Network representation learning method embedding linear and nonlinear network structures

Hu Zhang[1*], Jingjing Zhou[1], and Ru Li[1,2]

[1] *School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China.*

[2] *Key Laboratory of Computation Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan 030006, China.*

Abstract. With the rapid development of neural networks, more attention has focused on network embedding for complex network data, which aims to learn low-dimensional embedding of nodes in the network and how to effectively apply learned network representations to various graph-based analytical tasks. Two typical models are the shallow random walk network representation method and deep learning models like graph convolution networks (GCNs). The former can be used to capture the linear structure of the network using depth-first search (DFS) and width-first search (BFS). Hierarchical GCN (HGCN) is an unsupervised graph embedding that can be used to describe the global nonlinear structure of the network by aggregating node information. However, the two existing kinds of models cannot capture the nonlinear and linear structure information of nodes simultaneously. Therefore, the nodal characteristics of nonlinear and linear structures were examined in this study, and an unsupervised representation method based on HGCN that joins learning of shallow and deep models is proposed. Experiments on node classification and dimension reduction visualization were carried out on citation, language, and traffic networks. The results show that, compared with the existing shallow network representation model and deep network model, the proposed model achieves better performance in terms of micro-F1, macro-F1, and accuracy scores.

Keywords: Shallow random walk, Deep HGCN, Nonlinear structure, Linear structure, Network embedding

1. Introduction

The purpose of network embedding is to map high-dimensional sparse network data into low-dimensional, dense, real-value vector space, which can be used to adaptively extract features and facilitate analysis of downstream network tasks. These tasks can be performed using machine learning and other methods, such as node classification, link prediction, and dimension reduction visualization[1][2]. At present, the mainstream network embedding methods include a shallow random walk model and a semi-supervised deep learning model based on graph convolution networks (GCNs). In the shallow random walk method, the depth-first search (DFS) method captures the global network structure, and the width-first search (BFS) method captures the local network structure.

For example, the DeepWalk method uses a DFS-based random walk and uses second-order similarity to capture local community information, which can be used to observe the entire network in the absence of structural information. LINE considers both the first-order

* Corresponding author E-mail: zhanghu@sxu.edu.cn.

similarity and second-order similarity of nodes and stitches their trained similarity vectors to represent nodes. Meanwhile, node2vec and struc2vec also flexibly describe the structure of nodes in the network according to their different roles in the network. A random walk is used in the aforementioned shallow models because it can capture the linear structure of the network.

With the rapid development of graph neural networks, more semi-supervised methods based on GCNs have been used for graph embedding. In the real world, due to difficulties of privacy protection and labeling, many real networks do not contain node characteristics and labels, which results in the poor applicability of the original GCN semi-supervised method to a network with only structural information. Hierarchical GCN (HGCN) is proposed as it can be used to learn the structure of the network. In other words, a HGCN is used to construct an initial node embedding and pseudo label, and then the labels are updated and embedded by using a double GCN, which can describe the nonlinear structure of the network without attributes. However, the two existing types of models cannot capture the nonlinear and linear structure information of nodes simultaneously.

To overcome these limitations, we examine the nodal characteristics of network structures, improve the HGCN model, and a representation method based on unsupervised joint learning with shallow and deep learning models is proposed. Our model uses the shallow learning model to extract local and global linear structural features of nodes and obtains global nonlinear structural features of nodes by aggregating information from neighboring nodes using HGCN. More specifically, it fuses the two kinds of extracted features and applies them to downstream tasks in the network. We provide in depth experimental analyses in each stage of the proposed approach and compare our work with state-of-the-art approaches. The experimental results show that the proposed approach produces higher accuracy, micro-F1, and macro-F1 scores. A detailed flowchart of the proposed method is shown in Figure 1. Network embedding of the input graph is obtained by extracting nonlinear and linear structural features, which are then applied to downstream tasks such as node classification and dimension reduction visualization after fine-tuning.

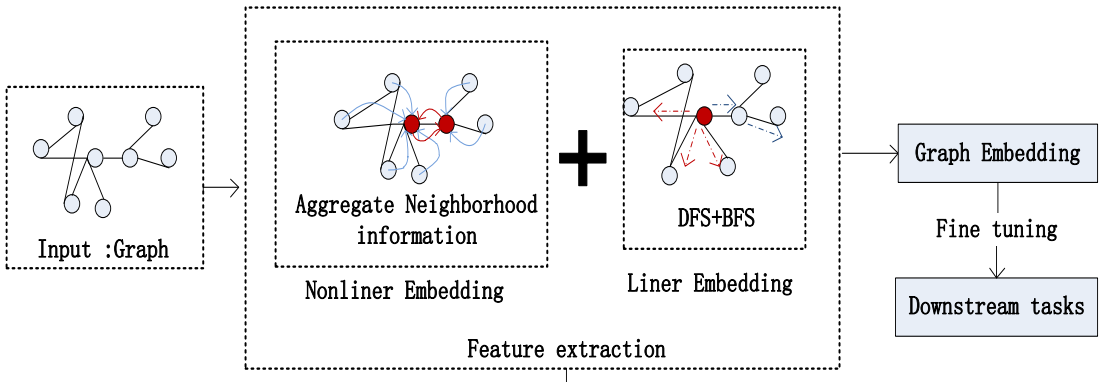


Fig.1. Model workflow.

The remainder of the paper is organized as follows. Related work on network embedding is introduced in Section 2. The proposed approach, including shallow linear structure representation, proposed HGCN, and improved HGCN, is presented in Section 3. Experimental results are presented in Section 4, and the paper is concluded in Section 5.

2. Related work

Network embedding methods have different classification systems. In this section, we briefly summarize existing work on network embedding from three perspectives. First, we introduce matrix methods, and then we review different network embedding methods in the field of machine learning. In the end, we classify network embedding models based on the structure of the embedding methods, and the disadvantages of existing network embedding methods are analyzed.

2.1 Matrix

As mentioned above, traditional network embedding methods are described in the context of dimensionality reduction. From the perspective of linear algebra, unsupervised feature learning methods are usually represented by various graph matrices, especially the Laplacian operator and adjacency matrix[3]. The classical dimensionality reduction techniques include principal component analysis (PCA)[4] and multidimensional scaling (MDS)[5]. In both methods, a graph is represented as an $n \times m$ matrix and transformed into an $n \times k$ matrix, where $k \ll M$. Both methods can capture the linear structural information, but they cannot be used to learn the nonlinear structure of the input data.

