A graph neural network fused with multi-head attention for text classification

Bing Ai¹, Yibing Wang¹, Liang Ji¹, Jia Yi¹, Ting Wang¹, Wentao Liu¹ and Hui Zhou²*

¹Big Data Center of State Grid Corporation of China, Beijing, 100052, China
²Department of Computer, North China Electric Power University, Baoding, Hebei, 071000, China
*email: 18730216812@163.com

Abstract. Graph neural network (GNN) has done a good job of processing intricate architecture and fusion of global messages, research has explored GNN technology for text classification. However, the model that fixed the entire corpus as a graph in the past faced many problems such as high memory consumption and the inability to modify the construction of the graph. We propose an improved model based on GNN to solve these problems. The model no longer fixes the entire corpus as a graph but constructs different graphs for each text. This method reduces memory consumption, but still retains global information. We conduct experiments on the R8, R52, and 20newsgroups data sets, and use accuracy as the experimental standard. Experiments show that even if it consumes less memory, our model accomplish higher than existing models on multiple text classification data sets.

1. Introduction

Text classification is a basic problem in natural language processing (NLP), which can be effectively applied to public opinion analysis, news filtering, and spam detection. The main process of text classification is text representation learning.

In recent years, deep learning has developed rapidly. Various neural network models have been applied to text classification problems, such as convolutional neural networks (CNN) [1] and recurrent neural networks (RNN) [2] and other neural networks. In these years, a new type of neural network has been developed rapidly, this type of network is called graph neural network (GNN) [3]. GNN was first proposed in [4] and has applied in many fields in natural language processing, including text classification [5], sequence labeling, neural Network machine translation and relational reasoning. Defferrard et al. [5] are the first to use graph convolutional neural networks (GCN) in text classification tasks and are superior to traditional CNN models. They build graphs through text nodes and weighted edges, and their experiments show that the model is superior to the most advanced text classification methods at the time. In addition, Yao et al. [6] improved the model.

2. Related work

We introduce in detail the work done by predecessors in the field of text classification and GNN in this section.
2.1. Text classification
Text classification is the most common and important task type in the field of NLP applications. This work has a major impact on people's lives, such as public opinion analysis, sentiment analysis, news classification, etc. Previous machine learning text classification methods mainly focused on machine learning algorithms and text information feature extraction, such as bag-of-words[7], n-gram and Topic Model. With the development of neural network technology, many deep learning models are applied to text classification. Kim[1] ; Liu used RNN and CNN for text classification and got better results than traditional machine learning models. On the other hand, many neural network models that based on graph have been developing fast. Yao [6] proposed Text-GCN and acquired the most advanced outcomes on many major data sets. Since Text-GCN builds a large image based on the entire corpus, it has disadvantages such as large memory consumption. Our model has achieved a good effect on the problems in Text-GCN.

2.2. Graph Neural Network
The graph neural network is a graph-based neural network. Compared with traditional neural networks, graph-based neural networks can model non-Euclidean data. Many data, such as knowledge graphs, social networks and many other research fields, use trees or graphs. Form of data. Therefore, People have a lot of interest in Graph Neural Networks (GNN) in these years. GCN has achieved rapid development.

3. Method
In this section, we will show our approach in detail. The neural network model in this figure consists of two key parts: the text information structure based on graphics and the readout function based on the multi-head attention mechanism. The architecture is shown in Figure 1. In this section, we will introduce in detail how to achieve these two goals and how they work.

3.1. Create a text information graph
We show all the words that appear in the text as nodes in the graph, and each edge starts from a word in the text and ends with a word in the adjacent window. The words in the text can be expressed as $T = \{r_1, \ldots, r_l, \ldots, r_t\}$, where $r_i$ represents the i-th word. $r_i$ is a vector initialized by d-dimensional word embedding. And the vector will be trained and updated in the model.

