Network Representation Learning Algorithm Combined with Node Text Information

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

Abstract. In a network data analysis task, network representation learning play an important role. It can obtain a low-dimensional representative vector from high dimensional features of nodes which will promote the efficiency of downstream analysis task. Many algorithms mainly based on the structure information of the network but for a complex network, there is a lot kinds of feature information available for representation learning which has an important impact on network data analysis task. Therefore, we propose a network representation learning algorithm that combines node text information. Before our algorithm, we encode the information of the nodes including structure and text, and the strategy of fusion is adopted in advance which splice structure and text information. And then we feed the splicing results into the neural network which is a multi-layer feedforward neural network with a tower structure to learn the nonlinear structure information of complex networks. Finally, we design the structure loss function that preserve structure similarity and text loss function that preserve text information respectively. And we use the joint training to make two aspects information interact continuously. We conduct experiment to evaluate our algorithm. The results on two real word networks show our algorithm has better performance compared with traditional algorithms.

Keywords. Network representation learning, deep neural network, complex network, node text information

1. Introduction
Complex network is abstracted from various complex relationships in real life, its edges and nodes have different meanings in different scenes, which is an important form to express the connection between things. Such as the social network can express users’ friendship in MicroBlog, the protein interaction networks can express the connections between proteins and the citation network the reference relationship etc. Its research has important academic value and application value [1]. However, in the research of networks, an important problem is that the adjacency matrix which is the traditional network representation has high sparsity, so the high-dimensional sparse vector requires more computing space and running time. Inspired by word representation learning techniques, the researchers represent the nodes with a low-dimensional dense vector instead of high-dimensional sparse vectors. In the network analysis task, the performance of the task can be effectively improved by using low dimensional dense vectors of network representation learning (NRL) algorithm, such as in node classification [2,3], link prediction [4,5], node clustering [6], network visualization [7,8] etc.
At present, most of the NRL algorithms are based on the networks’ structure and few researches on the networks with rich text information. But in real life, the network nodes also contain rich text information, which is significant to the research of the network. For example, two users in social network who have the same hobbies but do not have connection relationship and common friends. It is difficult to capture such information based on the network’s structure.

To solve this problem that only consider structure information, we propose a NRL algorithm which combines node text information and structure features (NTI_NRL). Firstly, we encode the structure and text information of the nodes, and then we spliced the two encoded information together in advance as the input vector of our model. In the cause of extracting the nonlinear features better, we use the multi-layer feedforward neural network with a tower structure to obtain the node representation vector of complex networks. After that in the model optimization process we design the structure loss function that preserve structure similarity and text loss function that preserve text information respectively. And we use the joint training to make two aspects information interact continuously, so that the final node representation vector has structure and text information.

We summary our contributions as follows:

- We propose NTI_NRL to perform NRL by preserving structure similarity and text similarity of network. So the representation vector we get has the structure and text information at the meantime.
- We use node classification and visualization on two citation networks. The experimental results show our algorithm has better performance compared with traditional algorithms.

The rest of this paper is organized as follows. We introduce some common NRL algorithms in section 2. In section 3, we present the detailed of our model. In section 4, we empirically evaluate NTI_NRL on node classification and visualization task on two data sets. In section 5, we give conclusions and high light some directions for future work.

2. Related work

There are many existing NRL algorithms. Spectral clustering is an early NRL algorithm, which obtains the k-dimensional representation of the node by calculating the first k feature vectors or singular vectors of the relationship matrix. Because the calculation time of feature vectors and singular vectors is nonlinear, so the time complexity of the spectral clustering method is high. And the entire relation matrix needs to be stored in the memory so the space complexity also needs to be considered.

With the wide application of neural network, the researchers propose many algorithms which base on shallow neural network. In [3], inspired by word representation algorithm Word2Vec, the authors propose DeepWalk, which obtain low-dimensional representation of nodes by using Skip_gram on the node sequence that can be get by using truncated random walk. In [9], in order to optimize the process of random walk, the authors proposed Node2Vec which introduce two hyper parameters $p$ and $q$ to control whether the random walk is breadth-first or depth-first. In [10], for large-scale directed weighted networks, the authors propose Line which propose first-order similarity to describe local structure features and second-order similarity to describe global structure features of the network.