Several linear (e.g., PCA) and nonlinear (e.g., Isomap) dimensionality reduction techniques[6] have computational and statistical performance defects. In terms of computational efficiency, feature decomposition of the input data matrix requires high computational power.

2.2 Machine learning

In representation learning for machine learning, network embedding is considered a subset of the set of representation learning technologies. To that extent, supervised machine learning methods need a set of node-distinguishing features. The typical methods primarily focus on specific professional fields, and manual feature extraction can be used to learn node representation, which leads to inaccurate feature extraction and poor robustness[7]. The other method primarily examines node representation by solving an optimization problem, which enables feature extraction with higher accuracy; however, the number of estimated parameters is large, and the process requires significant computation time[8].

In unsupervised learning (e.g., such as DeepWalk, node2vec, and struc2vec)[9], feature representation requires defining an objective function that is independent of downstream tasks to balance accuracy and computational efficiency. Compared with manual feature extraction, using an objective function for determining network representation can ensure more comprehensive features are extracted, which are closely related to the prediction network accuracy[10]. Therefore, unsupervised graph embedding improves the shortcomings of semi-supervised machine learning. Nevertheless, unsupervised graph embedding is difficult to expand and has high training complexity.

2.3 Structure

The structure of a neural network is related to the number of hidden layers. According to the number of hidden layers[11], network embedding methods are divided into shallow and deep learning methods. First, in the shallow learning model, inspired by successful application of skip-grams in natural language processing[12][13], a series of network structure coding models based on skip-grams have been proposed, such as DeepWalk, LINE, and node2vec. To be specific, DeepWalk[14] simulates word sequences, generates node sequences from each node randomly, and forms a "corpus" based on these node sequences. In addition, DeepWalk sets the size of the background window and then imports the "corpus" into the skip-gram model to obtain

the node embedding. LINE[15] optimizes the first-order and second-order similarity of direct connections, and the two similarities are combined as the node embedding. Node2vec[16] uses two additional parameters to control the direction of the random walk in the "corpus" generation step in DeepWalk. Struc2vec[17] defines nodes that are not structurally adjacent but have the same structural role. In summary, unsupervised machine learning is normally used in the shallow embedded model because it can better capture the linear structure of the network; however, it has the disadvantage of ignoring nodal features.

In the deep learning model, a mainstream method like GNN uses a deep learning model to capture the nonlinear relationship between nodes[18]. Typical deep learning models include SDNE, GCN, and a series of graph neural network models based on GCN. SDNE[19] uses an unsupervised learning method to automatically capture the local relationship between nodes, and the second-order neighborhood of nodes is used as the low-dimensional representation of input learning nodes. GCN[20] is a deep, semi-supervised graph embedding model, which integrates additional features and labeled data into the graph embedding learning process.

2.4 Graph neural networks

With the development of graph neural networks (GNNs), a series of GNN methods based on GCN have been applied to various types of networks to improve the applicability of GCN in the inductive task. Typical examples are graph attention network (GAT), N-GCN, and Fast-GCN. Specifically, GAT[21] uses an attention mechanism to address the shortcomings of prior methods based on graph convolutions or their approximations. N-GCN[22] improves the scalability of GCN on an entire graph by setting the size of the convolution kernel. Fast-GCN[23] is a batch training algorithm combining importance sampling; it can make GCN training more efficient provide generalized inference. Inspired by unsupervised methods and the idea of pre-training, a series of GCN-based training methods have emerged. Deep Graph Info Max [24] relies on maximizing mutual information between patch representations and corresponding high-level summaries of graphs, [while both are derived using established GCN architectures. The pre-trained GCN model proposed by Hu et al. [25] can capture generic graph structural information that is transferable across tasks. M3S [26] uses the correct extra feature and increases the number of labeled data via self-training to learn the undirected graph embedding. GMNN [27] also applies the correct extra feature and neighbor nodes to generate pseudo-labels for unsupervised learning. Consequently, one main challenge encountered when using the above GNN method is that it still cannot consider the feature and the label of the node at the same time.

3. Model

Suppose $G = (\mathrm{V}, E)$, where V is a node in the network, E is an edge in the network, and $\mathrm{n} = |\mathrm{V}|$ is the number of nodes. The shallow linear structure embedding of the network is E_s, and the deep nonlinear structure is E_D. Finally, the model embedding is E_{All}.

3.1 Shallow linear structure representation

Qi [31] showed that DeepWalk, LINE, PTE, and Node2vec are implicit matrix factorizations in theory. DeepWalk first generates a low rank transformation of the network's normalized Laplacian matrix, and then the eigenvector of the decomposed matrix is taken as the representation vector. Because the eigenvector of each dimension represents a linear subspace and observes characteristics of the entire network in the absence of structural information, DeepWalk captures a global linear representation. LINE is a

special case where the size of DeepWalk's window is "1". The first-order proximity similarity of LINE captures a representation of the local linear structure. The following definitions are given in this paper:

(1)Local structural information: edge information in the network is an observed first-order similarity. In general, the first-order similarity considers that the greater the edge weight between two vertices, the more similar the two vertices are. In this paper, the first-order similarity in the LINE method is used to describe the shallow linear structure. For each pair of undirected $(\mathrm{i},\mathrm{j}) \in \mathrm{E}$, the joint probability of (v_i, v_j) is defined as formula (1):

$$E_{S-C}(\mathrm{v_i}, \mathrm{v_j}) = \frac{1}{1+\exp(-\overrightarrow{\mathrm{u_i}} \cdot \overrightarrow{\mathrm{u_j}})} \quad (1)$$

Where $\overrightarrow{\mathrm{u_i}}$ is the vector of $\mathrm{v_i}$; $\mathrm{p}(.,.)$ is an $\mathrm{n} \times \mathrm{n}$ vector; and the objective function for first order similarity minimization of $(\mathrm{v_i},\mathrm{v_j})$ is defined as formula (2):

$$\mathrm{O_L} = -\sum_{(\mathrm{i},\mathrm{j})\in \mathrm{E}} \mathrm{w_{ij}} \log E_{S-C}(\mathrm{v_i}, \mathrm{v_j}) \quad (2)$$

Where $\mathrm{w_{ij}}$ is the weight of vertex $(\mathrm{v_i},\mathrm{v_j})$.

In the undirected graph, $\mathrm{w_{ij}} \equiv \mathrm{w_{ji}}$. If there are no edges directly connecting two vertices, then $\mathrm{w_{ij}} = 0$.

