A Representation Learning Algorithm Preserving Structure and Attribute Proximity

Yu Liu*, Rui Wang, Jiawang Chen
School of Computer Science, Shaanxi Normal University, Xi’an, China
*Corresponding author’s email: liuyuy@snnu.edu.cn

Abstract. Representation learning is the application of data mining on complex networks, which extracts high-dimensional information from nodes to form a low-dimensional representation vector, and offers a better input to downstream application. So it has attracted many researchers’ attention recently. In a complex network, there are many kinds of node information, including the attribute information and structure information which are of great significance for data mining because they determine the nature of the node. However, the limitation of popular methods is that only the structure information between the node is considered, the rich high-dimensional attribute information of nodes has been ignored. The relationship between these attribute information of nodes in complex network is also important reference in the process of node representation learning. To address this problem, we propose an algorithm that uses the autoencoder to extract features based on fusion with node structure features and attribute features, and gets a more representative node representation vector. We apply the new algorithm on different datasets compared with popular baseline algorithms. The results show that the method we proposed is effective and it has good performance in visualization and node classification.

Keywords. Deep learning, network representation learning, attribute complex network

1. Introduction
Complex network is abstracted from complex systems in real life and has diverse relationships. Such as the citation network, social network, and user purchase network etc. These networks use node attribute, edges and label information to indicate the interaction between different kinds of things. For example, a typical social network is composed of user nodes. The edges between users represent the friend relationship. User nodes are often accompanied by user attributes, such as user age, registration Date, region, label, user introduction etc. These multi-dimensional features become our main pointcut, and they are difficult to use when we do network data analysis tasks. The sparse representation of nodes is converted into low-dimensional dense vectors and the features of nodes are preserved, which can help us to deal with complex systems to a greater extent. NRL (Network Representation Learning) [1] is a bridge connecting the original network data and network application tasks. Its algorithms extract features of the network, to form the low-dimensional representation of the node which can be used as the node features for downstream network application tasks, such as node classification [2], link prediction [3] and community detection [4,5] etc, and significantly enhance the performance of these tasks. It further facilitates people to understand and process the systems represented by these complex networks.
1.1. Problem of general representation learning methods
In the early stage of research, NRL uses matrix decomposition to get low dimensional features vectors. In recent years, inspired by the technology of word embedding in the field of natural language processing [6], researchers have proposed some shallow neural network algorithm to extract the structure features of nodes.

As complex networks are abstracted from complex systems, there must be a variety of data and relationships. When embedding the nodes, both the structure information and the attribute features of nodes should be considered. In existing NRL methods, most of them only pay attention to the structure features of nodes for embedding. In the process of node embedding, we cannot get good representation vector of two nodes if there is no connection or common neighbour. In this paper, we focus on the relationship between its own attribute and use it as a basis to strengthen the node embedding representation.

1.2. Our improvements to the problem
Inspired by the idea of homogeneity of complex networks, we propose a different algorithm SFANE(network embedding of structure fusion attribute) for the existing typical representation learning algorithm, which combines the attribute and structure features of the nodes as input, and uses deep neural networks for feature extraction to enhance performance and reduce running time. In this method, we extract the attribute matrix of the dataset during pre-processing. In the model training, in order to better optimize our embedding problem, we construct the attribute similarity matrix of the nodes to measure the relationship between the nodes, which can strengthen the optimization process of node embedding.

1.3. Our work
- We use structure features and attribute features to optimize at the same time.
- We extract the similarity relationship between node attributes, and use the relationship between attributes to optimize the similarity loss function for node representation learning.
- We propose a network representation learning algorithm with the fusion of structure and attribute.

1.4. Structure of paper
The remaining of this paper is organized as follows. We use the section 2 to introduce the related work; the pre-processing work and the system model is described in section 3. In the section 4, we use the representation vector trained by the algorithm for visualize and node classification to measure the performance of our algorithm.

2. Related work

2.1. Research status of network representation learning
To sum up, we will introduce related work from different perspectives of the NRL methods. In particular, Peng. cu [7] provides a comprehensive analysis of many different NRL algorithms.

In these algorithms, the first consideration is the adjacency structure features of nodes, and the further is on rich attributes information with development on network representation, because the embedding method only structure features cannot meet the need of actual research.

2.2. NRL based on structure features
Based on matrix processing, LLE [8] (Local Linear Embedding) and ISOMap [9] (IsoMetric Mapping) have been evolved, which use the local structure of the node to obtain the node embedding.

