Soft Computing (2020) 24:8223–8231
https://doi.org/10.1007/s00500-019-04451-z

FOCUS

A network representation method based on edge information extraction

Wei Fan[1] · Hui Min Wang[1] · Yan Xing[1] · Rui Huang[1] · W. H. Ip[2,3] · Kai Leung Yung[4]

Published online: 20 November 2019

Abstract
In recent years, network representation learning has attracted extensive attention in the academic field due to its significant application potential. However, most of the methods cannot explore edge information in the network deeply, resulting in poor performance at downstream tasks such as classification, clustering and link prediction. In order to solve this problem, we propose a novel way to extract network information. First, the original network is transformed into an edge network with structure and edge information. Then, edge representation vectors can be obtained directly by using an existing network representation model with edge network as its input. Node representation vectors can also be obtained by utilizing the relationships between edges and nodes. Compared with the structure of original network, the edge network is denser, which can help solving the problems caused by sparseness. Extensive experiments on several real-world networks demonstrate that edge network outperforms original network in various graph mining tasks, i.e., node classification and node clustering.

Keywords Network representation learning · Edge network · Node representation vectors · Edge representation vectors

1 Introduction

With the development of Internet, information networks have become one of the most common data forms to preserve information. Analysis and research on networks have great academic value and high potential application value (Hoang et al. 2018). For example, user recommendation system is designed to explore potential relationships between users in social networks (Wang et al. 2017c; Zedan and Miller 2017; Hu et al. 2015; Wang et al. 2017a), and some online advertisement delivery systems also deliver similar advertisements to people in similar groups. An important issue in the network research is how to represent network information properly.

Communicated by Mu-Yen Chen.

✉ Yan Xing
yxing@cauc.edu.cn

[1] Department of Computer Science and Technology, Civil Aviation University of China, Tianjin, China

[2] Department of Industrial and Systems Engineering, Hong Kong Polytechnic University, Kowloon, China

[3] Department of Mechanical Engineering, University of Saskatchewan, Saskatoon, Canada

[4] Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

Network representation learning, or network embedding, is a promising way to explore information. It has been applied in many fields such as sociology and computer science. Network representation learning methods first identify and preserve valuable information in the original network automatically. Then, encode them into a low-dimension, dense and continuous vector space, so that the noise or redundant information can be reduced and the intrinsic structure information can be preserved. Node representation vectors obtained by the representation vector space can be applied to many downstream tasks, such as node classification (Perozzi et al. 2014), node clustering (Wang et al. 2017b), link prediction (Ou et al. 2016), visualization (Wang et al. 2016) and so on. Early network representation learning methods obtain node representation vectors by constructing a feature matrix. However, these methods usually have high computational complexity and poor performance when the scale of network is very large. With the development of deep learning, many representation learning methods based on neural networks have been proposed.

However, they do not make full use of edge information and ignore edge representation, which causes (1) information in the original network is not utilized sufficiently, (2) edge representation vectors are too dependent on node representation vectors. In order to preserve both network structure and edge information better, in this paper, we transform an original network into an edge network. During this process, first-order and second-order similarities between nodes and edge information in original network are preserved. Because of more nodes and edges, edge network will alleviate sparseness greatly. As shown in Fig. 1, edge network preserves the structure of original network well and becomes denser than that. Based on edge network, we input edge network into an existing network representation model to get edge representation vectors directly. Then, edge representation vectors are transformed into node representation vectors by using relationships between nodes and edges. In order to verify the effectiveness of the proposed method, we use the network representation learned by LINE on Polbooks dataset as the input to the visualization tool t-SNE (Van Der Maaten 2014) to compare the visualization effect. In Fig. 1, the left two graphs are the structure of the original network and the transformed edge network, respectively. The right two are the corresponding visualization results. Different clusters are represented by different colors. We can see that in the visualization result of original network, the nodes are uniformly distributed in whole space, which cannot reflect the cluster relationship of nodes. While in the edge network, nodes in the same cluster are located more closely to each other. It can be found that the network representation of edge network is better than that of original network. More downstream tasks of node classification and node clustering are conducted on several real-world networks. The results show that edge network outperforms original network in most cases.

The main contributions of this paper are summarized as follows:

1. We present a novel way to extract network information, which not only preserves network information better, but also alleviates sparseness greatly.
2. Edge representation vectors use original network information sufficiently and are not dependent on node representation vectors.
3. We conduct comprehensive node classification and node clustering experiments on several real-world networks to demonstrate the effectiveness of the proposed method.

2 Related work

Network representation learning, also known as network embedding, aims to find the low-dimensional vector space to better capture the information in the network. It has shown superior performance in various tasks, such as node classification, node clustering, link prediction. As a result, network representation learning has attracted more and more attention in recent years.

