idea of network embedding is to obtain the node presentations by preserving certain network properties. Comparing with the traditional feature extraction methods which require manual design of node features (e.g., in-degree and out-degree), one of the advantages of network embedding methods is automatically learning of these features[5].

In this article, we present two taxonomies to categorize existing network embedding methods. The first taxonomy focuses on the technical aspect, i.e., the specific techniques used to learn the node embeddings. Specially, we divide existing methods into two stages, i.e., context construction and objective design. The context construction stage aims to derive a context network from the original network, where the context information (e.g., the properties to preserve) for each node is encoded in the context network. In particular, we divide the widely-used context networks into three classes: original network, local neighborhood (which incorporates indirect nearest neighbors as additional context nodes), and walking network (which applies random walks on the original network). In the objective design stage, an objective function is designed over the constructed context to obtain the final node representations. We divide the existing objectives into reconstruction-oriented objective (which aims to reconstruct the context network), discrimination-oriented objective (which aims to distinguish the context nodes from the non-context nodes), and ranking-oriented objective (which aims to preserve the relative orders of edges).

The second taxonomy focuses on the non-technical aspect, and categorizes existing work based on whether they preserve certain special network properties, consider special network types, or incorporate

• Yaojing Wang, Yuan Yao, Feng Xu, and Jian Lu are with the State Key Laboratory for Novel Software Technology Nanjing University, Nanjing 210023, China. E-mail: wyj@smail.nju.edu.cn; {y.yao, xf, lj}@nju.edu.cn.
• Hanghang Tong is with the School of Computing, Informatics and Decision Systems Engineering, Arizona State University, AZ 85281, USA. E-mail: hanghang.tong@asu.edu.
∗ To whom correspondence should be addressed.
Manuscript received: 2018-05-10; accepted: 2018-05-25

additional inputs. Existing works commonly preserve the local neighborhood structure. By contrast, special global properties such as node roles (e.g., structural identity[6]) and global ranking[7] have also been considered. For network types, although numerous network embedding methods are designed for general networks, a few special treatments can be applied to special network types such as directed networks, signed networks, heterogeneous networks, and dynamic networks. In addition to the network topology, various additional inputs such as node attributes, community structures/group labels, and supervision labels can be utilized. Figure 1 summarizes the proposed two taxonomies.

For each taxonomy, we first review and categorize the existing network embedding methods accordingly. Then, we describe several representative methods in each category. Finally, we summarize the main findings and discuss possible future directions. We believe that the proposed taxonomies can benefit the research on network embedding in the following aspects.

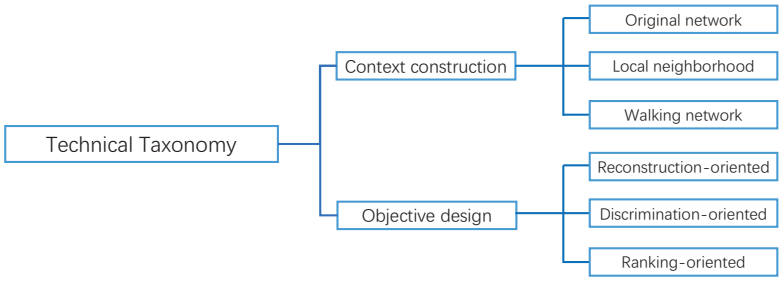
• Compared with the existing surveys (see the related work section for details), we provide a new technical taxonomy to categorize the techniques used for network embedding. The proposed technical taxonomy consists of two orthogonal dimensions which we can use as basis to develop new network embedding methods by different combinations of these dimensions. Further, we can systematically evaluate these combinations to understand the advantages and disadvantages of different choices in each dimension.

• The proposed non-technical taxonomy mainly considers the problem settings. That taxonomy can be first used to find the research gaps to fulfill (e.g., the problem settings that have not been considered before). Second, when researchers develop a new network embedding method, they can find suitable competitors by comparing their problem settings with those of the existing methods.

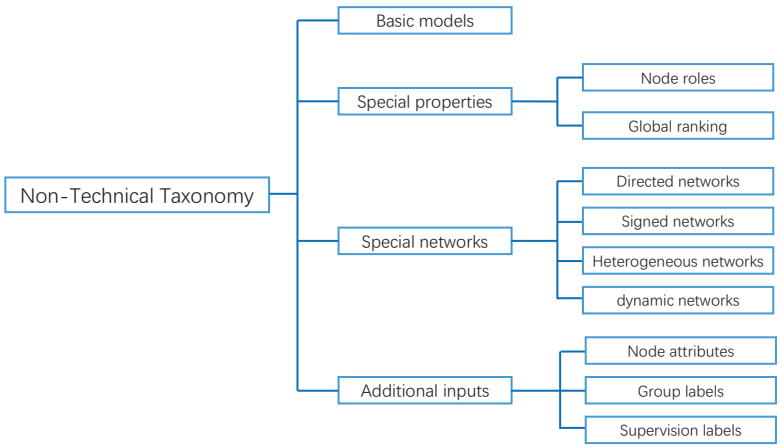
The rest of the article is organized as follows. Section 2 reviews the related network embedding surveys. Sections 3 and 4 present the proposed taxonomies. Section 5 concludes the paper with discussions of future directions.

2 Related Work

Several existing surveys center on network embedding. In the technical aspect, Hamilton et al.[8] divided existing network embedding methods into factorization-based methods, random walk methods, and other



(a) Technical taxonomy



(b) Non-technical taxonomy

Fig. 1 The proposed two taxonomies.

generalized methods (e.g., graph convolutional networks[9]); Goyal and Ferrara[10] divided them into factorization methods, random walks, and deep learning methods; Yao et al.[5] divided existing methods into neural network models, factorization based models, and regularization based models. These surveys focus on the specific techniques used during the network embedding process, whereas the non-technical aspect is ignored.

Considering the non-technical aspect of network embedding, Zhang et al.[11] categorized network embedding methods into unsupervised and semi-supervised methods. In each category, they further consider whether to incorporate content information in the embedding process. However, the used techniques are not well-categorized.

Other existing surveys consider both the technical and non-technical aspect. Cui et al.[12] categorized existing methods based on whether the methods consider side information or whether they are designed for specific tasks (in addition to structure preserving). With regard to the used techniques, similar to Goyal and Ferrara[10], they divided existing methods into matrix factorization, random walks, and deep neural networks. Cai et al.[13] also divided the existing network embedding methods based on the input (e.g., homogeneous networks v.s. heterogeneous networks). Considering the network embedding techniques, the authors divided the existing methods into matrix factorization, deep learning, and edge reconstruction. In contrast to the above two surveys, instead of dividing the existing methods based on the specific used techniques, we divide them into two orthogonal stages of context construction and objective design, where the second stage is similar to existing surveys but standing at a higher perspective. Additionally, we provide a wider non-technical taxonomy to cover different problem settings.