(2)Global structural information: Global network embedding can obtain more abundant node characteristics [32]. In a real network, many legitimate edges are not observed. While DeepWalk can be used to predict and learn the global structure of the network with higher accuracy, it is used to describe the global information of the network in the shallow linear structure. For an undirected graph, a random walk is used to obtain a random walk sequence with a specific length $\mathrm{w}(\mathrm{v_i}) = (\mathrm{w}(\mathrm{v_i})^1, \mathrm{w}(\mathrm{v_i})^2 \ldots \mathrm{w}(\mathrm{v_i})^{\mathrm{k}})$, where $\mathrm{w}(\mathrm{v_i})^{\mathrm{n}}$ is n_{th} node. Then the model uses the language model skip-gram to learn these sequences and determine the global structure embedding $\mathrm{E_{S-G}}$, which is as formula (3):

$$\mathrm{E_{S-G}}(\mathrm{w}(\mathrm{v_i})) = \mathrm{skip-gram}(\mathrm{w_i^u}) \quad (3)$$

Where, $\mathrm{w_i^u} = (\mathrm{w_{1,}w_{2,}} \ldots \mathrm{w_n})$ is the word order and $\mathrm{w_i}$ is a specific word in the word order.

3.2 Model on embedding linear and nonlinear network structure

In this section, we focus on the improved hierarchical GCN model. We first introduce GCN, then we introduce the HGCN nonlinear structure representation model to further justify our model. In addition, we further explain the reason for the improved model.

3.2.1 GCN model

GCN is a semi-supervised graph convolution model that integrates additional node attributes and some labeled nodes into the process of learning the node representation. Mathematically, graph convolution is defined as formula (4):

$$H^{(l+1)} = \sigma(PH^{(l)}W^{(l)}) \quad (4)$$

Where $P = D^{-\frac{1}{2}}(A+I)D^{-\frac{1}{2}}$ is the normalized Laplacian matrix of the graph G. $\mathrm{H}^{(l)}$is the input from the l_{th} hidden layer in the GCN, i.e., it is the output from the $(l-1)_{\mathrm{th}}$ hidden layer. $\mathrm{W}^{(l)}$ is the weight matrix in the l_{th} hidden layer that would be trained, A is the adjacency matrix of the undirected graph, and $\sigma(.)$ is the activation function.

For any given node, a GCN layer aggregates the previous layer's embedding of its neighbor with A, followed by a linear transformation $W^{(l)}$ and nonlinear activation $\sigma(.)$, so as to obtain a contextualized node representation. We denote $F_w(\cdot)$ as L-layer GCNs, which are parametrized by $\{w_i\}_{\mathrm{i=1}}^{L}$. For each given graph G = (V, A) with input features $H^{(0)}$, the node representations $F(\cdot)$ are defined as formula (5):

$$F_w(G) = \sigma(\mathrm{P}(\sigma(PH^{(1)}W^{(1)})..)W^{(L)}) \quad (5)$$

GCN is a deep neural network model. In each hidden layer in the GCN, a neighboring node representation for the aggregation center node is used as the input representation in the next layer, and the nonlinear activation function is

connected. From the overall structure of the GCN stacking nonlinear activation function, a GCN model captures the deep nonlinear structural information of the network.

3.2.2HGCN nonlinear structure representation model

In the real world, due to the lack of privacy and tagged data, GCN is not suitable for graphs with only structural information. HGCN improves the applicability of the GCN model in graphs with only structural information by constructing an initial node embedding vector and pseudo label. The specific model includes an input layer, an updated layer, and an output layer.

Detailed definitions are as follows:

(1)Input layer: first, HGCN will capture the global information of $E_{S-G}(\mathrm{w_{v_i}})$ as an additional feature of the node. Because the number of label categories is unknown, the model needs to choose an unsupervised clustering method that does not need to specify the number of clusters [33]. Then AP clustering is used to cluster and label the feature, and the pseudo label Y is given. $y_i \in Y$ is a pseudo label of vertex v_i. The undirected graph G in the input layer is defined as formula (6):

$$\mathrm{G} = (\mathrm{V}, \mathrm{E}, E_{S-G}(\mathrm{w_{v_i}}), \mathrm{Y}) \quad (6).$$

(2)Update layer: the update layer in the HGCN is composed of two GCN models, and each GCN model includes three hidden layers. When information from the input undirected graph is known, the first GCN model will be learned and embedded into E_{D1}. Let $H^{(0)} = E_{S-G}$, and according to Eq. (5), E_{D1} is defined as formula (7):

$$E_{D1}(\mathrm{w_{v_i}}) = \sigma(\mathrm{P}(..\sigma(PE_{S-G}(w(v_i))W^{(0)})..)W^{(2)}) \quad (7)$$

Meanwhile, node labels Y' are predicted, and the second GCN is used to learn the updated features E_{D2} and labels. E_{D2} is defined as formula (8):

$$E_{D2}(\mathrm{w_{v_i}}) = \sigma(\mathrm{P}(..\sigma(PE_{D1}(w(v_i))W^{(0)})..)W^{(2)}) \quad (8).$$

(3)Output layer: after the second training iteration, the trained GCN model will have an embedding that is too smooth, which will affect the downstream node classification results. Therefore, to prevent smoothing, we splice E_{D2} and E_{S-G}. Finally, E_D is obtained, which is the nonlinear structure representation. The detailed formula is shown in Eq. (9), where $\oplus$ is a vector splicing operation:

$$E_D = E_{D2}(\mathrm{w_{v_i}}) \oplus E_{S-G}(\mathrm{w_{v_i}}) \quad (9).$$

3.2.3 Fusion embedding model based on HGCN

Splicing E_{D2} and E_{S-G} in the output layer of the HGCN prevents smoothing of the embedding caused by iterative training. Meanwhile, global information of a shallow linear structure is fused into the learned embedding. The experimental results of HGCN show that the linear structure of the embedding in the shallow layer can achieve the above purpose.