In [8], SDNE is proposed which is first uses deep neural network. It use the first-order neighbor relations in the network node as the supervised information of autoencoder neural network and second-order neighbour relations as unsupervised information of autoencoder neural network to obtain the representation vector, so realizes the preservation of the local structure. In [11], the authors prove that DeepWalk actually decomposes a specific relation matrix. On the basis of this, they propose TADW, which embeds the text information of the node into the matrix decomposition process. However, TADW matrix factorization process consumes time and space, making it difficult to scale to large-scale networks. In [12], CENE integrates text modeling and structure modeling in a general framework, and considers text information as a particular node. In [13], CANE is used to learn the low-dimensional vector representation by considering different contexts of nodes. This algorithm link the structure-based and text-based embedding vector as the final representation vector, but CANE ignore the relationship of structure-based and text-based embedding.
3. Method

3.1. Related Definitions
In order to describe the proposed algorithms clearly, relevant definitions and their main symbols are shown in Table 1.

| Symbol | Definition |
|--------|------------|
| V      | set of network nodes |
| | | number of nodes |
| $v_i$  | node of number i, $v_i \in V$ |
| E      | set of network edges |
| $s_i$  | structure coding vector of $v_i$ |
| $t_i$  | text coding vector of $v_i$ |
| Nr     | neighbor node set of $v_i$ |
| $u_i$  | representation vector of node $v_i$, $u_i \in \mathbb{R}^d$ |
| U      | matrix of node represents vector |

**Definition1 (network)** Let $G = (V, E, T)$ represent a network which has rich text information, where $V = \{v_1, v_2, ..., v_n\}$ is the set of nodes; $E = \{e_{ij}\}$ is the set of edges between nodes, $e_{ij}$ represent $v_i$ has an edge with $v_j$; $T$ is the set which contain the text of nodes, $T = \{(v, txt) \mid v \in V; txt = \{W_i\}\}$, where $w_i$ is the word sequence $<w_1, w_2, ..., w_n>$ which has all words of the text information of the node $v$.

**Definition2 (NRL)** For a network $G = (V, E, T)$, the purpose of NRL is to allocate a low dimensional real-valued vector representation $u_i \in \mathbb{R}^d$ for $v_i \in V$, where $d << |V|$ and make the $u_i$ retain the structure and text information of the node $v_i$.

3.2. Model framework

In order to get the representation vector has structure and text information, we propose NIT_NRL algorithms which model framework is shown in Figure 1. We introduce our algorithm from three aspects that is data coding, neural network structure and loss function in detail.

**Figure 1. Model Framework**

3.2.1. Data coding. In order to facilitate the use of deep neural network for network representation learning, the node structure and text information are encoded respectively.

**Structure information coding.** For a network $G$, according to the edge set $E$ we construct the network’s adjacency matrix $S$ which contains all structure information of the network. We set $s_{ij} = 1$
when nodes $v_i$ and $v_j$ has an edge, otherwise $s_{ij} = 0$. We select the row vector \( S_i \) in \( S \) for node \( v_i \) as its structure coding vector. In our work, we only consider the unweighted network but the \( S \) can be constructed as a weight matrix for a weighted network.

**Text information coding.** According to the text set \( T \) we build a dictionary which contains the words that make up the text content of each node. Then we use the dictionary to construct the node text encoding vector, if the node text contains the word, its corresponding position is 1 otherwise it is 0. For example, a dictionary containing \( m \) words for a network \( G \), the text information encoding vector \( t_i \in \mathbb{R}^m \) of \( v_i \) is composed of 0 or 1, indicating whether there is the word in the dictionary.

### 3.2.2. Neural network structure.

For the design of neural network structure, a multilayer feedforward neural network is used. On the one hand, the universal approximation theorem [14] shows that the feedforward neural network shows a powerful ability in data fitting. On the other hand, the multi-layer feedforward neural network can effectively extract the complex nonlinear features of the complex network [8].