Inspired by word embedding technology, DeepWalk [10] generates a sequence of nodes by tructing as sentences according to the edges, is a kind of probabilistic model to predict correlation of two nodes.
Node2vec [11] optimizes the random walk of the DeepWalk which obtains a better walk sequence. Both of two algorithms focus on the user’s local structure to embed nodes.

LINE [12] considers more node structures so it defines the first-order and the second-order proximity of nodes. On the basis of optimizing the proximity loss function, the node representation vector is obtained. LINE is similar to the idea of SDNE [13]. The difference between the LINE is that the SDNE uses a deep neural network to extract the structure features on the basis of structure proximity. Grarep [14] defines the K-order proximity to enhance embedding effect of the network, but its matrix factorization method reduces the support for large network datasets.

2.3. NRL algorithms of adding learning attributes

TADW [15] regards text information as the attributes uses matrix decomposition that combines the text information to obtain a new embedding result; HSCA [16] is an enhancement of TADW, which uses the stronger first-order proximity of the network on the basis of TADW to get better effect.

CANE [17] uses the attention mechanism in the neural network and the attributes to get the context-aware network embedding;

AANE [18] decomposes the node attributes matrix together with the structure similarity loss function by forming a joint optimization problem of two convex functions to obtain the node embedding; SINE [19] focuses on the incomplete attributes in the network, and eliminates the impact of incomplete attributes on embedding as much as possible.

Most above-mentioned methods extract the attributes embedding separately and the final vector is obtained by splicing attribute vector and structure vector.

2.4. Our representation learning method

There is heterogeneous information in the attribute network. So we believe structure and attributes features of the node should work together in the input layer. The separation of attribute embedding and structure embedding dissorts the manifold relationship between nodes and fails to show the complex relationship between structure and attributes.

3. Network representation learning based on structural fusion attributes

In this section, we introduce the concept and related definitions of SFANE in section 3.1. In section 3.2, we introduce our proposed framework. Finally, we summarize the algorithm analysis and expand the direction of future work in Section 3.3.

3.1. Problem definition

| Notation | Definitions |
|----------|-------------|
| $|V|$ | number of vertexes |
| $|E|$ | number of edges |
| $F$ | attribute matrix of complex network |
| $S$ | attribute proximity matrix of network |
| $d$ | dimension of representation vector |
| $X$ | fusion of attribute and adjacent matrix |
| $zi$ | representation vector of $Vi$ |

In this section, the above Table 1. Marks the main symbols used in the article. Below we give some definitions used in the algorithm.
3.1.1 Definition 1: Attribute network An attribute network is defined as a graph $G=(V, E, A, F)$, where $V=\{v_1, v_2, \ldots, v_n\}$ represents the node set; $E=\{e_{ij}\}$ represents the edge set; $A=\{a_{ij}\}$ represents the adjacent matrix where $a_{ij}$ is the weight of the edge between the nodes. In the directed graph, $a_{ij}$ may not be equal to $a_{ji}$. The $f_i=\{x_1; x_2; x_3 \ldots; x_m\}$ summarizes all the attributes of the node, $F=\{f_1, f_2, f_3, \ldots, f_n\}\in \mathbb{R}^{m*|V|}$ is the attribute matrix of the node.

3.1.2 Definition 2: Attribute network represents learning Given the attribute network $G$, the main task of the algorithm is to learn the mapping function $f:G\rightarrow Z\in \mathbb{R}^{|V|*d}$ by processing the graph $G$, which can represent the node $v_i\in V$ in the network as a low-dimensional vector $z_i\in \mathbb{R}^d$, where $d<<|V|$. The embedding vector $z_i$ saves the structure and attribute features of the node after mapping.

3.1.3 Definition 3: Attribute proximity and its matrix In the process of network representation learning, in order to save the attribute features of nodes, it is necessary to calculate the attribute similarity of the nodes to form a matrix $S\in \mathbb{R}^{|V|*|V|}$, where the $S_{ij}$ is obtained by calculating the attribute similarity of nodes $v_i$ and $v_j$. Then we use it to optimize the embedding loss function.

