An Integration Model for Text Classification using Graph Convolutional Network and BERT

Bingxin Xue\textsuperscript{1}, Cui Zhu\textsuperscript{2}, Xuan Wang\textsuperscript{3}, Wenjun Zhu\textsuperscript{4}

\textsuperscript{1}Beijing University of Technology Faculty of Information Technology, Chaoyang District, Beijing, China
\textsuperscript{2}Beijing University of Technology Faculty of Information Technology, Chaoyang District, Beijing, China
\textsuperscript{3}Beijing University of Technology Faculty of Information Technology, Chaoyang District, Beijing, China
\textsuperscript{4}Beijing University of Technology Faculty of Information Technology, Chaoyang District, Beijing, China

*Corresponding author’s e-mail: xbx68@emails.bjut.edu.cn

Abstract. Recently, Graph Convolutional Neural Network (GCN) is widely used in text classification tasks, and has effectively completed tasks that are considered to have a rich relational structure. However, due to the sparse adjacency matrix constructed by GCN, GCN cannot make full use of context-dependent information in text classification, and cannot capture local information. The Bidirectional Encoder Representation from Transformers (BERT) has been shown to have the ability to capture the contextual information in a sentence or document, but its ability to capture global information about the vocabulary of a language is relatively limited. The latter is the advantage of GCN. Therefore, in this paper, Mutual Graph Convolution Networks (MGCN) is proposed to solve the above problems. It introduces semantic dictionary (WordNet), dependency and BERT. MGCN uses dependency to solve the problem of context dependence and WordNet to obtain more semantic information. Then the local information generated by BERT and the global information generated by GCN are interacted through the attention mechanism, so that they can influence each other and improve the classification effect of the model. The experimental results show that our model is more effective than previous research reports on three text classification data sets.

1. Introduction
Text classification is the basic task of natural language processing. Using text classification can scientifically and effectively manage a large amount of text, and plays an important role in application fields such as public opinion analysis, news classification, spam filtering, and opinion mining, etc. In the era of deep learning, text classification algorithms based on traditional machine learning have been more mature and achieved good results. Of course, it also has limitations, such as a large number of complicated manual features, and insufficient representation of the semantic features of the text, etc. However, deep learning methods can extract text features that are richer in semantic information, and can greatly save the cost of manual intervention. Existing text classification models based on deep neural networks include typical neural network methods (e.g. Convolutional Neural Network (CNN)[1], Recurrent Neural Network (RNN)[2], Capsule Network[3]), and their excellent variants (e.g. Long and
Short-Term Memory (LSTM)[4] Network, Gated Recurrent Unit (GRU)[5], Convolutional Recurrent Neural Network (CRNN)[6], and various models based on attention [7] (e.g. BERT[8]) , etc. These deep learning models can well capture the semantic and syntactic information in the local continuous word sequence, but most of them may ignore the information with discontinuous semantic words, and are limited in their ability to capture the global features of the information[9]. The Graph Convolutional Neural Network[10] method, a new method of solving graph classification developed in recent years, is one of the hot issues in the field of deep learning. It has the function of dealing with irregular matrices, which makes up for the limitations of CNN and other methods. This model performs graph convolution operations on the constructed topological relationship graph to obtain features, thereby realizing text classification. Good classification effects have been obtained in the fields of visual discovery and machine translation. In order to solve the classification problem, many variants of GCN[10-13] have been proposed and explored. Although GCN is gradually becoming a good choice for graph-based text classification, the current research still has some shortcomings that cannot be ignored.

In a recent study, researchers represented the text as a graph structure, and captured the structure information of the text through the Graph Convolutional Network, as well as the non-continuous and long-distance dependence between words. The most representative work is Graph Convolutional Network (GCN)[10] and its variant Text GCN[11]. However, the sequential structure of the text is ignored because the word is represented as a node and the neighborhood information of the node is aggregated through the adjacency matrix. As a result, contextual semantic information of the text is lost, which is important for understanding the meaning of the sentence. Look at this example, “You have a lot of hard work.”, “You have to work hard.”, where “work” will change its meaning depending on its position in the context sentence. By increasing the number of GCN layers, long-distance contextual semantic information can be captured to alleviate the problem of semantic information loss. However, the current research shows that multi-layer GCN used for text classification tasks will produce high space complexity[14]. At the same time, increasing the number of network layers will also cause excessive smoothness of node features, which will cause local features to converge to similar values. In addition to the problem of contextual semantic information, GCN may not be able to extract local feature information (e.g. key phrases) in text.

