Computing and Informatics, Vol. 41, 2022, 1563–1588, doi: 10.31577/cai_2022_6_1563

REPRESENTATION LEARNING METHOD OF GRAPH CONVOLUTIONAL NETWORK BASED ON STRUCTURE ENHANCEMENT

Ningchen Fu
Shanghai Engineering Research Center of Intelligent Education and Bigdata
Shanghai Normal University, Shanghai 200234, China

Qin Zhao*
Shanghai Engineering Research Center of Intelligent Education and Bigdata
Shanghai Normal University, Shanghai 200234, China
&
The Research Base of Online Education for Shanghai Middle and Primary Schools
Shanghai Normal University, Shanghai 200234, China
&
Key Laboratory of Embedded Systems and Service Computing
of Ministry of Education,Tongji University, Shanghai 200092, China
e-mail: q_zhao@shnu.edu.cn

Yaru Miao
Shanghai Engineering Research Center of Intelligent Education and Bigdata
Shanghai Normal University, Shanghai 200234, China

Bo Zhang
Shanghai Engineering Research Center of Intelligent Education and Bigdata
Shanghai Normal University, Shanghai 200234, China
&
The Research Base of Online Education for Shanghai Middle
and Primary Schools, Shanghai Normal University, Shanghai 200234, China

Dong Wang
School of Computer Science and Information Engineering
Shanghai Institute of Technology, Shanghai 201418, China
e-mail: dongwang@sit.edu.cn

* Corresponding author

Abstract. Network representation learning has attracted widespread attention as a pre-processing process for some machine learning and deep learning tasks. However, most existing methods only consider influence of nodes' low-order neighbors to represent them. Either nodes' high-order neighbor or the intrinsic characteristic attributes of nodes are ignored, leading to the effect of network representation learning that needs to be improved. This paper proposes a novel model named Structure Enhanced Graph Convolutional Network (SEGCN) to address these limitations. SEGCN consists of the following components, i.e., the network structure enhancement to transform weak relationship into strong relationship, the node feature aggregation to fuse high-order neighbor information. Hence, the SEGCN model can simultaneously integrate network structure information, attribute information, and high-order neighbor relationships together. Experimental results for node classification and node clustering on six datasets show that SEGCN achieves better effectiveness and efficiency than state-of-the-art baselines.

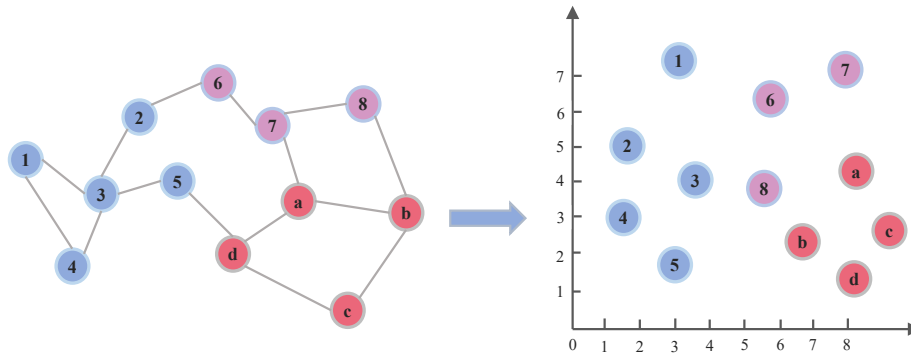
Keywords: Network representation learning, graph convolutional network, deep learning

Mathematics Subject Classification 2010: 68-T30

1 INTRODUCTION

Graphs are ubiquitous data structure employed in many deep learning tasks [1], such as paper citation networks [2], social networks [3], e-commerce networks [4], and so on. There are a large number of graph-based algorithms and applications have been derived, including information recommendation [5, 6], link prediction [7], and traffic risk forecast [8], etc. Obviously, as a prior machine learning task, how to efficiently and accurately learn and represent a complex network has become a top priority. The key of network representation learning (Figure 1) is to embed the nodes in the network with given network structure information or attribute information, and embed nodes of high-dimensional sparse space into a low-dimensional dense representation vector [9]. Traditional approaches usually use the structural information about given networks for embedding, which allows structurally similar nodes to have similar vector representations. However, these methods ignore the inherent characteristics of nodes. In recent years, one of the most popular deep learning methods is the graph convolutional neural network, which aggregates the information of a node's neighbors to improve its vector representation. Nevertheless, the deep-learning-based methods only consider nodes' local adjacency relationship but ignore the impact of their high-order neighbors.

Hence, this paper proposes the SEGCN framework to integrate the local structure, global structure and node characteristics of the network to improve the effect of network representation learning. The overall architecture of SEGCN is illustrated



a) Input: Complex network b) Output: Vertex representations

Figure 1. Network representation learning

in Figure 2. For an initial complex network, considering its structure and attributes, we compute the structural similarity and attribute similarity, respectively. The autoencoder is used to non-linearly fuse the two similarities, and then based on the similarity threshold comparison method, the structure of the whole network is enhanced, new edges arise in the network, and the original sparse network will become a dense network. The central node and its high-order neighbor nodes may be linked, and the graph convolutional neural network is further employed to finish the feature aggregation of nodes in the network, and a high-quality node vector representation is generated. Based on the structure enhancement algorithm, the graph's adjacency structure is enhanced to capture high-order neighbor relationships, and the enhanced weighted adjacency matrix is input into the graph convolutional neural network model, then the network's high-order topological structure and the inherent characteristics of the node are merged to realize the representation learning of the node. Experimental results on six datasets show the effectiveness and efficiency of the algorithm proposed in this paper.

In summary, this work makes several major contributions:

1. An autoencoder is used to fuse the nonlinear relationship between structural similarity and attribute similarity to obtain comprehensive similarity, and the similarity threshold method is further used to enhance the network structure to capture high-order neighborhood relationships.
2. Based on the structure enhancement algorithm and one-layer GCN network, the over-smoothing problem is alleviated and high-quality node representation learning is obtained.
3. We conduct extensive experiments on some datasets for node classification and node clustering, experiments on all of those datasets and tasks show that SEGCN model is consistently better than those state-of-the-art baselines or deep learning methods.