The linear structure embedding in the shallow layer can be divided into a global embedding vector and local embedding vector. HGCN integrates the global embedding vector of the network by splicing. Inspired by the different types embedding, the splicing operation in the output layer of the HGCN is improved. In general, there are two key points involved in the modified splicing operation:

(1)Fusion of local linear structure embedding (HGCN-L):

$\mathrm{E_{S-C}}$ is spliced with E_{D2} to prevent smoothing and fuse local information in the shallow linear structure. According to HGCN, global embedding is used to alleviate over smoothing. In our model, $E_{S-G}(\mathrm{w_{v_i}})$ is replaced by local embedding $E_{S-C}(\mathrm{w_{v_i}})$. Eq. (9) is modified to give Eq. (10) as follows:

$$E_{All} = E_{D2}(\mathrm{w_{v_i}}) \oplus E_{S-C}(\mathrm{w_{v_i}}) \quad (10).$$

(2)Fusion of local and global linear structure embedding (HGCN-DL):

First, the linear structure embedding E_S is

obtained by splicing E_{S-C} and E_{S-G}. Then the linear embedding and deep nonlinear structure embedding are spliced to prevent smoothing and integrate local and global structure embedding. In our model, $E_{S-G}(\mathrm{w_{v_i}})$ is replaced by linear structure embedding E_S. Finally, E_{All} is defined as the final embedding of a node. According to Eq. (9), it is improved to Eq. (11):

$$E_{All} = E_{D2}(\mathrm{w_{v_i}}) \oplus \mathrm{E_S}(\mathrm{w_{v_i}}) \quad (11).$$

A detailed model is shown in Figure 2. The graph embedding is composed of nonlinear and linear structure embedding. Among them, the nonlinear structure embedding is generated by the update layer in the HGCN, namely the double GCN. The linear structural embedding uses a random walk model and first-order proximity similarity to describe the global and local vectors, respectively. Finally, the nonlinear and linear structure embedding are spliced to obtain the final network structure embedding.

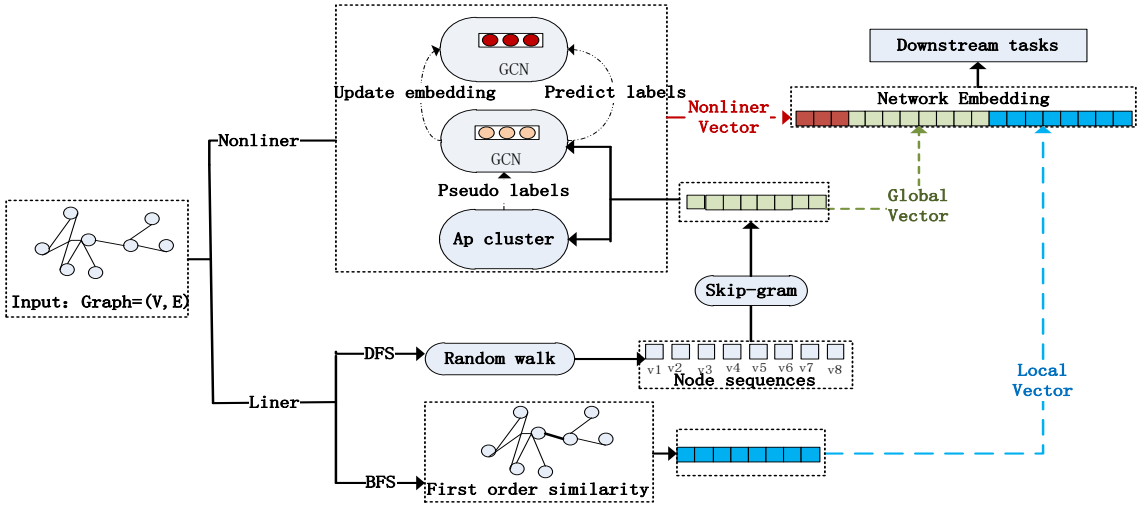


Fig. 2. Global model.

4. Experiments

In this section, the proposed approach is experimentally evaluated using three kinds of datasets. We first present the dataset, the experimental settings, and a comparison with state-of-the-art baseline results. We also evaluate the influence of shallow linear structural information for embedding and downstream analysis tasks.

4.1 Data sets

To verify the validity of the model, three kinds of undirected graph network data sets are selected: citation network [28], air-traffic network [17], and language network [29]. In this paper, the citation network includes the Cora and Citeseer data sets; the air traffic network includes European and American air traffic data sets; and the language network is a Wikipedia data set. Specifically, in this paper, the citation network is used as a network with only structural information, and the correct label is only used to verify downstream tasks. The details of these data sets are shown in Table 1.

Table 1

Experimental data sets

Type	citation network		language network	air traffic network	
Datasets	Cora	Citeseer	Wikipedia	American air traffic	European air traffic
Nodes	2708	3327	2405	1190	399
Edges	5429	4732	17981	13599	5995
Classes	7	6	17	4	4

4.2 Baselines

DeepWalk is a typical unsupervised graph embedding method that adopts the skip-gram language model.

LINE is also a popular unsupervised method that considers first-order and second-order proximity information.

Node2vec learns low-dimensional representations for nodes in a graph by optimizing a neighborhood-preserving objective. This method explores the structure of the network by controlling the p and q parameters, and d dimension feature representations can be learned by simulating biased random walks.

SDNE learns the d dimension node representation by capturing the nonlinear structure of the network using multi-layer nonlinear functions, which is a semi-supervised deep graph embedding model.

SN2vec is a model that integrates node structure and node content similarity. This model is our previous work. It is a shallow random walk model, which primarily captures linear structures. In this paper, the model is compared as a shallow linear baseline model.

4.3 Experimental setup

Due to the different sizes of the data sets and different experimental settings in different models, we introduce the parameters of the step size and step number in DeepWalk, the parameters p and q in Node2vec, and batch size in LINE separately. Parameter values are shown in Table 2.

Table 2

Baselines experimental setup

Model	Traffic datasets		Citation network		Language network	
DeepWalk	Step size	Step number	Step size	Step number	Step size	Step number
	10	80	10	80	10	80
Node2vec	p	q	p	q	p	q
	0.25	0.25	0.25	2	0.25	2
LINE	Training batch size		Training batch size		Training batch size	
	128		1024		1024	

The step number of random walks was set to 80 and the step size was set to 10 in DeepWalk. While for the European traffic network data set, the step size was set to 10, and the number of

random walk steps was set to 15 after comparing experimental results for different parameter values.

Table 2 shows that for the traffic network, the parameters p and q in Node2vec were set to 0.25 because the datasets are equivalent in structure and content. For the citation network, after comparing experimental results for different parameter values, the experimental results are better when p = 0.25 and q = 2. The sizes of the data sets for the language network and citation network are similar, so the same experimental parameters were used.

Regarding the training batch size in LINE, the training batch size for the European traffic data set was set to 128; the training batch size for the other data sets was set to 1024.