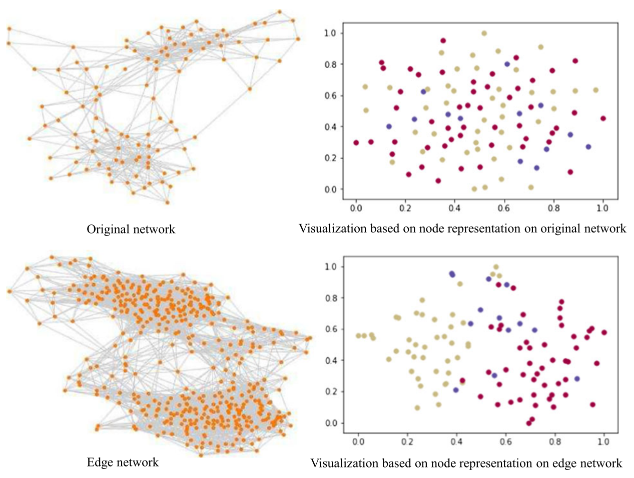

Fig. 1 An example of 2D visualization result of network representation on Polbooks dataset by t-SNE. Please see text for details

Although network representation learning has achieved good results, it still faces many challenges. (1) Network structure is highly nonlinear, which means it is very difficult to capture structural features completely (Luo et al. 2011); (2) most of the real-world networks are sparse, in other words, there are little edges can be observed. So, results will not be good if only the observed edges are used (Tang et al. 2015); (3) there is a lot of edge information in network, so how to make full use of it is also a key problem to be solved (Perozzi et al. 2014).

Traditional network representation learning methods usually use spectrum properties, such as eigenvalues and eigenvectors, also known as singular values and singular vectors. The input matrix, adjacency matrix or Laplacian matrix in most cases, is designed by a specific method. For example, locally linear embedding (LLE) (Roweis and Saul 2000) considers that each node can be constructed through a linear weighted combination of its neighbor nodes. Laplacian Eigenmaps (LE) (Belkin and Niyogi 2002) can reflect the intrinsic manifold structure of the data with an adjacency matrix as input. Unlike LLE, the feature vectors corresponding to the minimum K nonzero feature values of Laplacian matrix are used as network representation learning vectors. The network representation learning methods based on spectrum properties only consider network structure, and their computational complexity is high $O(n^2)$, so they are difficult to be applied to large scale networks.

With the development of deep learning, a large number of representation learning methods based on neural networks have emerged. Mikolov et al. propose an effective neural network framework to learn the distributed representation of words in natural language (Mikolov et al. 2013a, b). Inspired by this, Deepwalk (Perozzi et al. 2014) first obtains a series of node sequences by using random walk approach to generate network neighborhoods for nodes, which is analogical to a depth-first search. It treats each node sequence as a sentence, then inputs node sequences into the Skip-gram model and finally obtains a low-dimensional representation vector for each node. Based on Deepwalk, LINE (Tang et al. 2015) uses a breath-first search strategy to preserve first-order and second-order similarities in the network. However, both of them fail to offer any flexibility in node sampling from networks. Node2vec (Grover and Leskovec 2016) improves random walk approach with a more flexible approach, so that sampled paths can preserve the local and global properties of the network to a greater extent. Different from these shallow neural networks-based methods, SDNE (Wang et al. 2016) uses deep neural networks to capture highly nonlinear relationships between nodes.

However, a major limitation in all of the above works is that they only use the network structure for network representation. But for most of the real-world networks, other information such as node features, the supervised label information and heterogeneous information is also important. Matrix factorization-based network representation techniques can fuse structure and other information well. TADW (Yang et al. 2015) introduces the text features of nodes into network representation learning under a matrix decomposition framework. BANE (Yang et al. 2018) formulates a new Weisfeiler-Lehman matrix factorization learning function under the binary node representation constraint. Based on BANE, LQANR (Yang et al. 2019) compacts node representations with low bitwidth values and achieves high representation accuracy.

However, these methods suffer high time complexity when network scale is large. Since deep neural networks can integrate different kinds of information, MMDW (Tu et al. 2016), which is based on Deepwalk, incorporates label information into the network representation learning process. CANE (Tu et al. 2017a) encodes text information by using CNN and obtains context-aware network representation. SNE (Liao et al. 2018) learns node representation vectors in the attribute network by using neural networks. DANE (Hong et al. 2019) applies GCN to learn transferable node representation of attributed networks.

Despite node features and label information, deep neural networks can also fuse heterogeneous information into network representation process. MVE (Qu et al. 2017) proposes a multi-view representation learning approach, which promotes the collaboration of different views and lets them vote for the robust representations by using attention mechanism. Different from the voting process in MVE, MEGAN (Sun et al. 2019) employs a generator to integrate information about pair-wise links between nodes across all of the views.