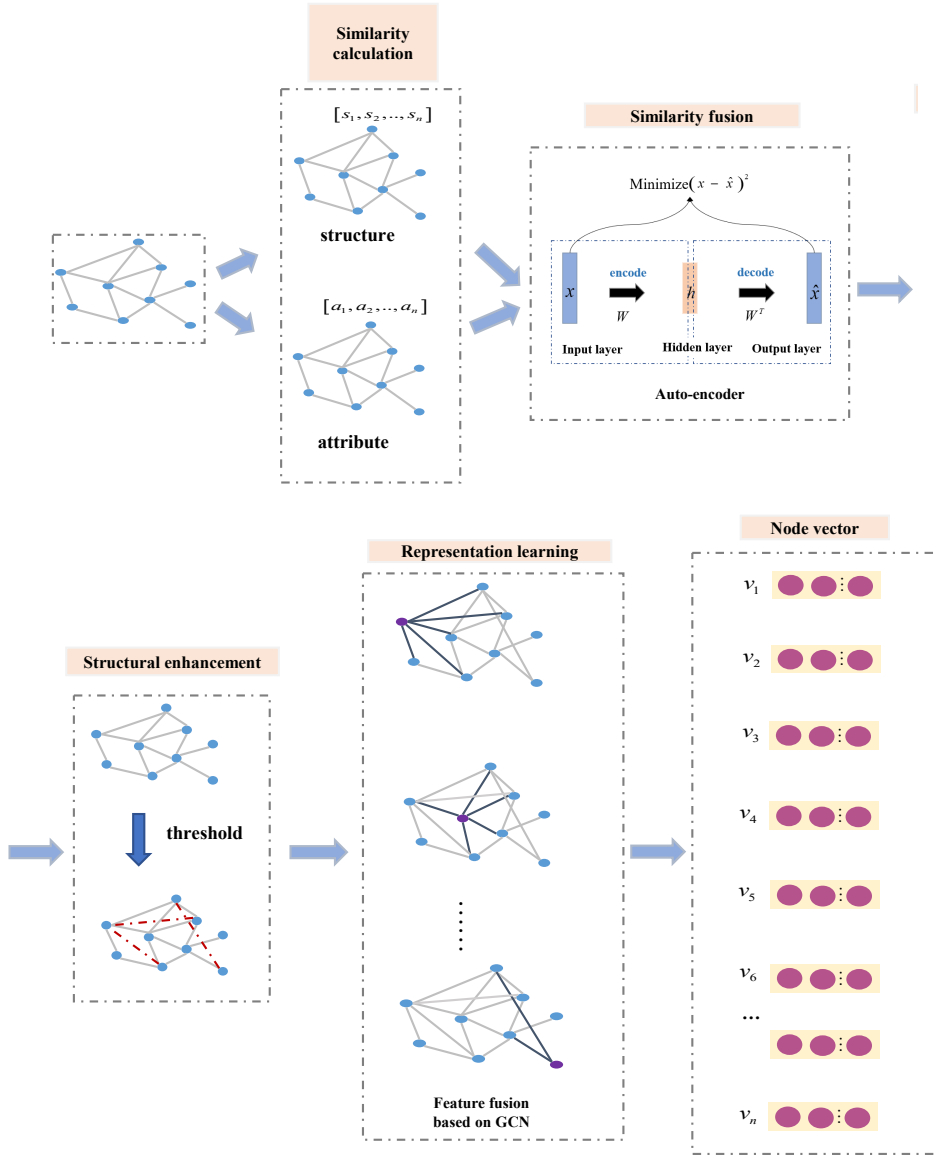


Figure 2. The overall architecture of SEGCN

The remainder of this article is structured as follows. Section 2 briefly introduces the methods for representation learning and the existing problems, Section 3 presents our methods and elaborates, Section 4 evaluates the proposed method and analyzes the experimental results. Section 5 discusses the conclusion and future works.

2 RELATED WORK

In this section, we mainly review some algorithms of network representation learning. They are organized into two subsections. Section 2.1 summarizes some traditional network representation learning methods, while Section 2.2 introduces some deep learning methods for graph embedding.

2.1 Traditional Representation Learning Methods

The purpose of network representation learning is to map a group of nodes in a high-dimensional space to a low-dimensional space, and each node is marked as a low-dimensional dense vector. The low dimensional node representation vector group can be located in the vector space and has the ability of representation and reasoning, which can be used for downstream machine learning or deep learning tasks, such as node classification, node clustering, link prediction, recommendation system, etc. Traditional network representation learning algorithms are mainly based on factorization or shallow network structure.

Some factorization based methods [10, 11] regard an adjacency matrix A as the adjacency relationship between any node in the network and other nodes. By decomposing this adjacency matrix, those similar nodes in the network structure can also have similar node representation after matrix decomposition. Locally Linear Embedding (LLE) [12] assumes that the nodes in the network are a set of linear representations of the neighbor node vectors in the network, and aims to find a set of normalized weight coefficients so that the node can be weighted by a group of neighbor nodes. However, this method is depending on the choice of neighbor nodes, and different neighbor nodes will have an important impact on the representation of the node. Laplacian Eigenmaps [13] regard the representation of the node as a problem of the eigenvalue decomposition of the Laplacian matrix, and the eigenvector obtained by decomposing the Laplacian matrix is regarded as the embedding representation of the node. However, the time complexity of this method is too high. When the network structure is large, such as an e-commerce network containing millions of nodes, it will have a large time cost, so it is not suitable for large-scale networks.

Methods based on shallow networks mostly consider the low-level topology of the network, and then combine some bag-of-words models or natural language processing methods to complete node embedding. Based on the idea of random walk, DeepWalk [14] uses a depth-first search strategy to find a node sequence $s = \{v_1, v_2, \ldots, v_k\}$ that contains the network topology, and then feeds the sequence

into the Skip-Gram model. The central node and the context node in the sequence generate the representation vector of the node. LINE [15] considers both the first-order similarity and the second-order similarity. The first-order similarity is regarded as the directly connected local similarity, and the second-order similarity is used to compensate for the sparsity of the first-order similarity, so that nodes with similar first-order similarity and second-order similarity also have similar node representations. Node2vec [16] is different from DeepWalk in that it uses a breadth-first search strategy to sample the sequence of nodes, while observing the microscopic view around the nodes, which can make the nodes with structural equivalence closer in the embedding space.

Although the above network representation learning methods can use the topology information in the network to embed and represent the nodes, some ignore the high-order neighbor relationship of the network structure or do not integrate the inherent characteristics of the node, resulting in the performance of the representation learning needs to be further improved.

2.2 Deep Learning for Representation Learning

The rapid development of deep learning in computer vision has also promoted the application of deep learning in natural language processing and social networking. Using deep learning technology to learn representations of nodes in the network has become a concern of many researchers. Structural Deep Network Embedding (SDNE) [17] uses unsupervised learning to obtain the latent representation of the vertices by forming an AutoEncoder based on the neighbor structure of the vertices, which will effectively retain the second-order similarity of the network. At the same time, in order to retain the first-order similarity, it is of great significance to use the method of Laplacian feature mapping to supervise and adjust the embedding representation of the node according to the prior knowledge represented by the adjacency matrix.

In the real world, many irregular data do not have a network structure that can only be represented in the form of graphs. Therefore, extending the idea of neural networks to graph structure data has attracted great attention from researchers. Convolutional neural networks for graph structure are mainly divided into two aspects: graph convolution based on spectral domain or based on spatial domain.

For spectral domain graph convolution, Bruna et al. [18] introduced the CNN on the graph for the first time to perform eigenvalue decomposition on the Laplacian matrix of the graph, and then regard the decomposed eigenvectors as a set of basis for Fourier Transform. However, when the network is large, the time cost for decomposition of the Laplacian matrix is too high. Because the K-order Chebyshev needs to multiply the Laplacian matrix on the graph for k times, the Laplacian matrix under the k-fold multiplication can represent the K-order neighbor relationship on the graph, so Defferrard et al. [19] introduced Chebyshev polynomials to approximate graph convolution operations in the spectral domain. In 2020, Abu-El-Haija

et al. [20] proposed to train multiple GCN instances on the node pairs found at different distances in the random walk, and learn to optimize the output combination of the instances of the classification target, and finally achieve better results.

For spatial graph convolution, the convolution operation is directly defined on a group of similar nodes in space, and the representation of a node can be regarded as a weighted combination of a group of neighbor nodes. The weight learning between the central node and neighbor nodes has recently attracted more and more attention. GAT [21] incorporates the attention mechanism into the aggregator function to take into account the relative importance of each neighbor's information from the target node's perspective.

Due to the natural ability to integrate topological structure and inherent features, the network representation learning method based on graph convolution has significantly improved the representation effect of nodes compared with traditional methods. Since each convolution of GCN only involves first-order neighbors, if we want to integrate the features of K-order neighbors, we need to perform k times convolution operations. However, high-level graph convolution will have overfitting problems, so most of the existing graph convolution models are within two or three layers. In other words, it can not effectively capture high-order neighbors' features.