3.2. The model

3.2.1 Dataset pre-processing and framework

![Diagram](image)

**Figure 1.** The model framework of structural fusion attributes

As shown in Figure 1, we first process original dataset of complex network, using the mapping function to process the relationship file to obtain the $E$. In order to get attribute feature matrix of nodes, we filter attribute feature file, remove the too discrete and unrepresentative attributes, such as the user’s height, weight and hair color etc to get feature matrix $F$. Then we construct the matrix of attribute proximity by using Cosine similarity to calculate attribute proximity between two nodes. The calculation formula of similarity as follows

$$S_{ij} = \frac{f_i \cdot f_j}{\|f_i\| \cdot \|f_j\|} \quad (1)$$

Where $f_i=\{f_1, f_2, f_3, \ldots f_m\}$ is the feature vector of node $v_i$. We encode and classify the feature vector, to maintain $f_i>0$ . In the similarity matrix $S$, $s_{ij}$ is the similarity value of node $i$ and node $j$ calculated by the Eq (1).

The structure and attribute information are heterogeneous information of the attribute complex network. The vector splicing after separate representation learning can also be used as the result as
well, but it is insufficient to describe the relationship between the structure and attributes of nodes. Inspired by Symmetric normalized Laplacian [20] and GCN [21,22], we use the adjacency matrix standardization and feature fusion to fuse the structure information and attribute information of the node and get new input for the neural network. The use of these two processes preserves structure and attribute information of neighbours the same time. The specific operation is shown as:

$$X = D^{-1/2}LD^{-1/2}F$$  \hspace{1cm} (2)

Where X is a matrix that combines structure and attribute features, as the input of the neural network, L is the adjacency matrix of the attribute network plus its own connection, and the D matrix is the degree matrix of the network, which is a diagonal matrix. The calculation method is shown as:

$$L = A + I$$ \hspace{1cm} (3)

$$d_{ii} = \sum_{j=1}^{n} a_{ij}$$ \hspace{1cm} (4)

Because the diagonals of the adjacency matrix are all 0, the inner product of the adjacency matrix and the attribute matrix is equivalent to a weighted sum. The value of the node feature becomes the weight of the adjacency matrix, and its own features are ignored. To avoid this situation, we can firstly add an identity matrix I to A.

The adjacency matrix A has not been normalized, so it is difficult to limit the data to the range we need. Through normalization, the data can be made comparable while maintaining the relationship between the data. In order to avoid the multiplication of adjacency matrix and attribute matrix change the original distribution of features. We need to standardize A and get the inner product of adjacency matrix and attribute matrix after the normalization. This operation can make the structure and attribute information fuse before input, and obtain the better input for our model.

Our task is to get the low dimension representation vector of the node and maintain the structure and attribute information of the node. For this purpose, we use the autoencoder structure [23] of the neural network, which can quickly and effectively extract the nonlinear feature relationship of the input data and get good representation learning of nodes. At the data respect, high-dimensional data can be encoded into a low-dimensional data through a multilayer neural network. For the method of the article, we use each row of the feature matrix X that combines structure and attribute information as the input of the neural network. The feature is compressed by the encoder, and the reconstruction loss is optimized when decoder is running out. And we can define better loss functions to get good optimization results. The expression of the hidden layer of the autoencoder is shown as:

$$h^{(i)} = \sigma(W^{(i)}x_i + b^{(i)})$$

$$h^{(k)} = \sigma(W^{(k)}z^{(k-1)}_i + b^{(k)}), k = 2, \ldots, L$$ \hspace{1cm} (5)

Where K represents the number of auto-encoder layers; $\sigma$ is the activation function which is Sigmoid function or Relu; W and B are the weight matrix and bias of each layer.

3.2.2 Loss function First, when optimizing the autoencoder, only the reconstruction loss is generally used as the objective loss function for optimization. For our method, due to the existence of structure and attribute information, we define a better loss function for the heterogeneous information. As shown in the following (6):

$$L_n = \sum_{i,j=1}^{n} (a_{i,j} + s_{i,j}) ||z^{k}_i - z^{k}_j||^2_2$$ \hspace{1cm} (6)
Where $a_{ij}$ is the edge between nodes in the adjacency matrix, and $s_{ij}$ is the weight of attribute similarity matrix that measures the proximity of attributes between nodes. Inspired by SDNE and Laplacian Eigenmaps [24], we use a separate loss function to form a semi-supervised model in the latent representation. For nodes with low similarity, penalties should be imposed. The two nodes with higher similarity should be better optimized.

Second, for autoencoders, the goal is to get a smaller reconstruction loss. Therefore, for the method of this article, we have a reconstruction loss for the input fusion feature matrix $X$, as shown in the following (7):

$$L_b = \sum_{i=1}^{n} \| (\hat{x}_i - x_i) \|^2_2$$

(7)

The reconstruction criterion can smoothly capture the data manifolds and thus preserve the similarity between samples. So the $\hat{x}_i$ is reconstruction of input.