In order to solve the above mentioned problems, this paper proposes an improved GCN (MGCN) for text classification. Based on the Vocabulary Graph Convolution (VGCN)[13], we rationally introduce WordNet and dependency relationship, and combine GCN with BERT through attention mechanism[15]. By building dependencies, GCN can make good use of contextual dependencies of text. WordNet was introduced to provide more useful connections between words. At the same time, BERT model is used to extract the local feature information of text. Subsequently, the attention mechanism is used to interact the local information obtained by BERT with the global information obtained by GCN, so that they can influence each other and jointly construct the final classification representation.

We call the proposed model MGCN. Experiments on three benchmarking datasets demonstrate that the problems in the current GCN-based methods can be effectively solved by MGCN which has achieved certain results. The main contributions of this paper are as follows:

- The dependency and semantic dictionary are fully integrated into MGCN. They can enrich the connections between words, effectively deal with context-dependent problems, and reduce the number of GCN layers partly. BERT can effectively extract local feature information. Through the attention mechanism, the local information and the global information are closely combined to obtain a text representation that contains more comprehensive text classification semantics.
- A large number of experimental results not only prove the rationality of integrating BERT, WordNet and dependencies, but also prove the efficiency of MGCN for text classification.

2. Related work

Traditional text classification methods are mainly based on feature engineering. Feature engineering includes bag-of-words model[16] and n-grams, etc. Later, there was research[17-18], which was to convert text into graphics and perform feature engineering on graphics. But these methods cannot
automatically learn the embedding representation of nodes. Compared with traditional text classification methods, deep learning-based text classification methods have the ability to automatically acquire features and perform end-to-end learning, and they can learn the deep semantic information of the text. In terms of deep learning models, Kim 2014[1] proposed to use text convolutional neural networks (TextCNN) for text classification, using multiple filters of different sizes to capture local feature information of text, and obtained good results on multiple data sets. Conneau et al. 2017[19] proposed VD-CNN based on character level (Character-level), which also achieved good results. Liu et al. 2016[20] proposed to use recurrent neural network for text classification, which can capture the historical information of the sequence. In order to encode the internal relationship between sentences in the document, Tang et al. 2015[21] proposed a gated recurrent neural network and used it for sentiment classification. In order to introduce position invariance into RNN, Wang 2018[22] used discontinuous recurrent neural network for text classification.

At the same time, the attention-based model had also been proposed, which was used to capture the weight of each word within a sentence. Yang et al. 2016[15] proposed to use hierarchical attention networks to model and classify documents. Wang et al. 2016[23] introduced the attention mechanism into LSTM, which provided a new idea for text classification. Lin et al. 2017[24] proposed a model for extracting interpretable sentence embeddings by introducing a self-attention mechanism, and applied it in sentiment classification and other fields, and achieved good results. Combining convolutional neural network and nested long short-term memory network, Liu et al. 2019[25] proposed a CNLSTM model based on attention mechanism for Chinese news classification. A neural network model based on the convolutional attention mechanism was proposed by Gu et al. 2020[26] to extract the local optimal emotional polarity of the text, capture the semantic information of the emotional polarity transfer of the text, and use it in the field of sentiment classification. In particular, Vaswani et al. 2017[7] proposed a self-attention mechanism, which was good at capturing the internal correlation between features and was less dependence on external information. By introducing the Bidirectional Encoder Representation from Transformers (BERT)[8] and the self-interaction attention mechanism, Dong et al., 2020[27] obtained a text representation that contains more comprehensive text classification semantics. Most attention-based deep neural networks mainly focus on local continuous word sequences, which provide local context information. However, the ability to capture the global characteristics of the relevant information may not be sufficient.

