FedUni ResearchOnline

https://researchonline.federation.edu.au

Copyright Notice

DINE: A Framework for Deep Incomplete Network Embedding

Ke Hou[1], Jiaying Liu[1], Yin Peng[1], Bo Xu[1(✉)], Ivan Lee[2], and Feng Xia[3]

[1] Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116620, China
`boxu@dlut.edu.cn`
[2] School of Information Technology and Mathematical Sciences, University of South Australia, Adelaide, SA 5095, Australia
[3] School of Science, Engineering and Information Technology, Federation University, Australia

Abstract. Network representation learning (NRL) plays a vital role in a variety of tasks such as node classification and link prediction. It aims to learn low-dimensional vector representations for nodes based on network structures or node attributes. While embedding techniques on complete networks have been intensively studied, in real-world applications, it is still a challenging task to collect complete networks. To bridge the gap, in this paper, we propose a Deep Incomplete Network Embedding method, namely DINE. Specifically, we first complete the missing part including both nodes and edges in a partially observable network by using the expectation-maximization framework. To improve the embedding performance, we consider both network structures and node attributes to learn node representations. Empirically, we evaluate DINE over three networks on multi-label classification and link prediction tasks. The results demonstrate the superiority of our proposed approach compared against state-of-the-art baselines.

Keywords: Incomplete network embedding · Network completion · Network representation learning · Deep learning.

1 Introduction

Information networks (e.g. citation networks, social networks, biological networks) contain different types of entities and intricate relations. Analyzing these networks plays an important role in many disciplines [29]. For example, in citation networks, we can find influential entities (i.e., scholars, papers) by calculating the importance of vertices [2, 5]. In social networks, clustering users into communities is useful for recommendation [25, 26]. In biological networks, measuring the similarity between proteins helps us better understand protein interactions [27]. However, with the increase of entities and relations in real-world networks, it is challenging to explore the underlying network structures.

To find an efficient way to model networks, researchers focus on network representation learning (NRL). NRL aims to learn latent, low-dimensional representations for nodes, with preserving not only network topologies but also
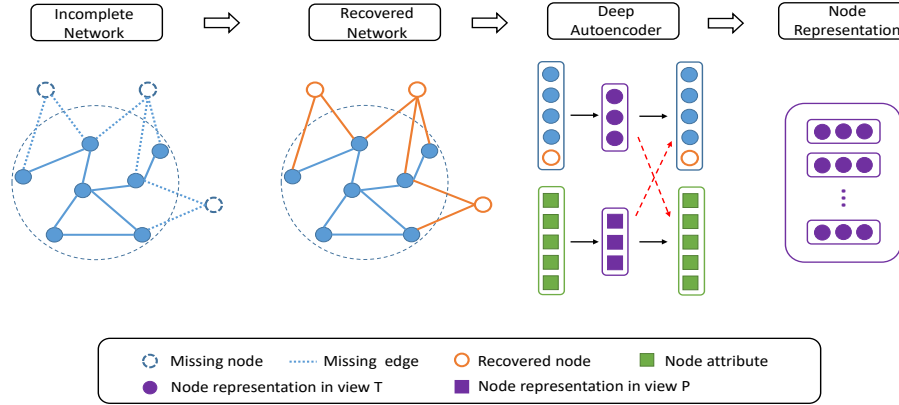
Fig. 1. The overview of DINE framework.

node contents. Perozzi et al. [19] first combine NRL with skip-gram and propose Deepwalk, which lays a solid foundation for future development in this area. Recent advances in NRL have witnessed powerful representations abilities such as DeepGL [20], DANE [7]. Taking advantage of its powerful representation ability to model complex structures, NRL achieves significant performance in downstream tasks such as node classification [3, 30], link prediction [8, 16], and network visualization [21].

In practice, many real-world networks are incomplete [14], which further complicates the embedding process. For example, citation networks are usually incomplete because it is impossible for academic search engines to collect every paper. In biological networks, there exist a huge amount of undiscovered links because of the complexity of gene expression. Analyzing incomplete network makes a deviation because only a part of links are observed, which alters our estimates of network-level statistics. To fill this gap, researchers focus on network completion problem, which makes use of observed connectivity patterns to infer the missing part. However, existing studies only pay attention to missing links inference [6,11], few of them focus on the incomplete networks with both missing nodes and edges [13].