**Input layer.** In order to let the node representation vector contains structure and text information, a very direct method is to embed the structure and text separately, and then join the two parts together as the node representation. In this post-fusion method, the two models trained separately cannot complement each other and restrict the two kinds information. On the contrary, we need only one model and adopt the strategy of fusion in advance, which can realize the continuous interaction of node structure and text information during the training process, and optimize all parameters at the same time. Therefore, the input vector \( x_i \) is combined with \( s_i \) and \( t_i \), expressed as:

\[
    x_i = s_i \oplus t_i. \tag{1}
\]

**Hidden layer.** The input vector \( x_i \) is fed into the multi-layer feedforward neural network. For the size of each hidden layer, we reduced it layer by layer so let the neural network has a tower structure. By using the multi-layer feedforward neural network with this structure, we can obtain node feature representation which are more abstract. We define the hidden layers’ output vectors as \( y^{(1)}, y^{(2)}, ..., y^{(n)} \), and the specific definition as follows:

\[
    y^{(k)} = \phi_k(W^{(k)}y^{(k-1)} + b^{(k)}), k \in \mathbb{Z}^+; \tag{2}
\]

where the \( W \) is the weight matrix of each layer, the \( b \) is the bias vector, \( n \) represents our model has the \( n \) hidden layers; \( \phi_k \) is the nonlinear activation function and we choose the tanh (hyperbolic tangent function) in our model. In this paper, our method obtain the the representation vector of each node by extracting the parameter matrix of the output layer and its preceding layer.

**Output layer.** We use softmax function to convert the \( y^{(n)} \) that is the output vector of the last hidden layer into a probability vector. The probability vector includes the probability that predict node \( v_i \) is connected to all other nodes in \( V \).

### 3.2.3. Loss function.

The purpose of NRL is to effectively retain the original network information, fully mine potential feature representations, and better serve downstream network data mining tasks. In order to obtain a more effective node representation vector, our algorithm should preserve the similarity of the node structure and text at the same time. Therefore, loss functions for structure and text are designed respectively.

**Loss function of structure similarity.** The neighbor nodes of a node are important in inferring the structure similarity relationship between nodes. Therefore, similar to the Skip_gram model [15,16] which use the central word to predict context words, we can maximize the probability of a node predicting its neighbor nodes to achieve similarity preservation of the network structure. The probability \( p(v_j|v_i) \) for node \( v_i \) to predict its neighbor node \( v_j \) is defined as follows using the softmax function:
\[ P(v_j | v_i) = \frac{\exp(u_i^T u_i)}{\sum_{k=1}^{n} \exp(u_k^T u_i)} \]  

Where \( u_i \) represents the representation vector of node \( v_i \) and \( u_i' \) represents the representation vector when it as neighbor nodes. For preserving the network structure similarity, it is necessary to maximize the prediction probability mentioned above, so the structure optimization loss function \( L_{str} \) is defined:

\[
L_{str} = \arg \max \sum_{v_i \in F} \sum_{v_j \in N(v_i)} \log P(v_j | v_i) = -\sum_{v_i \in F} \sum_{v_j \in N(v_i)} \log P(v_j | v_i)
\]

Loss function of text similarity. Two nodes \( v_i \) and \( v_j \) have similar text information. In the representation space, the representation vector of \( v_i \) should be close to \( v_j \). So we use Euclidean distance to measure text similarity and define the measurement function as follows:

\[
d(v_i, v_j) = \|u_i - u_j\|_2 = \sqrt{\sum_{k=1}^{n} |x_{ik} - x_{jk}|^2}
\]

It can be seen from the above formula that the text similarity between nodes is inversely proportional to the distance of the vector representation between nodes: the smaller distance, the higher similarity; the greater distance, the lower similarity. The homogeneity of the network indicates that the nodes with the connection relationship usually have similar text information. Therefore, design a loss function for text similarity preservation based on connection constraints:

\[
L_{txt} = \sum_{v_i \in F} \sum_{v_j \in N(v_i)} d(v_i, v_j)
\]

