Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 8, August 2019, pp. 2237-2248
DOI: 10.23940/ijpe.19.08.p24.22372248

Deep Walk Algorithm based on Improved Random Walk with Equal Probability

Zhonglin Ye[a,b,c,d], Haixing Zhao[a,b,c,d,*], Ke Zhang[a,c,d], Yanlin Yang[a,c,d], and Lei Meng[a,c,d]

[a]*College of Computer, Qinghai Normal University, Xining, 810008, China*
[b]*College of Computer Science, Shaanxi Normal University, Xi'an, 710119, China*
[c]*Tibetan Information Processing and Machine Translation Key Laboratory of Qinghai Province, Xining, 810008, China*
[d]*Key Laboratory of Tibetan Information Procedureing, Ministry of Education, Xining, 810008, China*

Abstract

Most of the existing network representation learning algorithms are mainly based on the DeepWalk algorithm. The main improvement is to modify DeepWalk's three-layer neural network to multi-layer neural network or to introduce the network's external attributes into the DeepWalk algorithm for joint representation learning. In network representation learning, the random walk strategy can be considered as network data preprocessing of network representation learning tasks. Learning the random walk sequences, including most of the network structure features, is very important for the network representation learning algorithm, because the subsequent training procedure of neural networks is to continuously adjust the representation of each node in the network based on the co-occurrence of node pairs. Therefore, the purpose of this paper is to improve the random walk strategy of the DeepWalk algorithm. We propose a novel DeepWalk algorithm based on the improved random walk with equal probability (EPDW). Although the next node in the random walk of the DeepWalk algorithm is chosen with equal probability, one of the neighbouring nodes of the current node is chosen as the next node of the random walk by the pseudo-random number. The improvement of EPDW is to choose the next hop node in the random walk using the roulette method of cumulative probability sum of random walk nodes. Although this method also chooses the next hop node with equal probability, it can choose the next hop random walk node more reasonably and effectively.

Keywords: network representation learning; network representation; network embedding; feature learning

(Submitted on June 11, 2019; Revised on July 5, 2019; Accepted on August 12, 2019)

© 2019 Totem Publisher, Inc. All rights reserved.

1. Introduction

The Word2Vec algorithm [1] is the most classical algorithm in word representation learning. It is mainly used in language modelling [2], in which the word vector is an intermediate result of the language model. Namely, the original idea of Word2Vec algorithm is not to obtain the representation vectors of each word, but to construct a language model by studying how to construct the word vector. Word2Vec greatly shortens the training time of language modeling. It simplifies the neural network model trained by multiple hidden layers to a neural network composed of input layer, single hidden layer, and output layer. However, the performance of Word2Vec in various natural language tasks has not declined, but it has been greatly improved.

Shallow neural network has achieved great success in language models [3-5]. Therefore, the idea of word representation learning is introduced into network representation learning [6] (NRL). Similar to word representation learning, network representation learning [7] compresses the network features, such as high-dimensional structure features, into low-dimensional representation vector space, which makes the words and nodes with strong correlation have closer space distance in the representation vector space. However, there has always been a huge gap between the network and language, and there are little common points between them. At this time, it is a feasible method to transform the network into language to train shallow neural network models. Perozzi et al. proposed the DeepWalk algorithm [8] based on the Word2Vec algorithm. In the Word2Vec algorithm, sentences are ordered sequences of words. If the network learning model is trained

* Corresponding author.
E-mail address: haixing_zhao@163.com

based on the Word2Vec algorithm, the network needs to be transformed into sentences similar to those in language modeling. Therefore, in the DeepWalk algorithm, the random walk sequence of network nodes is regarded as a "sentence". In order to obtain the random walk sequence of network nodes, many methods can be used, but one condition needs to be satisfied: the transformed "sentences" of network should have the same linguistic rules as the sentences in the language model. Perozzi et al. [8] used simple random walk to construct network "sentences". Then, they conducted a comparative study based on YouTube networks and Wikipedia text. They found that the frequency of each node constructed based on network random walk sequences obeys the power law distribution, which also exists in the language model. The frequency of every word in the Wikipedia corpus also obeys the power law distribution. Moreover, the power-law distributions of nodes and words in the two datasets are generally the same, so Perozzi et al. believed that "sentences" of networks can be generated by the random walk strategy. The network representation learning algorithm greatly improves the efficiency and effectiveness of network feature mining. NRL algorithms are also applied to machine learning tasks of various graph structures, such as network node classification [9], network visualization [10], network link prediction [10], and recommendation systems [11]. Based on the DeepWalk algorithm, some new network representation learning algorithms have been proposed, such as, Line [12], HARP [13], NEU [14], WALKLETS [15], TriDNR [16], and node2vec [17]. Other improved algorithms, such as CNRL [18] and MMDW [19], integrate the node's text, community, and label information of the vertex into the network representation vectors. The existing link prediction algorithms [20] are mainly based on the framework of matrix factorization, and the existing data mining methods [21-23] are mainly based on statistical machine learning methods. The performance of link prediction and data mining would be extremely improved if the network representation learning approaches are introduced into these two kinds of tasks.

In network theory, scale-free networks are a special type of complex networks. In the past few years, the study of scale-free characteristics of complex networks has produced many research results [24]. A typical feature of scale-free networks is that most nodes tend to establish links with nodes that have smaller degrees, while a small number of nodes have links with most nodes. Namely, a small number of nodes have many links, while most nodes only have very few links. In addition, a complex network whose degree distribution obeys the power law distribution is called a scale-free network. In real life, there are many scale-free networks, such as the Internet, social networks, and citation networks. At present, there are many research results on scale-free network growth models [25-26]. Among them, the degree-first link algorithm is the most commonly used method to construct scale-free networks. A node tends to build an edge with a node that has a larger degree value, but the establishment of such link is a probability relationship. It also has a chance to build a link with a node that has a smaller degree value. Therefore, in the procedure of next node selection, the roulette method based on the sum of probability accumulation is introduced into scale-free network to realize the procedure of building links between old and new nodes.

