Document and Word Representations Generated by Graph Convolutional Network and BERT for Short Text Classification

Zhihao Ye [1] and **Gongyao Jiang** [2] and **Ye Liu** [3] and **Zhiyong Li** [4,*] and **Jin Yuan** [5]

Abstract.
In many studies, the graph convolution neural networks were used to solve different natural language processing (NLP) problems. However, few researches employ graph convolutional network for text classification, especially for short text classification. In this work, a special text graph of the short-text corpus is created, and then a short-text graph convolutional network (STGCN) is developed. Specifically, different topic models for short text are employed, and a short text short-text graph based on the word co-occurrence, document word relations, and text topic information, is developed. The word and sentence representations generated by the STGCN are considered as the classification feature. In addition, a pre-trained word vector obtained by the BERTs hidden layer is employed, which greatly improves the classification effect of our model. The experimental results show that our model outperforms the state-of-the-art models on multiple short text datasets.

1 Introduction

Short text usually has a short length, and generally, it includes up to 140 characters. The short text classification is widely applied to the question-answer systems, dialogue systems, sentiment analysis systems, and other systems, and it is one of the most important tasks in natural language processing. Many different deep learning models, such as a convolutional neural network (CNN)[15] and a recurrent neural network (RNN)[13], have been used in the short text classification. Compared with the traditional methods, such as support vector machine (SVM)[27], the text classification model based on deep learning provides better results and achieve significant improvement. Recently, a new research direction called the graph neural networks[4, 2], especially the graph convolutional neural network[17] , has attracted wide great attention.

The graph convolutional neural networks have been applied to natural language processing (NLP) task, such as semantic role labeling [21], relation classification [18], and machine translation [1]. Yao and Mao [31] proposed a novel text graph convolutional neural network that could build a single text graph of a corpus based on the word co-occurrence and document word relations, then learn a text graph convolutional network (Text GCN) of the corpus.

However, none of the above-mentioned works applies the graph convolutional network to the classification of short texts. Moreover, Yaos model [31] performs worse than a CNN or RNN on the short text dataset, such as MR (Please refer to section 4.1 for more details).

In this paper, we propose a novel deep learning-based method for short text classification. Specifically, following Text GCN [31], we build short text graph of short text corpus. After that, due to the short length and sparse features, we use the topic models to extract the topic information of a short text and employ topic information to help construct the short text graph. Namely, the Text GCN ignores semantic information in word node representation and word orders that are very useful in the short text classification, and in the final input of the softmax classifier, only the document node is used. On the contrary, we input both word and document nodes trained by the graph convolutional network (GCN) into the bi-directional long short-term memory (BiLSTM) or other classification models to classify the short text further. In addition, we use the vector received by the BERT's hidden layer which can represent the context-sensitive word representation and we find that using the combination of the representation obtained by a short-text GCN and the pre-trained word vector obtained by BERT's hidden layer [9]]can greatly improve the performance of our model.

The contributions of this work can be summarized as follows.

(1) A novel graph neural network-based method for short text classification is proposed. The proposed method is validated by experiments on several different-sentence-length datasets, and it is found that the proposed model achieves the state-of-the-art effect on all datasets.

(2) For the first time, a topic model is used to obtain the global topic information on the short text, and the topic information is further used to assist the construction of the short-text graph and address the issues of sparse features of the short text.

(3) In order to make more effective use of the word and sentence representations generated by the short-text GCN, we input them into the BiLSTM classifier, and it is found that adding the word representations generated by the BERT can greatly improve the performance of short text classification.

2 Related Work

Recently, numerous researchers have applied deep learning [10] to text classification. Specifically, Kim [15] used CNN for text classification. The architecture is a direct application of CNNs as used in computer vision but with one dimensional convolutions. In [33, 7], the authors designed the character level CNNs and achieved promising results. In [25, 19, 20], the BiLSTM was employed to learn text

[1] Hunan University, China, email: zhihaoye.chn@qq.com
[2] Guangdong University of Technology, China, email: uni_coder99@163.com
[3] Sun Yat-sen University, China, email: jourkliu@163.com
[4] Hunan University, China, email: zhiyong.li@hnu.edu.cn
∗ Corresponding author.
[5] Hunan University, China, email: yuanjin@hnu.edu.cn

