Text Level Graph Neural Network for Text Classification

一、现有方法的问题：

基于 GNN 的模型通常采用为整个语料库构建一个图的方式。

—》构建单个语料库级别图

1、语料库级别的图，使用具有固定权重的边，极大地限制了边的表达能力，所以必须使用较大的连接窗口才能获得全局表示。

2、很难进行在线测试，因为图的结构和参数取决于语料，并且在训练后无法修改。

二、本文的方法：具有全局参数共享的每个输入文本构建图形，而不是为整个语料库构建单个图形

1、为给定文本构建文本级别的图； 文本级别的图的所有参数均取自某些全局共享矩阵。

2、在这些图上引入消息传递机制，以从上下文中获取信息。

3、根据学习的表示来预测给定文本的标签。

三、模型

1、建图

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

N 和 E 是图的节点集和边集，而 N 中的单词表示和 E 中的边权重均取自全局共享矩阵。

p 表示连接到图中每个单词的相邻单词的数量。

此外，将训练集中少于 k 次的边缘均匀地映射到“公共”边，以使参数得到充分训练。

2、消息传递

从相邻节点收集信息，并根据其原始表示形式和所收集的信息更新其表示形式。

$$\mathbf{M_n} = \max_{a \in \mathcal{N}_n^p} e_{an} \mathbf{r_a}, \quad (3)$$

节点n从其邻居收到的消息

$$\mathbf{r'_n} = (1 - \eta_n)\mathbf{M_n} + \eta_n \mathbf{r_n} \quad (4)$$

max 是归约函数，它将每个维上的最大值组合起来以形成新的矢量作为输出

N_n^p 表示代表原始文本中 n 的最近 p 个单词的节点

$e_{an} \in R^1$ 是从节点 a 到节点 n 的边缘权重，可以在训练过程中进行更新

$r_n \in R^d$ 表示节点 n 的前一个表示

$\eta_n \in R^1$ 是节点 n 的可训练变量，它指示应保留 r_n 多少信息

r_n' 表示节点 n 的更新表示

四、结论

相比于传统 CNN、RNN 的优点：

图结构允许存在不同数量的邻居节点，这使单词节点可以通过不同的搭配学习更准确的表示形式。此外，单词之间的关系可以记录在边缘权重中并在全局范围内共享。

相比于 Graph-CNN 的优点：

1、小窗口

2、边的权重可训练，使每个文本的单词表达方式有所不同

Text Level Graph Neural Network for Text Classification

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang and Houfeng WANG
MOE Key Lab of Computational Linguistics, Peking University, Beijing, 100871, China
{hlz, madehong, lisujian, zxdcs, wanghf}@pku.edu.cn

Abstract

Recently, researches have explored the graph neural network (GNN) techniques on text classification, since GNN does well in handling complex structures and preserving global information. However, previous methods based on GNN are mainly faced with the practical problems of fixed corpus level graph structure which do not support online testing and high memory consumption. To tackle the problems, we propose a new GNN based model that builds graphs for each input text with global parameters sharing instead of a single graph for the whole corpus. This method removes the burden of dependence between an individual text and entire corpus which support online testing, but still preserve global information. Besides, we build graphs by much smaller windows in the text, which not only extract more local features but also significantly reduce the edge numbers as well as memory consumption. Experiments show that our model outperforms existing models on several text classification datasets even with consuming less memory.

该模型为具有全局参数共享的每个输入文本构建图形，而不是为整个语料库构建单个图形

1、这种方法消除了支持在线测试但仍保留全局信息的单个文本与整个语料库之间的依赖负担。
2、通过文本中较小的窗口来构建图形，这不仅提取了更多的局部特征，而且还显着减少了边数以及内存消耗。

1 Introduction

Text classification is a fundamental problem of natural language processing (NLP), which has lots of applications like SPAM detection, news filtering, and so on (Jindal and Liu, 2007; Aggarwal and Zhai, 2012). The essential step for text classification is text representation learning.