All the above methods focus on the node representation, but ignore the edge representation. There are a few exiting methods which can obtain edge representation. Node2vec (Grover and Leskovec 2016) obtains edge representation vectors by using vector operations between the node representation vectors. TransNet (Tu et al. 2017b) uses translation mechanism to get edge representation vectors. HEER (Shi et al. 2018) obtains node representation vectors and edge representation vectors simultaneously by extracting the relationships in the heterogeneous network. However, they all obtain the node representation vectors first and then obtain the edge representation vectors through their relationships, which causes edge information in the original networks utilized insufficiently, and edge representation is deeply dependent on the node representation.

In order to solve these problems, we propose an information extraction method that transforms an original network into an edge network. By inputting the edge network into an existing network representation model, edge representation vectors can be obtained directly. Node representation can be further obtained by the edge representation vectors.

3 Network representation learning based on edge information extraction

In this section, we formally define the related problem of network representation learning and introduce our method. Figure 2 illustrates the framework of our method.

3.1 Problem definition

In this section, we will introduce notions and define the problems formally.

Definition 1 Network representation learning. Given network $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of nodes, $E = \{e_1, e_2, \ldots, e_m\}$ represents the set of edges. n and m are the number of nodes and edges in the network, respectively. A represents the adjacency matrix. If v_i and v_j are connected, $a_{ij} = 1$. Otherwise, $a_{ij} = 0$. The purpose of network representation is to learn a mapping function $f : v_i \rightarrow y_i \in R^d$, where d is the dimension of the vector and $d \ll |V|$. The relationship between node representation vectors y_i and y_j can reflect the relationship between nodes v_i and v_j in G.

Definition 2 First-order similarity. Given a graph $G = (V, E)$, for any two nodes v_i and v_j, if they are connected directly($a_{ij} = 1$), nodes v_i and v_j have first-order similarity (Grover and Leskovec 2016).

First-order similarity intuitively reflects the relationship between nodes in the network. If there is a connection between two nodes, these two nodes are more similar. For example, if a paper cites another one, then these two papers are more similar than others. However, a large number of

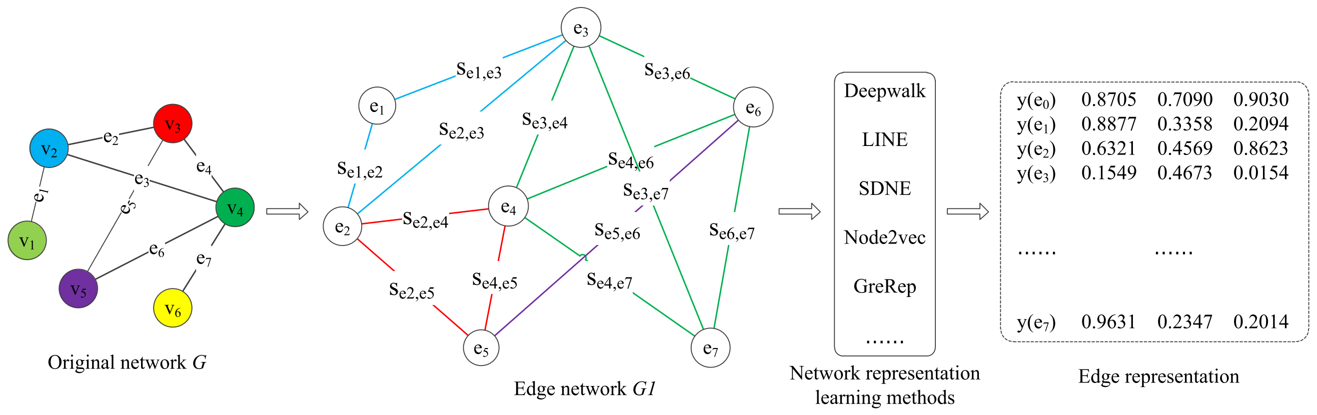
Original network G
Edge network G1
Deepwalk
LINE
SDNE
Node2vec
GreRep
......
Network representation learning methods
Edge representation
y(e0) 0.8705 0.7090 0.9030
y(e1) 0.8877 0.3358 0.2094
y(e2) 0.6321 0.4569 0.8623
y(e3) 0.1549 0.4673 0.0154
y(e7) 0.9631 0.2347 0.2014

Fig. 2 The framework of the proposed method. An original network G with 6 nodes and 7 edges is transformed into an edge network $G1$, which has 7 nodes and 13 edges. By inputting $G1$ into the existing network representation models (such as Deepwalk, LINE.), the node representation vectors in $G1$ (also edge representation vectors in G) can be obtained directly

node pairs do not have first-order similarity because of the sparseness in the real-world networks. Therefore, it is difficult to save the network structure well by relying only on the first-order similarity. Second-order similarity is used to solve this problem.

Definition 3 Second-order similarity. Given a graph $G = (V, E)$, for any two nodes v_i and v_j, if there is a common neighbor v_k between them, that is, $a_{ik} = 1, a_{jk} = 1$, v_i and v_j have second-order similarity (Grover and Leskovec 2016).