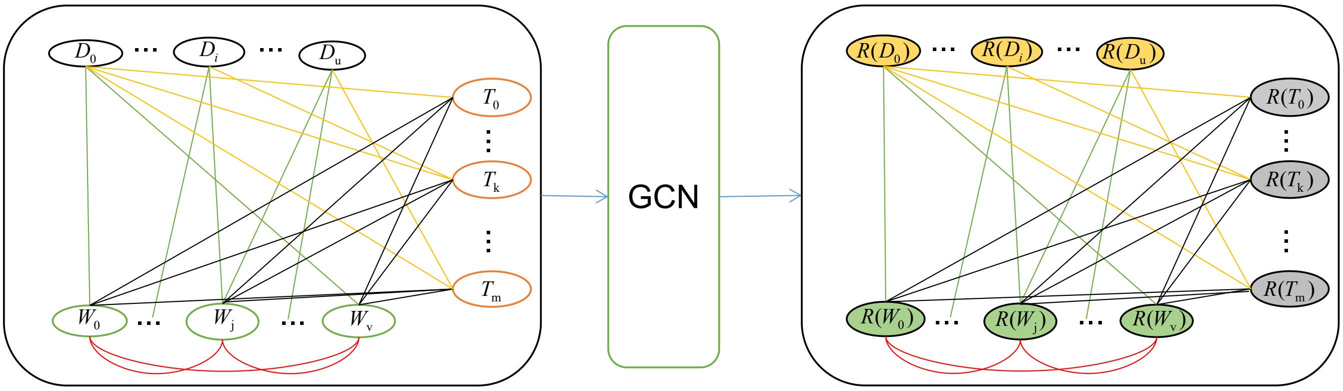


Figure 1. The overview of short text GCN. The left side represents the short-text graph, where D_i represent document i, T_k represent topic k, and w_j represent word j. v, u and m represent the total number of words, documents and topics respectively. Yellow lines represent the topic-to-document edges, green lines represent the document-to-word edges, red lines represent the word-to-word edges, and black lines represent the word-to-topic edges. After initialization and training of the GCN, the short-text graph presented on the right is obtained, where $R(X)$ represents the representation of X.

representation. Besides, in order to increase the representation flexibility of deep learning models, Yang [30] employed the attention mechanisms as an integral part of models for text classification.

To overcome the feature-sparsity problem of a short text, in [24, 6] the pre-trained topic mixtures learned by the LDA [3] were considered as a part of features to alleviate data sparsity issues. In [32], the authors encoded the topic representations in a memory mechanism where topics were induced jointly with the text classification in an end-to-end manner. The difference between our model and the previous works is that we employ the topic information extracted by the topic model, the word co-occurrence, and the document word relations to construct a short-text graph, and then a graph convolutional network is used to learn and train the short-text graph.

In [17], a kind of graph neural networks called the graph convolutional networks (GCNs) are introduced, and the state-of-the-art classification results were achieved on a number of benchmark graph datasets. The GCN was also explored in several NLP tasks such as semantic role labeling [21], relation classification [18], and machine translation [1]. The GCN for text classification can be categorized into two groups. In the first group [12, 8, 17, 34], a document or a sentence is considered as a graph of word nodes, and the not-routinely-available document citation relation is used to construct the graph. In the second group [31], the documents and words are regarded as nodes, and inter-document relations are not required. In this work, the documents and words are regarded as nodes. However, different from the previous works, our work aims at the problem of sparse features of a short text in the short text classification, and our model makes full use of representation generated by the graph convolutional networks and BERT for the short text classification.

3 Framework Overview

In this work, the short text classification process performed by the graph convolution network can be divided into two steps. In the first step, a special text graph is built for the short-text corpus, and the short-text graph is employed to learn and train the short-text graph convolutional network. The proposed short-text GCN is illustrated in Figure 1. In the second step, the word and document representations generated by the short-text GCN are inputted into the BiLSTM classifier to obtain the text category. Moreover, the word representations generated by the BERT are also added into the model which can help improve the model classification performance. The proposed classifier is illustrated in Figure 2..