In recent years, many studies have tried to apply convolution operations to graph data. Kipf and Welling 2017[10] proposed Graph Convolutional Networks (GCNs), and the model has achieved good results on node classification tasks. Velickovic et al. 2018[28] proposed Graph Attention Network (GAT), which uses the attention mechanism[29] to assign weights to the neighborhood information of nodes. Compared with the Graph Convolutional Network without adding weights, it has better promotion. Later, Yao et al. 2019[11] proposed the TextGCN model, which introduced the Graph Convolutional Network to text classification for the first time. By modeling the text into a graph, the nodes in the graph are composed of words and documents, and then applying the Graph Convolutional Network to classify the text on the graph, a good effect is finally obtained. Cavallari et al. 2019[30] introduced a new setting of graph embedding, which considers embedding communities instead of individual nodes. Tang et al. 2020[12] introduced a Bidirectional Long Short-Term Memory (BiLSTM) Network, part-of-speech (POS) information and dependencies to enhance the effect of GCN. Lu et al. 2020[13] integrated the vocabulary graph embedding module with BERT, and achieved good results in many public datasets.

Although GCN performs well in text classification, it still cannot cover more semantic information and cannot solve the problems of context dependence and insufficient local information extraction in text classification tasks. To solve these problems, an improved model based on GCN is proposed accordingly.
3. Proposed method
Through in-depth research on text classification based on neural networks, this paper proposes MGCN. The whole process of MGCN can be divided into the steps of extracting text features, constructing adjacency matrix based on normalized point-wise mutual information (NPMI)[31], WordNet and dependencies, training neural network, and final prediction. The architecture of MGCN is shown in figure 1. First, use BERT's word embedding to obtain text features to form the features required by MGCN. Then the adjacency matrix required by MGCN is constructed based on NPMI and WordNet. At the same time, the generated dependence relationship is used to solve the context dependence problem and construct another adjacency matrix required by MGCN. After that, the neural network is trained with input features and adjacency matrix, and the hidden state vector of the last layer is obtained. Next, the hidden state vector is combined with the text features obtained by BERT using the attention mechanism. Subsequently, final features will be generated to predict the category of the given text.

Figure 1. The architecture of MGCN.

3.1. Vocabulary graph
3.1.1. Vocab graph
In this article, we first use Wordnet and NPMI to construct a text map. NPMI, which is a commonly used measure of word association, is used to calculate the weight between two word nodes. But the calculation of NPMI depends on the corpus. If the probability of certain words appearing in the corpus is low, this method may cause the calculation result of NPMI to be very small, and it will be judged that the similarity of the two words is low. There are a lot of connections between words, and the most direct connection is the synonymy between some words. For example, the two words “omit” and “overlook” have very similar meanings and can be used interchangeably in most cases. If this synonymous relationship can be used, it will improve the accuracy of the calculation undoubtedly. The WordNet method can calculate the closeness of two words, but there are only four parts of speech in WordNet, namely noun, verb, adjective and adverb. Although these four words dominate natural language, they are not the only four words, such as conjunctions and pronouns. Therefore, we need to combine NPMI with semantic dictionary. For example, the word “omit” has a very low probability of appearing in the corpus. When calculating the co-occurrence probability of “omit” and other words, the values of P (omit & other words) and P (omit) are very small, even 0, it will cause the value of NPMI to be very small. Therefore, “omit” may be judged as having little or no semantic relevance with other words in the end.
But if you expand the synonyms of the word “omit” and other words, you can calculate the semantic similarity between them through WordNet.

For two words i and j, we use WUP method\cite{32} to calculate the semantic similarity between them. WUP method is a similarity measurement method based on path structure proposed by Wu and Palmer. The similarity is calculated according to the similarity degree of word meaning and the position of synonym set in the upper word tree relative to each other. If the results are large, meaning that they are interchangeable in most cases, then they are similar. Since a word can have multiple meanings, we need to compare each meaning of two words, so we need to expand the synonyms separately. We chose to use the maximum value of the comparison to determine the similarity between two words so that we could filter out words with more unrelated meanings. If the NPMI value of the two words to be calculated is very small, WordNet can be used for calculation. This can minimize the problems caused by the low frequency of certain words in the corpus. Formally, the edge weight between node i and node j is defined as:

\[
A_{ij} = \begin{cases} 
  d(S_1, S_2) & 0.5 \leq d(S_1, S_2) \leq 1, \text{NPMI}(i,j) < 0 \\
  \text{NPMI}(i,j) & i, j \text{ are words}, \text{NPMI}(i,j) > 0 \\
  1 & i = j \\
  0 & \text{otherwise}
\end{cases}
\]  

(1)

Where \(S_1\) and \(S_2\) are synonym expansion sets of i and j respectively. When the distance between the two word synonym expansion sets is between 0.5 and 1, the performance will be better. At this time, we create an edge.

