Computer Life (CPL) ISSN: 1819-4818
DELIVERING QUALITY SCIENCE TO THE WORLD

Web page classification based on graph neural network

Huawei Mei[1], Shuai Zhang[1,] * and Yonggang Liu[2]

[1]Computer Department, School of North China Electric Power University, China

[2]Shijiazhuang water supply Co., Ltd, China

Abstract: Graph neural network mainly includes graph embedding (based on random walk) and graph CNN (based on neighbor aggregation). This paper improves the original deepwalk algorithm and mainly studies the problem of web page classification. From the idea of depth first search to the idea of breadth first search, the accuracy has been improved.
Keywords: Classification, graph neural network, deepwalk, graph Embedding.

---

1. INTRODUCTION

In NLP tasks, word2vec is a commonly used word embedding method. Word2vec describes the co-occurrence relationship between words through sentence sequences in the corpus, and then learns the vector representation of words. The idea of graph embedding is similar to word2vec.

It uses the co-occurrence relationship between nodes in the graph to learn the vector representation of nodes. Graph embedding technology expresses the nodes in the graph in the form of low-dimensional dense vectors. It is required that similar nodes in the original graph (different methods have different definitions of similarity) are also close in the low-dimensional expression space. The obtained expression vector can be used for downstream tasks, such as node classification, link prediction, visualization or reconstruction of the original graph.

Use the co-occurrence relationship between nodes in the graph to learn the vector representation of nodes. Then the key problem is how to describe the co-occurrence relationship between nodes. The method given by deepwalk is to use random walk to sample nodes in the graph.

Randomwalk is a depth first traversal algorithm that can repeatedly access visited nodes. Given the current access starting node, randomly sample the node from its neighbors as the next access node, and repeat this process until the length of the access sequence meets the preset conditions.

After obtaining a sufficient number of node access sequences, use skip gram model for vector learning.

However, the processing speed and processing rate of deep walk are not ideal.

2. EXPERIMENTAL CONTENT

2.1 Overall thinking.

Here, the adjacency matrix of the graph is used for input. For the ith vertex, there is Xi = Si, and each

Si contains the neighbor structure information of vertex I. therefore, such a reconstruction process can make the vertices with similar structure have similar embedding representation vectors.

One problem here is that due to the sparsity of the graph, the non-zero elements in the adjacency matrix s are far less than the zero elements, so as long as all outputs 0 can achieve a good effect for the neural network.

2.2 Definitions.

The definition of similarity between vertices in a graph is different from that of deepwalk. The first-order similarity is used to describe the local similarity between pairs of vertices in a graph. The formal description is that if there is a direct edge between u and V, the edge weight wuV is the similarity between two vertices. If there is no direct edge, Then the first-order similarity is 0, and the second-order similarity is used to describe the case where there are no direct edges but have the same neighbor nodes.

When using the gradient descent method to optimize the parameters, w will be directly multiplied by the gradient. If the edge weight variance in the graph is large, it is difficult to choose an appropriate learning rate. If a larger learning rate is used, the gradient explosion may be caused for a larger edge weight, and a smaller learning rate for a smaller edge weight will lead to a too small gradient. For the above problems, if all edge weights are the same, it will be easy to choose an appropriate learning rate. Here, a method is adopted to split the weighted edge into equal weight edges. If an edge with weight of W is split into w edges with weight of 1. This can solve the problem of learning rate selection, but due to the increase of the number of edges, the demand for storage will also increase.

2.3 Formulas.

For the first-order similarity, the loss function is defined as follows:

$$L_{1st} = \sum_{i,j=1}^{n} s_{i,j} \left\| y_i^{(K)} - y_j^{(K)} \right\|_2^2 = \sum_{i,j=1}^{n} s_{i,j} \left\| y_i - y_j \right\|_2^2$$

The loss function can make the embedding vector corresponding to two adjacent vertices in the graph approach in the hidden space.