To solve the problem, we present a new framework, named DINE for deep incomplete network embedding. DINE intelligently combines network completion and NRL into a unified framework. As shown in Fig. 1, DINE contains two pivotal steps, including network recovery and network embedding. Specially, we first capture the connectivity patterns from the partially observable network and fit the generative graphs model to estimate missing components. To model the network more accurately, we consider both network structures and node attributes to learn the representations of the recovered network by using a deep autoencoder. Finally, we empirically verify the performance of the proposed framework on three real-world networks. Experimental results illustrate

the significant representation ability of DINE in partially observable networks. Our main contributions can be concluded as follows:

(1) We present a new framework, namely DINE, for deep incomplete network embedding. DINE intelligently combines network completion and NRL into a unified framework, which provides an effective solution for data missing.
(2) DINE considers not only topology structure but also node attributes for embedding. It can accurately and effectively model node proximity and underlying structure in the joint space.
(3) We extensively validate the framework on three real-world networks through multi-label classification and link prediction tasks. The results demonstrate the superiority of our proposed approach compared with state-of-the-art baselines.

The remainder of this paper is organized as follows. Section 2 summarizes related work. In section 3, we focus on problem definition. Section 4 introduces the implementation details of the proposed framework. Experimental results are provided in section 5. Finally, we conclude this work in section 6.

2   Related Work

The framework we proposed in this paper is related to two areas of research, including network completion and NRL techniques.

2.1   Network Completion

Network completion deals with the problem of inferring missing nodes and edges in networks. Network completion is similar to matrix completion [12], which aims to complete the matrix with elements missing. However, network completion is more arduous because of network diversity. For missing edges, it is an attractive way to recover the original network by calculating node similarity. Another way to complete missing edges is considering shared node neighbors [4]. In cases where both nodes and edges are missing, we can utilize a generative graphs model named KronFit [15] to generate complete networks whose structures are similar to real-world networks. Kim et al. [13] combine expectation-maximization into KronFit, and propose a powerful algorithm KronEM, which is more effective for recovering the network.

2.2   Network Representation Learning

NRL aims to embed each node in the network into a low-dimensional representation. Existing NRL algorithms can be divided into four categories. The first category is matrix factorization based methods. They first represent the connections between network vertices and use matrix factorization to obtain representations. IsoMAP [23] constructs an affinity network by feature vectors. It represents nodes with the solved leading eigenvectors. The second category is random walk based

Table 1. The description of notations

Notation	Description
N_G	number of nodes in the complete network
N'	number of nodes in the incomplete network
N_M	number of missing nodes
N_R	number of recovered nodes
xt	input of network structure view T
xp	input of node attribute view P
$x\hat{t}^t$	reconstruction output of xt by self-view
$x\hat{t}^p$	reconstruction output of xt by cross-view
$x\hat{p}^t$	reconstruction output of xp by self-view
$x\hat{p}^p$	reconstruction output of xp by cross-view
K	number of encoding layers
$yt^{(K)}$	representation in network structure view T
$yp^{(K)}$	representation in node attribute view P
α	balance reconstruction errors of self-view and cross-view
β	balance reconstruction errors of G' and G_M

methods. DeepWalk [19] utilizes random walk to learn structural information and uses skip-gram to obtain the representations. Node2vec [10] changes the strategy of random walk to capture a more global structure. The third category is edge modeling based methods. They utilize node-node connections to learn node representations directly. LINE [22] uses first-order proximity and second-order proximity to obtain local and global structure information. The fourth category is deep learning based methods. They could extract highly non-linear structure automatically by using deep learning techniques. SDNE [24] preserves first and second order proximities for highly non-linear structures via a deep autoencoder.

3   Preliminary

In this section, we first describe the notations used in this paper. We then formalize the problem of network embedding in an incomplete network.