4.4 Node classification experiments

Node classification is used to verify the model. The baseline models include the shallow linear structure model and deep nonlinear structure model. The evaluation indices used in this study are micro-F1, macro-F1, and accuracy.

In particular, the SN2vec model was used as a baseline model for exploring the shallow linear structure of undirected graph networks with equivalent structure and content. Among them, the results with * are experimental results obtained from OpenNE[30]. The specific experimental results are shown in Tables 3 and 4.

Table 3

Traffic network node classification results

	Traffic network					
Data sets	American air traffic			Europe		
Evaluating indicator	Micro-F1	Macro-F1	Accuracy	Micro-F1	Macro-F1	Accuracy
DeepWalk	0.523	0.518	0.523	0.395	290	0.395
LINE (2nd)	0.504	0.499	0.504	0.430	0.405	0.430
Node2vec	0.493	0.459	0.493	0.43	0.340	0.430
SN2vec	0.530	0.509	0.530	0.45	0.404	0.45
SDNE	0.580	0.56	0.58	---	---	---
HGCN	0.571	0.568	0.571	0.46	0.417	0.46
HGCN-L	0.584	0.581	0.584	0.500	0.477	0.500
HGCN-DL	**0.601**	**0.598**	**0.601**	**0.540**	**0.512**	**0.540**

Table 3 shows that on the American air traffic network, HGCN-L is better than the original HGCN model in terms of all three evaluation indicators. The same results are obtained for the Europe traffic dataset after adjusting the parameters, suggesting the local topological representation is more identifiable for nodes on datasets with the same structure and content. This also indicates that local structural information is more important for node classification than global structure information for such datasets.

Table 3 also shows that HGCN-DL, which fuses shallow linear local and global structural information, further improves the node classification results on HGCN-L in micro-F1, macro-F1, and accuracy. On the American traffic dataset, HGCN-DL is 1.7% higher than

HGCN-L on micro-F1 and accuracy, and 1.7% higher than HGCN-L on macro-F1. For the European dataset, HGCN-DL is 4% higher than HGCN-L on micro-F1 and accuracy, and 3.5% higher than HGCN-L on macro-F1. These results indicate that HGCN-DL can be used to capture and represent the topological structure of undirected graphs more fully.

Table 4

Classification results for citation and language network nodes

Type	citation network						language network	
Data sets	Cora			Citeseer			Wikipedia	
Evaluating indicator	Micro-F1	Macro-F1	Accuracy	Micro-F1	Macro-F1	Accuracy	Micro-F1	Macro-F1
DeepWalk	----	----	0.672*	----	----	0.432*	0.669*	0.560*
LINE (2nd)	0.704	0.684	0.704	0.444	0.399	0.444	0.576*	0.387*
Node2vec	0.771	0.765	0.771	0.567	0.486	0.567	0.651*	0.541*
SN2vec	0.786	0.773	0.786	---	---	---	0.571	0.394
SDNE	0.660	0.648	0.660	0.400	0.351	0.400	0.643*	0.498*
HGCN	0.797	0.780	0.797	0.568	0.518	0.568	0.642	0.590
HGCN-L	0.677	0.665	0.677	0.402	0.339	0.402	0.547	0.413
HGCN-DL	**0.814**	**0.804**	**0.814**	**0.574**	**0.517**	**0.574**	**0.647**	**0.604**

Table 4 shows that, HGCN-DL performs better than other models regarding micro-F1, macro-F1 and accuracy in node classification task. In the Cora dataset, it is 1.7% higher than HGCN in micro-F1 and accuracy, and 2.4% higher in macro-F1. For the Citeseer dataset, it is 0.6% higher than HGCN in micro-f1 and accuracy. Regarding the Wikipedia dataset, it is 0.5% higher than HGCN in micro-F1 and 1.4% higher in macro-F1. Table 4 further indicates that HGCN-DL, which combines shallow linear local and global structures, can also capture and represent undirected graph topology more fully on citation and language networks.

However, for the citation and language network datasets, our HGCN-L model performs slightly worse than HGCN regarding micro-F1, accuracy and macro-F1. This result may be related to the structure of the selected dataset. Compared with the Traffic network, the citation and language network datasets have sparser structures. This shows that in the citation and language network datasets, the integration of global information is more beneficial to downstream node classification tasks. Therefore, compared with the experimental results from the traffic network datasets, the citation and language networks fuse global information to make the classified nodes more distinguishable.

4.5 Experimental results comparison

In Section 3.1, we propose a linear structure representation for the model, including the local linear embedding of the first-order neighborhood and global linear embedding. We directly join the local linear embedding and global linear embedding to get the model, named DL. To verify whether the effect of HGCN-DL is entirely influenced by the shallow linear structure representation, the shallow linear model DL is compared in node classification task. The experimental results are shown in Figures 3 and 4.

By comparing the results in Figures 3 and 4, one finds that the HGCN-DL model presented in

this paper performs better than the shallow linear model DL in terms of micro-F1 and macro-F1 on all data sets. This shows that the effect of HGCN-DL is not completely affected by the shallow linear structure representation, that is to say, HGCN-DL can capture both linear and nonlinear network structures.

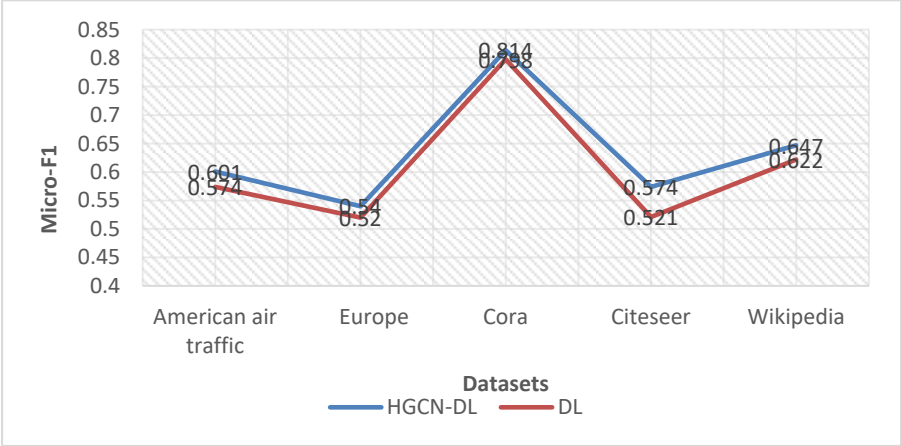


Fig. 3. Micro-F1 results for all datasets.