Second-order similarity assumes that the more common neighbors between two nodes, the higher similarity between them. For example, in the sentences *I like eating apple* and *I like eating meat*, the words *apple* and *meat* should have higher similarity because of the similar context. By the way, the problem of sparseness can be alleviated greatly by using second-order similarity.

Local and global information can be preserved finally in the network by using first-order and second-order similarities.

3.2 Edge information extraction

Different from the existing network representation learning methods, we focus on how to extract network information better. In order to solve this problem, an original network G is transformed into an edge network $G1$. In other words, the edges in G will become the nodes in $G1$. As shown in Fig. 2, G has 7 edges and $G1$ has 7 nodes. The similarity between nodes in $G1$ will be constructed by the first-order and the second-order similarities of the original network G. The similarity between nodes in $G1$ can be calculated by Cosine similarity (Salton 1970), also called Salton index, as shown in Eq. 1.

$$s_{pq} = s_{e_p,e_q} = \frac{\left|\mathcal{N}(v_i) \bigcap \mathcal{N}(v_j)\right|}{\sqrt{\left|\mathcal{N}(v_i)\right| \left|\mathcal{N}(v_j)\right|}} \tag{1}$$

Where $\mathcal{N}(v)$ represents the neighborhoods of node v. Node pair (v_i, v_k) is connected by edge e_p, and (v_j, v_k) is connected by e_q in original network G. For example, in Fig. 2, the way to calculate similarity between e_6 and e_7 is $s_{67} = s_{e_6,e_7} = \frac{|\mathcal{N}(v_5) \bigcap \mathcal{N}(v_6)|}{\sqrt{|\mathcal{N}(v_5)||\mathcal{N}(v_6)|}} = \frac{1}{\sqrt{2\times 1}} \approx 0.7071$. The detailed construct algorithm of edge network can be seen in Algorithm 1.

Algorithm 1 Transform original network to edge network.

```
Input: Original network $G$;
Output: Edge network $G1$;
 1: Generate an empty network $G1$;
 2: for $e$ in $G.edges()$ do
 3:     $G1.add\_nodes(e)$;
 4: end for
 5: for $e_p$ in $G1.nodes()$ do
 6:     for $e_q$ in $G1.nodes()$ do
 7:         get node pair $(v_i, v_k)$ connected by edge $e_p$ in $G$ (first-order similarity);
 8:         get node pair $(v_j, v_k)$ connected by edge $e_q$ in $G$ (first-order similarity);
 9:         calculate similarity between $e_p$ and $e_q$ by Equation 1 (second-order similarity)
10:         $G1.add\_edges(e_p, e_q, s_{pq})$;
11:     end for
12: end for
```

3.3 Network representation learning

Edge network can not only preserve network structure and edge information, but also alleviate the sparseness in the original network greatly. As shown in Fig. 2, an original network G is transformed into an denser edge network $G1$. Then, edge network is input into the existing network representation models. Finally, edge representation vectors can be obtained directly.

In order to evaluate our method on downstream tasks, we transform edge representation vectors into node representation vectors according to their relationships. As shown in Fig. 2, v_1 is connected to only one edge e_1 in G, so v_1's representation vector can be expressed as $y(v_1) = y(e_1)$. Node v_4 is connected to the edges e_3, e_4, e_6, e_7; then, v_4's representation vector can be expressed as $y(v_4) = \frac{y(e_3)+y(e_4)+y(e_6)+y(e_7)}{4}$.

3.4 Complexity analysis

In transform process, we need to calculate the similarities between nodes in the edge network. As seen in Algorithm 1, the time complexity is $O(m^2)$, where m is the number of nodes in the edge network. However, since a large number of node pairs have no connection, so in fact, the time complexity is much smaller than $O(m^2)$.

In addition, in order to evaluate our method on downstream tasks, edge representation vectors are transformed into node representation vectors. Therefore, we need to calculate the relationship between edges and nodes in G, so the time complexity is $O(n)$. In summary, the time complexity of our method is much smaller than $O(m^2)$.

4 Experiments

4.1 Experiment settings

Our experiments focus on two common tasks: node classification and node clustering. We evaluate our method on five different networks under different inputs (original network and edge network). Node classification is also conducted on different network representation models. In order to verify the dimensional sensitivity of edge network, we cluster node representation vectors with different dimensions and compare their node clustering performance. Lastly, we classify five different edge networks, which are constructed by using different similarity measuring methods, to verity the effect of similarity measuring methods.