3.1   Notations

We denote the complete network as $G = (V, A, P)$, where $V = \{v_1, v_2, ..., v_{|V|}\}$ indicates the nodes in the network. $A \in \mathbb{R}^{|V| \times |V|}$ represents the adjacency matrix and $P \in \mathbb{R}^{|V| \times |P|}$ denotes the node attribute matrix, where $|V|$ and $|P|$ represent the dimension of adjacency matrix and node attributes, respectively. Similarly, we define the incomplete network, the missing network, and recovered network as $G' = (V', A', P')$, $G_M = (V_M, A_M, P_M)$, $G_R = (V_R, A_R, P_R)$, respectively. Table 1 lists the meaning of the notations mainly used in this paper.

3.2   Problem Formulation

The purpose of network completion is to infer the missing part of the incomplete network, how to infer the missing network G_M from the observable network G' is crucial to the problem. If we use adjacency matrices to represent the network, then the network completion problem can be transformed into matrix completion problem. In general, classical matrix completion problem is to determine the value (0 or 1) of elements in the missing part in a binary matrix. In this paper, we assume that the number of missing nodes is known. If not, the standard methods for estimating the size of hidden (missing) populations can solve this problem [17].

Although network recovering helps in representing the incomplete network, there are some problems in the representation learning process. On the one hand, many network representation methods are shallow models. Network completeness is essential for extracting local or global topology information. On the other hand, most methods can't capture non-linear relations between nodes [24]. Thus, we need to consider not only topology information for non-linear relations but also node contents such as node attributes. Besides, A' and P' preserve the information of a network, which is used to represent the network in the joint space. Thus, nodes with similar topology structures or attributes will be closer in the representation dimension.

4   Design of DINE

In this section, we present a novel framework, namely DINE, to solve the problem of network embedding in incomplete networks. Our framework contains two crucial components, network recovery and network embedding. Firstly, we discuss how to recover the incomplete network. Then, we introduce the process of network representation learning, which considers both topology information and node attributes.

4.1   Recovery of Incomplete Network

To recover the network with nodes and edges missing, we model the incomplete network with the Kronecker graphs model [15]. In detail, we use the incomplete network to fit the Kronecker graphs model in network structure and estimate the missing part, and then re-estimate model parameters. These two steps are iterated until the model parameters converge. Finally, we obtain the missing part of the network.

The purpose of the network completion is to find the most likely structure of the missing part G_M. We connect the incomplete network and the missing network by network generation parameters Θ. Let σ denote the mapping among nodes in the recovered network, incomplete network, and missing network. The mapping σ indicates a permutation of set $\{1, ..., N_G\}$. The first N' elements of σ map the nodes of G_R to the incomplete network and the remaining N_M elements

of σ map the nodes of missing part G_M. The likelihood $P\left(G', G_M, \sigma | \Theta\right)$ can be represented as:

$$P\left(G', G_M, \sigma | \Theta\right) = \prod_{a_{uv}=1} \left[\Theta^k\right]_{\sigma(u)\sigma(v)} \prod_{a_{uv}=0} \left(1 - \left[\Theta^k\right]_{\sigma(u)\sigma(v)}\right) \qquad (1)$$

where Θ^k is the adjacency matrix generated by model parameters Θ. $\left[\Theta^k\right]_{\sigma(u)\sigma(v)}$ denotes the $(\sigma\left(u\right), \sigma\left(v\right))$-th element of matrix Θ^k. a_{uv} is the (u,v)-th element of A_R, which is the the adjacency matrix of the recovered network.