3.1 Topic Model for Short Text

Topic model is a statistical model that clustering latent semantic structure of corpus. However, directly applying conventional topic models (e.g. LDA[3] and PLSA) on short texts may not work well (it could be easily justified with the experiments in section 5.1). The fundamental reason lies in that conventional topic models implicitly capture the document-level word co-occurrence patterns to reveal topics, and thus suffer from the severe data sparsity in short documents. In this work, To solve the feature-sparsity problem of a short text, we use the biterm topic model (BTM) [29], which learns the topics by directly modeling the generation of word co-occurrence patterns (i.e. biterms) in the whole corpus.

BTM fully leverage the rich global word co-occurrence patterns to better reveal the latent topics. Specifically, For each topic t, BTM draws a topic-specific word distribution $\phi_t \sim Dir(\beta)$ and a topic distribution $\theta \sim Dir(\alpha)$ for the whole collection, where α and β are the Dirichlet priors. For each biterm b in the biterm set B, BTM draws a topic assignment $t \sim Multi(\theta)$ and two words: $w_i, w_j \sim Mulit(\phi_t)$. After that, the joint probability of a biterm $b = (w_i, w_j)$ can be written as:

$$P(b) = \sum_t P(t)P(w_i|t)P(w_j|t) = \sum_t \theta_t \phi_{i|t} \phi_{j|t} \tag{1}$$

Thus the likelihood of the whole corpus is:

$$P(B) = \prod_{i,j} \sum_t \theta_t \phi_{i|t} \phi_{j|t} \tag{2}$$

Finally, the BTM generates the topic-document matrix that represents the topic distribution of the document, and the topic-word matrix that represents the distribution of the specified k topics in the text and distribution of the word under each topic. Specifically, in this work, each topic-document weight in topic-document matrix represent a weight of the edge between a document node and a topic node;

each topic-word weight in topic-word matrix represent a weight of the edge between a word node and a topic node.

3.2 Short-Text Graph Convolutional Networks

In order to classify the short text effectively, we construct a special short-text graph. The nodes in the short-text graph consist of documents, unique words, and topics. The edges between the graph nodes are built based on the word occurrence in documents (documents-word edges), word co-occurrence in the whole corpus (word-word edges), the documents-to-topic weight (documents-topic edges) learned by the topic model, and the weight of topic-word (word-topic edges) learned by the topic model. Specifically, we use the term frequency-inverse document frequency (TF-IDF) where the term frequency denotes the number of times the word appears in the document, and the inverse document frequency is the logarithmically scaled inverse fraction of the number of documents that contain the word as weights of the edge between a document node and a word node. We employ the point-wise mutual information (PMI), which is a popular measure for word associations, to calculate the weights between two word nodes.

In addition, in order to solve the problem of sparse features of a short text, the topic extracted by the topic model is used as a node in the short-text graph. We use the document-topic weight learned by the topic model as a weight of the edge between a document node and a topic node. Similarly, the word-topic weight learned by the topic model is used as a weight of the edge between a word node and a topic node. Formally, the weight of the edge between node i and node j is defined as:

$$A_{ij} = \begin{cases} PMI(i,j) & i, j \text{ are words} \\ TF\text{-}IDF_{ij} & i \text{ is document, } j \text{ is word} \\ word\text{-}topic_{ij} & i \text{ is word, } j \text{ is topic} \\ doc\text{-}topic_{ij} & i \text{ is document, } j \text{ is topic} \\ 1 & i = j \\ 0 & otherwise \end{cases} \tag{3}$$

After building the short-text graph, a GCN that allows message passing between nodes that are at maximum two steps away is used to learn and train the short-text graph. We set a feature matrix $X = I$ as an identity matrix, which means every node in the graph represented as a one-hot vector is the input to the short-text GCN. The experimental results of a random one-hot vector were better than those of the pre-trained existing word vectors, such as Word2vec or Glove. Specifically, for a one-layer GCN, a new d-dimensional node feature matrix is expressed as:

$$L^{(1)} = \rho(\tilde{A}XW_0) \tag{4}$$

where $\tilde{A}$ represents a normalized symmetric adjacency matrix A, and it is calculated by (3), $X \in \mathbb{R}^{n\times m}$ denotes a matrix containing all n nodes with their features and $W_0 \in \mathbb{R}$ is a weight matrix; ρ is an activation function e.g., the sigmoidal function.