2.3 Existing Problems

While both conventional and deep learning-based approaches have made significant contributions to representation learning tasks, there are notable drawbacks.

1. It is challenging to capture high-order local interactions in some sparse networks. To use the citation network as an example, the majority of paper citation networks are sparse, meaning that each article is referenced in around a dozen or fewer documents, yet the total citation network may include thousands of documents. There may be no citation link between articles on the same subject, making it impossible to incorporate effective high-order neighborhood node information during representation learning tasks, resulting in low-quality representations.
2. In addition to the complex network's topological structure connection, the nodes in the network have a wealth of attribute qualities. The conventional approach of representation learning retrieves the node representation vector from the network topology, but ignores the influence of node attributes or intrinsic features on representation learning. While some deep learning algorithms use node characteristics to some degree, they neglect the effect of high-order neighbor nodes on the representation learning process.

As a result, this article proposes the SEGCN model. On the one hand, it is capable of capturing high-order neighborhood node information through structural enhancement, and on the other hand, it is capable of efficiently integrating the network's topological structure and node attribute information to achieve a higher representation learning effect.

3 PROPOSED METHOD

This paper proposes a Graph Convolutional Neural Network model named SEGCN based on the Structure Enhancement algorithm. The innovation of this model lies in the design of a structural enhancement algorithm. The algorithm can not only integrate the nonlinear relationship between structural similarity and attribute similarity, but also capture the influence of higher-order neighbor relationships on node representation. The model in this paper has been tested on multiple datasets, showing that the SEGCN model can outperform the mainstream network representation learning methods.

3.1 Basic Definition

Given an undirected graph $G = (V, E, A)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of nodes, E is an edge set that can be represented by adjacency matrix $E = (e_{ij})_{n\times n} \in R^{n\times n}$. $A = (a_1, a_2, \ldots, a_n)^{\mathrm{T}} \in R^{n\times d}$ is a feature representation matrix of all nodes, where $a_i \in R^d$ represents the feature representation vector of node i. $D = \{d_{ij}\} \in R^{n\times n}$ is a diagonal matrix, representing the degree of all nodes in the graph, which can reflect the local influence of the nodes in the network, where $D_{ii} = \sum_{j=1}^{n} d_{ij}$. If there is an edge between node v_i and v_j, then $d_{ij} = 1$; otherwise, $d_{ij} = 0$.

3.2 Network Structure Enhancement

The existing network representation learning is mainly based on strong relationships, namely first-order neighbor relationships. Since weak relationships [22] exist between nodes that are not directly connected, this kind of relationships forms an "information bridge", which makes the information of one node spreading to another node. This information dissemination will also affect the representation of the node. Therefore, in complex networks, the weak relationship based on higher-order neighbors is also worthy of attention. In this part, we use the autoencoder to integrate node similarity and attribute similarity to enhance the network structure. Specifically, given a similarity threshold, if the similarity between two nodes exceeds the threshold, it indicates a necessary mutual influence relationship between the two nodes, and a new edge is added between the two nodes. The weak relationship between the two nodes will transform into a strong relationship. Meanwhile, the similarity value represents the weight of the relationship between the two nodes.

To discover weak relationships among nodes, we use SimRank [23] to calculate the structural similarities among nodes by following formula:

$$sim_{stru}(v_i, v_j) = \begin{cases} 1, & i = j, \\ \frac{C}{|O(v_i)||O(v_j)|} \sum_{a\in O(v_i)} \sum_{b\in O(v_j)} sim_{stru}(a, b), & \text{otherwise}, \end{cases} \tag{1}$$

where $O(v_i)$ and $O(v_j)$ represent the out-degrees of the two nodes, respectively, and C is the damping coefficient.

We iteratively calculate the structural similarities among nodes. We only calculate the similarity among the unconnected nodes. After the process is over, for each node in the initial network, an n-dimensional vector $stru_i \in R^n$ will be obtained to indicate the similarity between a node in the network and the remaining $n-1$ nodes.

SimRank only calculates the structural information of nodes, but ignores the attribute information of them. Therefore, we use cosine similarity to calculate the attribute similarities among nodes as below. After the process is over, for each node in the initial network, an n-dimensional vector $attr_i \in R^n$ will be obtained to indicate the attribute similarity between a node in the network and the remaining $n-1$ nodes.

$$sim_{attr}(i,j) = \begin{cases} 1, & i = j, \\ \frac{\sum_{k=1}^{n}(X_i \cdot Y_i)}{\sqrt{\sum_{i=1}^{n}(X_i)^2} \cdot \sqrt{\sum_{i=1}^{n}(Y_i)^2}}, & \text{otherwise.} \end{cases} \tag{2}$$

After calculating the strucural and attribute similarities, we obtain the structural similarity and attribute similarity between nodes. Since structural similarity and attribute similarity are not linear, if the two parts of similarity are directly weighted and fused, on the one hand, it does not conduce to extracting the nonlinear relationship between them; on the other hand, it will cause information redundancy. Therefore, we use the autoencoder to capture the nonlinear relationship between structural similarity and attribute similarity, and to better integrate the two parts to obtain a comprehensive similarity. The process of using the autoencoder to integrate structure and attribute similarity to obtain the comprehensive similarity between nodes is shown in Figure 3.

The autoencoder network [24] (Figure 4) consists of an encoder and a decoder. The encoder converts the original input data x into an intermediate hidden layer vector h. It is a process of dimensionality reduction that can capture the nonlinear relationship between features. Then the hidden layer vector of the middle layer is remapped to the same space as the initial input dimension, and a vector $\hat{x}$ is obtained, which having the same dimension as x.

In our model, the encoder function and decoder function are as follows:

$$h_i = \sigma(W^H x_i + b^1), \tag{3}$$

$$\hat{x}_i = \sigma(W^H h_i + b^2) \tag{4}$$

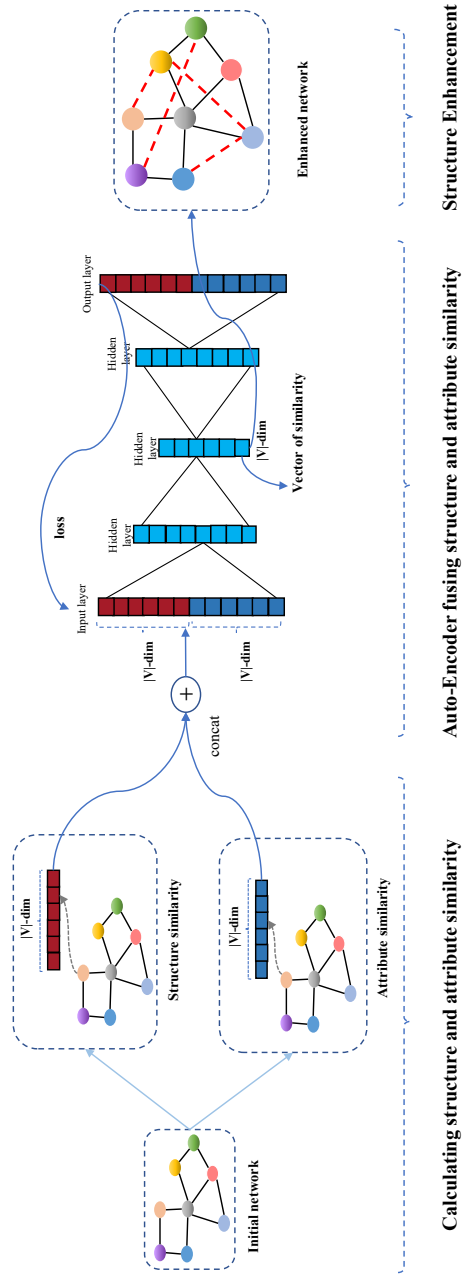


Figure 3. The process of structure enhancement

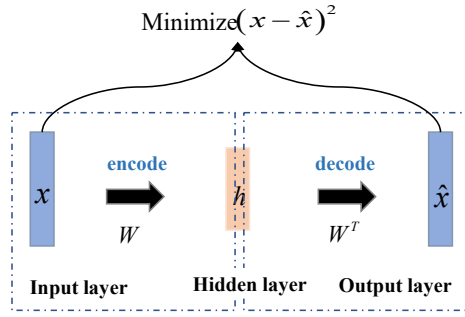


Figure 4. The process of structure enhancement

where W and b are network parameters, σ is the activation function, and ReLU is employed here. Meanwhile, we employ the Mean Square Error as the loss function of the autoencoder.