Next, we consider the edges in the missing part and σ as the latent variables. E-step is to sample the missing part and permutation. M-step aims to optimize the parameters Θ by stochastic gradient descent process. Then we iterate E-step and M-step until parameters Θ converge. The steps could be described as:

$E\text{-}step:$

$$(G_M^{(t)}, \sigma^{(t)}) \sim P(G_M, \sigma | G', \Theta^{(t)}) \qquad (2)$$

$M\text{-}step:$

$$\Theta^{(t+1)} = \underset{\Theta \in (0,1)^2}{\arg\max} \, \mathbb{E}[P(G_M^{(t)}, \sigma^{(t)}, G' | \Theta)]. \qquad (3)$$

In detail, we first initialize model parameters Θ and generate a stochastic network. Then we sample the missing part G_M and node mapping σ by Gibbs sampling, which can be considered to recover the missing part of the network. Besides, we optimize the model parameters Θ and iterate the above steps until the parameters converge. Finally, we obtain the most likely instances of the missing part and node mapping.

4.2 Recovered Network Embedding MVC-DNER

In terms of network representation, we consider not only network topology structure but also node attributes. Furthermore, inspired by MVC-DNE [28] which utilizes a deep autoencoder, we propose MVC-DNER to capture non-linear structures and node attributes in the recovered network. Fig. 2 shows that the embedding part has network structures view T and node attributes view P, which uses deep autoencoder to learn latent information in each view. We take the

Table 2. Layer structures of MVC-DNER on three datasets

Dataset	layers in view V	layers in view P
Citeseer	N_R-500-128	3,703-600-128
DBLP	N_R-800-128	9,662-900-128
BlogCatalog	N_R-500-128	39-4

adjacency matrix xt and the attribute matrix xp of the recovered network as

input. In the encoding process, input features of one view could encode some shared latent information reflecting the input of the other view. In the decoding process, latent representations in one view could reconstruct the input of another view. The loss function is defined as:

$$L(xt, xp; \theta) = L_t(xt, xp; \theta) + L_p(xt, xp; \theta) \tag{4}$$

$$L_t(xt, xp; \theta) = \beta \sum_{i=1}^{|V'|} ((1-\alpha) \left\| xt_i - x\hat{t}_i^t \right\|_2^2 + \alpha \left\| xt_i - x\hat{t}_i^p \right\|_2^2)$$
$$+ (1-\beta) \sum_{i=1}^{|V_M|} ((1-\alpha) \left\| xt_i - x\hat{t}_i^t \right\|_2^2 + \alpha \left\| xt_i - x\hat{t}_i^p \right\|_2^2) \tag{5}$$

$$L_p(xt, xp; \theta) = \beta \sum_{i=1}^{|V'|} ((1-\alpha) \left\| xp_i - x\hat{p}_i^t \right\|_2^2 + \alpha \left\| xp_i - x\hat{p}_i^p \right\|_2^2)$$
$$+ (1-\beta) \sum_{i=1}^{|V_M|} ((1-\alpha) \left\| xp_i - x\hat{p}_i^t \right\|_2^2 + \alpha \left\| xp_i - x\hat{p}_i^p \right\|_2^2) \tag{6}$$

where $x\hat{t}_i^t$ and $x\hat{t}_i^p$ are the reconstruction outputs of xt_i. $x\hat{p}_i^t$ and $x\hat{p}_i^p$ are the reconstruction vectors of xp_i. α and β are parameters to adjust the proportion of self-view and cross-view reconstruction errors, recovered nodes and observed nodes reconstruction errors, respectively. $\theta = \{W^{(l)}, b^{(l)}, \hat{W}^{(l)}, \hat{b}^{(l)}\}_{l=1}^{K}$ denotes parameters including the weights W and bias b in the deep autoencoder.
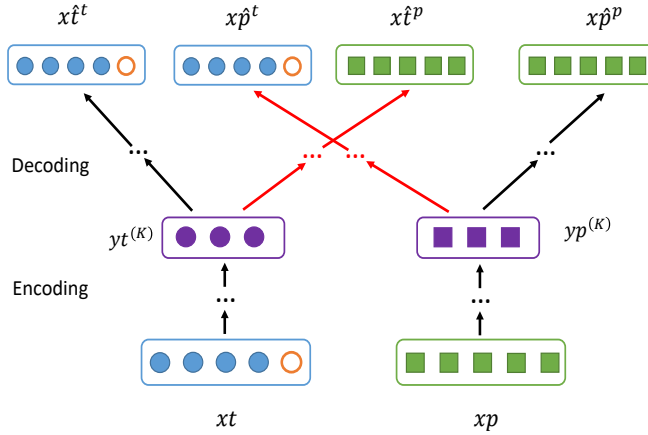

Fig. 2. Deep autoencoder MVC-DNER.