3.3. Model training
For purpose of retaining the original information of the network and realizing the fusion representation of the structure and text information, the training optimization loss function is defined as:

\[
L = \alpha L_{str} + (1-\alpha) L_{txt}
\]

Where \( \alpha \in (0, 1) \) is a parameter that adjusts the importance of structure information and text information. As \( \alpha \) increases, structure information is more considered. By adjusting the size of \( \alpha \), the importance of two kinds information can be dynamically adjusted to better adapt to different data scenarios. Let the model parameter set be \( \theta = \{W^{(k)}, b^{(k)}; k = 1, 2, ..., n\} \). The process of model training is to minimize the loss function \( L \), and update \( \theta \) through gradient back propagation.

4. Experiment
In this paper, we use two tasks that are node classification and visualization to evaluate the performance of NIT_NRL. We use node classification tasks to evaluate whether node representations can be effectively applied to subsequent network analysis tasks, and visualization to verify whether the node representation has the ability for clustering.

4.1. Experimental setup
In this part, the network data that we used, baseline algorithms and relevant parameter settings are introduced.

4.1.1. Data sets. In this paper, we conduct simulation experiments are on two real network data sets. And then we give the detailed description of these two data sets and the statistical indexes of them are shown in the Table 2. The introduction of them are as follows:
Table 2. Statistics of datasets

| Datasets | Cora | Citeseer |
|----------|------|----------|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
First of all, we changed the input of the SDNE algorithm, adding the text information encoding vector of the node based on the previous input that is the adjacency matrix. Using the input of our proposed algorithm, whether it is on Cora or Citeseer we can find the results of node classification are improved compared to the previous only considering the structure. It verifies the importance of combing the text information for NRL.

Secondly, it is obvious that our algorithm is higher than other comparison algorithms on Citeseer, whether it is Micro-F1 or Macro-F1. Compared with the traditional Node2vec and Line algorithms, we use a multi-layer feedforward neural network which can well capture the complex nonlinear relationships in the network make node classification more effective. But our algorithm has the same or slightly lower F1_score on Cora at different training rates compared with node2vec. Cora's network density is higher than Citeseer, that is, Cora has more edges, more connections between nodes, and richer structure information. Our algorithm performs better than node2vec on the Citeseer, indicating that considering the text information of nodes for sparse networks can make the learned representation vector more effective.

Finally, compared with the SDNE algorithm that also uses deep neural network, because our algorithm considers the text information of the nodes, and separately designs loss functions that retain structure similarity and text similarity, which can make continuous interaction of the information in the training process, so the node representation vector shows better performance in the node classification task.

### Table 3. F1_score of citeseer under different training rates

| Algorithm | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% |
|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Node2Vec  | 0.525 | 0.553 | 0.571 | 0.573 | 0.591 | 0.582 | 0.588 | 0.582 | 0.579 |
| Line      | 0.362 | 0.395 | 0.412 | 0.407 | 0.420 | 0.443 | 0.450 | 0.445 | 0.441 |
| Micro-F1  | 0.441 | 0.499 | 0.541 | 0.558 | 0.569 | 0.591 | 0.594 | 0.609 | 0.635 |
| SDNE      | 0.452 | 0.511 | 0.572 | 0.586 | 0.590 | 0.781 | 0.807 | 0.793 | 0.794 |
| SDNE+Text | 0.570 | 0.574 | 0.602 | 0.619 | 0.622 | 0.627 | 0.634 | 0.621 | 0.596 |
| NTI_NRL   | 0.524 | 0.526 | 0.544 | 0.569 | 0.571 | 0.581 | 0.585 | 0.574 | 0.554 |