$$\text{loss}(X, \hat{X}) = \min \sum_{i=1}^{n} (x_i - \hat{x}_i)^2. \tag{5}$$

The loss function converges when the initial input vector and output vector are infinitely close, and the autoencoder reaches the ideal state.

For node i, a structural similarity vector $stru_i \in R^n$ and an attribute similarity vector $attr_i \in R^n$ will be obtained. Concat the two to get the input $x_i \in R^{2n}$ of the autoencoder. When the autoencoder converges, the output of the hidden layer is the vector representation of the comprehensive similarity that combines the structure similarity $stru_i$ and the attribute similarity $attr_i$, which is $sim_i \in R^n$. When all the nodes are processed, a similarity matrix $SIM = (sim_{ij})_{n\times n} \in R^{n\times n}$, where sim_{ij} represents the comprehensive similarity between node i and j.

It is worth noting that for the original relationship in the network, the two nodes initially have connected edges, we consider it to be a strong relationship and do not deal with it. For node i and j which have no relationship between them at the beginning, with a given similarity threshold t, if $sim_{ij} \geqslant t$, it is considered that node i and node j are similar enough that they can influence each other, then an edge is added between the two nodes, and the weak relationship is transformed into a strong relationship. The weights of their relationship are calculated as follows:

$$rel_{ij} = \begin{cases} 1, & i = j, \\ 1, & attr_{ij} = 1, \\ sim_{ij}, & sim_{ij} > t, \\ 0, & \text{otherwise}. \end{cases} \tag{6}$$

After the above process, we have obtained a structure-enhanced network. Even though there may not be a strong relationship between two nodes in the initial

network, because they are sufficiently similar in structure and attributes, this weak relationship eventually turns into a strong relationship. For example, for node 2 and node 5 in the Figure 5, there is no connection between them, but after strengthening the structure, a new connection appears between them. After the processing of the structure enhancement algorithm, the adjacency relationship of the entire graph can be represented by $L = (rel_{ij})_{n\times n} \in R^{n\times n}$, combined with the initial feature matrix $A \in R^{n\times d}$ of the network, and sent to the graph convolutional neural network for representation learning. The overall structure of the model is shown in Figure 2.

3.3 Node Embedding Based on SEGCN

The existing Graph Convolutional Neural Network (GCN) have achieved good results in network representation learning tasks. However, since only the local topological structure of the initial adjacency matrix is considered, and the global topological structure information formed by high-order neighbors is ignored, the loss of information will eventually affect the performance of the network representation learning. Therefore, based on the similarity matrix calculated above, we follow Kipf and Welling [25] and propose a Graph Convolutional Neural Network based on Structure Enhancement (SEGCN). Figure 5 illustrates the process of utilizing GCN to complete node representation learning by using the generated structure-enhanced network.

In order to weaken the fluctuations caused by the excessive weight of some nodes, we first normalize the similarity matrix so that the aggregated features roughly maintain the same proportions as the input features, and also add self-loops to the similarity matrix to alleviate the over-smoothing problem.

$$S = \tilde{D}^{-1/2}\tilde{L}\tilde{D}^{-1/2} \tag{7}$$

where $\tilde{L} = L + I$, I is the identity matrix and $\tilde{D}$ is the degree matrix. The characteristics of the nodes are updated by fusing their own information and the characteristics of neighbor nodes, and use different weights to distinguish the contributions of neighbor nodes to the central node.

$$x_i^{l+1} = \left(s_{ii}x_i^l + \sum_{j\in N(i)} s_{ij}x_j^l \right) \tag{8}$$

where s_{ii} represents the weight of the node itself, and x_i^l is the output node representation in the upper layer network. $N(i)$ represents the set of neighbors of node i after structural enhancement, s_{ij} represents the similarity between node i and j, and is used here to represent the contribution of node j to node i. By fusing the information of the central node and neighbor nodes of the l^{th} layer, we get the information of the $(l+1)^{\text{st}}$ layer node. Since the adjacency matrix generated by the

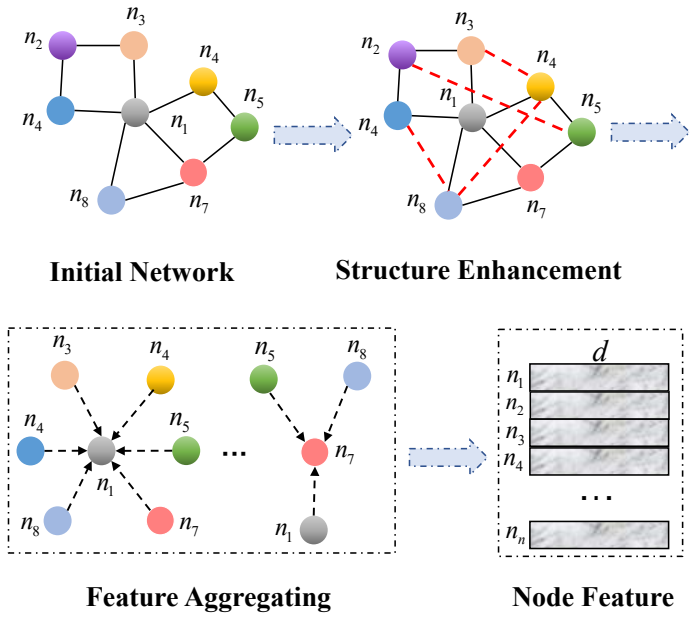


Figure 5. Use GCN to fuse feature information of neighbor nodes

structural enhancement algorithm can describe the relationship of high-order neighbors, a layer of GCN network can be used here to fuse the feature information of high-order neighbors, and at the same time, the over-smoothing problem caused by multi-layer GCN network can be alleviated. After that, the feature vectors of all nodes are spliced and transposed to obtain a feature representation matrix.

$$X^{(l+1)'} = Concat(x_1', x_2', \ldots, x_n')^T. \tag{9}$$

The feature representation matrix of the $(l+1)^{\text{st}}$ layer obtained by the above formula is high-dimensional and sparse, and then a linear transition matrix is introduced to reduce the dimension and use ReLU [26] as the activation function, and finally a feature representation matrix that can be obtained.

$$X^{(l+1)} = Relu\left(X^{(l+1)'}W\right). \tag{10}$$

Our model is finally trained on a semi-supervised learning task. We use the Cross Entropy function [27] as the loss function and use back propagation to update the parameters.

$$Loss = -\sum_{v \in V_{label}} \sum_{c=1}^{C} y_v[c] \cdot \log\ x_v^{l+1}[c] \tag{11}$$

where V_{label} is the set of nodes that have labels, C is the number of classes, y_v is the one-hot label vector of node v, and x_v^{l+1} is the predicted probability vector of node v.