文本分类的基本步骤是文本表示学习

With the development of deep learning, neural networks like Convolutional Neural Networks (CNN) (Kim, 2014) and Recurrent Neural Networks (RNN) (Hochreiter and Schmidhuber, 1997) have been employed for text representation. Recently, a new kind of neural network named Graph Neural Network (GNN) has attracted wide attention (Battaglia et al., 2018). GNN was first proposed in (Scarselli et al., 2009) and has been used in many tasks in NLP including text classification (Defferrard et al., 2016), sequence labeling (Zhang et al., 2018a), neural machine translation (Bastings et al., 2017), and relational reasoning (Battaglia et al., 2016). Defferrard et al. (2016) first employed Graph Convolutional Neural Network (GCN) in text classification task and outperformed the traditional CNN models. Further, Yao et al. (2019) improved Defferrard et al. (2016)'s work by applying article nodes and weighted edges in the graph, and their model outperformed the state-of-the-art text classification methods.

However, these GNN-based models usually adopt the way of building one graph for the whole corpus, which causes the following problems in practice. First, high memory consumption is required due to numerous edges. Because this kind of methods build a single graph for the whole corpus and use edges with fixed weights, which considerably limits the expression ability of edges, they have to use a large connection window to get a global representation. Second, it is difficult for this kind of models to conduct the online test, because the structure and parameters of their graph are dependent on the corpus and cannot be modified after training.

现有方法的问题：
基于GNN的模型通常采用为整个语料库构建一个图的方式。
—》构建单个语料库级别图

1、语料库级别的图，使用具有固定权重的边，极大地限制了边的表达能力，所以必须使用较大的连接窗口才能获得全局表示。
2、很难进行在线测试，因为图的结构和参数取决于语料，并且在训练后无法修改。

To address the above problems, we propose a new GNN based method for text classification. Instead of building a single corpus level graph, we produce a text level graph for each input text. For a text level graph, we connect word nodes within a reasonably small window in the text rather than directly fully connect all the word nodes. The representations of the same nodes and weights of edges are shared globally and can be updated in the text level graphs through a massage passing mechanism, where a node takes in the information from neighboring nodes to update its representation. Finally, we summarize the representations of all the

为每个输入文本生成一个文本级别图

nodes in the graph to predict the results. With our design, text level graphs remove the burden of dependency between a single input text and the entire corpus, which support online test. Besides, it has the benefit of consuming less memory by connecting words in a small contextual window, because it excludes a good many words that are far away in the text and have little relation with the current word and thus significantly reduces the number of edges. The message passing mechanism makes nodes in the graph perceive information around them to get precise meaning in a specific context.

In our experiments, our method achieves state-of-the-art results in several text classification datasets and consumes significantly fewer memory resources compared with previous methods.

2 Method

1、为给定文本构建文本级图； 文本级图的所有参数均取自某些全局共享矩阵。
2、在这些图上引入消息传递机制，以从上下文中获取信息。
3、根据学习的表示来预测给定文本的标签。

In this section, we will introduce our method in detail. First, we show how to build a text level graph for a given text; all the parameters for the text level graph are taken from some global-sharing matrices. Then, we introduce the message passing mechanism on these graphs to obtain information from the context. Finally, we depict how to predict the label for a given text based on the learned representations. The overall architecture of our model is shown in Figure 1.

2.1 Building Text Graph

We notate a text with l words as $T = \{\mathbf{r_1}, ...\mathbf{r_i}, ..., \mathbf{r_l}\}$, where $\mathbf{r_i}$ denotes the representation of the i_{th} word. $\mathbf{r_i}$ is a vector initialized by d dimension word embedding and can be updated by training. To build a graph for a given text, we regard all the words that appeared in the text as the nodes of the graph. Each edge starts from a word in the text and ends with its adjacent words. Concretely, the graph of text T is defined as:

$$N = \{\mathbf{r_i} | i \in [1, l]\}, \tag{1}$$

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

N和E是图的节点集和边集，而N中的单词表示和E中的边权重均取自全局共享矩阵。
p表示连接到图中每个单词的相邻单词的数量。
此外，我们将训练集中少于k次的边缘均匀地映射到“公共”边缘，以使参数得到充分训练。

where N and E are the node set and edge set of the graph, and word representations in N and edge weights in E are taken from global shared matrices. p denotes the number of adjacent words connected to each word in the graph. Besides, we uniformly map the edges that occur less than k times in the training set to a "public" edge to make parameters adequately trained.