The NPMI value of a pair of words i, j is calculated as:

\[
\text{NPMI}(i,j) = -\frac{1}{\log p(i,j)} \log \frac{p(i,j)}{p(i)p(j)}
\]

(2)

Where \(p(i,j) = \frac{\#w(i,j)}{\#w}\), \(p(i) = \frac{\#w(i)}{\#w}\), \(\#W(i)\) is the number of all sliding windows containing word i, \(\#W(i,j)\) is the number of all sliding windows containing word i and word j, and \(\#W\) is the total number of sliding windows. We set the window larger to gain long-term dependence. A positive NPMI value indicates that the semantic relevance between words is high, while a negative NPMI value indicates that there is little or no semantic relevance. Therefore, we only create an edge between pairs of words with a positive NPMI value.

### 3.1.2. Dependency graph

We are also trying to get more relationships for words. Therefore, the dependency relationship is introduced to reveal a clear syntactic structure to construct a map of word dependency. Generally, the weight of the edge between the document node and the word node is calculated using the frequency and inverse frequency term of the document (TFIDF) algorithm. But it lacks the combination of text semantics to understand the text content. In order to overcome the above problems, this article refers to\cite{33}, introduces dependencies, and uses an improved TFIDF-based weight calculation algorithm to understand and optimize text features.

Dependency syntax is mainly concerned with the relationship between the various words (sentence components) in a sentence, and it is not affected by the physical location of the components\cite{34}. Since the importance and relevance of various dependencies in a sentence are different, first of all, determine the importance of words to sentences, texts and even categories according to the different dependencies of words and predicates on the meaning of words. Then, according to the importance of different components to the sentence, the sentence composition is divided into eight levels.

The results of dependency analysis are usually represented by labeled directed graphs. In a directed graph, words (components) are represented by nodes (Node). The dependency between the head word and the auxiliary word is usually represented by an arrow line, starting from the head word and pointing to the auxiliary word. Finally, the label indicates the specific relationship between them. For example, as shown in figure 2, “root” represents the central word “has”. Examples of the relationship between the
performance head word and its subsidiary words are “has” and “performances”. The result of dependency analysis shows that the relationship between these two words is a direct object (dobj, direct object) relationship, and the result is also in line with the expected grammatical relationship.

Figure 2. Directed graph of dependency analysis.

After classifying the text features in the data set according to the dependency relationship, this paper uses the following dependency-based TFIDF weight calculation method. Specific steps are as follows:

For the word i, we calculate the number of times that the word i appears in the text and set it as n. Then, according to the result of the dependency syntax analysis implemented by Stanford Parser, it is obtained that the word i belongs to the m-th (1≤m≤n) sentence component in the text. And according to table 2 in the paper [33], classify the m-th occurrence of word i in the text into ki,m level, and assign weight w_{i,m} to it.

Then the improved frequency TF_{i} with the weight of word i in the text is calculated by formula (3). For word i, the improved TFIDF_{i} weight based on the dependency relationship is as shown in formula (4). Where s represents the total number of words in the text where the word i is located, pi represents the number of texts containing the word i, and D represents the total number of texts in the dataset. Where λ is a parameter, which is used to adjust the weight gap between feature levels, and the range is [0, 1].

According to the improved TFIDF_{i} weight formula based on the dependency relationship, the weight between the word and the document is obtained. Then the relationship between the word and the document is transformed into the relationship between the word and the word containing the document information and the dependent relationship, and then another text graph is constructed. The dependency-based text graph can provide both short-distance context dependency and long-distance context dependency at the same time, which can solve the context dependency problem more effectively.

3.2. Vocabulary GCN
The general GCN[10] is a multi-layer neural network that directly convolves on the graph and updates the embedding vector of the node based on the information of its neighbor nodes. We use the Vocabulary Graph Convolution (VGCN)[13], and the goal is to convolve related words. Therefore, for m documents of small batch processing, the single-layer convolution is:

\[ H = X \tilde{A} W \]
Where $\tilde{A}$ represents the vocabulary graph. $\tilde{X}\tilde{A}$ to extract part of vocabulary related to the input
matrix $X$. $W$ hold hidden state vector of a weight of a single document, dimension is $|V| \times h$.