In the procedure of complex network growth and building, it is necessary to choose a node from the existing nodes to establish the link relationships. In random walk, it is necessary to choose a node from the neighbouring nodes of the current node as the next hop node of random walk. Therefore, both these two tasks need to choose a node from an existing node. In this section, the roulette method is introduced into the random walk by referring to the rules of complex network growth, so as to choose the next hop node of random walk more reasonably. Based on this, we propose a novel DeepWalk algorithm based on the improved random walk with equal probability, also called the EPDW algorithm. The EPDW algorithm is a simple and excellent network representation learning algorithm. The bottom layer of the EPDW algorithm also uses the neural network model adopted by the DeepWalk algorithm, namely, the shallow neural network composed of an input layer, hidden layer, and output layer is used in the EPDW algorithm. Unlike the DeepWalk algorithm, the EPDW algorithm improves the node sampling method of random walk of the DeepWalk algorithm. The specific methods are as follows: first, after the EPDW algorithm obtains the neighbouring node of the current node, the probability of walking to each neighbouring node is set as 1. Then, these probability values are normalized according to the number of neighbouring nodes. Based on the normalized random walk probability, the cumulative sum of the probability is calculated according to the node number from smallest to largest, and the cumulative sum of the probability of the last node is 1. Finally, the roulette approach is used to choose the next hop node of the random walk from the cumulative sum, until all nodes in the network have completed the random walk procedure.

In summary, we first introduce the concept of network representation learning, the relationships between the DeepWalk algorithm and Word2Vec algorithm, and the random walk hypothesis of the DeepWalk algorithm. Then, we introduce how to improve the random walk sequence of the DeepWalk algorithm and propose the EPDW algorithm in this paper. In the following section, we introduce the model implementation of the DeepWalk algorithm and discuss how the EPDW algorithm improves the random walk procedure in detail. In the experimental sections, we compare and analyze the performance of the EPDW algorithm and some classical network representation learning algorithms in network node classification.

2. Model Framework

2.1. DeepWalk Algorithm

The DeepWalk algorithm is a classical algorithm for network representation learning that is mainly used to mine large-scale network structure features. It is mainly composed of three layers of neural networks: input layer, hidden layer, and output layer. It was proposed based on the Word2Vec algorithm, so the underlying implementation of the DeepWalk algorithm and Word2Vec algorithm is the same. The Word2Vec algorithm is a classical word representation learning algorithm that embeds the context structure relationships between current words and context words into a low-dimensional vector space, so that words with structural associations or similar semantics have a closer distance in the representation vector space. The underlying models of the DeepWalk and Word2Vec algorithms are composed of the CBOW model and Skip-Gram model. In addition, these two algorithms provide two different optimization methods: negative sampling and hierarchical softmax. The CBOW model predicts the current words by the context words of the current words, while the Skip-Gram model predicts the context words by the current words. Therefore, for the representation learning task of training networks or language models, one of these two training models can be used, or one of these two optimization methods can be adopted. Finally, there are four training models that can be selected to NRL tasks.

CBOW is the abbreviation of continuous bag-of-words model, which train the network representation model by predicting the context nodes of the current node. The learning objective is to maximize the logarithmic likelihood function:

$$L(v) = \sum_{v \in C} \sum_{\xi \in \{v\} \cup v \in NEG(v)} log\, p\,(\xi | Context(v)) \tag{1}$$

Where

$$p(\xi | Context(v)) = [\sigma(\boldsymbol{x}_v^T \theta^{\xi})]^{L^v(\xi)} \times [1 - \sigma(\boldsymbol{x}_v^T \theta^{\xi})]^{(1 - L^v(\xi)} \tag{2}$$

In the above equations, $\sigma(x)$ represents the sigmoid function, which has a value range of (0,1) and a parameter domain of $(-\infty, +\infty)$. $\boldsymbol{x}_v$ is the sum of the representation vectors of each node in *Context*(*x*), and θ^{ξ} is the vector to be trained for the current node ξ. *Context*(*x*) represents the context node of the current node *v*. *NEG*(*v*) represents negative sampling of the current node. For a given *Context*(*x*), the node v is a positive sample, the rest of the nodes of the network are negative samples, and $NEG(v) \neq \emptyset$. We use

$$L^v(u) = \begin{cases} 1, & u = v \\ 0, & u \neq v \end{cases} \tag{3}$$

as the sampling result of the node u. In Equation (3), the positive sample label of the node *u* is 1, and the negative sample label of the node *u* is 0.

By combining Equations (1) and (2), we can get

$$L(v) = \sum_{v \in C} \sum_{\xi \in \{v\} \cup v \in NEG(v)} \{L^v(\xi) \times log[\,\sigma(\boldsymbol{x}_v^T \theta^{\xi})] + (1 - L^v(\xi)) \times log[\,1 - \sigma(\boldsymbol{x}_v^T \theta^{\xi})]\} \tag{4}$$

Equations (1) and (4) are two commonly used objective functions of the CBOW model, and this objective function can be optimized using negative sampling. If the CBOW model uses hierarchical softmax to optimize the modeling procedure, it does not apply the above Equations (1) and (4). The EPDW algorithm proposed in this paper also adopts the CBOW model and the negative sampling optimization method to train network representation vectors. Therefore, the CBOW model is introduced in this part. As for negative sampling, its objective is that the selected probability of nodes with higher occurrence frequency is greater than that of nodes with lower occurrence frequency. The negative sampling algorithm firstly divides the interval [0,1] into non-equidistant segments. The size of each segment is the occurrence probability value of each node in the entire random walk sequences. These non-equidistant segments are then connected form head to end in the coordinate with length 1. Finally, a random number is generated to fall on the coordinate, so that the probability of the segment with larger distance is greater than that with smaller distance. This procedure is similar to the rule of priority link in the complex network growth model, namely, it is similar to the roulette method.