Specifically, the graphic definition of the text is:

$$N = \{r_i|i \in [1, l]\}$$  \hspace{1cm} (1)

$$E = \{e_{ij}|i \in [1, l]; j[i - p, i + p]\}$$  \hspace{1cm} (2)

Where $N$ is the node set of the graph and $E$ is the edge set of the graph, the word representation in $N$ and the edge weight in $E$ come from the global shared matrix. $l$ denotes the length of the sentence. $p$ represents the number of words in the window connected to individual word in the figure. In addition, we uniformly remove the edges that appear less than J times in the training set, so that the parameters can be fully trained. J is usually set to 2.
3.2. Multi-head attention mechanism

In this article, we use the Multi-Head Attention Mechanism (MHA) on each graph to aggregate contextual information. We hope that the model can learn different behaviors based on the same attention mechanism, and then combine different behaviors as knowledge, such as capturing various ranges of dependencies in the sequence (for example, short-distance dependence and long-distance dependence). Therefore, we allow the attention mechanism to combine different representation subspaces of queries, keys, and values. We will transform the query, key, and value with h different sets of linear projections obtained by independent learning. Then, the h groups of transformed queries, keys and values will be sent to the attention pool in parallel. Given a query \( q \in R_d \), key \( k \in R_d \) and value \( v \in R_d \), the calculation method of each attention head \( h_i \) (\( i=1,\ldots,h \)) is as follows:

\[
    h_i = f(w_{i}^{(q)} q, w_{i}^{(k)} k, w_{i}^{(v)} v) \quad (3)
\]

Among them, the learnable parameters include \( w_{i}^{(q)} \), \( w_{i}^{(k)} \), \( w_{i}^{(v)} \) and the function \( f \) representing the concentration of attention. \( f \) can be the additive attention. The output of multi-head attention needs to undergo another linear transformation, which corresponds to the result of hi heads concatenated.

MHA makes the representation of the node affected by different dependencies, which means that the representation can get information from the context. Therefore, even if it is a polysemous word, the exact definition in different circumstance can be defined through different dependencies. In addition, the parameters of the model are obtained from the shared matrix, which represents that it can also obtain global information like other GNN. Finally, we use this representation of all parameters in this model to forecast the tag of the data:

\[
    y_i = \text{softmax}(\text{Relu}(W \sum h_i + b)) \quad (4)
\]

Among them, \( W \) is the matrix that maps the vector to the output vacuity, \( h_i \) is the list of the head i, and \( b \) is the bias.

We use the cross-entropy loss between the real value and the predicted value to train the model, and the goal is to improve the network performance by minimizing the loss value. The cross-entropy loss function is

\[
    \text{Loss} = -g_i \log y_i \quad (5)
\]

where \( g_i \) is the vector representation of real value.
4. Experiments and results
Through experiments, we verify the validity of the text classification model based on graph neural network proposed in section 3.

4.1. Experimental setup
For the experiment, we used corpus including R8, R521, and 20newsgroups. Both R8 and R52 are subsets of the Reuters 21578 data set. The 20newsgroups data set is one of the international standard data sets used for text classification, text mining and information retrieval research. For all the above corpus, we stochastically choose 80% of the text from all the data to construct the training set, stochastically choose 10% of the text to construct the test set, and finally the remaining data to construct the validation set. The introduction of the corpus is shown in Table 1.

| Corpus       | #Train | #Test | #Valid | Species |
|--------------|--------|-------|--------|---------|
| R8           | 6140   | 767   | 767    | 8       |
| R52          | 7280   | 910   | 910    | 52      |
| 20newsgroups | 15076  | 1885  | 1885   | 20      |

We respectively use CNN, LSTM, and Text-GCN as the baseline model. Then, we compare them with the results of our own model experiments.

- CNN is used to classify text in [1], which has a deep structure and contains many convolutinal layers, it is a Feedforward Neural Networks.
- LSTM is a typical recurrent neural network and all its nodes are connected to form a chain. Its input is sequence data, and it recurses in the evolution direction of the sequence.
- Text-GCN proposed by Yao[6], it is a CNN with components on the graph.

4.2. Implementation details
We use random vectors or Glove to initialize the nodes, and the representation dimension of each node is set to 300 dimensions. Our model makes use of Adam optimizer, the initial learning rate is 0.001 and the L2 weight decay is set to 0.0001. We use 64 as our model batch size. In our model, dropout is set to 0.5. If the loss is not reduced for 10 consecutive cycles, we stop training.