L_{1st} can also be expressed as

$$L_{1st} = \sum_{i,j=1}^{n} \left\| y_i - y_j \right\|_2^2 = 2\,\mathrm{tr}\left(Y^T L Y\right)$$

Where L is the Laplace matrix corresponding to the graph, L = D-S, D is the degree matrix of the vertices in the graph, and S is the adjacency matrix.

Loss function definition: L_mix = L_2nd + alpha * L_1st + beta * L_reg

L_2nd, is the loss function corresponding to the second-order similarity, and the parameter， beta , controls the penalty term coefficient of non-zero elements. y_true and y_pred is the input adjacency matrix and the network reconstructed adjacency matrix respectively.

L_1st is the loss function corresponding to the first-order similarity, and the parameter alpha controls its proportion in the overall loss function.

2.4 Model definition.

create_model is created by the model. L1 and L2 are the regularization term coefficients of the model respectively. The input a of the model is the adjacency matrix and l is the Laplace matrix. Output a_ Is the reconstructed adjacency matrix, and Y is the embedding vector of the vertex.

```
def create_model(node_size, hidden_size, l1, l2):
    A, L = Input(shape=(node_size,)), Input(shape=(None,))
    fc = A
    for i in range(len(hidden_size)):
        if i == len(hidden_size) - 1:
            fc = Dense(hidden_size[i], activation='relu',
                       kernel_regularizer=l1_l2(l1, l2), name='1st')(fc)
        else:
            fc = Dense(hidden_size[i], activation='relu',
                       kernel_regularizer=l1_l2(l1, l2))(fc)
    Y = fc
    for i in reversed(range(len(hidden_size) - 1)):
        fc = Dense(hidden_size[i], activation='relu',
                   kernel_regularizer=l1_l2(l1, l2))(fc)
    A_ = Dense(node_size, 'relu', name='2nd')(fc)
    model = Model(inputs=[A, L], outputs=[A_, Y])
    emb = Model(inputs=A, outputs=Y)
    return model, emb
```

The key codes are as above.

2.5 Data set.

the data set contains the link relationship between 2405 web pages and 17981 web pages, as well as the category of each web page, and carries out node classification tasks and visualization tasks.

3. EXPERIMENTAL RESULT

3.1 Vertex classification task results.

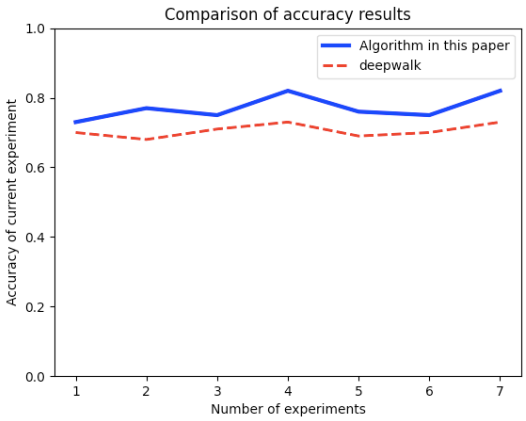

Figure1 Comparsion of accuracy results

As can be seen from the above figure1, the accuracy of the seven experiments is that the improved algorithm is slightly higher than the original algorithm.

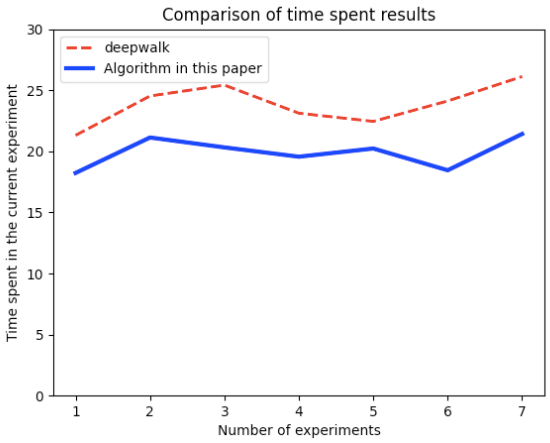

Figure2 Comparsion of time spent results

It can be seen from the figure2 above that in addition to the accuracy, the time spent in each experiment of the algorithm in this paper is less than that of the original algorithm.