Finally, we get joint loss function (8) as the optimization target by combining (6) and (7), as shown in following:

$$L_{total} = L_a + L_b + \lambda L_{reg}$$

(8)

Where $\lambda$ is the regularization parameter.

In the above, we use regularization parameters to strengthen the anti-interference ability of the model, and try to make every feature value produce the result influences. The regular terms are defined as

$$L_{reg} = \frac{1}{2} \sum_{k=1}^{K} (||W^{(k)}||_F^2 + ||\hat{W}^{(k)}||_F^2)$$

(9)

Where $||.||_F$ is the L2 norm, $W$ is the encoder weight, and is the decoder weight.

3.3 Discussion and following work

It is not hard to see that the training complexity of neural network is $O(ndIc)$, and the pre-processing complexity is $O(n^2)$. In the above two tasks, $n$ is the number of nodes, $d$ is the maximum dimension of the hidden layer, $c$ is the average degree of the network, and $I$ is the number of iterations. The parameter $d$ is usually related to the dimension of the representation vector $z_i$. For $c$, it can be regarded as a constant in practical applications.

In the following work, we will further improve its performance in heterogeneous information network. The method of this paper is based on our open project, which aims to extract the security attribute information of nodes in the network during the embedding process. However the datasets are mostly heterogeneous networks. We will further improve this method by considering the security attributes and heterogeneous network structure for presentation learning.

4. Experiment

In this section, we present the visualization and node classification results of the baseline algorithms to verify the effectiveness of our algorithm. Node classification task is one of the most commonly used downstream tasks in evaluating algorithm’s embedding performance.

4.1. Datasets

In this work, we use three real-world attribute networks, Soc-pokec, Citeseer and Cora, which can be publicly available download in the page of Stanford network dataset (http://snap.stanford.edu/data/).

And detailedly the statistics of the datasets are summarized in Table 2.
Table 2. Detail information of datasets

| Dataset     | Soc-pokec | Citeseer | Cora |
|-------------|-----------|----------|------|
| Nodes(|V|)    | 5000      | 2708     | 2708 |
| Edges(|E|)  | 6852      | 5278     | 5278 |
| Attributes(m)| 9         | 3703     | 1433 |
| Labels(l)  | 10        | 6        | 7    |

**Soc-pokec** is the most popular social network data set in Slovakia. Each user as a node has its attribute information file. In the algorithm verification, in order to verify the availability of the algorithm, we select subset of soc-pokec aggregate the too discrete values and retain as much node attribute information as possible. We use the user's region as the user label for classification tasks. The experiment completed the single-label classification task.

**Cora** is a citation network containing 7 fields as labels. Each paper contains a title and abstract content. The data set has a single label information, and the experiment has completed the single label classification task.

**Citeseer** is a citation network dataset, which contains 6 research fields, and each node of paper contains the same title information and abstract content as Cora. The dataset contains a single layer of label information and the experiment completed the single-label classification task.

The experiment of dataset in our paper uses python3. 6 interpreter, based on pytorch1. 2. 0 implementation, using a single GPU_NVIDIA GeForce 970m for acceleration. It runs on computer with Intel core i7-6700K, 32G memory operating system, Windows 10 educational (64 bit).

4.2. Node classification algorithm performance

In this section, the node classification is used to evaluate the performance of our algorithm. Specifically, we first use the method of the article to learn the representation of different networks. Then we can randomly select from 10% to 90% nodes to train the logistic regression classifier, use the labels of the nodes in the data set as the training labels, and the remaining nodes as the test set for classification.

Finally, we calculate the Marco-F1 and Micro-F1 values as the test results. This process is repeated 10 times and the average value is taken as the result of the final node classification. The classification performance values of different datasets are drawn in Table 3. and Table 4. below respectively.

The method proposed in the article using attributes and attribute similarity to strengthen representation learning shows good embedding performance on the Soc-pokec dataset. It can be seen that the performance of our method is better than the baseline representation learning method which only considers structure features. It is proved that our method performs well in the network which has aggregation of node property comparison and rich edge relation like soc-pokec.