$$\tilde{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \tag{5}$$

In (3), A denotes an adjacency matrix of text graph; we define our adjacency matrix of short text graph as shown in equation (3). $D_{ii} = \sum_j A_{ij}$ is degree matrix of A. When multiple GCN layers are stacked, the information about larger neighborhoods is integrated. Specifically, the node feature matrix of a multiple-layer GCN $L^{l+1} \in \mathbb{R}^{n\times k}$ is computed as:

$$L^{(l+1)} = \rho(\tilde{A}L^{(l)}W_l) \tag{6}$$

where l denotes the layer number and $L(0) = X$. In our experiment, two-layer GCN was used. The loss function is defined as the cross-entropy error over all labeled documents, which is given by:

$$\mathcal{L} = -\sum_{d\in y_D}\sum_{f=1}^{F} Y_{df} ln Z_{df} \tag{7}$$

where y_D denotes a set of document indices that have labels, and F is the dimension of the output feature, which is equal to the number of classes; Y is the label indicator matrix, and Z is the output matrix.

3.3 BERT Representation.

In order to improve the model performance further, we apply the word vectors generated by the state-of-the-art models, the Bidirectional Encoder Representations from Transformers (BERT) [9]. Specifically, the pre-trained BERT model is used to predict the text category, but the BERT results are not used as final text-classification results. The vector s' obtained by the BERT's hidden layer can represent the context-sensitive word embedding. The experimental result shows that combining word vectors generated by the BERT and the representation generated by the short-text GCN a much better classification performance was achieved than by using the BERT or short-text GCN alone.

3.4 Classifier

As for the final classifier, the classifier is determined by the specific application scenario, and in the experiments, we applied the BiLSTM [13] as a classifier because the BiLSTM has better performances than the other classification models such as CNN. Also, we concatenate word node representations $S = \{R(w_0)..R(w_i)..R(w_n)\}$ and BERT's word representations $S' = \{w'_0..w'_i..w'_n\}$, and fed them to the BiLSTM input; n denotes the text length. And then the document node representation $R(S_{seq})$ and the BiLSTM output vector S_{lstm} are fed together to the softmax layer to obtain the text category y. The proposed classifier is illustrated in Figure 2.

Training of the final classifier is performed using the cross-entropy loss, which is expressed as:

$$\mathcal{L} = CrossEntropy(y, y') \tag{8}$$

where y denotes the true label of short text, and y' represents the category predicted by our model.

4 Experiment Setup

4.1 Datasets

We conduct experiments on five text datasets, respectively: MR, Weibo, StackOverflow, Biomedical, and R8. They are described in the following.

MR. The MR dataset represents a movie review dataset for binary sentiment classification, where each review contains only one sentence [23]. The corpus has 5,331 positive and 5,331 negative reviews. We used the training/test split presented in [26].

Weibo. This dataset includes Chinese microblog data. The raw dataset represents a collection of messages posted in June 2014 on Weibo released by [11]. In the Weibo dataset, each message is labeled with a hashtag as its category, and there are 50 distinct hashtag labels in total.