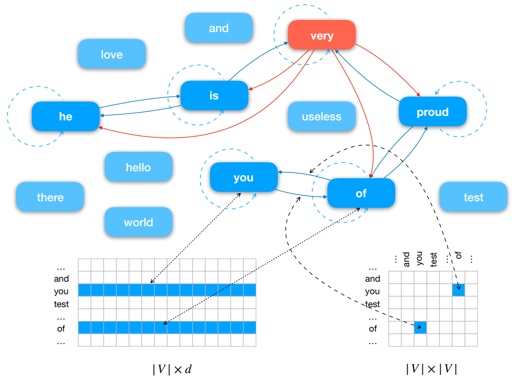

Figure 1: Structure of graph for a single text "he is very proud of you.". For the convenience of display, in this figure, we set $p = 2$ for the node "very" (nodes and edges are colored in red) and $p = 1$ for the other nodes(colored in blued). In actual situations, the value of p during a session is unique. All the parameters in the graph come from the global shared representation matrix, which is shown at the bottom of the figure.

Compared with the previous methods in building graph, our approach can exceedingly reduce the scale of the graph in terms of nodes and edges. That means that the text-level graph can consume much less GPU memory. Besides, their method is unfriendly to new-coming text, while our approach can solve this problem because the graph for each text is only dependent on its content.

2.2 Message Passing Mechanism

Convolution can extract information from local features (LeCun et al., 1989). In the graph domain, convolution is implemented by spectral approaches (Bruna et al., 2014; Henaff et al., 2015), or non-spectral approaches (Duvenaud et al., 2015). In this paper, a non-spectral method named message passing mechanism (MPM) (Gilmer et al., 2017) is employed for convolution. MPM first collects information from adjacent nodes and updates its representations based on its original representations and collected information, which is defined as:

MPM首先从相邻节点收集信息，并根据其原始表示形式和所收集的信息更新其表示形式

$$\mathbf{M_n} = \max_{a \in \mathcal{N}_n^p} e_{an}\mathbf{r_a}, \tag{3}$$

节点n从其邻居收到的消息

$$\mathbf{r'_n} = (1 - \eta_n)\mathbf{M_n} + \eta_n \mathbf{r_n} \tag{4}$$

where $\mathbf{M_n} \in \mathbb{R}^d$ is the messages that node n receives from its neighbors; max is a reduction function which combines the maximum values on each dimension to form a new vector as an output. $\mathcal{N}_n^p$ denotes nodes that represent the nearest p words of n in the original text; $e_{an} \in \mathbb{R}^1$ is the edge weight

max是归约函数，它将每个维上的最大值组合起来以形成新的矢量作为输出
Nn p表示代表原始文本中n的最近p个单词的节点

e an∈R 1是从节点a到节点n的边缘权重，可以在训练过程中进行更新
r n∈R d表示节点n的前一个表示
ηn∈R 1是节点n的可训练变量，它指示应保留r n多少信息
r n'表示节点n的更新表示

from node a to node n, and it can be updated during training; and $\mathbf{r_n} \in \mathbb{R}^d$ denotes the former representation of node n. $\eta_n \in \mathbb{R}^1$ is a trainable variable for node n that indicates how much information of $\mathbf{r_n}$ should be kept. $\mathbf{r'_n}$ denotes the updated representation of node n.

MPM makes the representations of nodes influenced by neighborhoods, which means the representations can bring the information from context. Therefore, even for polysemous words, the precise meaning in the context can be determined by the influence of weighted information from neighbors. Besides, the parameters of text level graphs are taken from global shared matrices, which means the representations can also bring global information as other graph-based models do.

Finally, the representations of all nodes in the text are used to predict the label of the text:

$$y_i = \text{softmax}(\text{Relu}(\mathbf{W} \sum_{n \in N_i} \mathbf{r'_n} + \mathbf{b})) \qquad (5)$$

where $W \in \mathbb{R}^{d \times c}$ is a matrix mapping the vector into an output space, N_i is the node set of text i and $\mathbf{b} \in \mathbb{R}^c$ is bias.