Table 3. Result of soc-pokec node classification on Soc-pokec dataset

| Metric       | Algorithm | Training percentage |
|--------------|-----------|---------------------|
|              | 10%  | 20%  | 30%  | 40%  | 50%  | 60%  | 70%  | 80%  | 90%  |
| Micro-F1     |       |       |      |      |      |      |      |      |      |
| Deepwalk     | 0.33  | 0.37  | 0.38 | 0.41 | 0.40 | 0.41 | 0.42 | 0.42 | 0.40 |
| Node2vec     | 0.40  | 0.42  | 0.44 | 0.45 | 0.44 | 0.44 | 0.45 | 0.44 | 0.45 |
| Line         | 0.36  | 0.37  | 0.37 | 0.40 | 0.40 | 0.41 | 0.40 | 0.40 | 0.41 |
| Sfane        | 0.42  | 0.50  | 0.53 | 0.55 | 0.56 | 0.60 | 0.57 | 0.61 | 0.65 |
| Macro-F1     |       |       |      |      |      |      |      |      |      |
| Deepwalk     | 0.23  | 0.26  | 0.27 | 0.30 | 0.28 | 0.28 | 0.30 | 0.29 | 0.26 |
| Node2vec     | 0.27  | 0.29  | 0.31 | 0.31 | 0.26 | 0.26 | 0.27 | 0.27 | 0.26 |
| Line         | 0.23  | 0.23  | 0.25 | 0.26 | 0.26 | 0.25 | 0.27 | 0.27 | 0.26 |
| Sfane        | 0.36  | 0.45  | 0.48 | 0.50 | 0.52 | 0.56 | 0.53 | 0.57 | 0.61 |
Table 4. Result of citeseer node classification on Citeseer dataset

| Metric   | Algorithm | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% |
|----------|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|          |           |     |     |     |     |     |     |     |     |     |
| Micro-F1 | Deepwalk  | 0.52| 0.55| 0.56| 0.57| 0.58| 0.58| 0.59| 0.59| 0.59|
|          | Node2vec  | 0.52| 0.55| 0.57| 0.57| 0.59| 0.59| 0.60| 0.60| 0.59|
|          | Line      | 0.36| 0.39| 0.41| 0.40| 0.42| 0.44| 0.45| 0.44| 0.48|
|          | Sfane     | 0.44| 0.49| 0.54| 0.55| 0.56| 0.59| 0.59| 0.60| 0.63|
| Macro-F1 | Deepwalk  | 0.48| 0.51| 0.52| 0.53| 0.55| 0.54| 0.53| 0.54| 0.57|
|          | Node2vec  | 0.46| 0.50| 0.52| 0.52| 0.53| 0.53| 0.54| 0.55| 0.54|
|          | Line      | 0.31| 0.34| 0.36| 0.36| 0.37| 0.40| 0.41| 0.38| 0.41|
|          | Sfane     | 0.38| 0.45| 0.48| 0.51| 0.52| 0.56| 0.57| 0.57| 0.58|

In the experiment of the citeseer data set, we use one-hot encoding of the node's keywords as attributes. Compared with the soc-pokec dataset, the attribute vector is longer and sparse. In the embedding training of the article algorithm, the performance curve’s upward trend of is more ideal, and it has strong competitiveness compared with the baseline method. But in the later stage, we consider to aggregate the attribute information of the text and input it into our model. In terms of training time, it is better than the baseline, and embedding vectors with good results can be trained in a short time. In the experiment of the Cora dataset, it also has similar results to citeseer, which can well capture the non-linear relationship between the structure and attribute of the node. This indicates that node attributes are importance in network representation learning, and fusion of attribute information can greatly improve the performance of network representation learning methods.

In summary of the experimental results in Table 3 and Table 4 and the above analysis, the algorithm in our paper performs well in node classification tasks, and has good performance compared with other methods. It can more accurately predict the labels of unknown nodes in the network, thereby more effectively extracting more network information from the original network.

4.3. Visualization

Network visualization is a method to display the link relationship between nodes in a complex network. Using t-SNE, the learned node embedding can be visualized to observe the spatial structure of the network. As shown below, Figure 2 is the visualization results of different data sets learned by the algorithm in this paper. The effectiveness of the article method can be proved by visualizing the network representation.
On the three data sets, we get different visualization effects. It can be seen that the citation networks cora and citeseer have good embedding effects in Figure 2 (a) and Figure 2 (b) respectively. On the soc-pokec dataset, we have also obtained distinctive results. According to the features of social networks, a considerable degree of user nodes have fewer friends and their own information integrity is very low. So we will get as shown in Figure 2 (c), there are multiple types of nodes concentrated in the right part of the image.