4 EXPERIMENTS

In this section, we perform extensive experiments to evaluate the effectiveness of our proposed SEGCN on six real graph datasets with ground-truth classes. Our algorithms are implemented in Python and Pytorch, and all experiments are conducted on Ubuntu 14.04.6 with GeForce GTX 1080 GPU and 512 GB main memory.

4.1 Datasets

In order to verify the effectiveness of the SEGCN model proposed in this paper, not only experimental verification is carried out on the citation network datasets Cora and Citeseer, but also four public real small attribute networks (Cornell, Texas, Washington and Wisconsin) are selected to verify the experimental effect of the model. Among them, for Cora and Citeseer, each paper in the dataset is represented as a node, and the citation relationship between papers is represented as an edge relationship between nodes. Each paper uses a high-dimensional one-hot encoded vector to describe the text content. Taking the Cora dataset as an example, 1 433 unique words are selected as the corpus. If a word appears in the paper, the index position of the word is 1, otherwise it is 0. Finally, a 1 433-dimensional paper text representation vector is obtained.

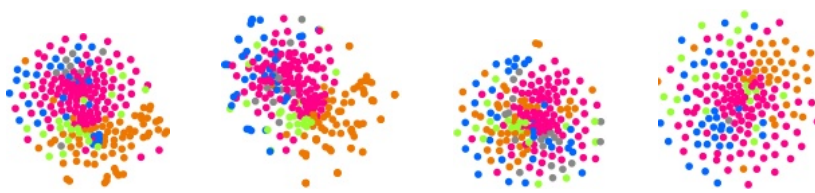


Figure 6. t-SNE embedding and visualization

We use the t-SNE [28] method to view the distribution of the original data. As can be seen from Figure 6, these nodes have begun to show clustering effects on the two-dimensional plane. The reason may be that the distribution of nodes with similar characteristics is denser, and the distribution of larger differences is also more scattered. Some statistics of our datasets are shown in Table 1.

Dataset	$\|V\|$	$\|E\|$	Dimensions	Clusters	Average Degree
Cornell	195	283	1 703	5	1.5 %
Texas	187	280	1 703	5	1.6 %
Washington	230	366	1 703	5	1.4 %
Wisconsin	265	459	1 703	5	1.3 %
Cora	2 708	5 278	1 433	7	0.14 %
Citeseer	3 327	4 552	3 703	6	0.8 %

Table 1. Datasets statistics. $|V|$: number of nodes, $|E|$: number of edges, dimensions: number of node attributes, clusters: number of classes.

4.2 Model Evaluation

In order to better evaluate our model, we have selected two evaluation indicators, Accuracy (AC) and Normal Mutual Information (NMI) [29]. AC is used to evaluate the accuracy of node classification. NMI is used to measure the clustering coefficient of nodes, which can reflect the quality of node clustering. The value range of NMI is 0 to 1. The larger the value of the NMI, the better the clustering effect.

4.3 Comparison Against State-of-the-Art Baselines

We compared SEGCN with six state-of-the-art algorithms, incluing SNMF [30], NC [31], PCL-DC [32], SCI [33], CDE [34], GCN [25]. These algorithms have achieved good results in some node classification or community detection. We mainly compare the AC and NMI indicators with these methods to detect the performance of our model for network representation learning.

SNMF. Use the idea of matrix decomposition to decompose the matrix, and obtain the probability that the node belongs to a certain community through continuous iterative optimization.

NC. Considering the network as a graph, and continuously divide it to maximize the similarity between nodes in the sub-graphs, while the similarity between the sub-graphs is kept to a minimum.

PCL-DC. The content analysis and links are considered together for community detection, and a discriminant model of content analysis is designed to reduce the influence of not wanting to manage content attributes.

SCI. Based on matrix decomposition, combined with the observed network structure and node attributes, only the adjacency matrix is decomposed, and no attention is paid to the decomposition of the node attribute matrix.

CDE. The inherent community structure is coded on the basis of the original network structure, and the community discovery problem is regarded as a non-negative matrix factorization problem.

GCN. The convolution operation is applied to unstructured graph data, and utilize the idea of message passing and information aggregation for node embedding.

	Dataset	Cornell	Texas	Washington	Wisconsin	Cora	Citeseer
AC	SNMF	0.3692	0.4019	0.3009	0.3773	0.4323	0.3079
	NC	0.3538	0.4545	0.4348	0.3170	0.2622	0.4094
	PCL-DC	0.3512	0.3850	0.4608	0.3773	0.5823	0.4682
	SCI	0.4769	0.6096	0.3773	0.5283	0.4121	0.3260
	CDE	**0.6154**	0.6150	0.5823	0.7321	0.6555	0.5827
	GCN	0.3692	0.6385	0.6122	0.4553	0.7782	0.6461
	SEGCN	0.4989	**0.8932**	**0.6547**	**0.8667**	**0.7961**	**0.6563**

Table 2. Quality evaluation (in terms of AC) on networks with ground-truth classes

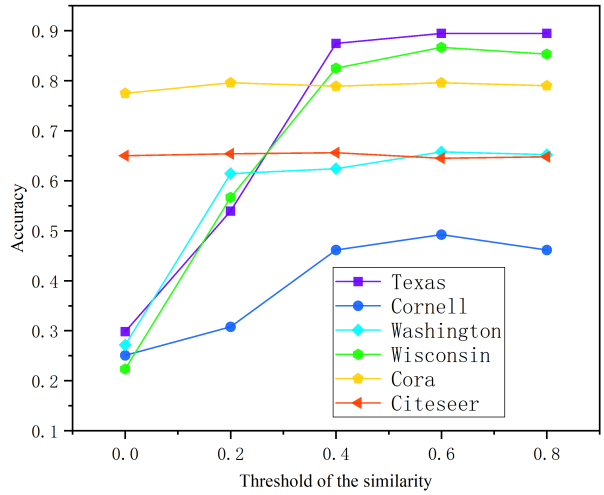
	Dataset	Cornell	Texas	Washington	Wisconsin	Cora	Citeseer
NMI	SNMF	0.0762	0.1022	0.0321	0.0842	0.2996	0.1044
	NC	0.0855	0.0706	0.0591	0.0507	0.1731	0.1998
	PCL-DC	0.0873	0.0729	0.1195	0.0778	0.4071	0.2246
	SCI	0.1516	0.2153	0.1304	0.1823	0.2138	0.0758
	CDE	**0.3403**	0.3208	**0.4079**	0.4284	0.5037	0.2985
	GCN	0.1476	0.2099	0.1708	0.1867	0.5625	0.3543
	SEGCN	0.2258	**0.7783**	0.2768	**0.6439**	**0.5913**	**0.3746**

Table 3. Quality evaluation (in terms of NMI) on networks with ground-truth classes

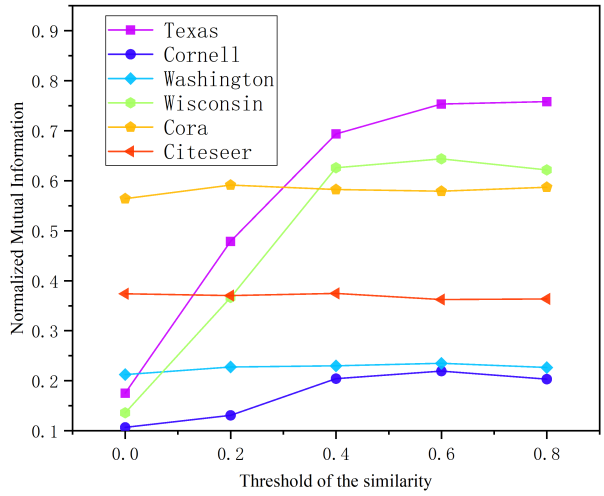
We compare the proposed SEGCN with baseline methods proposed above using six network datasets: Cornell, Texas, Washington, Wisconsin, Cora and Citeseer, with ground-truth classes on the Accuracy (AC) and Normalized Mutual Information (NMI) indicators.