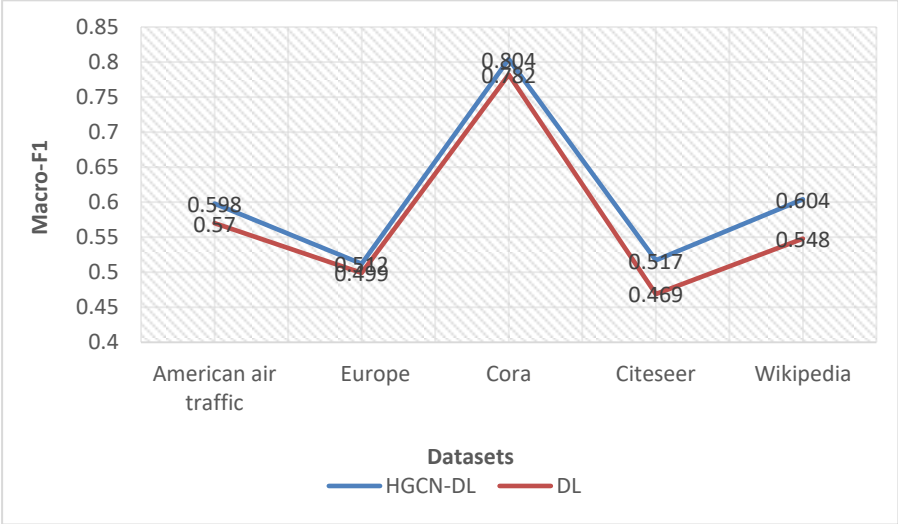


Fig.4. Macro-F1 results for all datasets.

4.6 t-SNE

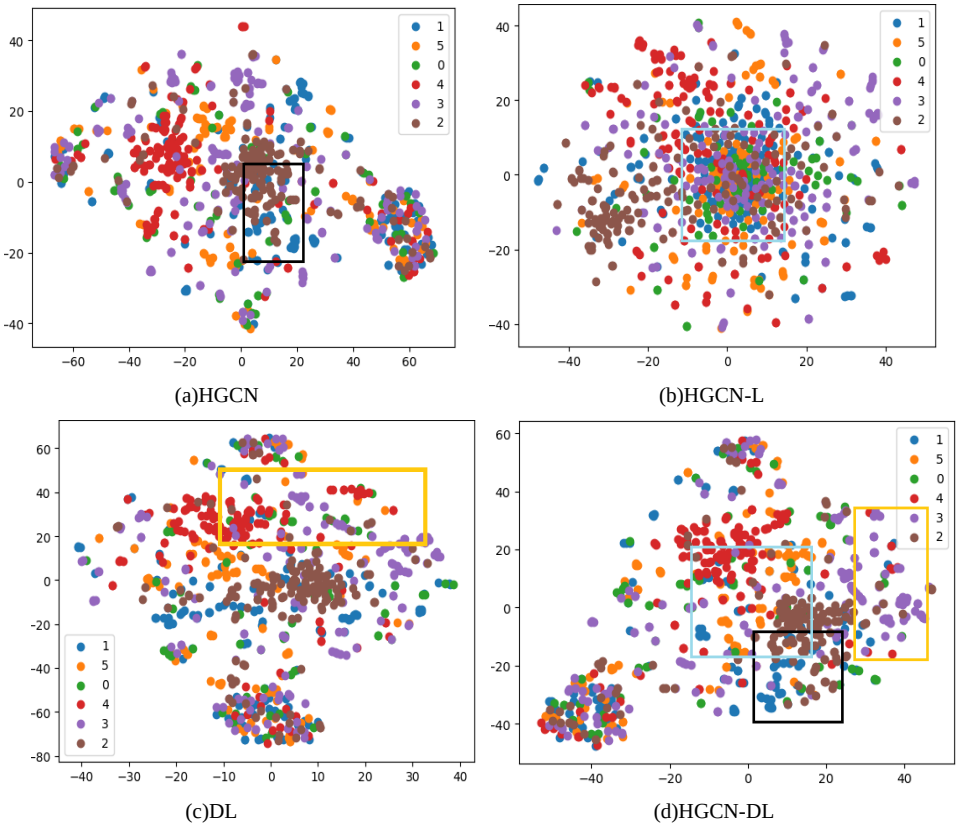
For a more intuitive comparison model, there are three different kinds of datasets, including the citation, traffic, and language networks, which are visualized in a reduced dimension in Sections 4.6.1-4.6.3, respectively. The results are shown in Figures 5-7.

4.6.1 Citeseer Network

Figure 5 (a) shows the dimensionality reduction for the original HGCN model, where categories 1-3 in the black framed parts are not obvious. Figure 5 (b) shows the model with local linear features. The edge category features are clear, while the six categories in the middle have no obvious boundaries. Compared with Figure 5 (d), the HGCN-DL model proposed in this paper can refine some categories in the Citeseer dataset, particularly data in the intermediate categories in Figure 5 (b).

Compared with the shallow linear model, the t-SNE results of DL are given. As shown in Figure 5 (c), the shallow linear structure causes features in categories 3 and 4 in the Citeseer dataset to appear mixed. Compared with the corresponding observations in Figure 5 (d), HGCN-DL, which integrates the nonlinear structure, successfully distinguishes categories 3

and 4. This indicates that the fusion of the deep nonlinear and shallow linear structures can describe the features of nodes more comprehensively.



(a)HGCN (b)HGCN-L

(c)DL (d)HGCN-DL

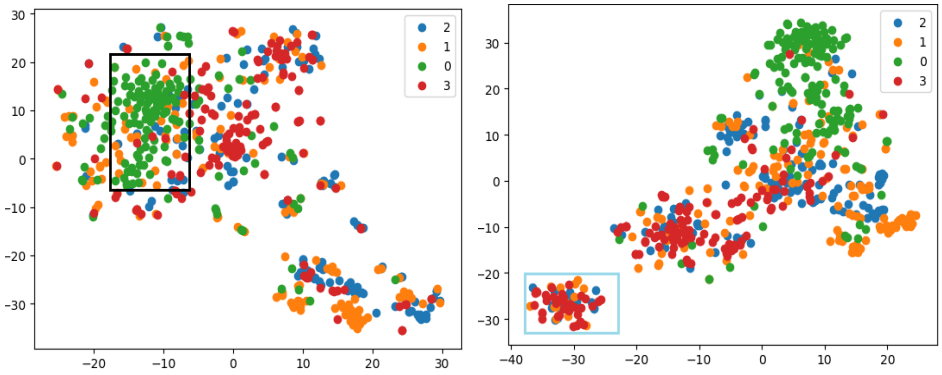
Fig.5. Dimensionality reduction results for the Citeseer network dataset in citation network.