The corresponding two-level GCN with LeakyReLU function is:

$$GCN = \text{LeakyReLU}(X_{mv}\tilde{X}_{vv}W_{vh})W_{hc}$$ (6)

Where $m$ is the mini-batch size, $v$ is the vocabulary size, $h$ is the hidden layer size, and $c$ is the class
size or sentence embedding size. $W$ holds the weight of the hidden state vector of a single document.
Each row of $X_{mv}$ is a vector containing document features, which is the word embedding of BERT. The
above equation aims to produce a convolutional layer of the graph that captures the input-related part of
the graph. Then a 2-layer convolution is performed to combine the words in the input sentence with the
related words in the vocabulary.

Specifically, we perform one layer of graph convolution for the two text graphs generated in the
previous section, then add the generated two hidden layer vectors, and finally perform another layer of
graph convolution.

In the training process of the original GCN, ReLU is used as the activation function of the GCN. In
this paper, the non-linear function LeakyReLU is selected as the activation function of GCN, which
overcomes the problem of gradient disappearance and speeds up training. In addition, L2 regularization
is also used to reduce overfitting. The LeakyReLU function is as follows:

$$\text{LeakyReLU}(x) = \begin{cases} 0.2x, & x < 0 \\ x, & x \geq 0 \end{cases}$$ (7)

3.3. Feature fusion and classification

In order to further extract the local information of the text, we use the word vector generated by the
BERT model. Specifically, the pre-trained BERT model is used to predict the text category, but the
BERT result is not used as the final text classification result. The vector obtained by the hidden layer of
BERT can represent context-sensitive word embedding. Experimental results show that combining the
word vector generated by BERT with the word vector generated by GCN can achieve better
classification performance than using BERT or GCN alone.

We use the attention mechanism to combine GCN and BERT. Specifically, the global features
generated by the GCN are used to calculate the weights using the attention mechanism[15], which makes
the network focus on the internal relations of the features themselves. Then combine the calculated
weights with the local features generated by BERT. This method not only preserves the order of words
in the sentence, but also uses the global information obtained by GCN, which can improve the expressive
ability of the model. Therefore, the new attention layer can be defined as follows:

$$u_t = \tanh(W_{wh}h_t + b_w)$$ (8)
$$a_t = \frac{\exp(a_t^T h_t)}{\sum_t \exp(a_t^T h_t)}$$ (9)
$$d = \sum_t a_t \cdot fp_t = a \cdot fp$$ (10)

Where $t$ is the number of words, $h_t$ is the hidden state vector of the output of GCN layer, and $fp_t$ is
the output of BERT layer. $a_t$ is the contribution of the $t$-th node to text classification, and $u_t^T h_t$ is
the sum of the similarity between the $t$-th node and all nodes in the sentence. $d$ is the final feature
representation of the sentence.

Finally, the final feature are sent to Softmax for classification. Different from the traditional attention
mechanism, this provides greater combination and potential, and can achieve better performance.

4. Experiments

In this section, this paper first introduces how to set up experiments and analyze the experimental results
of different models. Subsequently, the ablation study is conducted to further prove the performance of
each component.
4.1. Datasets
We conducted experiments on the following three text datasets:

- **SST-2**: This dataset is the Stanford sentiment tree library data set, which is a binary single sentence classification task, which consists of sentences extracted from movie reviews and artificial annotations of their emotions[35]. We use public version, which has 6920 training samples, 872 verification samples, and 1821 test samples. There were 4,963 positive comments and 4,650 negative comments in total. The average length of reviews is 19.3 words.
- **MR**: A binary sentiment classification dataset of movie reviews. Each comment contains only one sentence[36]. We used the public version in[37]. It has 5331 positive comments and 5331 negative comments. The average length of reviews is 21.0 words.
- **CoLA**: A dataset about grammar released by New York University[38]. This task is mainly to determine whether a given sentence is grammatically correct, which belongs to the task of text binary classification of a single sentence. We use the public version containing 8,551 training data and 1,043 training data, with a total of 6,744 positive examples and 2,850 negative examples. The average length is 7.7 words.