The goal of training is to minimize the cross-entropy loss between ground truth label and predicted label:

$$\textbf{loss} = -g_i \log y_i, \qquad (6)$$

where g_i is the "one-hot vector" of ground truth label.

3 Experiments

In this section, we describe our experimental setup and report our experimental results.

3.1 Experimental Setup

For experiments, we utilize datasets including R8, R52[1], and Ohsumed[2]. R8 and R52 are both the subsets of Reuters 21578 datasets. Ohsumed corpus is extracted from MEDLINE database. MEDLINE is designed for multi-label classification, we remove the text with two or more labels. For all the datasets above, we randomly select 10% text from the training set to build validation set. The overview of datasets is listed in Table 1.

We compare our method with the following baseline models. It is noted that the results of some models are directly taken from (Yao et al., 2019).

[1] https://www.cs.umb.edu/~smimarog/textmining/datasets/
[2] http://disi.unitn.it/moschitti/corpora.htm

Datasets	# Train	# Test	Categories	Avg. Length
R8	5485	2189	8	65.72
R52	6532	2568	52	69.82
Ohsumed	3357	4043	23	135.82

Table 1: Datasets overview.

- **CNN** Proposed by (Kim, 2014), perform convolution and max pooling operation on word embeddings to get representation of text.
- **LSTM** Defined in (Liu et al., 2016), use the last hidden state as the representation of the text. Bi-LSTM is a bi-directional LSTM.
- **fastText** Proposed by (Joulin et al., 2017), average word or n-gram embeddings as documents embeddings.
- **Graph-CNN** Operate convolution over word embedding similarity graphs by fourier filter, proposed by (Defferrard et al., 2016).
- **Text-GCN** A graph based text classification model proposed by (Yao et al., 2019), which builds a single large graph for whole corpus.

3.2 Implementation Details

We set the dimension of node representation as 300 and initialize with random vectors or Glove (Pennington et al., 2014). k discussed in Section 2.1 is set to 2. We use the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 10^{-3}, and L2 weight decay is set to 10^{-4}. Dropout with a keep probability of 0.5 is applied after the dense layer. The batch size of our model is 32. We stop training if the validation loss does not decrease for 10 consecutive epochs.

参数设置：
节点表示的维数为300，并使用随机向量或Glove进行初始化
k设置为2
使用Adam优化器
初始学习率为10 −3
L2权重衰减设置为10 −4
在dense层之后应用保持概率为0.5的dropout
模型的批量大小为32。如果验证损失连续10个周期没有减少，将停止训练。

For baseline models, we use default parameter settings as in their original papers or implementations. For models using pre-trained word embeddings, we used 300-dimensional GloVe word embeddings.

3.3 Experimental Results

Table 2 reports the results of our models against other baseline methods. We can see that our model can achieve the state-of-the-art result.

We note that the results of graph-based models are better than traditional models like CNN, LSTM, and fastTest. That is likely due to the characteristics of the graph structure. Graph structure

Model	R8	R52	Ohsumed
CNN	94.0 ± 0.5	85.3 ± 0.5	43.9 ± 1.0
LSTM	93.7 ± 0.8	85.6 ± 1.0	41.1 ± 1.0
Graph-CNN	97.0 ± 0.2	92.8 ± 0.2	63.9 ± 0.5
Text-GCN	97.1 ± 0.1	93.6 ± 0.2	68.4 ± 0.6
CNN*	95.7 ± 0.5	87.6 ± 0.5	58.4 ± 1.0
LSTM*	96.1 ± 0.2	90.5 ± 0.8	51.1 ± 1.5
Bi-LSTM*	96.3 ± 0.3	90.5 ± 0.9	49.3 ± 1.0
fastText*	96.1 ± 0.2	92.8 ± 0.1	57.7 ± 0.5
Text-GCN*	97.0 ± 0.1	93.7 ± 0.1	67.7 ± 0.3
Our Model*	**97.8 ± 0.2**	**94.6 ± 0.3**	**69.4 ± 0.6**

Table 2: Accuracy on several text classification datasets. Model with "*" means that all word vectors are initialized by Glove word embeddings. We run all models 10 times and report mean results.