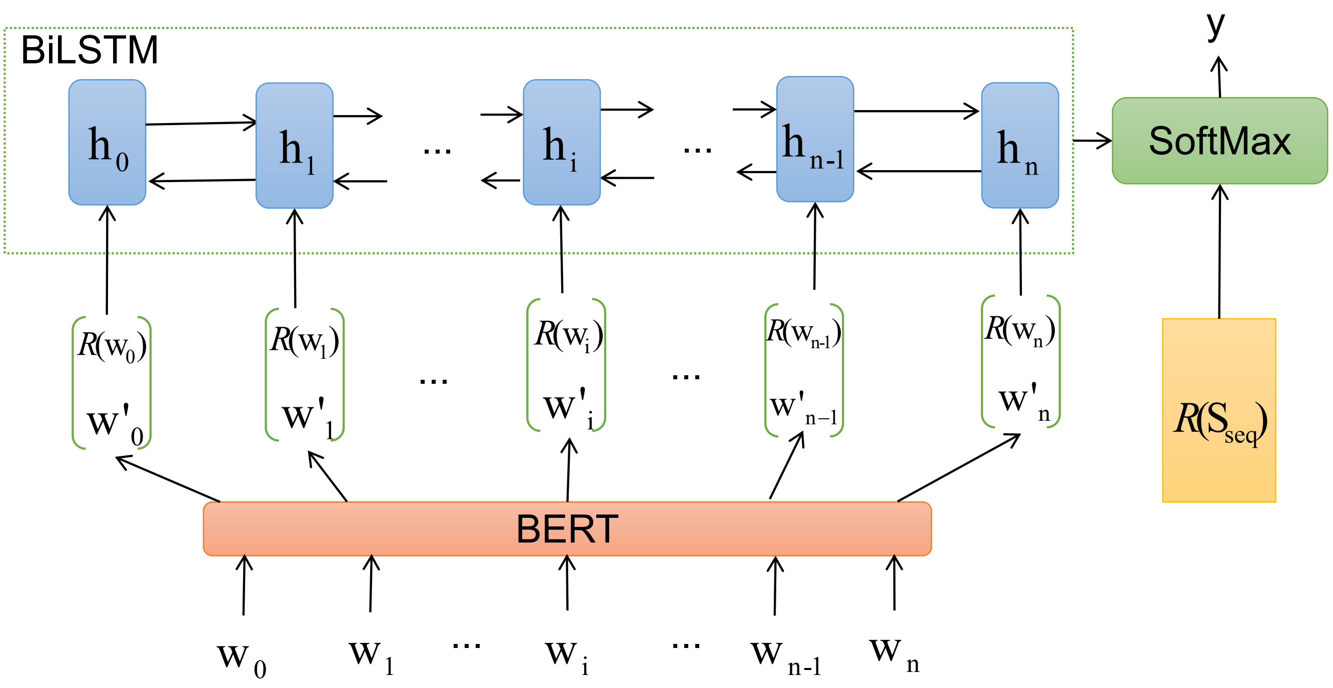


Figure 2. The overview of classifier. The pre-training BERT is used to generate the corresponding word representation features $S' = \{w'_0..w'_i..w'_n\}$. We combined the word node representations $S = \{R(w_0)..R(w_i)..R(w_n)\}$ and BERT's word representations and fed them to a BiLTSM. Finally, the ouput of BiLTSM and document node representation $R(S_{seq})$ are fed to a softmax layer to obtain the text category.

StackOverflow. This dataset denotes the competition data published by kaggle.com. The raw dataset included selected questions and the corresponding labels posted on stackoverflow.com from July 31, 2012, to August 14, 2012. It composed of 3,370,528 samples. Following [28], in the experiment, we randomly selected 20,000 question titles including 20 different tags, e.g., excel, svn, and ajax.

Biomedical. We use the challenge data related to biomedicine published in BioASQ's official website4, an internationally renowned biomedical platform. Following [28], in our experiment, we also randomly select 20, 000 paper titles from 20 different MeSH5 major topics,e.g., chemistry, cats, and lung.

R8. This dataset represents a subset of the Reuters 21578 dataset. This dataset included eight categories and was split into 5,485 training and 2,189 test documents.

Table 1. Experimental dataset characteristics.

Dataset	#Docs	#Classes	#Avg	Vocab
			len	size
MR	10,662	2	20	18,764
Weibo	35,000	50	7	10,220
StackOverflow	20,000	20	8	6,762
Biomedical	20,000	20	18	6,004
R8	7674	8	66	7,688

4.2 Experimental Methods

In the experiment, different models for short text classification were used, and their performances were compared. The models used in the experiment were as follows.

TF-IDF+LR: This is a classic baseline bag-of-words model that includes the weighting of the term frequency-inverse document frequency. Logistic Regression is used as a classifier.

CNN: We use convolutional neural network(CNN), a common model of deep learning, as our baseline. In our experiment, we used the pre-trained word embedding word2vec as a CNN input

Bi-LSTM: A bi-directional LSTM [19, 30] is commonly used in text classification. We input the pre-trained word embedding to the Bi-LSTM.

FastText: This is a simple and efficient text classification method [14], which treats the average of word/n-grams embeddings as document embeddings, then feeds document embeddings into a linear classifier. We evaluated it with bigrams.

Text GCN: Text GCN [31] which employed GCN to classify text is one of the state-of-the-art models for text classification.

Fine-tuning BERT: In the original Bidirectional Encoder Representations from Transformers (BERT) [9] we applied a small learning rate and our experimental data to fine-tune the BERT, and its output is considered as a text category.

STGCN: The short-text GCN (STGCN) was used alone to classify the text. Just like the setting of Text GCN [31], the document node representation generated by the short-text GCN was fed directly to the softmax layer to get the text category.