Less related to this work, some other surveys also exist. For example, Wang et al.[14] contributed to knowledge graph embedding only, Fu and Ma[15] focused on the traditional graph embedding methods (e.g., Isomap[16] and local linear embedding[17]).

3 Technical Taxonomy

In this section, we first present the proposed technical taxonomy of existing network embedding methods, and then discuss some representative methods under this taxonomy. The discussion is followed by a brief summary.

3.1 Taxonomy

We categorize existing methods based on two orthogonal stages of context construction and objective design. Table 1 shows the categorization results. Specifically, we mainly categorize the unsupervised or semi-supervised network embedding methods. The supervised methods designed for specific tasks such as Refs. [58–60] are not covered.

As presented in the table, the first stage constructs the context network where each node can be characterized by a set of context nodes. Such context contains the network properties that different network embedding methods aim to preserve. In literature, the existing

Table 1 A unified technical framework of network embedding methods.

	J1: reconstruction-oriented	J2: discrimination-oriented	J3: ranking-oriented
N1: original network	SVD, SDNE[18], AANE[23], LANE[24], DANE[29], SNEA[28], MVC-DNE[33]	LINE[19], HNE[20], PTE[25], CENE[26], EOE[30], MVE[31], IIRL[34]	BPR[21], EP[22], SiNE[27], SNEA[28], DynamicTriad[32]
N2: local neighborhood	SVD#, LLE[17], Isomap[16], M-NMF[35]	LINE[19], PRUNE[7]	
N3: walking network	SVD##[36], GraRep[37], TADW[40], HOPE[41], DNGR[44], NetMF[45], URGE[48], UltimateWalk[49]	DeepWalk[38], APP[39], node2vec[42], GENE[43], TriDNR[46], Planetoid[47], metapath2vec[50], HIN2Vec[51], struc2vec[6], SNS[52], DP-Walker[53], MINES[54], ANE[55], SIDE[56], GraphSAGE[57]	

methods mainly use the following classes of context networks.

• *N1: original network*. The first class directly uses the original network as the context network. That is, the connected nodes fully define the context for each node.

• *N2: local neighborhood*. Local neighborhood incorporates both direct and several indirect neighbors (i.e., neighbors of neighbors) as context nodes. Earlier network embedding methods such as Isomap[16] and LLE[17] also belong to this category.

• *N3: walking network*. The walking network can be constructed by applying random walks on the original network. The main difference between walking network and local neighborhood is that the former usually incorporates neighbors several steps away.

Objective design is the second stage in the table: it is used to preserve the properties of the constructed context matrix. In this article, we divide existing objectives into *reconstruction-oriented*, *discrimination-oriented*, and *ranking-oriented* objectives. When discussing the objective design, we only consider the objective function of the network structure. For example, if two objective functions are separately used on the network structure and node attributes (we will discuss the details about node attributes in the next section) in a method, we categorize this method based on the first objective.

• *J1: reconstruction-oriented objective*. Reconstruction-oriented objectives aim to reconstruct the edges of the context network. Usually, different types of matrix low-rank approximations are used to reconstruct the networks.

• *J2: discrimination-oriented objective*. The discrimination-oriented objective aims to distinguish the context nodes from the non-context ones.

• *J3: ranking-oriented objective*. The ranking-oriented objective aims to optimize the relative orders of a set of edges. For example, a ranking-oriented objective is used in EP[22] to ensure that the connected nodes are closer than the unconnected ones.

In the following, as the construction of the context network is relatively straightforward, we mainly discuss the representative methods in terms of the three types of objectives.

3.2 Representatives of J1

For the reconstruction-oriented objective, we further divide existing work into two classes. The first class uses different types of matrix low-rank approximations such as SVD, Probabilistic Matrix Factorization (PMF), and Non-negative Matrix Factorization (NMF). For example, SVD##[36] and NetMF[45] directly use SVD, TADW[40] and HOPE[41] apply a variant of SVD by keeping the Frobenius norm, M-NMF[35] further adds non-negativity constraints, etc. Other examples include Refs. [23, 24, 28, 29, 37, 48, 49, 61, 62].

The basic idea of these methods is to first construct a node-context matrix, and then use the low-rank approximation matrices as the embeddings. Consider GraRep[37] as an example. As shown in Fig. 2, GraRep consists of three steps. Step one, GraRep computes all the possible paths between nodes via k-hop multiplications of the normalized adjacency matrix. Next, for each k value, GraRep applies SVD on the shifted Pointwise Mutual Information (PMI) matrix[36]; the PMI matrix is defined below:

$$\mathrm{PMI}(i, j) = \log\left(\frac{\#(i, j) \cdot |D|}{\#(i) \cdot \#(j)}\right) \tag{1}$$

where $\#(i, j)$ denotes the number of occurrences that nodes i and j are in the same context, $\#(i)$ ($\#(j)$) refers to the number of occurrences of node i (j), and $|D| = \sum_i \sum_j \#(i, j)$. Finally, GraRep concatenates the resulting embeddings from all k values.

The second class of reconstruction-oriented objectives adopt autoencoders. The basic idea here is to first map the context matrix into embeddings, and then use the embeddings to reconstruct the context matrix. Examples include SDNE[18], DNGR[44], and MVC-DNE[33]. For example, SDNE inputs the neighbor vector of each node into the autoencoder, uses the hidden state as the embedding, and further regulates the embeddings of connected nodes to be close to each other.

3.3 Representatives of J2

For the discrimination-oriented objectives, the key idea is to learn the node embeddings by designing the objectives to distinguish between the context and non-context nodes. A recent trend involves the application of the skip-gram model in word2vec[63, 64] to learn the node embeddings. Another common practice uses the logistic regression model for classification purpose. In fact, nearly all the existing methods that fall into this category use either the skip-gram model (or its variant) or the logistic-like objective. Examples include Refs. [6, 7, 19, 20, 25, 26, 30, 31, 34, 38, 39, 42, 43, 46, 47, 50–57].