The loss function is minimized by stochastic gradient descent. Thus, the learning representations preserve not only network structures information but also node attributes information.

5   Experiments

In this section, we evaluate our framework on three datasets through multi-label classification and link prediction tasks. We first introduce three datasets and baseline methods. Then we describe evaluation metrics and parameter settings of the methods. Finally, we present the performance of DINE and compare it against state-of-the-art baselines.

5.1   Datasets

We use three datasets including two academic datasets (Citeseer[4] and DBLP[5]) and a social dataset BlogCatalog[6].

(1) **Citeseer** contains citation information of papers. In the citation network, each node represents a paper and edges reflect citation relationship. The citation network constructed by Citeseer contains 3,312 papers divided into six classes including Agents, AI, DB, IR, ML, and HCI. Besides, the attribute feature of each paper is a 3,703 dimensional binary vector based on the topics.

(2) **DBLP** is also a citation dataset which covers useful information on papers such as authors, year, publisher, and title. It provides open bibliographic information of major computer science journals and proceedings. We choose 8,192 papers from 10 research domains. We choose the title of papers as the attribute and use a 9,662 dimensional binary vector to represent the attribute feature.

(3) **BlogCatalog** is a social blog directory that manages bloggers and their blogs. We choose some data which contains 4,096 nodes, 38,983 edges, and 39 groups. Nodes represent bloggers and edges represent the friendship between bloggers. Besides, each blogger belongs to one or several groups based on interesting.

5.2   Baseline Methods

We use the following methods as our baseline methods. We choose four network representation learning methods based on matrix factorization, random walk, and deep learning, respectively.

(1) **GF** [1] is a matrix factorization based method. It relies on partitioning a graph to minimize the number of neighboring vertices. In addition, it preserves first order proximity and allows for linear scalability.

(2) **HOPE** [18] is also based on matrix factorization. It provides an efficient way to preserve high-order proximities of large-scale graphs. It is also capable of capturing the asymmetric transitivity.

---

(3) **Node2vec** [10] is a shallow model. It designs a flexible neighbor sampling
strategy based on Deepwalk. It can preserve both local structure and global
structure to learn network representations.
(4) **SDNE** [24] is the first network representation learning method based on deep
learning. The deep autoencoder captures the non-linear network structure.
It also can preserve the local and global network structure.

5.3 Parameter Settings

Our framework consists of network completion and recovered network repre-
sentation learning. In the network completion, the Kronecker parameter Θ is
random initialization. The neural network structure of MVC-DNER is listed in
Table 2. We set the learning rate as 0.001. The mini-batch size of optimization
is 50. The parameters for balancing the importance of self-view and cross-view
α , recovered nodes and observed nodes β are set to 0.5 and 0.8, respectively.

The parameter settings of these baseline methods including GF, HOPE,
Node2vec, and SDNE follow a NRL survey [9]. The learning rate of SGD is
0.0001, and max iterations are 5,000 in GF. The higher-order coefficient of HOPE
is 0.01. In Node2vec, we set the window size as 10, the walk length as 40, walks
per node as 40. The dimension of network learning representation is 128 for all
methods.

Table 3. Multi-label classification results(macro-F1) on two datasets with the portion
of missing nodes

Datasets	M_r	GF	HOPE	Node2vec	SDNE	DINE
Citeseer	0.05	0.253	0.265	0.431	0.366	**0.642**
	0.10	0.231	0.253	0.423	0.365	**0.636**
	0.15	0.202	0.257	0.412	0.358	**0.627**
	0.20	0.203	0.244	0.419	0.352	**0.629**
	0.25	0.235	0.249	0.403	0.346	**0.617**
	0.30	0.225	0.243	0.388	0.335	**0.614**
DBLP	0.05	0.579	0.575	0.582	0.585	**0.595**
	0.10	0.575	0.574	0.594	0.558	**0.601**
	0.15	0.541	0.534	0.584	0.571	**0.594**
	0.20	0.576	0.587	0.564	0.566	**0.591**
	0.25	0.574	0.580	0.561	0.577	**0.593**
	0.30	0.572	0.571	0.558	0.573	**0.590**