The framed parts regions in (d) correspond to the observation results in (a)-(c), respectively.

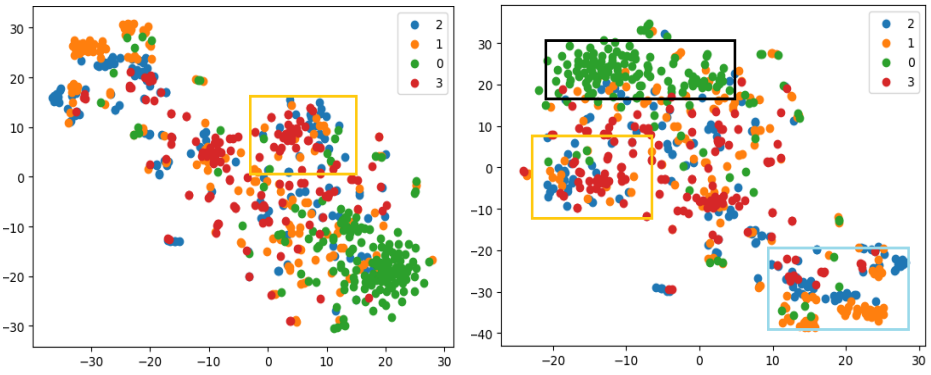
4.6.2 American air traffic network

The American air traffic network comes from struc2vec [17], where air traffic in the US is divided into four categories based on airport activity. Figure 6 (a) shows that the original HGCN model does not distinguish category 0 from category 1, while Figure 6 (d) displays that HGCN-DL separates category 1 from category 0. Compared with HGCN-L in Figure 6 (b), HGCN-DL makes the dataset more obvious in local distribution.

Similarly, compared with DL in Figure 6 (c), Figure 6 (d) shows that HGCN-DL, the addition of a nonlinear structure, can successfully separate the label nodes in category 2 in the edge portion of the graph, which makes the distribution boundaries in different categories of labels more obvious.



(a)HGCN (b) HGCN-L



(c)DL (d)HGCN-DL

Fig.6. Dimensionality reduction visualization of the American-air-traffic network dataset in traffic network.

The framed parts regions in (d) correspond to the observation results in (a)-(c), respectively.

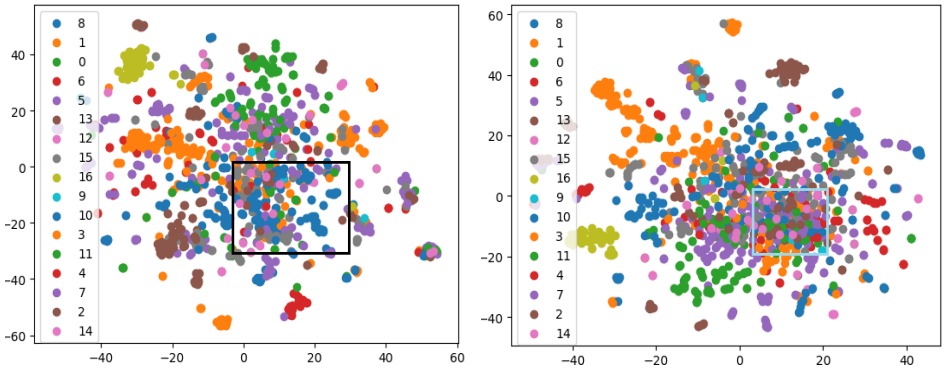
4.6.3 Wikipedia Network

The Wikipedia language network dataset consists of 17 categories, 2405 Wikipedia pages, and 17981 links, and it is denser than the citation and traffic network datasets.

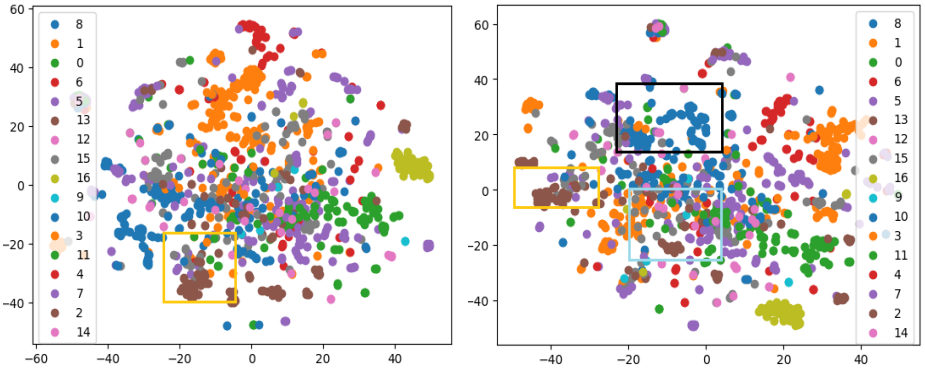
In Figure 7(a), the results determined with HGCN show that the boundary between nodes for category 10 is near the middle edge, and the nodes in partial category 3 are blurred. However, the observations in Figure 7(d) show that HGCN-DL can be used to successfully separate the nodes of category 3.

In Figure 7(b), HGCN-L with shallow linear local information can better describe the node labels at the edge of the dataset. For the denser nodes in the middle of the Wikipedia, HGCN-L has a weak ability to describe. The above problem is also seen in HGCN-DL, but compared with HGCN-L in Figure 7(b), HGCN-DL improves the representation ability of HGCN-L for dense node distributions.

Similarly, the t-SNE results obtained with DL are given. Through the comparison between Figure 7(d) and Figure 7(c), it is found that HGCN-DL can capture some category 3 nodes in the language network so that it can be successfully distinguished from nodes in category 5 and category 13 at the edge of the distribution.



(a)HGCN (b)HGCN-L



(c)DL (d)HGCN-DL

Fig. 7. Dimensionality reduction visualization of the Wikipedia network dataset in language network

The framed parts regions in (d) correspond to the observation results in (a)-(c), respectively.