We next briefly explain the basic assumption of the skip-gram model. Given a node u in a network G =

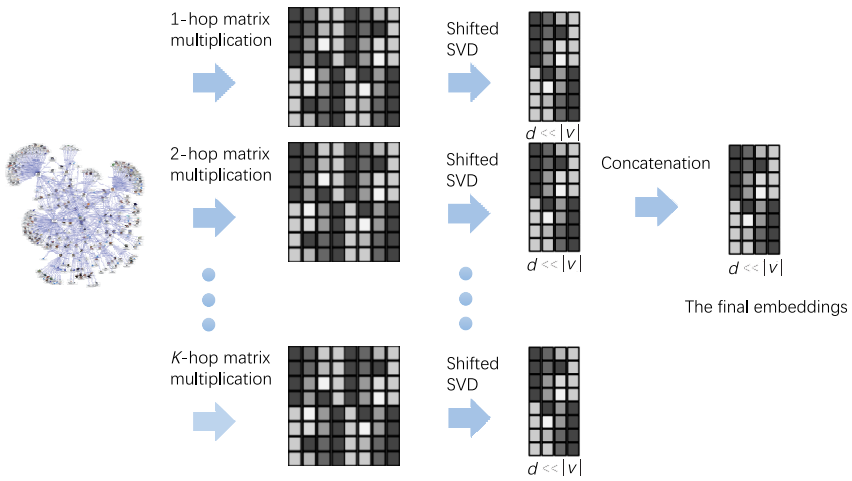


Fig. 2 The illustration of GraRep[37].

(V, E), and its neighbors $N(u)$, the skip-gram model aims to maximize the following objective,

$$\sum_{u \in V} \log Pr(N(u)|f(u)) \tag{2}$$

where $f(u)$ denotes the embedding of node u. Then, by assuming a conditional independence, the above equation can be written as follows:

$$Pr(N(u)|f(u)) = \prod_{v \in N(u)} Pr(v|f(u)) \tag{3}$$

where the softmax function is usually used to estimate the conditional probability:

$$Pr(v|f(u)) = \frac{\exp(g(v) \cdot f(u))}{\sum_{v' \in V} \exp(g(v') \cdot f(u))} \tag{4}$$

Here, "$\cdot$" denotes the inner product, f stands for the embedding function for central nodes, and g refers to the embedding function for neighbor nodes. The denominator of the above equation is expensive to compute for large networks. Therefore, negative sampling techniques are commonly used to approximate the denominator. Based on the above model, we can learn the functions f and g which are used to map nodes to their embeddings.

Based on the skip-gram model, different methods use various extensions and modifications to further enhance the performance. For example, node2vec[42] combines DFS and BFS with normal random walk; DP-Walker[53] finds that edges can be formed by social rank (i.e., the richer get richer phenomenon), and modifies the walking process of DeepWalk to reflect the real proximity between nodes.

3.4 Representatives of J3

Existing ranking-oriented objectives mainly aim to optimize the relative orders of a pair of edges. Different from the discrimination-oriented objective which aims to identify the difference between a positive edge (connecting to a context node) and a negative edge (connecting to a non-context node), the ranking-oriented objective ensures that the proximity of the positive edge is larger than that of the negative edge.

Consider BPR[21] as an example. BPR is originally proposed for recommender systems. The intuition behind this method is that the user preference on an observed item should be higher than that on an unobserved item:

$$\max_{\boldsymbol{U},\boldsymbol{V}} \sum_{(u,i,j) \in \mathcal{D}} \ln \sigma(\hat{\boldsymbol{R}}(u,j) - \hat{\boldsymbol{R}}(u,i)) \tag{5}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ contain the learned embeddings for users and items, respectively, $\mathcal{D}$ contains the (u,i,j) triples with observed (u,i) from user u to item i and unobserved (u,j) from user u to item j, $\hat{\boldsymbol{R}}(u,i) = \boldsymbol{U}(u,:)\boldsymbol{V}(i,:)'$ is the estimated preference of user u on item i, and $\sigma(x) = \frac{1}{1+\mathrm{e}^{-x}}$. We ignore the regularization terms for brevity. Given the above formulation, when the estimated preference of an unobserved item (i.e., item j) is higher than that of an observed item (i.e., item i), a larger penalty is added on the objective function. We can then adapt the above BPR formulation for pairwise network embedding as follows:

$$\max_{\boldsymbol{U},\boldsymbol{V}} \sum_{(u,v,c) \in \mathcal{D}} \ln \sigma(\hat{\boldsymbol{S}}(u,v) - \hat{\boldsymbol{S}}(u,c)) \tag{6}$$

where the (u,v,c) triple consists of a positive edge (u,v) and a negative edge (u,c), and $\hat{\boldsymbol{S}}(u,v) = \boldsymbol{U}(u,:)\boldsymbol{V}(v,:)'$ is the estimated edge weight corresponding to the edge $\boldsymbol{S}(u,v)$ in the context network $\boldsymbol{S}$.

Several network embedding methods also follow

this pairwise objective. For example, SiNE[27] and SNEA[28] are proposed for signed networks, and they concentrate on the triad structure and propose a pairwise objective where the similarity of positively-linked nodes should be higher than that of negatively-linked nodes. Similarly, DynamicTriad[32] also works on the triad structure, and it uses such structure for dynamic network embedding. EP[22] proposes a message passing framework to propagate embeddings, and adopts the above pairwise ranking loss at the top layer. SNEA[28] appears in both the ranking-oriented column and the reconstruction-oriented column as it simultaneously uses two types of methods to embed the network.

3.5 Summary and discussion

Based on the information in Table 1, we can infer the following observations. First, the three types of context networks (especially the original network and the walking network) have already been widely used by the existing methods. This finding implies that no absolute conclusion decides which context network performs better, and the performance may depend on how the context network is combined with the objective design.

Second, most of the existing methods select the reconstruction-oriented and discrimination-oriented objectives. This condition probably results from the popularity of low-rank approximation model (i.e., reconstruction-oriented objective) and skip-gram model (i.e., discrimination-oriented objective). This observation also encourages us to consider other models such as ranking-oriented models in the future research. Additionally, as mentioned above, SNEA[28] appears in both the ranking-oriented column and the reconstruction-oriented column. Thus we may simultaneously use multiple context networks and multiple objectives to obtain high-quality node embeddings.

Third, considering the combinations of context network and objective design, the table still contains vacancies (e.g., walking network with ranking-oriented objective). Future efforts can be spared to explore these areas. We also observe that many of the existing methods are applicable to different context networks (e.g., SVD, SVD#, and SVD## are variants of SVD applied to the three types of context networks). Exploring such case is also an interesting future direction.

4 Non-Technical Taxonomy

In this section, we categorize the existing network embedding methods with a non-technical taxonomy. We first present the taxonomy and several representative methods in each category, and then discuss the observations from the categorization results.