5.4 Experimental Results

Multi-label classification. We aim to learn representations in an incomplete
network. To achieve this goal, we need to remove 5%-30% nodes and the corre-
sponding edges. We first learn the representations based on remaining nodes and

take the representations as the input of the classification model. Then we divide the labeled nodes into training set and testing set. The portion ratio of training nodes varies from 10% to 90%. We use macro-F1 to evaluate the performance of the classification model. Besides, the experiment runs 10 times, and we take the average of results as the final results. Table 3 lists the classification results for each method, where M_r is the portion ratio of missing nodes.

From the table, we can observe that the performance of DINE is better than any other baseline methods, especially in the Citeseer dataset. The performances of these methods gradually become worse as the portion of missing nodes increasing. Besides, two matrix factorization methods have terrible performance in Citeseer. Most methods have a relatively better performance in the DBLP dataset.

Link prediction. Similar to the task of multi-label classification, we also remove partial nodes and the corresponding edges. Then we remove 20% edges of the remainder network as links for prediction and consider them as positive samples. Besides, we randomly select unconnected node pairs as negative samples. The number of negative samples is the same as positive samples. From the results of link prediction presented in Fig. 3, we can see that DINE achieves significant improvements in AUC over the baselines in all datasets. As the portion of missing nodes increasing, the performances of these methods have a downward trend.
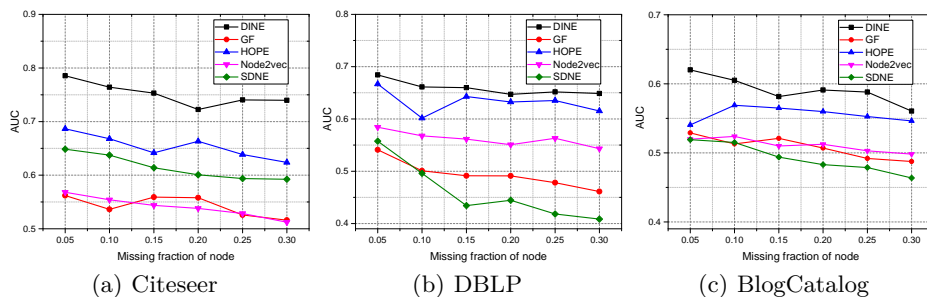

| (a) Citeseer | (b) DBLP | (c) BlogCatalog |

Fig. 3. Link prediction results (AUC) on three datasets with the portion of missing nodes.

6   Conclusion

In this paper, we have presented a framework named DINE, which aims to learn node representations in incomplete networks. The framework is divided into two parts: network completion and recovered network representation learning. Specifically, we recover the missing part of the incomplete network based on the combination of EM approach and Kronecker graphs model. After recovering

the incomplete network, we propose an algorithm named MVC-DNER to learn node representations for the recovered network. MVC-DNER uses the deep autoencoder to learn representations, which preserves both network structures and node attributes. Experimental results on three real-world network datasets show the significant performance of our proposed method. The future work is primarily on extending DINE to heterogeneous networks containing different types of nodes and edges.