5. Conclusion
In this paper, we propose a model, named SFANE, for network representation learning to perform the node embedding vector. It can capture nonlinear relation of node. Specifically, our method takes advantage of the attribute information of the node, optimizes the loss function by using the similarity of node attribute, further improves the structure embedding based on the node attributes, and enhances the relationship between the node embedding vector and the node features. The experimental results on three real network datasets show that our method can improve the performance of node embedding, especially when the attribute features are relatively complete, the embedding effect is greatly improved. So it has a certain competitiveness in the processing of complex network representation learning.

References
[1] Zhang D, Yin J, Zhu X, et al. 2018 Network representation learning A survey J. (IEEE transactions on Big Data) pp 3-28.
[2] Santos L. D, Piwowarski B, Denoyer L and Gallinari P. 2018 Representation learning for classification in heterogeneous graphs with application to social networks vol 12 no 5 (J. ACM Transactions on Knowledge Discovery from Data (TKDD)) pp 1-33.
[3] Wei X, Xu L, Cao B and Yu P S. 2017 Cross view link prediction by learning noise-resilient representation consensus (In Proceedings of the 26th International Conference on World Wide Web) pp 1611-1619.
[4] Tu C, Zeng X, Wang H, Zhang Z, Liu Z, Sun M, ... and Lin, L. 2018. A unified framework for community detection and network representation learning vol 31 no 6 (J. IEEE Transactions on Knowledge and Data Engineering) pp 1051-1065.
[5] Cavallari S, Zheng V. W, Cai H, Chang K. C. C and Cambria E. 2017 Learning community embedding with community detection and node embedding on graphs. In Proceedings of the 2017 (ACM on Conference on Information and Knowledge Management) pp 377-386.
[6] Goldberg Y, and Levy, O. 2014 Word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method (Arxiv Preprint Arxiv) pp 1402, 3722.
[7] Cui P, Wang X, Pei J, et al. 2019 *A survey on network embedding* vol 31 no 5 (J. IEEE Transactions on Knowledge and Data Engineering) pp 833-852.

[8] Roweis S T and Saul L. K. 2000 *Nonlinear dimensionality reduction by locally linear embedding* vol 290 no 5500 (J. Science) pp 2323-2326.

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

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

[11] Grover A and Leskovec J. 2016 *Node2vec: scalable feature learning for networks* (In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining) p 855-864.

[12] Tang J, Qu M, Wang M, Zhang M, Yan J and Mei Q. 2015 *Line: large-scale information network embedding* (In Proceedings of the 24th International Conference on World Wide Web) pp 1067-1077.

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

[14] Cao S, Lu W and Xu Q. 2015 *Grarep: learning graph representations with global structural information* (In Proceedings of the 24th ACM international on Conference on Information and Knowledge Management) pp 891-900.

[15] Yang C, Liu Z, Zhao D, Sun M and Chang E. Y. 2015 *Network representation learning with rich text information* vol 2015 (In International Joint Conference on Artificial Intelligence) pp 609-618.

[16] Zhang D, Yin J, Zhu X and Zhang C. 2016 *Homophily, structure, and content augmented network representation learning* (In 2016 IEEE 16th International Conference on Data Mining) pp 609-618.

[17] Tu C, Liu H, Liu Z and Sun M. 2017 *Cane: context-aware network embedding for relation modeling* vol 1 (In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics) pp 1722-1731.

[18] Huang X, Li J and Hu, X. 2017 *Accelerated attributed network embedding. In Proceedings of the 2017 SIAM international conference on data mining* (Society for Industrial and Applied Mathematics) pp 633-641.

[19] Zhang D, Yin J, Zhu X and Zhang C. 2018 *Sine: scalable incomplete network embedding* (In 2018 IEEE International Conference on Data Mining) pp 737-746.

[20] Von L, U. 2007 *A tutorial on spectral clustering* vol 17 no 4 (J. Statistics and Computing) pp 395-416.

[21] Yang Z, Cohen W and Salakhudinov R. 2016 *Revisiting semi-supervised learning with graph embeddings* (In International Conference on Machine Learning) pp 40-48.

[22] Kipf T. N and Welling M. 2016 *Semi-supervised classification with graph convolutional networks* (J. International Conference on Learning Representations (ICLR 2017)).

[23] Hinton G. E and Salakhutdinov R R. 2006 *Reducing the dimensionality of data with neural networks* vol 313 no 5796 (J. Science) pp 504-507.

[24] Belkin M and Niyogi P. 2001 *Laplacian eigenmaps and spectral techniques for embedding and clustering* vol 14 (J. Advances in Neural Information Processing Systems) pp 585-591.