STGCN+BiLSTM: After obtaining the document and word nodes, they were inputted into a BiLSTM to get the text category. Specifically, we used the word representation a BiLSTM input, and then the document node representation and BiLSTM output vector were fed together to the softmax layer to get the text category.

STGCN+BERT+BiLSTM: After geting document node representations and word node representations from GCN and BERT's

Table 2. Classification accuracy of different models on different datasets.

Models	MR	Weibo	StackOverflow	Biomedical	R8
TF-IDF+LR	0.746	0.501	—	—	0.937
CNN	0.777	0.524	0.823	0.701	0.957
LSTM	0.751	0.514	0.821	0.702	0.937
Bi-LSTM	0.777	0.522	0.821	0.705	0.963
fastText	0.751	0.524	—	—	0.961
Text GCN	0.767	0.534	0.814	0.680	0.970
Fine-tuning BERT	0.857	0.522	0.908	0.726	0.982
STGCN	0.782	0.542	0.835	0.690	0.972
STGCN+BiLSTM	0.785	0.546	0.857	0.728	—
STGCN+BERT+BiLSTM	**0.864**	**0.550**	**0.914**	**0.740**	**0.985**

word representations, we concatenated word node representations and word representations obtained by the BERT and input them into the BiLSTM. And then, we input the document node representations and BiLSTM output vector together into softmax layer to get the final text category.

4.3 Model Settings

In the short-text GCN, the first layer included 200 neurons, and the window size was 20. The initial graph nodes were randomly initialized 200-dimensional vectors, and in the baseline models, the pre-trained word embedding was adapted; we used 300-dimensional gloVe word embeddings. All the words that appeared more than three times were used to form the vocabulary, and then 10% of training data was randomly selected as validation data. In the training process, the short-text GCN training included a maximum of 200 epochs; the adam optimizer [16] was used in the training process, and the learning rate and the dropout rate were set to 0.01 and 0.5, respectively.

The BiLSTM size was 256, and the adam optimizer [16] was used. The learning rate was 0.01. We trained BiLSTM model for 10 epochs. In the BERT model, we applied different pre-training models for different datasets. Specifically, for English MR, Biomedical, R8, and StackOverflow datasets, we adopted the pre-trained uncased BERT-Base model, while for Chinese Weibo dataset, we applied the pre-trained BERT-Base, Chinese model [6]. Finally, for BERT fine-tuning, we trained BERT for about 3 epochs using our data.

5 Experimental Results

The experimental results of our model on different datasets are shown in Table 2. According to the experimental results, we can draw the following conclusions.

Our model can effectively classify short texts.

As can be seen in Table 2, compared with the other models, our model achieved a significant improvement regarding the classification accuracy on the MR, Biomedical, StackOverflow, and Weibo datasets. The reason why the STGCN had worse performance on the R8 dataset than on the MR dataset is that the text length of the dataset R8 was relatively long and topic information was not very helpful to the text classification.

The shorter the text, the more obvious the performance improvement is.

As shown in Table 2, at a shorter average text length, the performance improvement of our model was more obvious, especially for the datasets with the average text length less than 10, such as Weibo and StackOverflow. When the average text length was relatively long, the improvement of our model was relatively small compared with the other models. For instance, on the R8 dataset with the average text length of 66, the improvement effect of our model was relatively small compared with that on the other datasets.

The topic information of short texts helps to build the short-text graph.

The experimental results of the Text GCN and STGCN show that the classification effect of a short text can be improved by using the topic information.

The full use of document node and word node representations can improve the text classification results.

The experimental results of STGCN and STGCN+BiLSTM on short text datasets prove that making full use of the document node and word node representations can improve the classification results; this is why our model achieved better classification performance; namely, it made full use of the document node and word node representations.

Adding the pre-training BERTs word vector can improve the model performance.

In Table 2, it can be seen that after adding the pre-trained BERTs word vector, the performance of our model was greatly improved compared with the STGCN and fine-tuning BERT. This shows that adding the pre-trained BERTs word vectors is effective and greatly improves the classification ability of our model.

5.1 Impact of topic model on classification performance.

In our experiment, we tried several topic models to extract topic information from the short text. We use the topic model LDA[3], which is one of the most widely studied topic models, to obtain the topic information on the short text as an extended feature of the short text. With the recent development of neural networks, there have been more and more researches on the neural topic models. Neural topic model not only can model better texts, but also can embed better into other neural networks, and can be well trained together with neural network models. In our experiments, we also employed neural topic models (NTM)[22] and the neural variational document model(NVDM) [5] to extract the topic information of the short texts.