4.2. Parameter settings
The WordNet threshold of all datasets in this paper is set to 0.5. When calculating NPMI on the dataset, the entire sentence is used as a text window to construct a vocabulary graph. The NPMI threshold of all datasets is set to 0.2 to filter out meaningless relationships between words. In the MGCN model, the graph embedding output size is set to 40, and the hidden dimension of graph embedding is set to 128. We use the pre-trained BERT model provided by Google and set the maximum sequence length to 200. The dropout rate is set to 0.2. We train the model in 10 epochs. The activation function of the GCN layer is LeakyReLU. The parameters set by the baseline method are the same as those in the original paper.

Additional parameter Settings for different data sets are shown in table 1.

| Dataset | Batch size | Learning rate | Loss weight decay |
|---------|------------|---------------|------------------|
| SST-2   | 16         | 1e-5          | 1e-5             |
| MR      | 32         | 8e-6          | 0.001            |
| CoLA    | 16         | 8e-6          | 0.001            |

4.3. Baselines
In order to comprehensively evaluate the model, weighted average F1-Score and macro F1-Score are selected as evaluation indicators, and MGCN is compared with the following baseline models.

- **Bi-LSTM[39]**: Bidirectional LSTM is often used for text classification. Use BERT pre-trained word embeddings as input to the Bi-LSTM model.
- **Text-GCN[11]**: A text classification method that uses graphs to model text. The words and documents in the text are regarded as nodes, where the edges of the document and the words are based on the appearance information of the words in the document, and the edges of the words and the words are based on the global word co-occurrence information of the words. It has many improvements over many of the most advanced models.
- **VGCN[13]**: The model only uses VGCN, and the output dimension is the class size. The pre-trained word embedding of BERT is used as input. The output of VGCN is relayed to a fully connected layer with Softmax function to generate classification scores. This model only uses the global information of the vocabulary graph.
- **BERT[8]**: We use a small version (BERT-base-uncase) of pre-trained BERT.
• VGCN-BERT[13]: Based on the word co-occurrence information, a graph convolutional network is constructed on the vocabulary graph, and then the graph embedding and word embedding are provided to the self-attention encoder in BERT.

4.4. Main experimental results
The main results on weighted average F1-Score and macro F1-Score on test sets are presented in table 2. The experimental results on three benchmark datasets confirm that the performance of MGCN is basically better than other baseline models, which further proves the effectiveness and robustness of MGCN in text classification.

| Model     | SST-2 Weighted avg F1 | SST-2 Macro F1 | MR Weighted avg F1 | MR Macro F1 | CoLA Weighted avg F1 | CoLA Macro F1 |
|-----------|------------------------|----------------|-------------------|-------------|----------------------|--------------|
| Text-GCN  | 80.45                  | 80.45          | 75.67             | 75.67       | 56.18                | 52.30        |
| Bi-LSTM   | 81.32                  | 81.32          | 76.39             | 76.39       | 62.88                | 55.25        |
| VGCN      | 81.64                  | 81.64          | 76.42             | 76.42       | 63.59                | 54.82        |
| BERT      | 91.49                  | 91.49          | 86.24             | 86.24       | 81.22                | 77.02        |
| VGCN-BERT | 91.93                  | 91.93          | 86.35             | 86.35       | 83.68                | **80.46**    |
| Our model | **92.31**              | **92.31**      | **87.37**         | **87.37**   | **83.73**            | 80.34        |

It can be seen from the table that MGCN has a good classification effect compared with the baseline models, which proves the effectiveness of the model in this paper. In the models VGCN and Text-GCN that use vocabulary graphs, the performance of the two models is equivalent. Compared with the VGCN and BERT models, the model in this paper has a certain degree of effect improvement, indicating that the introduction of dependencies, semantic dictionaries and the BERT model in this model does enrich the contextual semantic information and local feature information of GCN, making the model in this paper compared to Other models have better classification performance. At the same time, compared with VGCN-BERT, MGCN usually performs better. All this shows that the combination of each component of this paper is useful.