2.2. Random Selection of Walk Nodes with Equal Probability

The DeepWalk algorithm was proposed based on the Word2Vec algorithm. In order to utilize the underlying model and optimization framework of the Word2Vec algorithm, DeepWalk needs to transform the network structure into a language structure. Perozzi et al. took the random walk sequences of the network as the "sentence" in the language model through the random walk algorithm. This "sentence" in the network model and the sentence in the language model have statistically similar attribute characteristics. The random walk strategy used in the DeepWalk algorithm is the simplest random node sampling method. In this random walk strategy, the next hop node of the current node is the neighbouring node of the current node, and the selected probabilities of these neighbouring nodes are equal, namely, the next hop node is selected with equal probability. In order to elaborate the random walk procedure of the DeepWalk algorithm, we first give a simple network as shown in Figure 1.

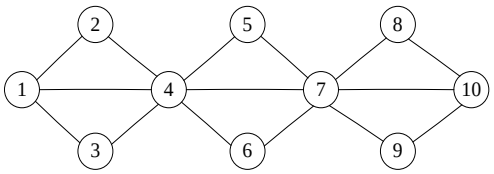


Figure 1. The network example

As shown in Figure 1, the network consists of 10 nodes and12 edges. The DeepWalk algorithm needs to set the random walk number and random walk length of each node. The random walk number refers to the number of random walk paths starting from the source node, which is generally set as 10. The random walk length refers to the path length from the first node to the last node, which is generally set as 40. In order to visualize the procedure in more detail, we set node 1 as the source node based on the network in Figure 1, the random walk number as 3, and the random walk length as 5. Then, the random walk sequences are shown as follows:

As shown in Figure 2, a random walk sequence starts at source node 1. There can exist multiple random walk sequences, but Figure 2 shows only three of them. In Figure 2, the first walk sequence and the second walk sequence do not return to the previous node, but a random walk strategy similar to depth-first search is adopted. In the third sequence, the random walk algorithm starts from source node 1 and then passes through nodes 4, 2, 1, and 3. The length of passing random walk is 5, and the number of random walks is 3. Therefore, when the random walk procedure is completed for all the nodes in Figure 1, the total random walk sequences is 30 and the length of each of them is 3. There are two main methods to generate random walk sequences for all nodes: (1) first, we complete first random walk for all nodes, and then we perform second random walk for all nodes and so on, until the number of random walks reaches the setting value. (2) Random walk is performed on the first node until the number of random walks reaches the setting value. Then, the second node is randomly walk until the number of random walks reaches the setting value.

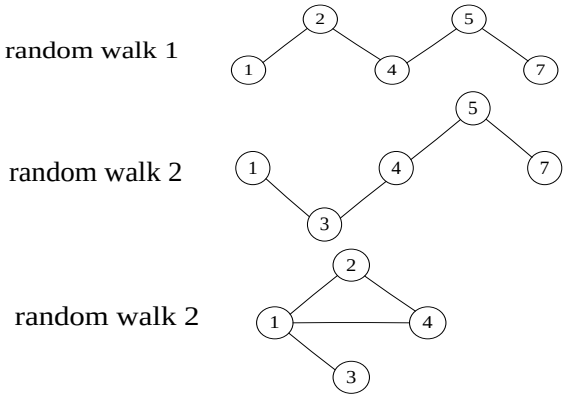


Figure 2. Random walk sequence example

The probability of selecting the next hop node in the DeepWalk algorithm is the same. The node can be either the neighboring node of the current node or the last one that has been traveled in the walk sequence. Therefore, how to choose the next hop node with equal probability from the neighboring nodes of the current node is the research focus of this paper. By analyzing the source code of the DeepWalk algorithm, it is found that DeepWalk uses the "choice" function provided by the random package in Python to randomly choose the next hop node. The result is the random item in list, tuple, or string. If the "choice" function does not specify the selected probability of each item, then the selected probability of each item is

equal. Moreover, the selected number of each node in multiple selection follows a uniform distribution. If the selected probability of each item is specified in the "choice" function, the random item with the specified probability distribution is returned. In addition, through the analysis of the source code of the "random" function in Python, it can be found that the "choice" function first produces a random integer, which is the subscript of the input element of the "choice" function, and then it returns the item from the input list or tuple through the subscript. In summary, DeepWalk returns a neighboring node as the next hop of the random walk through the random integer.

In random walk, although the next hop is randomly chosen, the selected probability of the node with larger degree is greater than that of the node with smaller degree. In the following section of experimental analysis, we conduct a detailed simulation on this phenomenon. The results show that the selected probability of node in random walk is proportional to the degree of node.

In scale-free networks, the roulette method based on the sum of degree accumulations conforms to some characteristics of scale-free networks. For example, the degree value of node obeys the power law distribution, most nodes have fewer links, and fewer nodes have huge links. If the random walk procedure refers to the growth law of the network or introduces the algorithm that follows the growth law of the network into the random walk procedure, the random walk procedure will choose a more reasonable next hop node. Therefore, this paper introduces the roulette method of the sum of degree accumulations into the random walk strategy. In order to compare with the equal probability selection method adopted in the DeepWalk algorithm, this paper ignores the degree value of each node, namely, the probability of walking from the current node to its neighboring node is equal. If there are four neighboring nodes for the current node (ignoring of the probability of returning to the previous node), the roulette method using the cumulative sum of degree values is as follows:

As shown in Figure 3, there are four neighboring nodes: node 1, 2, 3, and 4. If the random walk particle is in the position of node 0, then the random walk algorithm has four next hop nodes. The DeepWalk algorithm uses the random selection strategy to choose one of four neighboring nodes as the next hop node. The EPDW algorithm proposed in this paper avoids randomly selecting one node from the neighboring nodes as the next hop node as DeepWalk. Instead, the roulette method is used to assign an equal probability to each node. In Figure 3(a), the probability of random walk from the current node 0 to nodes 1, 2, 3, and 4 is set to 1. In Figure 3(b), the probability of random walk is normalized, i.e., 0.25. In Figure 3(c), the nodes in the network are sorted according to node number, and then the cumulative probability of each node is calculated according to the node number from smallest to largest. For example, the cumulative probability of node 1 is 0.25, and the cumulative probability of node 2 is the probability of node 2 plus the probability of node 1, namely, the cumulative probability of node 2 is 0.5. Consequently, the cumulative probabilities of nodes 3 and 4 are 0.75 and 1, respectively. Then, the cumulative probability of each node is transformed into a coordinate form as shown in Figure 4 above. The roulette method then randomly produces a random number between 0 and 1; the random number is a pseudo-random. The generated random number matches the coordinate values in Figure 4. If the random number is 0.3, then node 2 is chosen as the next hop node of random walk.