4.1 Taxonomy

In our non-technical taxonomy, our categorization is built upon the basic problem setting, that is, learning the node embeddings for general networks via preserving the local structures, with only the network topology as input. Based on such basic models, we further consider the cases when special global network properties are preserved, special network types are embedded, or additional inputs are used.

- *Basic Models*. Basic network embedding models only utilize the original network as the input. Additionally, these models consider the case of general networks and basic neighborhood properties.
- *Considering Special Network Properties*. Several existing network embedding methods incorporate special network properties that go beyond the local neighborhood. Examples include node roles (e.g., structural identity[6] and graphlet similarity[52]) and global ranking[7].
- *Considering Special Network Types*. To handle different network types, special treatments can be applied. Existing work mainly considers four types of special networks including directed, signed, heterogeneous, and dynamic networks.
- *Considering Additional Inputs*. In addition to the input network topology, various additional inputs can be utilized to enhance the network embedding quality. Examples include node attributes, community structures/group labels, and supervision labels.

Table 2 summarizes the existing network embedding methods based on the above non-technical taxonomy. In the following, we mainly discuss the representative methods in each category. One embedding method may belong to multiple categories (e.g., considering special network types while introducing additional inputs), whereas another embedding method can consider multiple aspects in each category (e.g., considering signed and directed networks).

4.2 Basic models

We start with the basic models. Most basic models for network embedding are based on the skip-gram model

Table 2 Non-technical taxonomy of network embedding methods.

Method	Special network property	Special network type	Additional input
DeepWalk[38]	—	—	—
SVD##[36]	—	—	—
GraRep[37]	—	—	—
LINE[19]	—	—	—
PTE[25]	-	Heterogeneous networks	Node attributes, supervision labels
TADW[40]	—	—	Node attributes
HNE[20]	—	—	Node attributes
node2vec[42]	—	—	—
DNGR[44]	—	—	—
SDNE[18]	—	—	—
CENE[26]	—	—	Node attributes
HOPE[41]	-	Directed networks	—
GENE[43]	—	—	Group labels
TriDNR[46]	-	Heterogeneous networks	Node attributes, supervision labels
Planetoid[47]	—	—	Node attributes, supervision labels
AANE[23]	—	—	Node attributes
APP[39]	-	Directed networks	—
M-NMF[35]	—	—	Group labels
LANE[24]	—	—	Node attributes, supervision labels
struc2vec[6]	Structural identity	—	—
metapath2vec[50]	—	Heterogeneous networks	—
HIN2Vec[51]	—	Heterogeneous networks	—
SiNE[27]	—	Signed networks	—
EP[22]	—	—	Node attributes
DANE[29]	—	Dynamic networks	Node attributes
SNEA[28]	—	Signed networks	Node attributes
EOE[30]	—	Heterogeneous networks	—
MVE[31]	—	Heterogeneous networks	—
MVC-DNE[33]	—	—	Node attributes
PRUNE[7]	Global ranking	—	—
URGE[48]	—	Uncertain networks	—
UltimateWalk[49]	—	—	—
SNS[52]	Graphlet similarity	—	—
GCN[9]	—	—	Supervision labels
GraphSAGE[57]	—	—	Node attributes
DynamicTriad[32]	—	Dynamic networks	—
IIRL[34]	—	—	Node attributes
NetMF[45]	—	—	—
DP-Walker[53]	—	—	—
MINES[54]	—	Heterogeneous networks	Hierarchical structure
ANE[55]	—	—	—
DepthLGP[65]	—	Dynamic networks	—
SIDE[56]	—	Signed networks, directed networks	—

and the matrix factorization models. Typical examples include DeepWalk[38], LINE[19], node2vec[42], and DP-Walker[53] which are all built upon the skip-gram model; as well as SVD##[36], GraRep[37], UltimateWalk[49], and NetMF[45] which are built upon matrix factorization models.

In addition to the above types of basic models, other models have also been adopted. For example, SDNE[18] and DNGR[44] use autoencoders to reconstruct the context network; ANE[55] and GraphGAN[66] adopt adversarial networks by, e.g., simultaneously training a structure preserving component and an adversarial

component.

4.3 Considering special network properties

Although basic models typically consider the local neighborhood structure, other methods further incorporate the global structure into the embedding process. For example, struc2vec[6] considers the structural identities of nodes (e.g., hubs and authorities), and computes the proximity of structural identities; SNS[52] considers the graphlets[67] to compute the similarities between local subgraphs of two nodes. Different from the above methods, PRUNE[7] preserves the global ranking (e.g., PageRank value) in the model and develops a multi-task neural network structure; RaRE[68] considers a Bayesian perspective, and assumes that edges are formed by either proximity or the social rank, similar to the global ranking concept.

4.4 Considering special network types

4.4.1 Directed networks

Several researchers pay special attention to directed networks[39,41,56] as the proximity computation can encode the asymmetric information which cannot be encoded by methods working on general networks. For example, when computing the proximities between nodes, HOPE[41] handles high-order proximity measurements, such as Katz Index and Adamic-Adar in a matrix factorization framework; APP[39] and SIDE[56] are built upon the skip-gram model, and they primarily focus on how to generate the random walks that are suitable for directed networks.

4.4.2 Signed networks

Different from the normal unsigned networks (with only positive edges), signed networks contain both positive and negative edges. A key issue here is to deal with the negative edges. As an example, SiNE[27] concentrates on the triad structure where the similarity of positively-linked nodes should be higher than that of negatively-linked nodes. SNE[69] is another embedding method designed for signed networks, which adopt the log-bilinear model to linearly combine the embeddings of the context nodes for a given central node. Other examples include SNEA[28] which embeds signed networks with node attributes, and SIDE[56] which embeds signed and directed networks.

4.4.3 Heterogeneous networks

Heterogeneous networks contain nodes/edges of different types[20,30,31,46,50,51,54]. To tackle heterogeneity, one line of existing work is based on the concept of metapaths[70], which consists of multiple types of nodes or edges. For example, metapath2vec[50] defines metapaths and learns the embeddings in a manner similar to DeepWalk; HIN2Vec[51] defines metapaths and adopts a logistic classification method to learn the embeddings. Another line of existing work is based on the multi-view learning framework where each view deals with a certain type of edge. For example, given the multiple views of networks, MVE[31] learns the embeddings for each view and combines them via attention mechanism; Ma et al.[71] learnt the network embedding and multi-view clustering via tensor factorization.