The comparison results of the Text GCN, STGCN with LDA, STGCN with NTM, STGCN with NVDM and STGCN with BTM models on three datasets are given in Table 3. We observe that neural topic network NVDM and NTM achieved better results than the

[6] https://github.com/google-research/bert

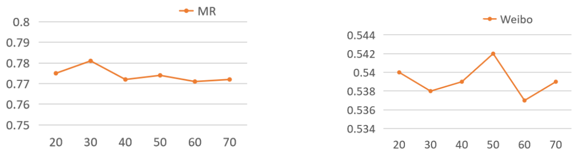
LDA, which implied the effectiveness of inducing the topic models by neural networks. In addition, the experimental results of the BTM were better than of the other models, which indicated that the BTM could extract the topic information on short texts more effectively

Table 3. Comparison of experimental results of different topic models on three datasets.

Model	MR	Weibo	StackOverflow
Text GCN	0.767	0.534	0.814
LDA	0.772	0.530	0.815
NTM	0.776	0.540	0.825
NVDM	0.773	0.541	0.822
BTM	0.782	0.542	0.835

5.2 Impact of topic number on classification performance.

The classification accuracy of our model at different K, the number of topics, on different experimental datasets is presented in Figure 3. In Figure 3, it can be observed that different numbers of topics led to different classification effects on different data sets. For instance, 30 topics provided the best effect on the MR dataset, and 50 topics provided the best effect on the Weibo dataset.



(a) MR

(b) Weibo

Figure 3. Experimental results on different datasets and at different topic numbers. The horizontal axis denotes the number of topics, and the vertical axis denotes the classification accuracy

6 Discussion and Future Work

In this work, the topic model is employed to extract the short text topic information, and a short-text graph is constructed by word co-occurrence and document word relations. Also, a graph convolutional neural network is used to construct and train the short-text graph. The word nodes and document nodes trained by the GCN and vector generated by the BERT's hidden layer are fed together to the BiLSTM classifier for short text classification. In the experiment, our model achieved state-of-the-art performance on different short text datasets. The experimental results show that our short-text graph can effectively model short text data, and combining the representation obtained by using the short-text GCN and the pre-trained word vector obtained by the BERT hidden layer can greatly improve the classification performance of our model.

However, our model consumes more memory and has longer training time than the other models used in the comparison. In our future work, we will explore how to simplify the proposed model while achieving the same classification effect.

7 Acknowledgements

This work was partially supported by the National Key Research and Development Program of China (No. 2018YFB1308604), National Natural Science Foundation of China (No.61672215, 61976086), Hunan Science and Technology Innovation Project (No. 2017XK2102), the Foundation of Guangdong Provincial Key Laboratory of Big Data Analysis and Processing (2017017, 201805), and the Research Project in the Data Center of Flamingo Network Co., Ltd.

REFERENCES

[1] Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Sima'an, 'Graph convolutional encoders for syntax-aware neural machine translation', *arXiv preprint arXiv:1704.04675*, (2017).

[2] 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., 'Relational inductive biases, deep learning, and graph networks', *arXiv preprint arXiv:1806.01261*, (2018).

[3] David M Blei, Andrew Y Ng, and Michael I Jordan, 'Latent dirichlet allocation', *Journal of machine Learning research*, **3**(Jan), 993–1022, (2003).

[4] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang, 'A comprehensive survey of graph embedding: Problems, techniques, and applications', *IEEE Transactions on Knowledge and Data Engineering*, **30**(9), 1616–1637, (2018).

[5] Dallas Card, Chenhao Tan, and Noah A Smith, 'A neural framework for generalized topic models', *stat*, **1050**, 25, (2017).

[6] Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang, 'Recurrent attention network on memory for aspect sentiment analysis', in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 452–461, (2017).

[7] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun, 'Very deep convolutional networks for text classification', *arXiv preprint arXiv:1606.01781*, (2016).

[8] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst, 'Convolutional neural networks on graphs with fast localized spectral filtering', in *Advances in neural information processing systems*, pp. 3844–3852, (2016).