4.3. Experimental Results
We use accuracy as an evaluation indicator. The calculation formula of accuracy is as follows:

\[
\text{accuracy} = \frac{(TP + TN)}{(P + N)}
\]  

Among them, \(TP, TN, P, N\) respectively represent the following meanings:

- True positives (TP): The number of negative cases correctly classified.
- True negatives (TN): The number of negative examples that are correctly classified,
- Positives (P): The number of positive examples divided by the classifier.
- Negatives (N): The number of cases classified as negative.

We use Table 2 to show the experimental data of our model and other models. We can find that our model can acquire the best effect.

We found that the experimental data of neural network model built on the graph is better than CNN and LSTM. This may be because the graph structure allows different numbers of neighbor nodes, so in this way, the precise meaning of words can be obtained through different combinations. In addition, we use the weight of the edge to represent the connection of the context. It is difficult for traditional models to realize these.

Experimental results show that our model performs better than Text-GCN. Text-CNN uses a bag-of-words model to represent documents, which is like ours, but their word nodes are located in a large range with external weighted edges, so that the significance of different words cannot be distinguished.
Our model uses a trainable side weight, which makes the expression of words different in the face of various collocations. In addition, the parameters in our model are globally shared, which implies that they can receive training by all texts containing the same collocation in the entire corpus.

Table 2. Experimental results of different models.

| Model          | R8   | R52   | 20newsgroups |
|----------------|------|-------|--------------|
| CNN            | 94.1±0.5 | 85.2±0.5 | 62.3±1.0    |
| LSTM           | 93.7±0.7 | 85.6±0.8 | 69.6±0.5    |
| Text-GCN       | 97.0±0.1 | 93.5±0.3 | 76.4±0.5    |
| CNN+Glove      | 95.7±0.5 | 87.7±0.5 | 63.4±1.0    |
| LSTM+Glove     | 96.0±0.2 | 90.4±0.8 | 70.1±0.5    |
| Text-GCN+Glove | 97.1±0.1 | 93.7±0.3 | 76.7±0.3    |
| **Our Model+Glove** | **97.7±0.3** | **94.6±0.3** | **78.8±0.5** |

On the other hand, this may be due to the different construction modes of the graph, our model performs better than Text-GCN. Text-GCN constructs a graph through the entire corpus, and our graph is constructed based on the context in a context window, just like traditional word embedding. So, our model can combine the advantages of pre-trained word embeddings and obtain better experimental results.

4.4. Memory consumption study
The nodes in our model have only relationship to neighboring nodes in the model, but Text-GCN constructed the entire corpus-level graph to analyze and connect nodes in a reasonably big distance. Since Text-GCN takes relationships (co-occurrence messages) as fixed weight, the distance must be enlarged to obtain more accurate co-occurrence weights. Therefore, we will get an edge weight matrix that is sparser than Text-GCN. In addition, Text-GCN puts all the word nodes in the corpus on one graph at a time, and we only put some of the nodes in the text on the graph at a time, which effectively reduces the memory consumption.

5. Conclusion
In this treatise, we propose a new neural network model that is applied to text classification problems in the world of NLP. Compared with Text-GCN, our model separately combines each text into a graph, and then classifies it, instead of building the entire corpus into a graph. Experimental outcomes performs that our model has effectively reduce memory consumption and achieved the best performance on several data sets.

Acknowledgments
This work is supported by the Science and Technology Project of Big Data Center of State Grid Corporation of China "Research on unstructured recognition technology based on NLP natural language analysis and keyword extraction technology service data Think-Tank construction"

References
[1] Yoon Kim. (2014) Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751.
[2] Sepp Hochreiter and Jurgen Schmidhuber. (1997) Long short-term memory. Neural computation,9(8):1735–1780.
[3] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. (2018) Relational inductive biases, deep learning, and graph networks. arXiv

5
preprint arXiv:1806.01261.

[4] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. (2009) The graph neural network model. IEEE Transactions on Neural Networks, 20(1):61–80.

[5] Michael Defferrard, Xavier Bresson, and Pierre Vandergheynst. (2016) Convolutional neural networks on graphs with fast localized spectral filtering. In Advances in neural information processing systems, pages 3844–3852.

[6] Liang Yao, Chengsheng Mao, and Yuan Luo. (2019) Graph convolutional networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 7370–7377.

[7] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. (2010) Understanding bag-of-words model: a statistical framework. International Journal of Machine Learning and Cybernetics, 1(1-4):43–52.