In addition to the above two lines of work, other types of heterogeneous network embedding methods have also been proposed. For example, PTE[25] extends the LINE model to embed heterogeneous networks: LINE can be considered as an embedding method for homogeneous networks (e.g., word-word networks), whereas PTE further considers word-document networks and word-label networks. Similarly, TriDNR[46] exploits node-node networks, node-content correlation, and label-content correspondence. Different from PTE and TriDNR, Chang et al.[20] considered the heterogeneous network where each node may include different types of attributes; Xu et al.[30] considered the heterogeneous networks with two different but related networks connected by inter-network edges; MINES[54] also deals with heterogeneous networks with multiple relations between nodes.

4.4.4 Dynamic networks

Several researchers proposed embedding methods for dynamic networks where the network evolves over time. For example, DANE[29] focuses on attributed networks using eigen decomposition and learns the embeddings of new nodes using neural networks on latent factors to approximate the mapping from nodes to their embeddings; DynamicTriad[32] learns the embeddings via triadic closure process (e.g., if an open triad closes, then the terminal nodes of the new connection should be closer to each other). DepthLGP[65] assumes f to be the mapping from nodes to their embeddings and introduces a function h to map each node to its hidden state. Then, DepthLGP learns a neural network to transfer h to f. By doing so, DepthLGP can learn the embeddings of new nodes.

Although not explicitly mentioned in Table 2, Planetoid[47], MVC-DNE[33], GraphSAGE[57], and DyRep[72] can also potentially predict the embeddings of new nodes, as they develop inductive models for the embedding problem.

4.4.5 Others

In addition to the above network types, other interesting network types have also been studied. For example, Hu et al.[48] embedded uncertain networks where each edge exists with a probability; IGE[73] considers bipartite networks; Tu et al.[74] embedded hyper-networks where a hyperedge involves more than two nodes.

4.5 Considering additional inputs

4.5.1 Node attributes

The node attribute is the first additional input that has been widely used by existing network embedding methods. The first way to use the node attribute is to model it as a special type of node in a heterogenous network[25,46]. The second way is to learn the representations of node attributes via, e.g., neural networks or matrix factorization. These learned attributes can help to regulate the node embeddings. Examples include Refs. [20, 23, 24, 26, 28, 29]. The third way is to develop an inductive network embedding method which can directly map the node features to node embeddings. Examples include TADW[40], Planetoid[47], MVC-DNE[33], GraphSAGE[57], and DepthLGP[65]. Other methods include EP[22] which propagates node embeddings (derived from node attributes) along neighbors and updates the embeddings backwards, and IIRL[34] which learns the node embeddings and the link types together where links can be formed by structure homophily and attribute homophily.

Notably, we can derive the node features (e.g., node degrees) from the network topology, and use these features to substitute node attributes. Therefore, a number of network embedding methods such as Planetoid[47] and GraphSAGE[57] can also be used in the settings without available node attributes. In Table 2, we still put "node attributes" for these methods as they can handle both cases.

4.5.2 Group labels

Several researchers propose to incorporate the community structures or the group labels when such information is available[35,43,75]. For example, M-NMF[35] adopts the factorization based method while preserving the community structure of the network. M-NMF also collectively factorizes the node similarity matrix and the community indicator matrix and further adds an optimization term to maximize the modularity[76]. GENE[43] generalizes the DeepWalk model: DeepWalk generates the embedding of a given node from its neighbors, whereas GENE performs the same from the neighbors and the group embedding. ComE[75] simultaneously learns the community embedding and the node embedding.

4.5.3 Supervision labels

The supervision information is also used by existing network embedding methods[9,24,25,46,47,62]. For example, PTE[25] and TriNDR[46] incorporate the known labels as input when learning the embeddings, and use these embeddings to predict the unknown labels. MMDW[62] introduces the max-margin idea in Support Vector Machine (SVM) and jointly optimizes the max-margin based classifier and the network embedding formulation. GCN[9] introduces graph convolutional networks for network embedding and optimizes the cross entropy.

We mainly review the semi-supervised network embedding methods as their main goal is to learn the node representations. Several supervised methods also use network embedding as an intermediate step. For example, Zhang and Hasan[59] used pairwise ranking on the person-person network, person-document network, and doc-doc network to perform name disambiguation; SHINE[60] works on signed heterogeneous networks for sentiment link prediction.

4.5.4 Others

Other inputs have also been used. For example, MINES[54] incorporates hierarchical structures among nodes (e.g., items are organized by categories in e-commerce networks).

4.6 Summary and discussion

Overall, we can draw several observations from Table 2. First, the local neighborhood is mostly used by existing network embedding methods. A few exceptions may compute the similarity of nodes far away in the network. The lesson learned from these exceptions is as follows. Although the basic assumption of network embedding is to preserve the local neighborhood, we consider that the global structure may benefit certain types of downstream prediction tasks where the node labels are more dependent on their roles over the whole

network.

Second, recent works consider various special network types (e.g., heterogeneous networks and dynamic networks) and additional inputs (e.g., community structures and node attributes), as well as the combinations of these extensions. From this perspective, several future directions can be considered by trying different combinations.

Finally, other network embedding methods exist in addition to those in Tables 1 and 2. For example, Misra and Bhatia[77] proposed binary embedding which requires that the learned embeddings are binary values; HARP[78] proposes the strategy to learn node embeddings from smaller networks (which approximates the global structure) to larger ones and shows that this strategy improves the performance of existing methods such as DeepWalk and node2vec. In addition to learning the embeddings for nodes, other researchers proposed to learn the embeddings of communities/subgraphs[79,80]. These methods are beyond the scope of this article.

5 Conclusion

In this article, we have reviewed the existing network embedding methods with two taxonomies. Technical taxonomy divides the existing methods into context construction and objective design. Specifically, context network includes original network, local neighborhood, and walking network, whereas reconstruction-oriented, discrimination-oriented, and ranking-oriented objectives are included. On the other hand, non-technical taxonomy divides existing work based on whether to preserve special network properties, to consider special network types, or to incorporate additional inputs. Several findings and future directions are discussed based on the taxonomy results. Thus, this article may benefit the future network embedding research in terms of understanding the advantages and disadvantages of design choices and providing suitable competitors.

Several interesting future directions should be considered. First, although the features from manual feature extraction methods provide clear meanings, network embedding methods provide features with vague meanings. A future direction is to interpret/explain the learned embeddings. Second, although generating features in a dynamic setting has been studied by several researchers, developing an efficient and accurate dynamic embedding method remains a challenging problem. Third, with the fast development of network embedding research, studies should aim to develop a platform where the results of benchmark methods on benchmark data sets with benchmark tasks can be shared and compared.