The results reported in Table 2 and Table 3 show that SEGCN clearly outperforms other six baseline methods on most datasets with significant improvements. In terms of AC indicator, we can see that in addition to the Cornell dataset, our model is better than other methods in the node classification accuracy. In some cases, the effect of our model can be improved by 10–25 % for Texas and Wisconsin. Under the NMI indicator, it is obvious that in addition to the Cornell and Washington, the effect of our SEGCN model on several other datasets has been greatly improved. As a result of the fact that the initial state of the dataset is different, the effect of our model on different datasets is more different.

In our experiment, in order to evaluate the impact of the similarity threshold between nodes on the SEGCN model, we changed the similarity threshold from 0 to 0.8 to evaluate the impact on node classification and node clustering. It can be seen from Figure 7 that when the dataset is small (Cornell, Texas, Washington, Wisconsin), the change of the similarity threshold has a greater impact on AC and NMI. This may be that when the dataset is small, the number of edges in the structure-enhanced network increases significantly, and most edges with smaller weights cause the phenomenon of excessive smoothness when the node features are fused. When the similarity threshold becomes larger, this part of the edge with smaller weight is eliminated, and the edge with larger weight is left. The nodes

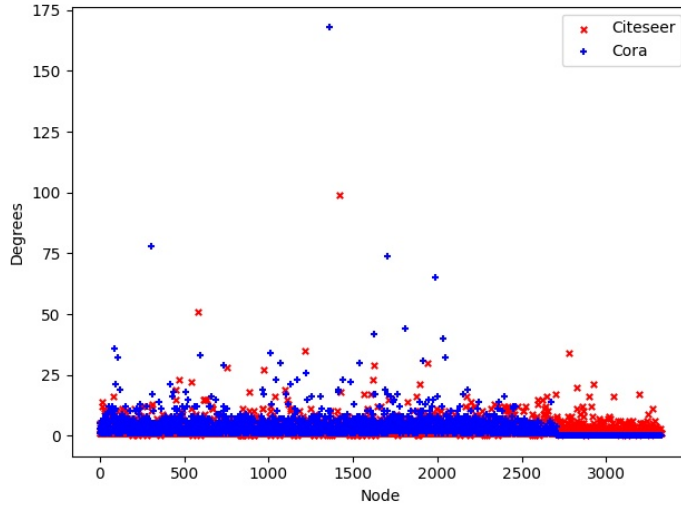


a) Node classification

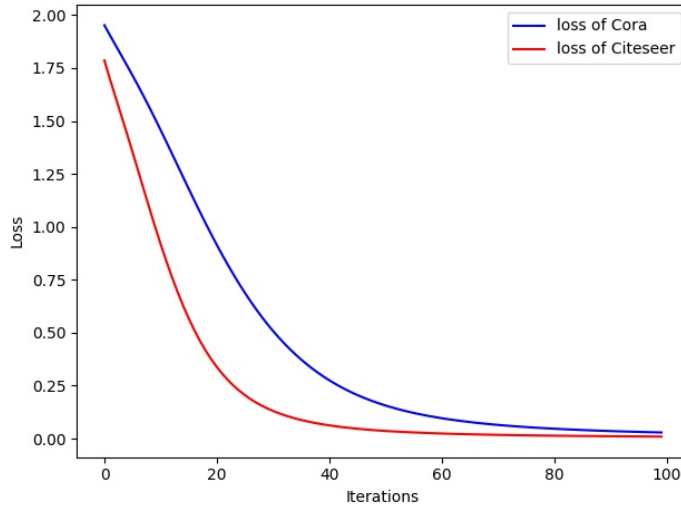


b) Node clustering

Figure 7. Parameter sensitivity analysis of similarity threshold



a) Degree distribution



b) Training loss

Figure 8. Data distribution and visualization of training process

connected by this part of the edge have higher similarity, which is more conducive to the feature fusion of the nodes, so when the similarity threshold becomes larger, the effect is significantly improved. On the Cora and Citeseer datasets, when the similarity threshold changes to 0.4, the effect of the model tends to stabilize. This may be that most of the newly added edges have weights below 0.6. When the similarity threshold is greater than 0.6, the network structure is not much different from the previous one, so the effect of SEGCN under the AC and NMI indicators tends to be stable. In addition, from Figure 8 a), we can see that the degree of nodes in the network is mostly below 25, that is, the connections of nodes are relatively sparse. After we use the structure enhancement algorithm, the edges between nodes become dense. Combined with the adjustment of the similarity threshold, only the edges with larger weights are retained and the edges with smaller weights are deleted, which plays a positive role in obtaining high-quality node representation to a certain extent.

At the same time, we also explored the influence of the embedding dimension of nodes in network representation learning on the model. We control the embedding dimension of the node between 64 and 512, respectively and verify it on node classification and node clustering tasks. It can be seen from Figure 9 that when the embedding dimension of the node is 64 or 128, the model has the best effect. And the effect of the model is not linearly related to the embedding dimension of the node. Considering that the dimensions of the initial features of Cornell and other datasets are around 1700 dimensions, when the node is embedded in 64 or 128 dimensions, in this vector space, the representation vector of the node will be relatively low-dimensional and dense, compared to high-dimensional and sparse representation of the node, which can better characterize a node, so it has the best effect on node classification and clustering tasks.

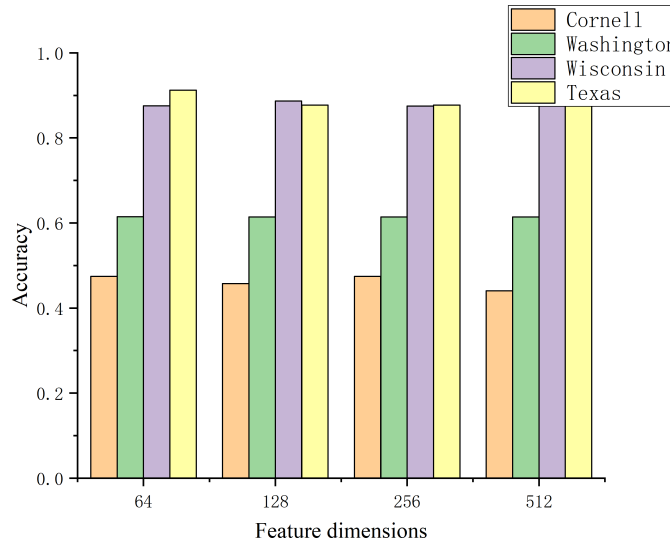
	Training Ratio	20 %	40 %	60 %	80 %
AC	Cornell	0.3633	0.4419	0.4989	0.4929
	Texas	0.5321	0.7431	0.8743	0.8932
	Wisconsin	0.6214	0.7853	0.8232	0.8667
	Washington	0.3245	0.7432	0.6132	0.6547