4.1.1 Datasets

In the experiments, we use five different real-world networks, which are processed into connected networks during preprocessing process. The detailed information has been summarized as follows:

– Polbooks[1] is a co-sold relationship network of American political books, consisting of 105 nodes and 441 edges. Nodes represent books. If two books were bought by one person, they are connected. The nodes are divided into three categories: *liberal*, *conservative* and *central*.
– Football[2] is a complex social network consisting of 115 nodes and 613 edges. Nodes represent football teams, and edges represent the two teams had a match. The nodes are divided into 12 categories.
– WebKB[3] consists of four independent subnetworks—Cornell, Texas, Wisconsin and Washington. The number of nodes and edges for each of these subnetworks are listed in Table 1. Nodes represent the site ID, and edges represent reference relationship between the sites. The nodes are divided into five categories: *course*, *faculty*, *student*, *project* and *staff*.
– ColiInter[4] is a transcription network consisting of 328 nodes and 497 edges. The edges are divided into three categories.
– Protein[5] is a network of protein interactions in yeast. It consists of 1458 nodes and 1993 edges. Nodes represent protein, and edges represent metabolic interaction between two proteins.

4.1.2 Adopted methods and experiment setup

In order to evaluate the effect of the proposed method, we use five state-of-the-art network representation models as intermediate algorithms. The details of these methods are as follows:

– Deepwalk (Perozzi et al. 2014) is the first method to introduce deep learning into network representation learning. It obtains sequences of nodes by applying random walks first and then inputs the sequences into the skip-gram model to learn a low-dimensional vector representation for each node.
– LINE (Tang et al. 2015) preserves the first-order and second-order similarities between node pairs and minimizes the KL distance between the probability distribution and the empirical distribution.
– SDNE (Wang et al. 2016) is different from the previous shallow neural networks, and it uses deep neural networks to capture the high nonlinearity between nodes and uses intermediate layer in the deep self-encoder as node representation.

1 http://www.orgnet.com/.

2 http://www-personal.umich.edu/~mejn/netdata/.

3 http://linqs.cs.umd.edu/projects/projects/lbc.

4 http://www.weizmann.ac.il/mcb/UriAlon/e-coli-transcription-network.

5 http://moreno.ss.uci.edu/data.html#Pro-pro.

Table 1 Information between original and edge network, where n is the number of nodes, m is the number of edges, C is the number of categories, D is the average degree, and P is the increasing rate of average degree between edge and original network

Method	Original network				Edge network			
	n	m	C	D	n	m	D	P
Polbooks	105	441	3	8.40	441	4822	21.87	2.60
Football	115	613	12	10.66	613	5967	19.47	1.83
WebKB-Cornell	183	280	5	3.06	280	5072	36.23	11.84
WebKB-Texas	183	295	5	3.22	295	6184	41.93	13.00
WebKB-Wisconsin	251	466	5	3.71	466	9113	39.11	10.53
WebKB-Washington	215	402	5	3.74	402	8448	42.03	11.24
ColiInter	328	497	3	3.03	497	5504	22.15	7.31
Protein	1458	1993	–	2.73	1993	12152	12.19	4.46

- Node2vec (Grover and Leskovec 2016) is an improved version of Deepwalk. It controls how to choose neighbors of a node by using two parameters.
- GraRep (Cao et al. 2015) extends to high-order proximity and uses SVD to train the model. It also directly concatenates the representations of first-order and high-order similarity.

We set the dimension size $d = 128$ for all the models, and other parameters are set to the default values presented in the publicly available implementations.

4.2 Comparison between original network and edge network

In this section, we will compare network information between original and edge networks. Average degree is used to measure the network density. The detailed information can be seen in Table 1.

As we can see, the density changes of edge network on Polbooks and Football increase less (2.6 and 1.83, respectively), which may because the original network is relatively dense, so the changes are not obvious compared to other networks. In Cornell, Texas, Wisconsin and Washington, the original networks are very sparse, and the density changes are more obvious (11.84, 13.00, 10.53 and 11.24, respectively). Both ColiInter and Protein are sparse networks with a large number of nodes. After transformation, they become denser (7.31 and 4.46, respectively).

In summary, our method can be applied to sparse networks with different node numbers and different densities, which alleviates the sparseness greatly.

4.3 Node classification

Node classification is a common task for measuring performance of node features. In order to minimize the impact of different classifiers and indicators, we use LR as classifier and accuracy as indicator. Specifically, TP, FP, TN and FN

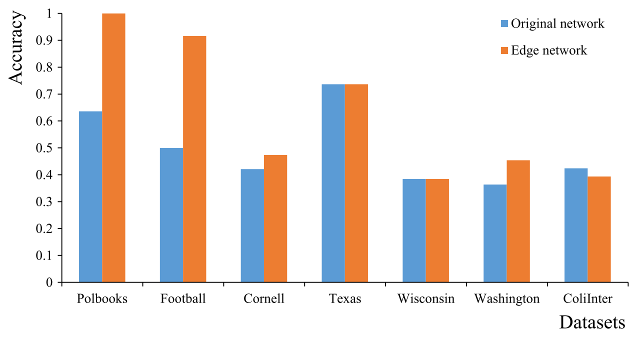

Fig. 3 Classification accuracy on different datasets with LINE model

are the number of true positives, false positives, true negatives and false negatives, respectively. Then, the accuracy is defined as Eq. 2.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{2}$$

We randomly select 90% nodes as the training set and rest as the testing set. Since ColiInter has edge labels instead of node labels, we transform node representation vectors into the edge representation vectors. In addition, there is no label in Protein, so we do not make experiments on it. Figure 3 shows the classification performance of different datasets between original and edge network under the LINE model.