References

[1] S. Bhagat, G. Cormode, and S. Muthukrishnan, Node classification in social networks, in *Social Network Data Analytics*, C. C. Aggarwal, ed. Boston, MA, USA: Springer, 2011, pp. 115–148.

[2] D. Liben-Nowell and J. Kleinberg, The link-prediction problem for social networks, *J. Am. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, 2007.

[3] S. E. Schaeffer, Graph clustering, *Comput. Sci. Rev.*, vol. 1, no. 1, pp. 27–64, 2007.

[4] L. Akoglu, M. McGlohon, and C. Faloutsos, Oddball: Spotting anomalies in weighted graphs, in *Proc. 14^{th} Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining*, Hyderabad, India, 2010, pp. 410–421.

[5] Y. Yao, H. Tong, F. Xu, and J. Lu, Feature generation for graphs and networks, in *Feature Engineering for Machine Learning and Data Analytics*, C. Z. Dong and H. Liu, eds. CRC Press, 2018, pp. 167–188.

[6] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, *struc2vec*: Learning node representations from structural identity, in *Proc. 23^{rd} ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, Halifax, Canada, 2017, pp. 385–394.

[7] Y. A. Lai, C. C. Hsu, W. H. Chen, M. Y. Yeh, and S. D. Lin, Prune: Preserving proximity and global ranking for network embedding, in *Proc. 31^{st} Conf. and Workshop on Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, 2017, pp. 5263–5272.

[8] W. L. Hamilton, R. Ying, and J. Leskovec, Representation learning on graphs: Methods and applications, arXiv preprint arXiv: 1709.05584, 2017.

[9] T. Kipf and M. Welling, Semi-supervised classification with graph convolutional networks, in *Int. Conf. Learning Representations (ICLR)*, Toulon, France, 2017.

[10] P. Goyal and E. Ferrara, Graph embedding techniques, applications, and performance: A survey, arXiv preprint arXiv: 1705.02801, 2017.

[11] D. K. Zhang, J. Yin, X. Q. Zhu, and C. Q. Zhang, Network representation learning: A survey, arXiv preprint arXiv: 1801.05852, 2017.

[12] P. Cui, X. Wang, J. Pei, and W. W. Zhu, A survey on network embedding, arXiv preprint arXiv: 1711.08752, 2017.

[13] H. Y. Cai, V. W. Zheng, and K. C. C. Chang, A comprehensive survey of graph embedding: Problems, techniques, and applications, *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, 2018.

[14] Q. Wang, Z. D. Mao, B. Wang, and L. Guo, Knowledge graph embedding: A survey of approaches and applications, *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, 2017.

[15] Y. Fu and Y. Q. Ma, *Graph Embedding for Pattern*

Analysis. New York, NY, USA: Springer Science & Business Media, 2012.

[16] J. B. Tenenbaum, V. De Silva, and J. C. Langford, A global geometric framework for nonlinear dimensionality reduction, *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[17] S. T. Roweis and L. K. Saul, Nonlinear dimensionality reduction by locally linear embedding, *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[18] D. X. Wang, P. Cui, and W. W. Zhu, Structural deep network embedding, in *Proc. 22^{nd} ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, 2016, pp. 1225–1234.

[19] J. Tang, M. Qu, M. Z. Wang, M. Zhang, J. Yan, and Q. Z. Mei, LINE: Large-scale information network embedding, in *Proc. 24^{th} Int. Conf. World Wide Web (WWW)*, Florence, Italy, 2015, pp. 1067–1077.

[20] S. Y. Chang, W. Han, J. L. Tang, G. J. Qi, C. C. Aggarwal, and T. S. Huang, Heterogeneous network embedding via deep architectures, in *Proc. 21^{th} ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, Sydney, Australia, 2015, pp. 119–128.

[21] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in *Proc. 25^{th} Conf. Uncertainty in Artificial Intelligence*, Montreal, Canada, 2009, pp. 452–461.

[22] A. G. Duran and M. Niepert, Learning graph representations with embedding propagation, in *Advances in Neural Information Processing Systems 30 (NIPS)*, 2017, pp. 5125–5136.

[23] X. Huang, J. D. Li, and X. Hu, Accelerated attributed network embedding, in *Proc. 17^{th} SIAM Int. Conf. Data Mining (SDM)*, Houston, TX, USA, 2017.

[24] X. Huang, J. D. Li, and X. Hu, Label informed attributed network embedding, in *Proc. 10^{th} ACM Int. Conf. Web Search and Data Mining (WSDM)*, Cambridge, United Kingdom, 2017.

[25] J. Tang, M. Qu, and Q. Z. Mei, PTE: Predictive text embedding through large-scale heterogeneous text networks, in *Proc. 21^{th} ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD)*, Sydney, Australia, 2015, pp. 1165–1174.

[26] X. F. Sun, J. Guo, X. Ding, and T. Liu, A general framework for content-enhanced network representation learning, arXiv preprint arXiv: 1610.02906, 2016.

[27] S. H. Wang, J. L. Tang, C. Aggarwal, Y. Chang, and H. Liu, Signed network embedding in social media, in *Proc. 17^{th} SIAM Int. Conf. Data Mining (SDM)*, Houston, TX, USA, 2017.

[28] S. H. Wang, C. Aggarwal, J. L. Tang, and H. Liu, Attributed signed network embedding, in *Proc. 26^{th} Int. Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 137–146.

[29] J. D. Li, H. Dani, X. Hu, J. L. Tang, Y. Chang, and H. Liu, Attributed network embedding for learning in a dynamic environment, in *Proc. 2017 Int. Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 387–396.

[30] L. C. Xu, X. K. Wei, J. N. Cao, and P. S. Yu, Embedding of embedding (EOE): Joint embedding for coupled heterogeneous networks, in *Proc. 10^{th} Int. Conf. Web Search and Data Mining (WSDM)*, Cambridge, United Kingdom, 2017, pp. 741–749.

[31] M. Qu, J. Tang, J. B. Shang, X. Ren, M. Zhang, and J. W. Han, An attention-based collaboration framework for multi-view network representation learning, in *Proc. 2017 Int. Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 1767–1776.

[32] L. K. Zhou, Y. Yang, X. Ren, F. Wu, and Y. T. Zhuang, Dynamic network embedding by modeling triadic closure process, in *Proc. 32^{nd} AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.

[33] D. J. Yang, S. Z. Wang, C. Z. Li, X. M. Zhang, and Z. J. Li, From properties to links: Deep network embedding on incomplete graphs, in *Proc. 2017 Int. Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 367–376.