4.5. Ablation study
In order to further study the benefits of the various components of MGCN, this paper conducted ablation experiments on the effectiveness of the constructed model, and the results are shown in table 3. The experimental results of MGCN and three variant models prove that the three components used in this paper can improve the performance of MGCN.

| Model     | SST-2 Weighted avg F1 | SST-2 Macro F1 | MR Weighted avg F1 | MR Macro F1 | CoLA Weighted avg F1 | CoLA Macro F1 |
|-----------|------------------------|----------------|-------------------|-------------|----------------------|--------------|
| Our model | **92.31**              | **92.31**      | 87.37             | 87.37       | **83.73**            | **80.34**    |
| MRDS      | 92.14                  | 92.14          | **87.40**         | **87.40**   | 82.32                | 79.12        |
| MRDW      | 92.09                  | 92.09          | 87.24             | 87.24       | 81.68                | 78.37        |
| MRDWA     | 91.38                  | 91.38          | 86.49             | 86.49       | 80.70                | 76.30        |
As shown in table 3, we can see that the model that deletes the corresponding module has a certain degree of performance degradation compared to the overall model, indicating that the modules are complementary to each other. It can be observed that different modules have different functions in different datasets.

![Figure 3](image.png)

Figure 3. Histogram of experimental results of ablation study.

First of all, we removed the dependency relationship, and constructed a model called MRDS only by learning the dependencies between nodes and WordNet. As shown in figure 3, compared with the F1 score of MGCN, the F1 score of the SST-2, MR and CoLA datasets on the MRDS model all have different degrees of decline. Relatively speaking, the score of the CoLA dataset has dropped more. This is because the CoLA dataset is a grammar-related dataset, and the dependency relationship is established through the dependency syntax. This indicates that dependencies are related to graph-based text classification.

Then, we deleted WordNet, and the performance of the model named MRDW would also decrease. As shown in figure 3, the F1 score of MRDW on the three data sets are lower than the F1 score of MGCN and MRDS. Although the gap is not high, it also proves that the introduction of WordNet can cover other useful connections between words.

Finally, after removing the attention mechanism, as shown in figure 3, the performance of the model named MRDWA also decreases. The decline in its F1 score proves that the combination of global information and local information through the attention mechanism can make them affect each other and help the model better represent text information, which further illustrates the effectiveness of the module.

5. Conclusion and future work

In this research, we propose a new MGCN model that uses the dependency relationship to capture the context dependency, and the semantic dictionary can increase the useful connection between words to supplement the shortcomings of the graph convolutional network in text representation. At the same time, the attention mechanism is combined with the BERT model to further improve the model's ability to extract local feature information, enrich the model's representation of text information, and improve the effect of text classification. We have conducted experiments on three datasets, which have achieved a certain degree of improvement compared with the baseline model.

In future work, we will further explore dependencies and construct graph features richer in semantic information to improve the performance of the model.

References

[1] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. EMNLP.
[2] Zhang, H., Xiao, L., Wang, Y., & Jin, Y. (2017). A Generalized Recurrent Neural Architecture for Text Classification with Multi-Task Learning. IJCAI.
[3] Zhao, W., Peng, H., Eger, S., Cambria, E., & Yang, M. (2019). Towards Scalable and Reliable Capsule Networks for Challenging NLP Applications. ACL.
[4] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9,
1735-1780.

[5] Cho, K., Merrienboer, B.V., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. EMNLP.

[6] Wang, R., Li, Z., Cao, J., Chen, T., & Wang, L. (2019). Convolutional Recurrent Neural Networks for Text Classification. 2019 International Joint Conference on Neural Networks (IJCNN), 1-6.

[7] Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. ArXiv, abs/1706.03762.

[8] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL.

[9] Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V.F., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gülçehre, Ç., Song, H.F., Ballard, A.J., Gilmer, J., Dahl, G.E., Vaswani, A., Allen, K.R., Nash, C., Langston, V., Dyer, C., Heess, N.M., Wierstra, D., Kohli, P., Botvinick, M.M., Vinyals, O., Li, Y., & Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. ArXiv, abs/1806.01261.

[10] Kipf, T., & Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks. ArXiv, abs/1609.02907.

[11] Yao, L., Mao, C., & Luo, Y. (2019). Graph Convolutional Networks for Text Classification. ArXiv, abs/1809.05679.