5. Conclusion

In this study, we explore a graph embedding approach on combining linear and nonlinear network structures. Because existing shallow and deep learning models cannot capture the nonlinear and linear structural information of nodes simultaneously, we introduce hierarchical graph convolution network to propose the HGCN-L and HGCN-DL methods respectively. To verify the effectiveness of the model, node classification experiments were conducted for citation network, traffic network, and language network data sets. Dimensionality reduction visualization experiments for the three kinds of data sets were conducted.

Besides, we come to the following conclusions:

(1)From the results of the node classification experiment, for the traffic network data set, the results of HGCN-L show that the local topological structure indicates that nodes to be classified are more significant; the proposed HGCN-DL model can capture and represent the topology of the undirected graph more comprehensively.

(2)Compared with the linear model DL, our model HGCN-DL performs better in node classification, which indicates that our model can further capture the depth nonlinear structure.

(3)The dimensionality reduction visualization experiment shows that HGCN-DL can capture and distinguish easily confused class label nodes in both sparse traffic networks, dense citation networks, and denser language networks.

In general, HGCN-DL can represent the topology structure of undirected graphs effectively. However, compared with HGCN, HGCN-L performs well in the traffic network, performs poorly in citation and language networks. We suspect that this is related to the structure of the dataset itself. In the future, we will provide evidence for this conjecture.

Acknowledgment

This research was supported by the National Key Research and Development Program of China (No. 2018YFB1005103), the National Natural Science Fund of China (No. 61936012), and the Natural Science Foundation of Shanxi Province (No: 201901D111028).

References

[1] P Goyal, E Ferrar: Network embedding Techniques, Applications, and Performance: A Survey[J]. Knowledge-Based Systems, 2(17),155-164(2017).

[2] L Tang, H Liu: Leveraging social media networks for classification[J]. Data Mining and Knowledge Discovery, 23(3),447-478(2011).

[3] W L Hamiltion, R Ying, J Leskovec: Representation learning on graphs: Methods and applications[J]. IEEE Data Engineering Bulletin, 2017.

[4] Y Bengio, A Courville, P Vincent: Representation learning: A review and new perspectives[J].IEEE TPAMI, 35(8),1798-1828(2013).

[5] M D Ou, P Cui, P Pei: Asymmetric transitivity preserving network embedding. In: Proceedings of ACM, 2016.

[6] J Pennington, R Sorcher: Manning. GloVe: Global vectors for word respresentation. In: Proceedings of KDD, 2014.

[7] B Mikhail, N Partha: Laplacian eigenmaps and spectral techniques for embedding and clustering. Neural Information Processing Systems,pp.585−591(2002).

[8] S Wold, K Esbensen, P Geladi: Principal component analysis. Chemometrics and intelligent laboratory systems, , 2(1-3),37–52(1987).

[9] Joseph B Kruskal, Myron Wish: Multidimensional scaling, volume 11. Sage, 1978.

[10] C C TU, C YANG, Z Y LIU: A summary of network represents learning. Chinese Science: Information Science, 47, 980-996(2017).

[11] J Zhou, G Cui, Z. Zhang et.al: Graph neural networks: A review of methods and applications[J], CoRR, abs/1812.08434(2018) .

[12] M Tomas, S Ilya, K Chen: Distributed representations of words and phrases and their compositionality, Neural Information Processing Systems, pp. 3111--3119(2013).

[13] T. Lei, L Huan: Leveraging social media networks for classification, Data Mining and Knowledge Discovery, 23(3), 447—478(2011).

[14] P Bryan, Al R Rami, S Steven: Deepwalk: Online learning of social representations, In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701—710(2014).

[15] J Tang, M Qu, M Z Wang: Line: Large-scale information network embedding, In: Proceedings of the 24th international conference on World wide web, pp. 1067—1077(2015).

[16] G Aditya, L Jure: node2vec: Scalable feature learning for networks, In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855—864(2016).

[17] R Leonardo FR, S Pedro HP, F Daniel R: struc2vec: Learning node representations from structural identity, In: Proceedings of the 23rd ACM SIGKDD international conference on Knowledge discovery and data Mining, pp.385—394(2017).

[18] J Y Xie, R Girshick, A Farhadi: Unsupervised deep embedding for clustering analysis, International conference on machine learning, pp. 478—487(2016).

[19] D X Wang, P Cui, W W Zhu: Structural deep network embedding, In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD), pp. 1225–1234(2016).

[20] T Kipf, M Welling, Semi-supervised classification with graph convolutional networks, In: Proceedings of ICLR, 2017.

[21] P Velkovi, C Guillem, C Arantxa: Graph attention networks, In: Proceedings of ICLR, 2018.

[22] S Abuelhaija, A Kapoor, B Perozzi: N-GCN: multi-scale graph convolution for semi-supervised node classification, In: Proceedings of PMLR, pp. 841-851(2020).

[23] J Chen, T F Ma, C Xiao: FastGCN: fast learning with graph convolutional networks via importance sampling, In: Proceedings of ICLR, 2018.

[24] P Velickovic, W Fedus, W L Hamilton, P Lio : Deep graph infomax, In: Proceedings of the International Conference on Learning Representations (ICLR), NewOrleans, LA, USA, May 2019.

[25] Z N Hu, C J Fan, T Chen: Pre-training graph neural networks for generic structural feature extraction, In: Proceedings of ICLR, 2020.

[26] K Sun, Z X Zhu, Z C Lin: Multi-stage self-supervised learning for graph convolutional networks, In: Proceedings of AAAI, pp. 5892-5899(2020).

[27] M Qu, Y Bengio, J Tang: GMNN: graph markov neural networks, In: Proceedings of ICML,2019.

[28] A K McCallum, K Nigam, J Rennie: Automating the construction of internet portals with machine learning, Information Retrieval, v3(2), 127–163(2000).

[29] P Sen, G Namata, M Bilgic: Collective c lassification in network data[J], AI Magazine, 29(3), 93–106(2008).

[30]Qinghua University, OpenNE, https://github.com/thunlp/OpenNE.

[31] J Qiu, Y Dong, H Ma: Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec, In: Proceedings of the Eleventh ACM International Conference on Web Search and Data

Mining, pp.459—467(2018).

[32] V Armanda, G Darioa, P Ferrana: Studying the impact of the Full-Network embedding on multimodal pipelines[J], Semantic Web, 10(5), 909-923(2019).

[33] R Giuseppe, A Claudia, F Nicola: An unsupervised approach to disjointness learning based on terminological cluster trees[J], Semantic Web, Pre-press, 1-25(2020).