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 'Bert: Pre-training of deep bidirectional transformers for language understanding', *arXiv preprint arXiv:1810.04805*, (2018).

[10] Cicero Dos Santos and Maira Gatti, 'Deep convolutional neural networks for sentiment analysis of short texts', in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 69–78, (2014).

[11] Yulan He, 'Extracting topical phrases from clinical documents', in *Thirtieth AAAI Conference on Artificial Intelligence*, (2016).

[12] Mikael Henaff, Joan Bruna, and Yann LeCun, 'Deep convolutional networks on graph-structured data', *arXiv preprint arXiv:1506.05163*, (2015).

[13] Sepp Hochreiter and Jürgen Schmidhuber, 'Long short-term memory', *Neural computation*, **9**(8), 1735–1780, (1997).

[14] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov, 'Bag of tricks for efficient text classification', *arXiv preprint arXiv:1607.01759*, (2016).

[15] Yoon Kim, 'Convolutional neural networks for sentence classification', *arXiv preprint arXiv:1408.5882*, (2014).

[16] Diederik P Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*, (2014).

[17] Thomas N Kipf and Max Welling, 'Semi-supervised classification with graph convolutional networks', *arXiv preprint arXiv:1609.02907*, (2016).

[18] Yifu Li, Ran Jin, and Yuan Luo, 'Classifying relations in clinical narratives using segment graph convolutional and recurrent neural networks (seg-gcrns)', *Journal of the American Medical Informatics Association*, **26**(3), 262–268, (2018).

[19] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang, 'Recurrent neural network for text classification with multi-task learning', *arXiv preprint arXiv:1605.05101*, (2016).

[20] Yuan Luo, ‘Recurrent neural networks for classifying relations in clinical notes’, *Journal of biomedical informatics*, **72**, 85–95, (2017).

[21] Diego Marcheggiani and Ivan Titov, ‘Encoding sentences with graph convolutional networks for semantic role labeling’, *arXiv preprint arXiv:1703.04826*, (2017).

[22] Yishu Miao, Edward Grefenstette, and Phil Blunsom, ‘Discovering discrete latent topics with neural variational inference’, in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2410–2419. JMLR. org, (2017).

[23] Bo Pang and Lillian Lee, ‘Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales’, in *Proceedings of the 43rd annual meeting on association for computational linguistics*, pp. 115–124. Association for Computational Linguistics, (2005).

[24] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi, ‘Learning to classify short and sparse text & web with hidden topics from large-scale data collections’, in *Proceedings of the 17th international conference on World Wide Web*, pp. 91–100. ACM, (2008).

[25] Kai Sheng Tai, Richard Socher, and Christopher D Manning, ‘Improved semantic representations from tree-structured long short-term memory networks’, *arXiv preprint arXiv:1503.00075*, (2015).

[26] Jian Tang, Meng Qu, and Qiaozhu Mei, ‘Pte: Predictive text embedding through large-scale heterogeneous text networks’, in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1165–1174. ACM, (2015).

[27] Sida Wang and Christopher D Manning, ‘Baselines and bigrams: Simple, good sentiment and topic classification’, in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pp. 90–94. Association for Computational Linguistics, (2012).

[28] Jiaming Xu, Bo Xu, Peng Wang, Suncong Zheng, Guanhua Tian, and Jun Zhao, ‘Self-taught convolutional neural networks for short text clustering’, *Neural Networks*, **88**, 22–31, (2017).

[29] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng, ‘A biterm topic model for short texts’, in *Proceedings of the 22nd international conference on World Wide Web*, pp. 1445–1456. ACM, (2013).

[30] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy, ‘Hierarchical attention networks for document classification’, in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1480–1489, (2016).

[31] Liang Yao, Chengsheng Mao, and Yuan Luo, ‘Graph convolutional networks for text classification’, in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 7370–7377, (2019).

[32] Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R Lyu, and Irwin King, ‘Topic memory networks for short text classification’, *arXiv preprint arXiv:1809.03664*, (2018).

[33] Xiang Zhang, Junbo Zhao, and Yann LeCun, ‘Character-level convolutional networks for text classification’, in *Advances in neural information processing systems*, pp. 649–657, (2015).

[34] Yue Zhang, Qi Liu, and Linfeng Song, ‘Sentence-state lstm for text representation’, *arXiv preprint arXiv:1805.02474*, (2018).