[12] Tang, H., Mi, Y., Xue, F., & Cao, Y. (2020). An Integration Model Based on Graph Convolutional Network for Text Classification. IEEE Access, 8, 148865-148876.

[13] Lu, Z., Du, P., & Nie, J. (2020). VGCN-BERT: Augmenting BERT with Graph Embedding for Text Classification. Advances in Information Retrieval, 12035, 369 - 382.

[14] Li, G., Müller, M., Thabet, A.K., & Ghanem, B. (2019). DeepGCNs: Can GCNs Go As Deep As CNNs? 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 9266-9275.

[15] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E.H. (2016). Hierarchical Attention Networks for Document Classification. NAACL.

[16] Harris, Z.S. (1981). Distributional Structure.

[17] Rousseau, F., Kiagias, E., & Vazirgiannis, M. (2015). Text Categorization as a Graph Classification Problem. ACL.

[18] Luo, Y., Uzuner, O., & Szolovits, P. (2017). Bridging semantics and syntax with graph algorithms - state-of-the-art of extracting biomedical relations. Briefings in bioinformatics, 18 1, 160-178.

[19] Schwenk, H., Barrault, L., Conneau, A., & LeCun, Y. (2017). Very Deep Convolutional Networks for Text Classification. EACL.

[20] Liu, P., Qiu, X., & Huang, X. (2016). Recurrent Neural Network for Text Classification with Multi-Task Learning. ArXiv, abs/1605.05101.

[21] Tang, D., Qin, B., & Liu, T. (2015). Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. EMNLP.

[22] Wang, B. (2018). Disconnected Recurrent Neural Networks for Text Categorization. ACL.

[23] Wang, Y., Huang, M., Zhu, X., & Zhao, L. (2016). Attention-based LSTM for Aspect-level Sentiment Classification. EMNLP.

[24] Lin, Z., Feng, M., Santos, C.N., Yu, M., Xiang, B., Zhou, B., & Bengio, Y. (2017). A Structured Self-attentive Sentence Embedding. ArXiv, abs/1703.03130.

[25] Liu, Y., Zhai, D., & Ren, Q. (2019). News text classification based on cnlstm model with attention mechanism. Computer Engineering.

[26] Jun-Hua, G. U., Peng, W. T., Na-Na, L. I., & Dong, Y. F. (2020). Sentiment classification method based on convolution attention mechanism. Computer Engineering and Design.

[27] Dong, Y., Liu, P., Zhu, Z., Wang, Q., & Zhang, Q. (2020). A Fusion Model-Based Label Embedding and Self-Interaction Attention for Text Classification. IEEE Access, 8, 30548-30559.
[28] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio’, P., & Bengio, Y. (2018). Graph Attention Networks. ArXiv, abs/1710.10903.
[29] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A.C., Salakhutdinov, R., Zemel, R.S., & Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. ICML.
[30] Cavallari, S., Cambria, E., Cai, H., Chang, K.C., & Zheng, V.W. (2019). Embedding Both Finite and Infinite Communities on Graphs [Application Notes]. IEEE Comput. Intell. Mag., 14, 39-50.
[31] Bouma, G. (2009). Normalized (pointwise) mutual information in collocation extraction.
[32] Wu, Z., & Palmer, M. (1994). Verb Semantics and Lexical Selection. ArXiv, abs/cmp-lg/9406033.
[33] Zhu, X., Xu, Q., Chen, Y., Chen, H., & Wu, T. (2020). A Novel Class-Center Vector Model for Text Classification Using Dependencies and a Semantic Dictionary. IEEE Access, 8, 24990-25000.
[34] Tesnière, L. (1959). Éléments de syntaxe structurale.
[35] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. EMNLP.
[36] Pang, B., & Lee, L. (2005). Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. ACL.
[37] Tang, J., Qu, M., & Mei, Q. (2015). PTE: Predictive Text Embedding through Large-scale Heterogeneous Text Networks. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
[38] Warstadt, A., Singh, A., & Bowman, S.R. (2019). Neural Network Acceptability Judgments. Transactions of the Association for Computational Linguistics, 7, 625-641.
[39] Graves, A., Mohamed, A., & Hinton, G.E. (2013). Speech recognition with deep recurrent neural networks. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 6645-6649.