As we can see in Fig. 3, the accuracy of edge network on Polbooks, Football, Cornell and Washington is certainly better than the original network. But, the accuracy of edge network on Texas, Wisconsin and ColiInter is not so good. Maybe because edge network only preserves the first-order and second-order similarities in the original network instead of higher order similarity.

Table 2 shows accuracy on Polbooks with different models. As we can see, accuracy of the edge network on Deepwalk and GraRep is the same as original network (100% both). Accuracy of LINE model in edge network is better than original network (100% and 64%, respectively). Accuracy of edge network under Node2vec is the same as original

Table 2 Classification accuracy on Polbooks dataset with different network representation models

	Deepwalk	LINE	SDNE	Node2vec	GraRep
Original network	100%	64%	**100%**	91%	100%
Edge network	**100%**	**100%**	73%	**91%**	**100%**

The best values are presented in bold

Table 3 Node clustering on different datasets with LINE model

	Cluster number	Original network	Edge network	Increase
Polbooks	3	0.0084	**0.0578**	**0.0494**
Football	12	−0.0016	**0.1912**	**0.1928**
Cornell	5	0.0066	**0.0233**	**0.0167**
Texas	5	0.0071	**0.0327**	**0.0256**
Wisconsin	5	0.0082	**0.0681**	**0.0599**
Washington	5	0.0030	**0.0605**	**0.0575**
ColiInter	3	0.0087	**0.0438**	**0.0351**
Protein	2	0.0096	**0.5248**	**0.5152**

The best values are presented in bold

network (91%), maybe because random walk sequences in Node2vec do not capture the structure of edge network well. In the SDNE model, the accuracy of the original network is better than edge network. This maybe because SDNE is a deep-learning model, which can better preserve the high-order nonlinear relationship between nodes.

4.4 Node clustering

Node clustering is also one of the important issues in network analysis. It divides the nodes into several clusters, so that nodes in the same cluster are similar and nodes in different clusters are different. We use K-means, a classical clustering method, to perform results and Silhouette Coefficient is used as evaluation indicator. Silhouette Coefficient ranges from -1 to 1, and a larger value indicates better clustering result. We also set different cluster numbers among different datasets. Experimental settings are seen in Table 3.

Table 3 shows the clustering performance of different inputs in the different networks under LINE model. On Polbooks, Cornell, Texas, Wisconsin, Washington and ColiInter, compare to original network, the increase in Silhouette Coefficient of the edge network is approximately 0.05, 0.02, 0.03, 0.06, 0.06 and 0.04, respectively, which indicates that the edge network preserves the network structure better. Football's Silhouette Coefficient of original network is negative, which maybe because the number of clusters is too large so K-means cannot distinguish the difference between nodes well. However, Silhouette Coefficient of the edge network (0.1912) is significantly better than the original network (-0.0016). This maybe because the edge network preserves rich edge information in the original network, so that K-means can better distinguish the differences between the

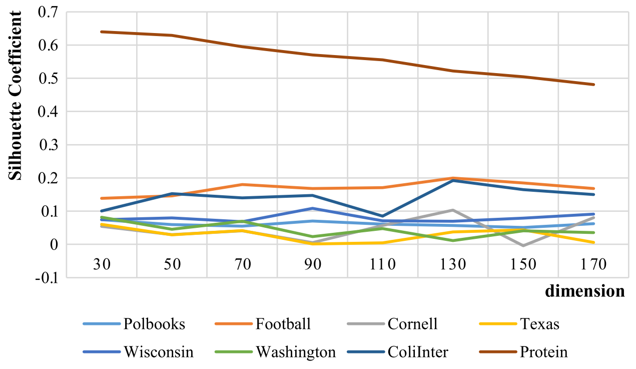

Fig. 4 Node clustering of Polbooks's edge network with different dimensions under LINE model

nodes. On Protein, Silhouette Coefficient of the edge networks is greater than original network (increasing 0.5152). This maybe because the number of nodes is large, so edge network contains richer structure and edge information.

4.5 Parameter analysis

In order to verify the sensitivity of parameters, edge networks are input into LINE model with different dimensions. Figure 4 shows the impact of different dimensions on node clustering.