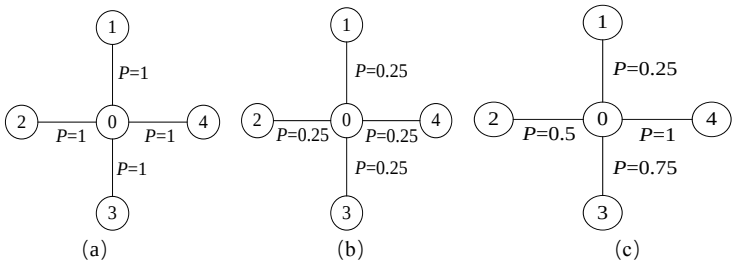


Figure 3. The cumulative sum of neighboring node probabilities

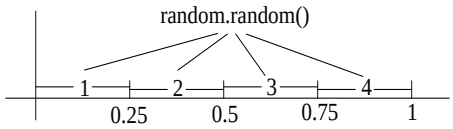


Figure 4. Roulette method of selecting next hop node

The above procedure is the main procedure to choose the next hop node for the EPDW algorithm. In addition, EPDW generates random walk sequences on the whole network. EPDW can get the "sentence" with the approach of random walk on the network. If the "sentence" is input into the underlying model of Word2Vec algorithm, the network representations of each node in the "sentence" can be obtained.

3. Experimental Results

3.1. Datasets

Network representation learning algorithms generally use social network or citation network datasets to verify their performance with network node classification tasks. Therefore, this paper uses three types of citation network datasets: Citeseer, Cora, and DBLP (V4). Some indicators for these three datasets are shown in Table 1 as follows:

Table 1. Dataset descriptions

Datasets	Vertex	Edge	Category	Average degree	Network diameter	Average path length	Density	Average aggregation coefficient
Citeseer	3312	4732	6	2.857	28	9.036	0.001	0.257
Cora	2708	5429	7	4.01	19	6.31	0.001	0.293
DBLP	3119	39516	4	25.339	14	4.199	0.008	0.259

As shown in Table 1 above, Citeseer, Cora, and DBLP have almost the same number of nodes, but the number of edges is different. DBLP has larger edges than Citeseer and Cora. Because the number of nodes is almost the same but the number of edges is quite different, the average degree of network increases with the growth of the edge number. Meanwhile, the average path length decreases with the growth of the edge number. In addition, Table 1 shows that Citeseer and Cora are sparse networks compared with DBLP, and DBLP is a dense network compared with Citeseer and Cora. In conclusion, the Citeseer, Cora, and DBLP datasets used in this paper can simulate different types of networks, which makes it possible to fully verify the performance of various network representation learning algorithms. For the DBLP dataset, we delete all separate nodes, so that the number of nodes in each category remains is about 800.

3.2. Contrast Algorithms

DeepWalk: The DeepWalk algorithm is applied to large-scale network representation learning tasks. This algorithm uses the CBOW and Skip-Gram models to train the relationships between the current nodes and context nodes, and it uses negative sampling and hierarchical softmax to optimize and speed up the training procedure.

Line: The line algorithm is mainly used for large-scale networks to represent learning tasks. Because the DeepWalk algorithm needs to obtain the walk sequence on network nodes, it is time-consuming and not suitable for large-scale networks to represent learning tasks. The line algorithm only calculates the first-order and second-order similarity between network nodes, so it is superior to the DeepWalk algorithm in time complexity.

HARP (DeepWalk): The HARP algorithm uses the network shrinking algorithm to compress the network. The HARP algorithm trains the network representation vectors of different scales after shrinking the network. Network representations of different scales are integrated into a unified network representation vector space. For each layer of network, the existing network representation learning algorithm can be used to obtain the representation vector. For example, the DeepWalk algorithm is used to train the representation vector of each layer of network, and this method is named HARP (DeepWalk).

DeepWalk+NEU: The NEU algorithm is a matrix transformation method that transforms low-order network representation vectors into higher-order network representation vectors. DeepWalk+NEU method transforms the network representations trained by the DeepWalk algorithm into a higher-order network representation. The NEU algorithm avoids the procedure of retraining the higher-order network representation learning model, but it directly transforms the low-order representation vectors into higher-order network representations through matrix factorization.

GraRep (K = 3): The GraRep algorithm explicitly constructs the feature matrix of different orders of the network, and it also encodes the higher-order features of the network into network representation vectors. For the feature matrix of each order, low-dimensional network representation is obtained by using the matrix factorization method, and finally the network representation of different orders is concatenated to obtain a unified network representation.

Node2vec: Node2vec improves the random walk procedure of the DeepWalk algorithm. In the node2vec algorithm, three types of walk probability are set: the probability of returning the previous node, the probability of walking to the node that has an edge with the previous node, and the probability of walking to the node that does not has an edge with the previous node.

3.3. Experimental Settings

In the experiment of network node classification in this paper, we set the length of the network representation vector to 100

for the DeepWalk, DeepWalk+NEU, Line, HARP (DeepWalk), and node 2vec algorithms. The GraRep algorithm sets the order as 3, and the length of the representation vector is set to 100 for each order of network representation. In addition, we set the random walk length of the DeepWalk algorithm and node2vec algorithm to 40. Except for the line algorithm and GraRep algorithm, the CBOW model is used for all these algorithms. Negative sampling is also used to optimize these algorithms, and the negative sampling size is set to 5. Node2vec achieves better network node classification performance when both the breadth-first walk control parameters and depth-first walk parameter are 0.25, but it makes the network obtain the next hop node with equal probability and random strategy. Therefore, the breadth-first walk parameter is set to 0.5, and the depth-first walk parameter is set to 0.25. In order to verify the performance of the network representation learning algorithm under different training set proportions, we set the training set proportions from 10% to 90%, and the interval is 10%. For the proposed EPDW algorithm, we give the classification performances of network nodes with random walk length as 40, 60, and 80. The experiment is repeated ten times, and the average accuracy is taken as the final result.