[34] L. C. Xu, X. K. Wei, J. N. Cao, and P. S. Yu, On exploring semantic meanings of links for embedding social networks, in *Proc. 2018 Web Conference (WWW)*, Lyon, France, 2018, pp. 479–488.

[35] X. Wang, P. Cui, J. Wang, J. Pei, W. W. Zhu, and S. Q. Yang, Community preserving network embedding, in *Proc. 31^{st} AAAI Conf. Artificial Intelligence*, San Francisco, CA, USA, 2017, pp. 203–209.

[36] O. Levy and Y. Goldberg, Neural word embedding as implicit matrix factorization, in *Proc. 27^{th} Int. Conf. and Workshop on Neural Information Processing Systems (NIPS)*), Montreal, Canada, 2014, pp. 2177–2185.

[37] S. S. Cao, W. Lu, and Q. K. Xu, GraRep: Learning graph representations with global structural information, in *Proc. 24^{th} Int. Conf. Information and Knowledge Management (CIKM)*, Melbourne, Australia, 2015, pp. 891–900.

[38] B. Perozzi, R. Al-Rfou, and S. Skiena, Deepwalk: Online learning of social representations, in *Proc. 20^{th} ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD)*, New York, NY, USA, 2014, pp. 701–710.

[39] C. Zhou, Y. Q. Liu, X. F. Liu, Z. Y. Liu, and J. Gao, Scalable graph embedding for asymmetric proximity, in *Proc. 31^{st} AAAI Conf. Artificial Intelligence*, San Francisco, CA, USA, 2017, pp. 2942–2948.

[40] C. Yang, Z. Y. Liu, D. L. Zhao, M. S. Sun, and E. Y. Chang, Network representation learning with rich text information, in *Proc. 24^{th} Int. Joint Conf. Artificial Intelligence (IJCAI)*, Buenos Aires, Argentina, 2015, pp. 2111–2117.

[41] M. D. Ou, P. Cui, J. Pei, Z. W. Zhang, and W. W. Zhu, Asymmetric transitivity preserving graph embedding, in *Proc. 22^{nd} ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, 2016, pp. 1105–1114.

[42] A. Grover and J. Leskovec, node2vec: Scalable feature learning for networks, in *Proc. 22^{nd} ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, 2016, pp. 855–864.

[43] J. F. Chen, Q. Zhang, and X. J. Huang, Incorporate group information to enhance network embedding, in *Proc. 25^{th} ACM Int. Conf. Information and Knowledge Management (CIKM)*, Indianapolis, IN, USA, 2016, pp. 1901–1904.

[44] S. S. Cao, W. Lu, and Q. K. Xu, Deep neural networks for

learning graph representations, in *Proc. 30th AAAI Conf. Artificial Intelligence*, Phoenix, AZ, USA, 2016, pp. 1145–1152.

[45] J. Z. Qiu, Y. X. Dong, H. Ma, J. Li, K. S. Wang, and J. Tang, Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec, in *Proc. 11th Int. Conf. Web Search and Data Mining (WSDM)*, Marina Del Rey, CA, USA, 2018.

[46] S. R. Pan, J. Wu, X. Q. Zhu, C. Q. Zhang, and Y. Wang, Tri-party deep network representation, in *Proc. 25th Int. Joint Conf. Artificial Intelligence (IJCAI)*, New York, NY, USA, 2016, pp. 1895–1901.

[47] Z. L. Yang, W. W. Cohen, and R. Salakhutdinov, Revisiting semi-supervised learning with graph embeddings, in *Proc. 33rd Int. Conf. Machine Learning*, New York, NY, USA, 2016, pp. 40–48.

[48] J. F. Hu, R. Cheng, Z. P. Huang, Y. Fang, and S. Q. Luo, On embedding uncertain graphs, in *Proc. 2017 ACM on Int. Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 157–166.

[49] S. H. Chen, S. F. Niu, L. Akoglu, J. Kovaevi, and C. Faloutsos, Fast, warped graph embedding: Unifying framework and one-click algorithm, arXiv preprint arXiv: 1702.05764, 2017.

[50] Y. X. Dong, N. V. Chawla, and A. Swami, metapath2vec: Scalable representation learning for heterogeneous networks, in *Proc. 23rd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, Halifax, Canada, 2017, pp. 135–144.

[51] T. Y. Fu, W. C. Lee, and Z. Lei, HIN2Vec: Explore meta-paths in heterogeneous information networks for representation learning, in *Proc. 2017 Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 1797–1806.

[52] T. Lyu, Y. Zhang, and Y. Zhang, Enhancing the network embedding quality with structural similarity, in *Proc. 2017 ACM on Conf. Information and Knowledge Management (CIKM)*, 2017, pp. 147–156.

[53] R. Feng, Y. Yang, W. J. Hu, F. Wu, and Y. T. Zhang, Representation learning for scale-free networks, in *Proc. 32nd AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.

[54] Y. Ma, Z. C. Ren, Z. H. Jiang, J. L. Tang, and D. W. Yin, Multi-dimensional network embedding with hierarchical structure, in *Proc. 11th Int. Conf. Web Search and Data Mining (WSDM)*, Marina Del Rey, CA, USA, 2018.

[55] Q. Y. Dai, Q. Li, J. Tang, and D. Wang, Adversarial network embedding, in *Proc. 2018 AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.

[56] J. Kim, H. Park, J. E. Lee, and U. Kang, SIDE: Representation learning in signed directed networks, in *Proc. 2018 ACM World Wide Web Conf. (WWW)*, Lyon, France, 2018, pp. 509–518.

[57] W. Hamilton, Z. T. Ying, and J. Leskovec, Inductive representation learning on large graphs, in *Advances in Neural Information Processing Systems 30 (NIPS)*, 2017, pp. 1025–1035.

[58] T. Chen and Y. Z. Sun, Task-guided and path-augmented heterogeneous network embedding for author identification, in *Proc. 10th ACM Int. Conf. Web Search and Data Mining (WSDM)*, Cambridge, United Kingdom, 2017, pp. 295–304.

[59] B. C. Zhang and M. Al Hasan, Name disambiguation in anonymized graphs using network embedding, in *Proc. 2017 ACM on Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 1239–1248.

[60] H. W. Wang, F. Z. Zhang, M. Hou, X. Xie, M. Y. Guo, and Q. Liu, SHINE: Signed heterogeneous information network embedding for sentiment link prediction, in *Proc. 11th Int. Conf. Web Search and Data Mining (WSDM)*, Marina Del Rey, CA, USA, 2018, pp. 592–600.