As dimension increases, Protein's Silhouette Coefficient decreases, which shows that Protein can achieve good results by using low-dimensional representation vectors. As the dimension increases, Football's Silhouette Coefficient fluctuates continuously, but it shows a slow upward trend. Silhouette Coefficient of ColiInter fluctuates greatly with the increase in the dimension, and gets the best performance when dimension is 130. The trends of Polbooks and

Table 4 Comparison among original network and five different edge networks constructed by using five similarity measuring methods

	Original network	Edge network				
		Salton	AA	HPI	HDI	Sorenson
Accuracy	64%	**100%**	72%	**100%**	**100%**	**100%**

The best values are presented in bold

four subnetworks of WebKB (Texas, Cornell, Wisconsin and Washington) are similar. As the dimension increases, Silhouette Coefficient fluctuates slightly, which indicates that Polbooks and WebKB are less sensitive to the dimensions.

4.6 Different similarity measuring methods analysis

In the process of transforming an original network G into an edge network $G1$, different similarity measuring methods lead to different edge networks. In order to verify the effect of similarity measuring methods, five different similarity indices, as shown in Eqs. 1, 3–6, are compared under LINE model on Polbooks dataset. We use the LR as classifier and accuracy as indicator. Table 4 shows the classification performance of original network and five different edge networks constructed by using five different similarity methods.

$$\text{AA:} \quad s_{pq} = \sum\nolimits_{v_z \in \mathcal{N}(v_i) \cap \mathcal{N}(v_j)} \frac{1}{log\, |\mathcal{N}(v_z)|} \tag{3}$$

$$\text{HPI:} \quad s_{pq} = \frac{|\mathcal{N}(v_i) \bigcap \mathcal{N}(v_j)|}{min\{|\mathcal{N}(v_i)|\,,\, |\mathcal{N}(v_j)|\}} \tag{4}$$

$$\text{HDI:} \quad s_{pq} = \frac{|\mathcal{N}(v_i) \bigcap \mathcal{N}(v_j)|}{max\{|\mathcal{N}(v_i)|\,,\, |\mathcal{N}(v_j)|\}} \tag{5}$$

$$\text{Sorenson:} \quad s_{pq} = \frac{|\mathcal{N}(v_i) \bigcap \mathcal{N}(v_j)|}{|\mathcal{N}(v_i)| + |\mathcal{N}(v_j)|} \tag{6}$$

Node pair (v_i, v_k) is connected by edge e_p, and (v_j, v_k) is connected by edge e_q. Equation 3 (Salton 1970) is Adamic–Adar index(AA). Equation 4 (Ravasz et al. 2002) is Hub Promoted Index(HPI). Equation 5 (Leicht et al. 2006) is Hub Depressed Index(HDI). Equation 6 (Li et al. 2014) is Sorensen index.

As shown in Table 4, the performance of edge network constructed by using Salton, HPI, HDI and Sorenson is great(100%) with increasing rate of 56.25% compared to original network (64%). While accuracy of edge network constructed by using AA is 72% with increasing rate of 12.5% compared to original network (64%). This maybe because Salton, HPI, HDI and Sorensen consider the common neighbors, and AA also considers degree information of common neighbors, while nodes v_i and v_j have a few common neighborhoods in the dataset; therefore, AA cannot obtain enough valuable information.

5 Conclusion

We proposed a network representation learning method based on edge information extraction, which not only can preserve the structure and edge information in the original network, but also alleviate the sparseness. First, an original network is transformed into an edge network, and then, input edge network into an existing network representation model. Finally, edge representation vectors of original network can be obtained directly. Evaluation on real-world datasets demonstrates that edge network can achieve better performance than original network in most cases.

In the future, we will explore the following directions:

1. It can be seen in the experimental part that high-order similarity plays an important role in learning representation of sparse networks, but our method fails to preserve high-order similarity between nodes in the original network. So, how to better preserve the network structure is one of our future work directions.
2. We mainly analyze the undirected and unweighted network in this paper, but there are also a large number of directed and weighted real-world networks. Therefore, the analysis of those networks is necessary to understand complex networks.
3. Heterogeneous networks have more complex structures, and many real-world networks are heterogeneous. These networks also contain rich information, such as attribute information, tag information, text information and so on. How to save complex structural information and other information in heterogeneous networks is also an important issue to be considered.