References

1. Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., Smola, A.J.: Distributed large-scale natural graph factorization. In: Proceedings of the 22nd international conference on World Wide Web. pp. 37–48. ACM (2013)
2. Bai, X., Lee, I., Ning, Z., Tolba, A., Xia, F.: The role of positive and negative citations in scientific evaluation. IEEE Access **5**, 17607–17617 (2017)
3. Bhagat, S., Cormode, G., Muthukrishnan, S.: Node classification in social networks. In: Social network data analytics, pp. 115–148. Springer (2011)
4. Buccafurri, F., Lax, G., Nocera, A., Ursino, D.: Discovering missing me edges across social networks. Information Sciences **319**, 18–37 (2015)
5. Cai, L., Tian, J., Liu, J., Bai, X., Lee, I., Kong, X., Xia, F.: Scholarly impact assessment: a survey of citation weighting solutions. Scientometrics **118**(2), 453–478 (2019)
6. Clauset, A., Moore, C., Newman, M.E.: Hierarchical structure and the prediction of missing links in networks. Nature **453**(7191), 98 (2008)
7. Gao, H., Huang, H.: Deep attributed network embedding. In: IJCAI. vol. 18, pp. 3364–3370 (2018)
8. Gao, S., Denoyer, L., Gallinari, P.: Temporal link prediction by integrating content and structure information. In: Proceedings of the 20th ACM international conference on Information and knowledge management. pp. 1169–1174. ACM (2011)
9. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems **151**, 78–94 (2018)
10. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864. ACM (2016)
11. Guimerà, R., Sales-Pardo, M.: Missing and spurious interactions and the reconstruction of complex networks. Proceedings of the National Academy of Sciences **106**(52), 22073–22078 (2009)
12. Keshavan, R.H., Montanari, A., Oh, S.: Matrix completion from a few entries. IEEE transactions on information theory **56**(6), 2980–2998 (2010)
13. Kim, M., Leskovec, J.: The network completion problem: Inferring missing nodes and edges in networks. In: Proceedings of the 2011 SIAM International Conference on Data Mining. pp. 47–58. SIAM (2011)
14. Kossinets, G.: Effects of missing data in social networks. Social networks **28**(3), 247–268 (2006)
15. Leskovec, J., Chakrabarti, D., Kleinberg, J., Faloutsos, C., Ghahramani, Z.: Kronecker graphs: An approach to modeling networks. Journal of Machine Learning Research **11**(Feb), 985–1042 (2010)
16. Lü, L., Zhou, T.: Link prediction in complex networks: A survey. Physica A: statistical mechanics and its applications **390**(6), 1150–1170 (2011)

17. McCormick, T.H., Salganik, M.J., Zheng, T.: How many people do you know?: Efficiently estimating personal network size. Journal of the American Statistical Association **105**(489), 59–70 (2010)
18. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1105–1114. ACM (2016)
19. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710. ACM (2014)
20. Rossi, R.A., Zhou, R., Ahmed, N.K.: Deep inductive network representation learning. In: Companion Proceedings of the The Web Conference 2018. pp. 953–960. International World Wide Web Conferences Steering Committee (2018)
21. Tang, J., Liu, J., Zhang, M., Mei, Q.: Visualizing large-scale and high-dimensional data. In: Proceedings of the 25th international conference on world wide web. pp. 287–297. International World Wide Web Conferences Steering Committee (2016)
22. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web. pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015)
23. Tenenbaum, J.B., De Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. science **290**(5500), 2319–2323 (2000)
24. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1225–1234. ACM (2016)
25. Wang, W., Liu, J., Yang, Z., Kong, X., Xia, F.: Sustainable collaborator recommendation based on conference closure. IEEE Transactions on Computational Social Systems **6**(2), 311–322 (2019)
26. Xia, F., Asabere, N.Y., Liu, H., Chen, Z., Wang, W.: Socially aware conference participant recommendation with personality traits. IEEE Systems Journal **11**(4), 2255–2266 (2014)
27. Xu, B., Li, K., Liu, X., Liu, D., Zhang, Y., Lin, H., Yang, Z., Wang, J., Xia, F.: Protein complexes detection based on global network representation learning. In: 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). pp. 210–213. IEEE (2018)
28. Yang, D., Wang, S., Li, C., Zhang, X., Li, Z.: From properties to links: Deep network embedding on incomplete graphs. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 367–376. ACM (2017)
29. Zhang, D., Yin, J., Zhu, X., Zhang, C.: Network representation learning: A survey. IEEE transactions on Big Data (2018)
30. Zhu, S., Yu, K., Chi, Y., Gong, Y.: Combining content and link for classification using matrix factorization. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 487–494. ACM (2007)