3.4. Experimental Results and Analysis

Network representation learning mainly uses network node classification tasks to measure performance. The proposed EPDW algorithm uses the Citeseer, Cora, and DBLP datasets to classify network nodes. The node numbers of the three datasets are almost the same, but the numbers of edges gradually increase, which can simulate networks of different densities. In order to compare and analyze the algorithms fairly, we use the CBOW model to train the DeepWalk, HARP (DeepWalk), NEU, and noe2vec algorithms, and we use the same random walk length and number. The EPDW algorithm proposed in this paper mainly improves the random walk procedure of the DeepWalk algorithm. Therefore, this paper chooses some comparative algorithms highly related to the DeepWalk algorithm, but we do not use some network representation learning algorithms based on deep neural network and network representation learning algorithms with external data. The performance of the network node classification experiments is shown in Tables 2-4 as follows:

Table 2. The network node classification performance comparison on the Citeseer dataset (%)

Algorithm name	10%	20%	30%	40%	50%	60%	70%	80%	90%	Average
DeepWalk	47.55	50.24	51.89	52.32	53.66	53.16	53.75	53.91	54.62	52.34
Line	41.22	44.61	47.88	49.15	52.18	53.49	53.87	53.25	53.91	49.51
HARP(DeepWalk)	48.92	50.28	50.84	50.67	51.31	51.27	50.31	51.75	52.99	50.93
DeepWalk+NEU	48.49	51.23	52.48	53.90	53.54	54.65	54.57	54.43	55.86	53.24
GraRep (K = 3)	45.07	50.95	53.40	54.21	54.87	55.75	55.54	55.15	54.22	53.24
node2vec	50.83	52.58	54.28	54.45	55.71	56.18	55.58	56.22	56.62	54.72
EPDW (l = 40)	51.92	53.84	55.16	55.81	56.86	57.09	57.52	57.90	58.33	56.05
EPDW (l = 60)	51.26	53.40	54.93	55.76	56.59	57.09	57.06	57.22	58.56	55.76
EPDW (l = 80)	50.53	53.57	54.64	55.08	55.66	56.20	56.34	57.15	57.33	55.16

Table 3. The network node classification performance comparison on the Cora dataset (%)

Algorithm name	10%	20%	30%	40%	50%	60%	70%	80%	90%	Average
DeepWalk	67.60	72.09	74.47	75.07	76.68	76.74	77.44	78.08	77.70	75.10
Line	64.25	68.38	70.11	71.34	73.26	75.81	75.62	77.73	79.51	68.75
HARP(DeepWalk)	65.58	68.53	70.79	70.99	70.85	70.82	71.21	72.81	72.92	70.50
DeepWalk+NEU	69.29	74.74	76.08	77.34	77.76	78.59	78.83	79.37	79.07	76.79
GraRep (K = 3)	72.60	77.34	78.34	79.39	79.43	80.30	80.32	80.68	79.88	78.70
node2vec	69.31	73.24	74.13	75.60	76.13	76.58	76.45	77.45	77.44	75.15
EPDW (l = 40)	72.50	75.86	76.84	77.64	77.98	78.65	78.94	79.23	79.65	77.48
EPDW (l = 60)	72.35	75.40	76.46	77.52	77.42	77.96	77.96	78.13	78.78	76.89
EPDW (l = 80)	71.85	74.89	76.07	76.72	77.08	77.27	78.01	78.43	78.25	76.51

Table 4. The network node classification performance comparison on the DBLP dataset (%)

Algorithm name	10%	20%	30%	40%	50%	60%	70%	80%	90%	Average
DeepWalk	76.73	79.47	80.78	81.24	82.11	81.60	82.60	83.24	82.60	81.15
Line	73.28	75.15	76.93	77.42	78.06	78.65	78.92	80.13	80.52	77.67
HARP(DeepWalk)	78.72	80.08	80.82	81.06	80.78	81.25	81.11	81.00	81.80	80.74
DeepWalk+NEU	80.86	81.64	82.10	83.84	83.89	83.79	83.89	84.45	84.02	83.16
GraRep (K =3)	81.59	83.13	84.33	84.07	84.03	84.42	85.24	85.51	85.05	84.15
node2vec	83.17	83.10	83.32	83.62	84.84	84.91	84.29	84.82	84.82	84.10
EPDW (l = 40)	84.06	84.51	84.82	85.31	85.04	85.32	85.60	85.91	86.20	85.20
EPDW (l = 60)	84.50	85.32	85.74	85.99	86.12	86.09	85.91	86.18	86.44	85.81
EPDW (l = 80)	84.05	85.04	85.44	85.66	85.69	85.97	86.16	86.55	86.54	85.68

Based on the data in Tables 2-4, we have the following observations:

(1) On the Citeseer, Cora, and DBLP datasets, the proposed EPDW algorithm achieves the best network node classification performance. Compared with the DeepWalk algorithm, the proposed EPDW algorithm achieves the highest performance improvement rates of 7.08%, 3.16%, and 5.74% on the three datasets and the lowest performance improvement rates of 5.38%, 1.88%, and 4.99%. This fully demonstrates that the improvement of this paper is effective in selecting the next hop node with moderate probability of random walk, and the roulette method using the sum of the cumulative probability can more effectively choose the next hop node of random walk procedure.