Table 4. Quality evaluation (in terms of AC) under different training ratios

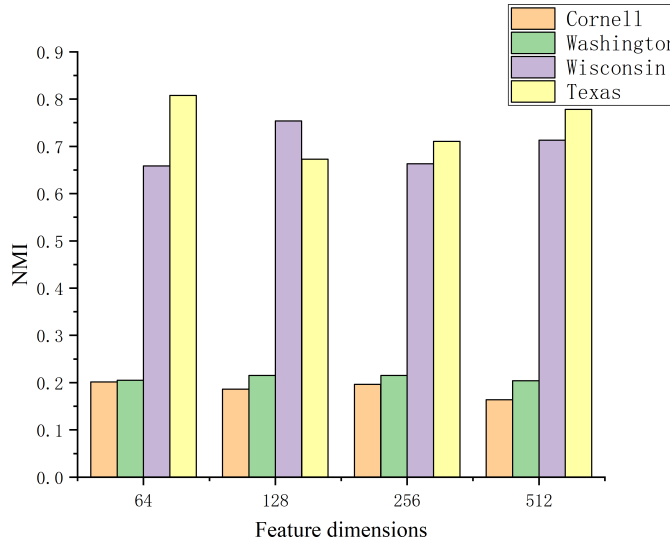
	Training Ratio	20 %	40 %	60 %	80 %
NMI	Cornell	0.1454	0.1832	0.2258	0.2154
	Texas	0.3245	0.4214	0.5673	0.7783
	Wisconsin	0.3512	0.3850	0.4608	0.6439
	Washington	0.0989	0.1665	0.2234	0.2768

Table 5. Quality evaluation (in terms of NMI) under different training ratios

In addition, we also compare the effects of different training ratios and different training iterations on the experimental results.



a) Node classification



b) Node clustering

Figure 9. Parameter sensitivity analysis of feature dimensions

Epoch	Cora		Citeseer	
	AC	*NMI*	*AC*	*NMI*
50	0.5876	0.3123	0.5783	0.3214
100	0.6421	0.5123	0.6234	0.3345
200	0.7961	0.5913	0.6563	0.3746
300	0.7945	0.59	0.6542	0.3702

Table 6. Quality evaluation (in terms of AC and NMI) under different training epochs

It can be seen from Table 4 and Table 5 that with the increase of the training ratio, the AC index and the NMI index also have a relatively obvious improvement. When the training ratio reaches more than 60 %, the improvement of the AC index starts to decrease slowly, while the improvement of the NMI index is still relatively obvious. This may be because NMI is more dependent on the local topology of the network. When the training ratio increases, the more nodes participating in the training, the tighter the local network, and the more obvious the improvement of the NMI index. As shown in Table 6, under the two datasets of Cora and Citeseer, the AC and NMI indicators increase significantly with the increase of the number of training iterations, and the optimal results are achieved when the number of iterations reaches 200. It shows that the SEGCN model proposed in this paper can quickly converge and reach a stable state.

5 CONCLUSION AND FUTURE WORKS

In this paper, we proposed a novel Structure Enhanced Graph Convolutional Network (SEGCN) to address some limitations of existing methods of network representation learning, such as ignoring the influence of higher-order neighbors, or discarding the node characteristics. To be specific, SEGCN contains the following parts:

1. Structure enhancement to capture high-order neighbor relationships,
2. Combine graph convolutional neural network for node feature aggregation to complete node representation learning,
3. Verify effectiveness on node classification and node clustering tasks.

Furthermore, we have thoroughly validated the performance of SEGCN on six real-world datasets, from the experimental results, we can see that due to the ability to capture the global structure and integrate high-order neighbor features, our proposed model SEGCN can improve the effectiveness of node classification and node clustering. At the same time, our model has a fast convergence rate. As can be seen from Figure 8 b), the SEGCN model can reach the convergence state after only one hundred iterations of training. However, our model also has some shortcomings, such as the instability of our model on large and small datasets, which is also the problem we need to solve in the next step.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61702333, in part by the Opening Topic of the Key Laboratory of Embedded Systems and Service Computing of Ministry of Education under Grant ESSCKF 2019-03, in part by the Natural Science Foundation of Shanghai under Grant No. 20ZR1455600, and in part by the Key Innovation Group of Digital Humanities Resource and Research of Shanghai Normal University.

REFERENCES

[1] Junghanns, M.—Petermann, A.—Neumann, M.—Rahm, E.: Management and Analysis of Big Graph Data: Current Systems and Open Challenges. In: Zomaya, A. Y., Sakr, S. (Eds.): Handbook of Big Data Technologies. Springer, Cham, 2017, pp. 457–505, doi: 10.1007/978-3-319-49340-4_14.

[2] Eto, M.: Extended Co-Citation Search: Graph-Based Document Retrieval on a Co-Citation Network Containing Citation Context Information. Information Processing and Management, Vol. 56, 2019, No. 6, Art. No. 102046, doi: 10.1016/j.ipm.2019.05.007.

[3] Chu, C.—Li, Z.—Xin, B.—Peng, F.—Liu, C.—Rohs, R.—Luo, Q.—Zhou, J.: Deep Graph Embedding for Ranking Optimization in E-Commerce. Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18), 2018, pp. 2007–2015, doi: 10.1145/3269206.3272028.

[4] Tang, D.—Qin, B.—Liu, T.: Learning Semantic Representations of Users and Products for Document Level Sentiment Classification. In: Zong, C., Strube, M. (Eds.): Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). ACL, 2015, pp. 1014–1023, doi: 10.3115/v1/P15-1098.

[5] Zhao, Q.—Wang, C.—Wang, P.—Zhou, M.—Jiang, C.: A Novel Method on Information Recommendation via Hybrid Similarity. IEEE Transactions on Systems, Man, and Cybernetics: Systems, Vol. 48, 2018, No. 3, pp. 448–459, doi: 10.1109/TSMC.2016.2633573.

[6] Zhao, Q.—He, Y.—Jiang, C.—Wang, P.—Qi, M.—Li, M.: Integration of Link and Semantic Relations for Information Recommendation. Computing and Informatics, Vol. 35, 2016, No. 1, pp. 30–54, https://www.cai.sk/ojs/index.php/cai/article/view/2600.

[7] Zhang, M.—Chen, Y.: Link Prediction Based on Graph Neural Networks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 31 (NIPS 2018). Curran Associates Inc., 2018, pp. 5165–5175, doi: 10.48550/arXiv.1802.09691.

[8] Zhang, Y.—Dong, X.—Shang, L.—Zhang, D.—Wang, D.: A Multi-Modal Graph Neural Network Approach to Traffic Risk Forecasting in Smart Urban Sensing. 2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), 2020, pp. 1–9, doi: 10.1109/SECON48991.2020.9158447.

[9] Zhang, D.—Yin, J.—Zhu, X.—Zhang, C.: Network Representation Learning: A Survey. IEEE Transactions on Big Data, Vol. 6, 2020, No. 1, pp. 3–28, doi: 10.1109/TBDATA.2018.2850013.

[10] Ahmed, A.—Shervashidze, N.—Narayanamurthy, S.—Josifovski, V.—Smola, A. J.: Distributed Large-Scale Natural Graph Factorization. Proceedings of the 22nd International Conference on World Wide Web (WWW '13), ACM, 2013, pp. 37–48, doi: 10.1145/2488388.2488393.

[11] Xiao, Y.—Zhu, Z.—Zhao, Y.—Wei, Y.—Wei, S.—Li, X.: Topographic NMF for Data Representation. IEEE Transactions on Cybernetics, Vol. 44, 2013, No. 10, pp. 1762–1771, doi: 10.1109/TCYB.2013.2294215.