Acknowledgements This work is partially supported by Grants from the National Natural Science Foundation of China (U1333109), Fundamental Research Funds for the Central Universities of Civil Aviation University of China (3122018C020, 3122018C021), Scientific Research Foundation of Civil Aviation University of China (600/600001050115, 600/600001050117) and a fund from the Hong Kong Polytechnic University, Department of Industrial and Systems Engineering (H-ZG3K).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Belkin M, Niyogi P (2002) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in neural information processing systems (NIPS), pp 585–591

Cao S, Lu W, Xu Q (2015) Grarep: learning graph representations with global structural information. In: Proceedings of the 2015 ACM on conference on information and knowledge management (CIKM), ACM, pp 891–900

Grover A, Leskovec J (2016) Node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (KDD), ACM, pp 855–864

Hoang DT, Nguyen NT, Tran VC, Hwang D (2018) Research collaboration model in academic social networks. Enterp Inf Syst 13(7–8):1023–1045

Hong R, He Y, Wu L, Ge Y, Wu X (2019) Deep attributed network embedding by preserving structure and attribute information. IEEE Trans Syst Man Cybern Syst pp 1–12

Hu W, Gong Z, LH U, Guo J (2015) Identifying influential user communities on the social network. Enterp Inf Syst 9(7):709–724

Leicht EA, Holme P, Newman ME (2006) Vertex similarity in networks. Phys Rev E 73(2):026120

Li Y, Luo P, Wu C (2014) Information loss method to measure node similarity in networks. Phys A 410:439–449

Liao L, He X, Zhang H, Chua TS (2018) Attributed social network embedding. IEEE Trans Knowl Data Eng 30(12):2257–2270

Luo D, Nie F, Huang H, Ding CH (2011) Cauchy graph embedding. In: Proceedings of the 28th international conference on machine learning (ICML), ACM, pp 553–560

Mikolov T, Chen K, Corrado G, Dean J (2013a) Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013b) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems (NIPS), pp 3111–3119

Ou M, Cui P, Pei J, Zhang Z, Zhu W (2016) Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (KDD), ACM, pp 1105–1114

Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD), ACM, pp 701–710

Qu M, Tang J, Shang J, Ren X, Zhang M, Han J (2017) An attention-based collaboration framework for multi-view network representation learning. In: Proceedings of the 2017 ACM on conference on information and knowledge management (CIKM), ACM, pp 1767–1776

Ravasz E, Somera AL, Mongru DA, Oltvai ZN, Barabási AL (2002) Hierarchical organization of modularity in metabolic networks. Science 297(5586):1551–1555

Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326

Salton G (1970) Automatic text analysis. Science 168(3929):335–343

Shi Y, Zhu Q, Guo F, Zhang C, Han J (2018) Easing embedding learning by comprehensive transcription of heterogeneous information networks. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining (KDD), ACM, pp 2190–2199

Sun Y, Wang S, Hsieh TY, Tang X, Honavar V (2019) Megan: a generative adversarial network for multi-view network embedding. In: Proceedings of the 28th international joint conference on artificial intelligence (IJCAI), AAAI Press, pp 3527–3533

Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q (2015) Line: large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web (WWW), ACM, pp 1067–1077

Tu C, Zhang W, Liu Z, Sun M, et al. (2016) Max-margin deepwalk: discriminative learning of network representation. In: Proceedings of the 25th international joint conference on artificial intelligence (IJCAI), AAAI Press, pp 3889–3895

Tu C, Liu H, Liu Z, Sun M (2017a) Cane: context-aware network embedding for relation modeling. In: Proceedings of the 55th annual meeting of the association for computational linguistics (ACL), pp 1722–1731

Tu C, Zhang Z, Liu Z, Sun M (2017b) Transnet: translation-based network representation learning for social relation extraction. In: Proceedings of the 26th international joint conference on artificial intelligence (IJCAI), AAAI Press, pp 2864–2870

Van Der Maaten L (2014) Accelerating t-sne using tree-based algorithms. J Mach Learn Res 15(1):3221–3245

Wang D, Cui P, Zhu W (2016) Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (KDD), ACM, pp 1225–1234

Wang S, Hu L, Cao L (2017a) Perceiving the next choice with comprehensive transaction embeddings for online recommendation. In: Joint European conference on machine learning and knowledge discovery in databases (ECML), Springer, pp 285–302

Wang X, Cui P, Wang J, Pei J, Zhu W, Yang S (2017b) Community preserving network embedding. In: Proceddings of the 31st AAAI conference on artificial intelligence (AAAI), pp 203–209

Wang Y, Ding M, Chen Z, Luo L (2017c) Caching placement with recommendation systems for cache-enabled mobile social networks. IEEE Commun Lett 21:2266–2269

Yang C, Liu Z, Zhao D, Sun M, Chang E (2015) Network representation learning with rich text information. In: Proceedings of the 24th international joint conference on artificial intelligence (IJCAI), AAAI Press, pp 2111–2117

Yang H, Pan S, Zhang P, Chen L, Lian D, Zhang C (2018) Binarized attributed network embedding. In: Proceedings of the 2018 IEEE international conference on data mining (ICDM), IEEE, pp 1476–1481

Yang H, Pan S, Chen L, Zhou C, Zhang P (2019) Low-bit quantization for attributed network representation learning. In: Proceedings of the 28th international joint conference on artificial intelligence (IJCAI), AAAI Press, pp 4047–4053

Zedan S, Miller W (2017) Using social network analysis to identify stakeholders' influence on energy efficiency of housing. Int J Eng Bus Manag 9:1–2