(2) In EPDW, we set the random walk lengths as 40, 60, and 80. The experimental results show that the classification accuracy of network nodes is better when the random walk length is small in the sparse networks, such as Citeseer and Cora, while the classification accuracy of network nodes is worse when the random walk length is larger. The network diameters of the Citeseer and Cora datasets are 28 and 19, respectively, so the longer the random walk length, the greater the probability of adding other types of nodes into a random walk sequence. If all the nodes in the random walk sequence belong to the same category, the classification accuracy of network nodes is the best. In the DBLP dataset, the network diameter is 14 and the average degree of the network is 25.339. Even if the random walk length is set to 80, the random walk particles still sample the next hop node near the current node with a higher probability, and the probability of walking to other types of nodes is lower. Therefore, in sparse networks, the length of random walk should be set relatively shorter, while in dense networks, the length of random walk should be set relatively longer. It should be noted that there is no fixed optimal length of random walk sequence in any network. Only experimental simulation can find the optimal length of random walk for the network.

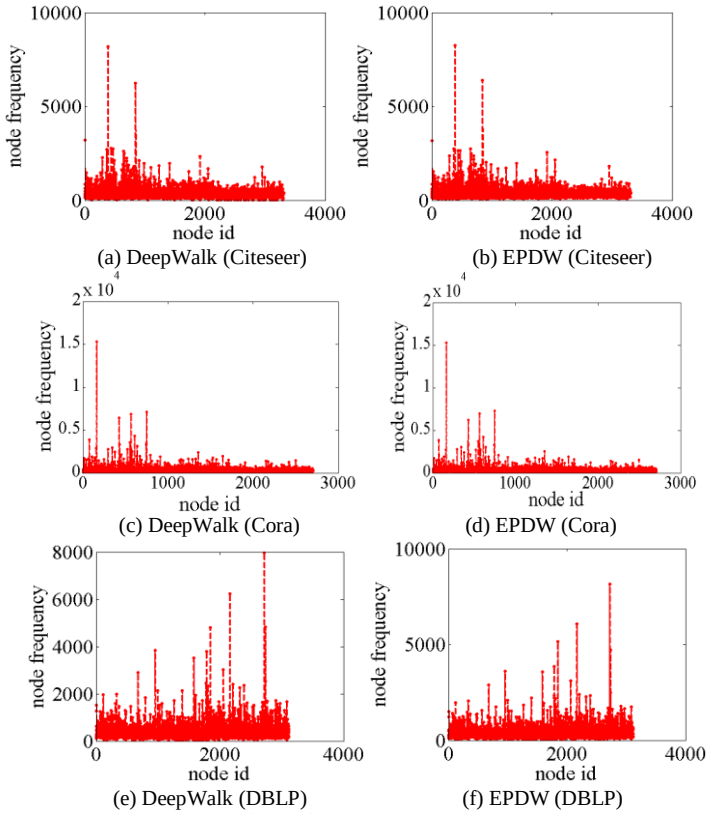
(3) The noe2vec algorithm chooses the next node of the walking particle by setting three different kinds of random walk probabilities: the probability of returning the previous node, the probability of walking to the node that has an edge with the previous node, and the probability of walking to the node that does not has an edge with the previous node. For the latter two probabilities, there is at least one optional walk path under each probability. By setting the probabilities of the first and the third kind of random walk, the random walk particles can travel to the next hop node with breadth-first or depth-first walk. Therefore, the node2vec algorithm is similar to the EPDW algorithm proposed in this paper. It improves the random walk procedure of the DeepWalk algorithm, but its performance is inferior to that of the EPDW algorithm. The main reason is that it is very difficult to find a moderate parameter between breadth-first walk and depth-first walk. Many simulation experiments need to be conducted in order to adjust these two parameters, and different parameters need to be set on different datasets. The EPDW algorithm proposed in this paper adopts the strategy of equal probability random selection of nodes, which can effectively avoid this challenge.

(4) The line algorithm is mainly proposed to solve large-scale network representation learning tasks. Therefore, this algorithm mainly trains network representation learning model through first-order similarity and second-order similarity of nodes. Therefore, the line algorithm improves the network representation learning speed by sacrificing the classification accuracy. Higher-order feature integration algorithms based on network structure (high-order approximate network representation learning) embed higher-order features into network representation vectors through different ways. For example, the HARP algorithm obtains different order network structures by shrinking the network continuously. GraRep explicitly constructs different order network structure feature matrices based on the adjacency matrix. The NEU algorithm transforms the low-order network representations through matrix factorization. These high-order network representation learning algorithms are all dedicated to fully mining the structural features of the network, so as to improve the performance of network representation learning. However, in this experiment, the performance of these high-order network representation learning algorithms is inferior to that of the proposed EPDW algorithm. The main reason is that it is difficult to determine how to give accurate weights to different order features and how to embed different order network structure features into a unified network representation vector space. Sometimes, simple representation concatenation methods are not superior to the network representation learning algorithm with appropriate random walk strategy.

3.5. Vertex Frequency Analysis

The EPDW algorithm proposed mainly improves the random walk procedure of the DeepWalk algorithm, which also uses the equal probability walk method of nodes. For the neighbouring nodes of the current node, although the selection is equal probability, the EPDW algorithm adopts the same probability random selection method that is different from DeepWalk. EPDW introduces the roulette method. Although the DeepWalk algorithm uses the same probability random method to choose the next hop node, it is very different from the computer implementation of EPDW. Therefore, the random walk sequence obtained by the two methods on the whole network is also different. In this experiment, we count the number of appearances of each node in all random walk sequences, and then we visualize the number of appearances of each node in the order of node number from smallest to largest on the Citeseer, Cora, and DBLP datasets. The specific results are shown in Figure 5.

As shown in Figure 5, the abscissa of the graph is the ID number of the node, which is sorted from smallest to largest, and the ordinate is the number of appearances of each node in the random walk sequence. Figures 5(a), 5(c), and 5(e) show the number of occurrences of each node in the walk sequence generated by the DeepWalk algorithm. Figures 5(b), 5(d), and 5(f) are the number of occurrences of each node in the walk sequence generated by the EPDW algorithm. Through horizontal comparison, it is found that the number of occurrences of each node in the walk sequence has a similar distribution in the random walk sequence generated by the DeepWalk algorithm and EPDW algorithm. This phenomenon shows that the probability of selecting the next node in a random walk with equal probability is approximate.