3.2 Vertex vector visualization.

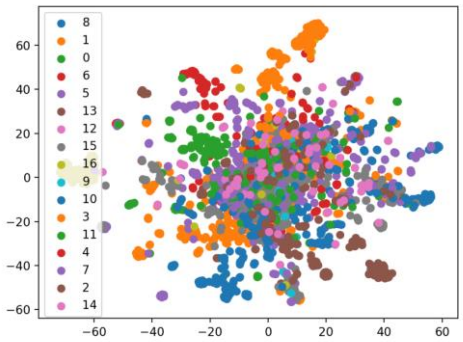

Figure3 Results of deepwalk

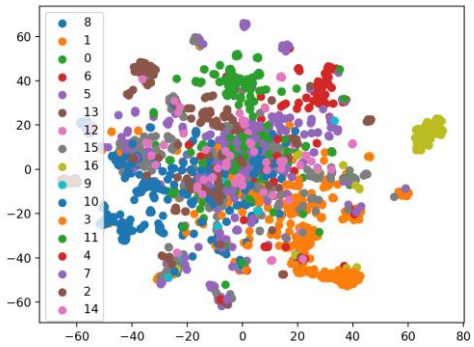

Figure 4 Results of the improved algorithm

As shown in the figure 3&4 above, the visualization results obtained from the original deepwalk are on the left and the visualization results obtained in this paper are on the right.

From the visual results of the obtained vertex vectors, compared with the vectors produced by deepwalk, the vectors obtained in this paper are indeed more able to gather similar vertices and distinguish different types of vertices.

4. CONCLUSION

This algorithm changes the idea of depth first search to the idea of breadth first search, and optimizes the first-order and second-order similarity at the same time. The learned vector representation can retain the local and global structure, and is robust to sparse networks. Compared with the original algorithm, it not only reduces the time cost, but also improves the accuracy in web page classification. In addition to web page classification, in the real world, the vertices in the graph also contain several attribute information, such as user portrait information in social networks, text information in citation networks, etc. for this kind of information, the method based on graph embedding usually splices the attribute features into the vertex vector for subsequent tasks.

REFERENCES

[1] Qi Ke, Li Wenkang Fault diagnosis method and medium of multi-source time series data based on graph neural network:, cn112783940a [P] two thousand and twenty-one.
[2] Tao Peng, fan Ningjun, Chen Derong Research on neural network classification method of hyperspectral images based on regional characteristic spectrum [J] Electronic devices, 2008, 31 .
[3] Wu Dongdong Research on multi label image classification based on graph neural network.
[4] Zhou Yuqing, Zhi Gaofeng, sun Weifang A tool state image classification method based on edge marker map neural network: cn112017204a [P] two thousand and twenty.
[5] Ning Xin, Dong Xiaoli, Tian Weijuan, et al Construction method and device of simplex neural network for image classification: cn112508183a [P] two thousand and twenty-one.
[6] Wang Li Peng Problem list classification method, equipment and storage medium based on graph neural network:, cn111694957a [P] two thousand and twenty.
[7] Yang Hongfei, Jin Xia, Han Ruifeng A table structure recognition method based on graph neural network: cn111597943a [P] two thousand and twenty.
[8] Zhang Xiaodan, Liang Bing An improved graph neural network big data classification method for scientific and technological documents:, cn112231476a [P] two thousand and twenty-one.
[9] Li G , Luo J , Xiao Q , et al. Predicting MicroRNA-Disease Associations using Network Topological Similarity based on DeepWalk[J]. IEEE Access, 2017, PP:1-1.