[61] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, Distributed large-scale natural graph factorization, in *Proc. 22nd Int. Conf. World Wide Web (WWW)*, Rio de Janeiro, Brazil, 2013, pp. 37–48.

[62] C. C. Tu, W. C. Zhang, Z. Y. Liu, and M. S. Sun, Max-margin deepwalk: Discriminative learning of network representation, in *Proc. 25th Int. Joint Conf. Artificial Intelligence (IJCAI)*, New York, NY, USA, 2016, pp. 3889–3895.

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

[64] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, Distributed representations of words and phrases and their compositionality, in *Proc. 26th Int. Conf. Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, USA, 2013, pp. 3111–3119.

[65] J. X. Ma, P. Cui, and W. W. Zhu, DepthLGP: Learning embeddings of out-of-sample nodes in dynamic networks, in *Proc. 32nd AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.

[66] H. W. Wang, J. Wang, J. L. Wang, M. Zhao, W. N. Zhang, F. Z. Zhang, X. Xie, and M. Y. Guo, GraphGAN: Graph representation learning with generative adversarial nets, in *Proc. 32nd AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.

[67] T. Hoevar and J. Demšar, A combinatorial approach to graphlet counting, *Bioinformatics*, vol. 30, no. 4, pp. 559–565, 2014.

[68] Y. P. Gu, Y. Z. Sun, Y. E. Li, and Y. Yang, Rare: Social rank regulated large-scale network embedding, in *Proc. 2018 World Wide Web Conf. (WWW)*, Lyon, France, 2018, pp. 359–368.

[69] S. H. Yuan, X. T. Wu, and Y. Xiang, SNE: Signed network embedding, in *Proc. 21st Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD)*, 2017, pp. 183–195.

[70] Y. Z. Sun, J. W. Han, X. F. Yan, P. S. Yu, and T. Y. Wu, PathSim: Meta path-based top-K similarity search in heterogeneous information networks, *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[71] G. X. Ma, L. F. He, C. T. Lu, W. X. Shao, P. S. Yu, A. D. Leow, and A. B. Ragin, Multi-view clustering with graph embedding for connectome analysis, in *Proc. 2017 ACM on Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 127–136.

[72] R. Trivedi, M. Farajtabar, P. Biswal, and H. Y. Zha, Representation learning over dynamic graphs, arXiv preprint arXiv: 1803.04051, 2018.

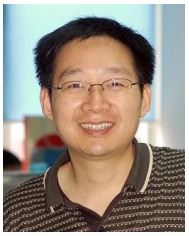
[73] Y. Zhang, Y. Xiong, X. N. Kong, and Y. Y. Zhu, Learning node embeddings in interaction graphs, in *Proc. 2017 ACM on Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 397–406.

[74] K. Tu, P. Cui, X. Wang, F. Wang, and W. W. Zhu, Structural deep embedding for hyper-networks, in *Proc. 32^{nd} AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.
[75] S. Cavallari, V. W. Zheng, H. Y. Cai, K. C. C. Chang, and E. Cambria, Learning community embedding with community detection and node embedding on graphs, in *Proc. 2017 ACM on Conf. Information and Knowledge Management (CIKM)*, Singapore, 2017, pp. 377–386.
[76] M. E. J. Newman, Modularity and community structure in networks, *Proc. Natl. Acad. Sci. U.S.A.*, vol. 103, no. 23, pp. 8577–8582, 2006.
[77] V. Misra and S. Bhatia, Bernoulli embeddings for graphs, in *Proc. 32^{nd} AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.
[78] H. C. Chen, B. Perozzi, Y. F. Hu, and S. Skiena, Harp: Hierarchical representation learning for networks, in *Proc. 32^{nd} AAAI Conf. Artificial Intelligence*, New Orleans, LA, USA, 2018.
[79] P. Yanardag and S. V. N. Vishwanathan, Deep graph kernels, in *Proc. 21^{th} ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, Sydney, Australia, 2015, pp. 1365–1374.
[80] A. Narayanan, M. Chandramohan, L. H. Chen, Y. Liu, and S. Saminathan, subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs, arXiv preprint arXiv: 1606.08928, 2016.

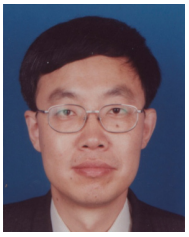
Yaojing Wang is a PhD student in the Department of Computer Science and Technology, Nanjing University, China. His research interests include software repository mining and network embedding. He received the BSc degree from Nanjing University in 2014.

Yuan Yao is currently an assistant researcher in the Department of Computer Science and Technology, Nanjing University, China. He received the PhD degree in computer science from Nanjing University in 2015. His research interests include social media mining, software repository mining, and software intelligence.

Hanghang Tong has been an assistant professor at School of Computing, Informatics, and Decision Systems Engineering (CIDSE), Arizona State University since August 2014. Before that, he was an assistant professor at Computer Science Department, City College, City University of New York, a research staff member at IBM T.J. Watson Research Center and a Post-doctoral fellow in Carnegie Mellon University. He received the MSc and PhD degrees from Carnegie Mellon University in 2008 and 2009, respectively, both majored in machine learning. His research interest is in large scale data mining for graphs and multimedia. He has received several awards, including NSF CAREER award (2017), ICDM 2015 Highest-Impact Paper Award, four best paper awards (TUP14, CIKM12, SDM08, ICDM06), five bests of conference (KDD16, SDM15, ICDM15, SDM11 and ICDM10), and one best demo, honorable mention (SIGMOD17). He has published over 100 referred articles. He is an associated editor of *SIGKDD Explorations* (ACM), an action editor of *Data Mining and Knowledge Discovery* (Springer), and an associate editor of *Neurocomping Journal* (Elsevier); and has served as a program committee member in multiple data mining, databases, and artificial intelligence venues (e.g., SIGKDD, SIGMOD, AAAI, WWW, CIKM, etc).



Feng Xu received the BS and MS degrees from Hohai University in 1997 and 2000, respectively. He received the PhD degree from Nanjing University in 2003. He is a professor in the Department of Computer Science and Technology at Nanjing University. His research interests include trust management, trusted computing, and software reliability.



Jian Lu received the BS and PhD degrees in Computer Science from Nanjing University, China, in 1982 and 1988, respectively. He is currently a professor in the Department of Computer Science and Technology and the Director of the State Key Laboratory for Novel Software Technology at Nanjing University. He serves on the Board of the International Institute for Software Technology of the United Nations University (UNU-IIST). He also serves as the director of the Software Engineering Technical Committee of the China Computer Federation. His research interests include software methodologies, software automation, software agents, and middleware systems.