(a) DeepWalk (Citeseer) (b) EPDW (Citeseer)

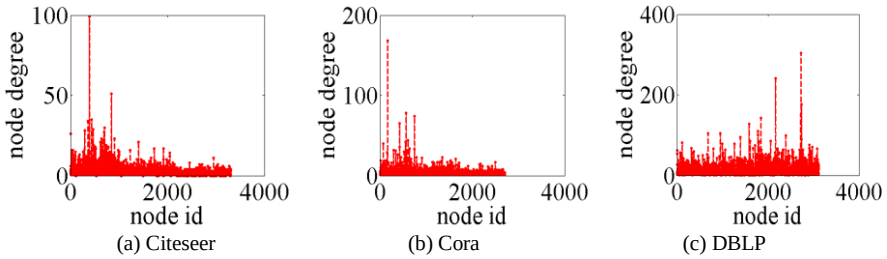
(c) DeepWalk (Cora) (d) EPDW (Cora)

(e) DeepWalk (DBLP) (f) EPDW (DBLP)

Figure 5. Statistics of node frequency in random walk sequences

In order to verify whether the selected probability of nodes in the walk sequence is related to the degree of nodes, we also count the degree value of each node. In Figure 6, the degree values of nodes in the Citeseer, Cora, and DBLP datasets are shown from smallest to largest according to their ID number value. The specific results are shown in Figure 6.

As shown in Figure 6, we calculate the degree of each node on each dataset. Among them, the abscissa is the same as the abscissa in Figure 5, namely, the number ID of each node, which is arranged from smallest to largest. By comparing Figure 5 with Figure 6, it can be found that the degree of nodes and the number of nodes selected in random walk show similar distributions. This phenomenon shows that greater the degree of the node in the network, the greater the probability that the node will be selected as the next hop node in the random walk. At the same time, it also shows that even if the equal probability random walk strategy is adopted in random walk, the nodes with large degrees still have certain advantages.



(a) Citeseer (b) Cora (c) DBLP

Figure 6. Node degree visualization

4. Visualization

The network representation learning algorithm can also verify its performance by visualizing tasks. In the visualization experiment, the learning performance can be judged by observing whether the boundary between different kinds of nodes is obvious. Therefore, in this experiment, we visualize the learnt network representation generated by the DeepWalk and EPDW algorithms on the Citeseer, Cora, and DBLP datasets. In order to show different kinds of nodes more intuitively and clearly, we use different colors to represent different kinds of nodes. In addition, we randomly choose four categories from all categories for visualization experiments and randomly extract 200 nodes from each category. Finally, the network representations of the 800 nodes are projected to 2D visualization space using the t-SNE algorithm. The specific results are shown in Figure 7.

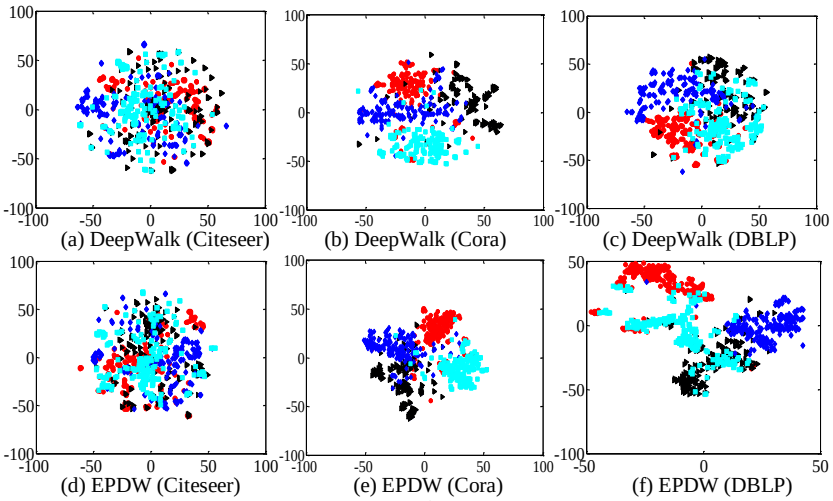


Figure 7. Visualization of network partial nodes

As shown in Figure 7, the DeepWalk algorithm has poor visualization results on Citeseer, and it is difficult to observe clearer boundaries. In the visualization of the DeepWalk algorithm on the Cora and DBLP datasets, there are obvious boundaries between nodes in the four categories. The visualization result of the EPDW algorithm on the Cteseer dataset is also poor, but the red, blue, and black nodes generated by the EPDW algorithm have initially shown more obvious clustering phenomenon compared with the DeepWalk algorithm on the Citeseer dataset. The EPDW algorithm shows better clustering ability and clear boundaries on the Cora dataset, the boundaries between different classes of nodes are clear, and the cohesion ability between the same classes of nodes is also stronger. On the DBLP dataset, the EPDW algorithm also shows similar phenomena with the Cora dataset. In summary, the network representation vectors generated by the EPDW proposed in this paper have stronger discriminant ability, and the algorithm can demonstrate better performance in network visualization tasks.

5. Conclusions

Network representation learning is a method of embedding characteristic information, such as network structure, into low-dimensional vector representation space. Most of the network representation learning algorithms based on shallow neural networks obtain context node pairs through the random walk strategy, and then they input the context node pairs into the neural network for training. In the random strategy, random walk randomly selects the next hop node from the neighbouring nodes of the current node. The EPDW algorithm proposed in this paper aims to improve the random walk strategy adopted by the DeepWalk algorithm. The improvement of the EPDW algorithm is to choose the next hop of random walk from the current node's neighbouring nodes with equal probability by using the roulette method of cumulative probability sum. Although the DeepWalk algorithm also chooses the next hop of random walk from the current node's neighbouring nodes with equal probability, the EPDW algorithm's equal probability node selection method demonstrates good performance in network node classification tasks. Although EPDW adopts the equal probability node selection method, its network node classification performance is also better than that of node2vec and other NRL algorithms, and it is also better than some higher-order network representation learning algorithms. The experimental results show that network node classification performance will inevitably be greatly improved as long as the appropriate random sequence is adopted in the NLR procedure.