[12] Roweis, S. T.—Saul, L. K.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science, Vol. 290, 2000, No. 5500, pp. 2323–2326, doi: 10.1126/science.290.5500.2323.

[13] Belkin, M.—Niyogi, P.: Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. In: Dietterich, T., Becker, S., Ghahramani, Z. (Eds.): Advances in Neural Information Processing Systems 14 (NIPS 2001). MIT Press, 2001, pp. 585–591.

[14] Perozzi, B.—Al-Rfou, R.—Skiena, S.: DeepWalk: Online Learning of Social Representations. Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14), 2014, pp. 701–710, doi: 10.1145/2623330.2623732.

[15] Tang, J.—Qu, M.—Wang, M.—Zhang, M.—Yan, J.—Mei, Q.: LINE: Large-Scale Information Network Embedding. Proceedings of the 24th International Conference on World Wide Web (WWW '15), ACM, 2015, pp. 1067–1077, doi: 10.1145/2736277.2741093.

[16] Grover, A.—Leskovec, J.: Node2vec: Scalable Feature Learning for Networks. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16), 2016, pp. 855–864, doi: 10.1145/2939672.2939754.

[17] Wang, D.—Cui, P.—Zhu, W.: Structural Deep Network Embedding. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16), 2016, pp. 1225–1234, doi: 10.1145/2939672.2939753.

[18] Bruna, J.—Zaremba, W.—Szlam, A.—LeCun, Y.: Spectral Networks and Locally Connected Networks on Graphs. 2013, doi: 10.48550/arXiv.1312.6203.

[19] Defferrard, M.—Bresson, X.—Vandergheynst, P.: Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 29 (NIPS 2016). Curran Associates, Inc., 2016, pp. 3844–3852, doi: 10.48550/arXiv.1606.09375.

[20] Abu-El-Haija, S.—Kapoor, A.—Perozzi, B.—Lee, J.: N-GCN: Multi-Scale Graph Convolution for Semi-Supervised Node Classification. In: Adams, R. P., Gogate, V. (Eds.): Proceedings of the 35th Uncertainty in Artificial Intelligence Conference. Proceedings of Machine Learning Research (PMLR), Vol. 115, 2020, pp. 841–851, doi: 10.48550/arXiv.1802.08888.

[21] Wang, X.—Ji, H.—Shi, C.—Wang, B.—Ye, Y.—Cui, P.—Yu, P. S.: Heterogeneous Graph Attention Network. The World Wide Web Conference (WWW '19), ACM, 2019, pp. 2022–2032, doi: 10.1145/3308558.3313562.

[22] Weng, L.—Karsai, M.—Perra, N.—Menczer, F.—Flammini, A.: Attention on Weak Ties in Social and Communication Networks. In: Lehmann, S., Ahn, Y. Y. (Eds.): Complex Spreading Phenomena in Social Systems: Influence and Contagion in Real-World Social Networks. Springer, Cham, Computational Social Sciences, 2018, pp. 213–228, doi: 10.1007/978-3-319-77332-2_12.

[23] Jeh, G.—Widom, J.: SimRank: A Measure of Structural-Context Similarity. Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '02), 2002, pp. 538–543, doi: 10.1145/775047.775126.

[24] Mao, X.—Shen, C.—Yang, Y. B.: Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 29 (NIPS 2016). Curran Associates, Inc., 2016, pp. 2802–2810, doi: 10.48550/arXiv.1603.09056.

[25] Kipf, T. N.—Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks. 2016, doi: 10.48550/arXiv.1609.02907.

[26] Zhang, Y. D.—Pan, C.—Sun, J.—Tang, C.: Multiple Sclerosis Identification by Convolutional Neural Network with Dropout and Parametric ReLU. Journal of Computational Science, Vol. 28, 2018, pp. 1–10, doi: 10.1016/j.jocs.2018.07.003.

[27] Zhang, Z.—Sabuncu, M.: Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 31 (NIPS 2018). Curran Associates Inc., 2018, pp. 8778–8788, doi: 10.48550/arXiv.1805.07836.

[28] van der Maaten, L.—Hinton, G.: Visualizing Data Using t-SNE. Journal of Machine Learning Research, Vol. 9, 2008, No. 86, pp. 2579–2605, https://jmlr.org/papers/v9/vandermaaten08a.html.

[29] Liu, H.—Wu, Z.: Non-Negative Matrix Factorization with Constraints. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 24, 2010, No. 1, pp. 506–511, doi: 10.1609/aaai.v24i1.7681.

[30] Wang, F.—Li, T.—Wang, X.—Zhu, S.—Ding, C.: Community Discovery Using Nonnegative Matrix Factorization. Data Mining and Knowledge Discovery, Vol. 22, 2011, No. 3, pp. 493–521, doi: 10.1007/s10618-010-0181-y.

[31] Shi, J.—Malik, J.: Normalized Cuts and Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, 2000, No. 8, pp. 888–905, doi: 10.1109/34.868688.

[32] Yang, T.—Jin, R.—Chi, Y.—Zhu, S.: Combining Link and Content for Community Detection: A Discriminative Approach. Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09), 2009, pp. 927–936, doi: 10.1145/1557019.1557120.

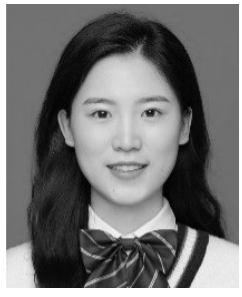
[33] Wang, X.—Jin, D.—Cao, X.—Yang, L.—Zhang, W.: Semantic Community Identification in Large Attribute Networks. Proceedings of the AAAI Conference on

Artificial Intelligence, Vol. 30, 2016, No. 1, pp. 265–271, doi: 10.1609/aaai.v30i1.9977.

[34] Li, Y.—Sha, C.—Huang, X.—Zhang, Y.: Community Detection in Attributed Graphs: An Embedding Approach. Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, 2018, No. 1, pp. 338–345, doi: 10.1609/aaai.v32i1.11274.

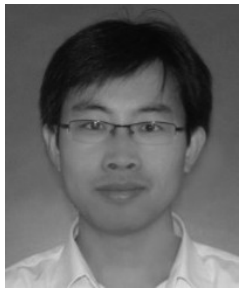
Ningchen Fu received his B.Eng. degree from the Department of Software Engineering, Shandong University of Technology, in 2019. He is currently a master student in Department of Computer at Shanghai Normal University, China. His research interests include social networks, data mining, and machine learning.

Qin Zhao received his Ph.D. degree in computer science from the Department of Computer Science and Technology, Tongji University, Shanghai, China, in 2016. Currently, he is Associate Professor with the Department of Computer, Shanghai Normal University, Shanghai, China. His research interests include data mining, social network analysis, and artificial intelligence.



Yaru Miao received her B.Eng. degree from the Department of Computer Science and Technology, Shanghai Normal University. She is currently pursuing the M.Sc. degree in the Department of Computer at Shanghai Normal University, China. Her research interests include community detection, natural language processing and sentiment analysis.

Bo Zhang received his Ph.D. degree in computer science from the College of Electronics and Information Engineering, Tongji University, in 2009, and he finished his PostDoc research work in the Tongji University in 2012. He is now Professor in the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai, China. His current research interests include trust computation and social network analysis.



Dong Wang received the Ph.D. degree in computer science from the Tongji University in 2013. He works at the Shanghai Institute of Technology, Shanghai, China, where he works on intelligent decision support systems, information services, trusted computing, and social networks. His research centers around machine learning from relational knowledge representations and graph-structured data as well as its applications in artificial intelligence and cognitive science.