### Table 4. F1_score of Citeseer Under Different Training Rates

| Algorithm | 10% | 20% | 30% | 40% | 50% | 60% | 70% | 80% | 90% |
|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Node2Vec  | 0.525 | 0.553 | 0.571 | 0.573 | 0.591 | 0.582 | 0.588 | 0.582 | 0.579 |
| Line      | 0.362 | 0.395 | 0.412 | 0.407 | 0.420 | 0.443 | 0.450 | 0.445 | 0.441 |
| Micro-F1  | 0.441 | 0.499 | 0.541 | 0.558 | 0.569 | 0.591 | 0.594 | 0.609 | 0.635 |
| SDNE      | 0.452 | 0.511 | 0.572 | 0.586 | 0.590 | 0.781 | 0.807 | 0.793 | 0.794 |
| SDNE+Text | 0.570 | 0.574 | 0.602 | 0.619 | 0.622 | 0.627 | 0.634 | 0.621 | 0.596 |
| NTI_NRL   | 0.524 | 0.526 | 0.544 | 0.569 | 0.571 | 0.581 | 0.585 | 0.574 | 0.554 |

4.2.2. Visualization. In order to verify whether the representation vector generated by the algorithm has the ability for clustering, we use the tool t_SNE [18] for visualization shown in Figure 2 and figure 3. Different labels use different color indicating that the node representation obtained by our algorithm has good clustering ability.
5. Conclusion
In this paper, we propose an algorithm, named NTI_NRL, which uses both node text information and structure features to obtain the representation vector for downstream analysis task. Specifically, our method takes advantage of the text information on the basis of network structure information so that the final node low-dimensional vector retains both these two features. Moreover, we use the deep neural network which can obtain the complex nonlinear relationships of the network effectively. The experimental results on two real network data sets show that our algorithm can boost the performance of node representation in node classification and visualization tasks. Especially our method definitely prove that the node text information is significant for network representation learning. On the basis of this, we will consider the connection between words and the semantic relationship of the context lately.

6. References
[1] Cui P, Wang X, Pei J. and Zhu W. 2018 A survey on network embedding vol 31 no 5 (J. IEEE Transactions on Knowledge and Data Engineering) pp 833-852.
[2] Sen P, Namata G, Bilgic M, Getoor, L, Galligher, B. and Eliassi-Rad, T. 2008 Collective classification in network data vol 29 no 3 (J. AI Magazine) pp 93-93.
[3] 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.
[4] Liben-Nowell, D. and Kleinberg J. 2007 The link prediction problem for social networks vol 58 no 7 (J. Journal of the American Society for Information Science and Technology) pp 1019-1031.

[5] Ou M, Cui P, Pei J, Zhang Z. and Zhu W. 2016 Asymmetric transitivity preserving graph embedding (In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining) pp 1105-1114.

[6] Wang X, Cui P, Wang J, Pei J, Zhu W. and Yang S. 2017 Community preserving network embedding vol 17 (In AAAI) pp 203-209.

[7] Herman I, Melançon G. and Marshall M. S. 2000 Graph visualization and navigation in information visualization: A survey vol 6 no 1 (J. IEEE Transactions on Visualization and Computer Graphics) pp 24-43.

[8] 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.

[9] 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) pp 855-864.

[10] 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.

[11] Yang C, Liu Z, Zhao D, Sun M, and Chang E. Y. 2015 Network representation learning with rich text information vol 2015 (In IJCAI) pp 2111-2117.

[12] Sun X, Guo J, Ding X. and Liu T. 2016 A general framework for content-enhanced network representation learning. (J. Arxiv preprint Arxiv) p 1610.

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

[14] Hornik K, Stinchombe M. and White H. 1989 Multilayer feedforward networks are universal approximators vol 2 no 5 (J. the Journal of Machine Learning Research) pp 359-366.

[15] Mikolov T, Sutskever I, Chen K, Corrado G. S. and Dean J. 2013 Distributed representations of words and phrases and their compositionality (In Advances in Neural Information Processing Systems) pp 3111-3119.

[16] Mikolov T, Chen K, Corrado G. and Dean J. 2013 Efficient estimation of word representations in vector space vol 1301 (Arxiv Preprint Arxiv) p 3781.

[17] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, ... and Vanderplas J. 2011 Scikit-learn: Machine learning in Python vol 12 (J. the Journal of Machine Learning Research) pp 2825-2830.

[18] van der Maaten L. and Hinton, G. 2008 Visualizing data using t-SNE vol 9 (J. journal of Machine Learning Research).