References

1. T. Mikolov, I. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," in *Proceedings of Twenty-Seventh Conference on Neural Information Procedureing System*, pp. 3111-3119, 2013
2. Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A Neural Probabilistic Language Model," *The Journal of Machine Learning Research*, No. 3, pp. 1137-1155, 2003
3. S. Lai, K. Liu, L. Xu, and J. Zhao, "How to Generate a Good Word Embedding?" *Intelligent Systems IEEE*, Vol. III, No. 2, pp. 1, 2015
4. R. Collobert and J. Weston, "A Unified Architecture for Natural Language Procedureing: Deep Neural Networks with Multitask Learning," in *Proceedings of International Conference on Machine Learning*, pp. 160-167, 2008
5. G. E. Hinton, "Learning Distributed Representations of Concepts," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, pp. 1-12, 1986
6. D. Zhang, Y. Jie, X. Zhu, and C. Zhang, "Network Representation Learning: A Survey," *IEEE Transactions on Big Data*, No. 99, pp. 1-1, 2017
7. C. C. Tu, W. C. Zhang, Z. Y. Liu, and M. S. Sun, "Max-Margin Deepwalk: Discriminative Learning of Network Representation," in *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 3889-3895, 2016
8. B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online Learning of Social Representations," in *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701-710, 2014
9. S. T. Tu, C. Yang, Z. Y. Liu, and M. S. Sun, "Summaries of Network Representation Learning," *SCIENTIA SINICA*, No. 8, pp. 32-48, 2017
10. "Predictive Network Representation Learning for Link Prediction," (http://101.96.10.64/www4.comp.polyu.edu.hk/~csztwang/ paper/ pnrl.pdf, last accessed on November 17, 2018)
11. W. X. Zhao, J. Huang, and J. R. Wen, "Learning Distributed Representations for Recommender Systems with a Network Embedding Approach," in *Proceedings of Asia Information Retrieval Symposium*, pp. 224-236, 2016
12. J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-Scale Information Network Embedding," in *Proceedings of International Conference on World Wide Web*, pp. 1067-1077, 2015
13. H. Chen, B. Perozzi, Y. Hu, and S. Skiena, "HARP: Hierarchical Representation Learning for Networks," in *Proceedings of National Conference on Artificial Intelligence*, pp. 2127-2134, 2018
14. C. Yang, M. Sun, Z. Liu, and C. Tu, "Fast Network Embedding Enhancement Via High Order Proximity Approximation," in *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 3894-3900, 2017
15. B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena, "Don't Walk, Skip! Online Learning of Multi-Scale Network Embeddings," *Advances in Social Networks Analysis and Mining*, pp. 258-265, 2017
16. S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-Party Deep Network Representation," in *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 1895-1901, 2016
17. A. Grover and J. Leskovec, "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
18. C. C. Tu, H. Wang, X. K. Zeng, Z. Y. Liu, and M. S. Sun, "Community-Enhanced Network Representation Learning for Network Analysis," ArXiv: 1611. 06645, 2016
19. C. C. Tu, W. C. Zhang, Z. Y. Liu, and M. S. Sun, "Max-Margin Deepwalk: Discriminative Learning of Network Representation," in *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 3889-3895, 2016
20. M. Li, Z. B. Niu, X. P. Chen, P. Zhong, F. X. Wu, and Y. Pan, "A Reliable Neighbor-based Method for Identifying Essential Proteins by Integrating Gene Expressions, Orthology, and Subcellular Localization Information," *Tsinghua Science and Technology*, Vol. 12, No. 6, pp. 668-677, 2016
21. Q. L. Han, S. Liang, and H. L. Zhang, "Mobile Cloud Sensing, Big Data, and 5G Networks Make an Intelligent and Smart World," *IEEE Network*, Vol. 29, No. 2, pp. 40-45, 2015
22. L. L. Gao, H. W. Pan, X. Q. Xie, Z. Q. Zhang, Q. Li, and Q. L. Han, "Graph Modeling and Mining Methods for Brain Images," *Multimedia Tools and Applications*, Vol. 75, No. 15, pp. 9333-9369, 2016
23. Q. Q. Shi, H. Z. Wang, D. Li, X. F. Shi, C. Ye, and H. Gao, "Maximal Influence Spread for Social Network based on MapReduce," in *Proceedings of International Conference of Young Computer Scientists, Engineers and Educators*, pp. 128-136, 2015
24. F. Hu, F. X. Li, and H. X. Zhao, "The Research on Scale-Free Characteristics of Hypernetwork," *Scientia Sinica Physica, Mechanica & Astronomica*, Vol. 47, No. 6, pp. 17-22, 2017
25. F. Hu, H. X. Zhao, J. B. He, F. X. Li, S. L. Li, and Z. K. Zhang, "An Evolving Model for Hypergraph-Structure-based Scientific Collaboration Networks," *Acta Physica Sinica*, Vol. 62, No. 19, pp. 101-107, 2013
26. F. Hu, H. X. Zhao, and X. J. Ma, "An Evolving Hypernetwork Model and its Properties," *Scientia Sinica Physica, Mechanica & Astronomica*, Vol. 43, No. 1, pp. 16-22, 2013

Zhonglin Ye is a graduate student in the School of Computer Science in Shaanxi Normal University. His research interests include data mining, knowledge discovery, and representation learning and understanding.

Haixing Zhao is a professor and Ph.D. supervisor. He received his Doctor of Engineering Degree from the School of

Computer Science at Northwestern Polytechnical University in 2004. He also received his Doctor of Science Degree from Twente University in Holland. His research interests include complex networks, semantic networks, machine translation, hypergraph theory and databases, and network reliability.

Ke Zhang is a Ph.D. candidate. His research interests include complex networks.

Yanlin Yang is an M.S. candidate. Her research interests include complex networks and link prediction.

Lei Meng is an M.S. candidate. His research interests include data mining and hyper networks.