在几个文本分类数据集上的准确性。 带“ *”的模型表示所有词向量都由Glove词embedding初始化。 运行所有模型10次，并报告平均结果。

相比于传统CNN、RNN的优点：图结构允许存在不同数量的邻居节点，这使单词节点可以通过不同的搭配学习更准确的表示形式。此外，单词之间的关系可以记录在边缘权重中并在全局范围内共享。

allows a different number of neighbor nodes to exist, which enables word nodes to learn more accurate representations through different collocations. Besides, the relationship between words can be recorded in the edge weights and shared globally. These are all impossible for traditional models.

相比于Graph-CNN的优点：
1、小窗口
2、边的权重可训练，使每个文本的单词表达方式有所不同

We also find that our model performs better than graph-based models like Graph-CNN. Graph-CNN represents documents using the bag-of-word model, which is similar to ours, but they connect word nodes within a large window without weighted edges, which cannot distinguish the importance between different words. While our model employed trainable edge weights, which let words express themselves differently when faced with various collocation. Besides, the weights are shared globally which means they can be trained by all the text contains the same collocation in the entire corpus.

We also note that our model performs better than former state-of-the-art model Text-GCN. That is likely due to more expressive edges, which have been discussed before, and the difference of representations learning. Text-GCN learns word representations by corpus level co-occurrence while our model is trained within a contextual window like traditional word embeddings. Therefore our model can benefit from pre-trained word embeddings and achieve better results.

3.4 Analysis of Memory Consumption

Table 3 reports the comparison of memory consumption and edges numbers between Text-GCN and our model. Results show that our model has a significant advantage in memory consumption.

Datasets	Text-GCN	Our Model
R8	9,979M(2,841,760)	954M(250,623)
R52	8,699M(3,574,162)	951M(316,669)
Ohsumed	13,510M(6,867,490)	1,167M(419,583)

Table 3: Comparison of memory consuming. The number of edges in the whole model is in parentheses.

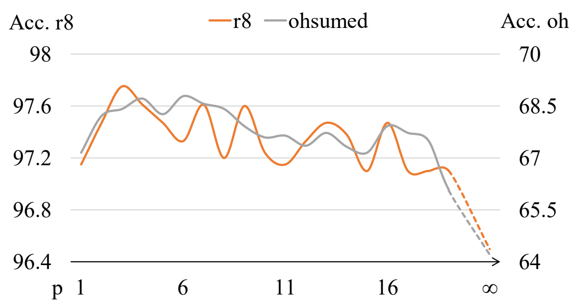

Figure 2: Model performance using p from 1 to 19 and "∞" (fully-connected). All hyperparameters are set the same except p. The left and right ordinate indicate the accuracy on the r8 and ohsumed dataset respectively.

As discussed in 2.1, the words in our model are only connected to adjacent words in the texts, while Text-GCN , which is based on the corpus level graph, connects nodes within a reasonably large window. Because Text-GCN uses co-occurrence information as fixed weights, it has to enlarge the window size to get a more accurate co-occurrence weight. Therefore, we will get a much more sparse edge weights matrix than Text-GCN. Also, since the representation of a text is calculated by the sum of the representations of word nodes in the text, there is no text node in our model, which also reduces memory consumption.

3.5 Analysis of Edges

To understand the difference of various connecting windows, we compared the performance of the R8 and ohsumed datasets with different p values, the result is reported in Figure 2. We find that the accuracy increases as p becomes larger and achieves the best performance when connected with about 3 neighborhoods. Then the accuracy decreases volatility as p increases. This suggests that when connected only with the nearest neighborhood, nodes cannot understand the dependencies that span multiple words in the context, while connected with neighborhoods far away (much larger p), the graphs become more and more similar with fully connected graphs which ignore the local features. In addition, the fewer edges, the

Setting	R8	R52	Ohsumed
Original	**97.8 ± 0.2**	**94.6 ± 0.3**	**69.4 ± 0.6**
(1)Fixed PMI Edges W.	97.7 ± 0.2	94.0 ± 0.2	67.6 ± 0.5
(2)Mean Reduction	97.7 ± 0.1	94.5 ± 0.3	62.6 ± 0.2
(3)Random Word Emb.	97.4 ± 0.2	93.7 ± 0.2	67.3 ± 0.5

Table 4: Results of ablation studies. We run all models for 5 times and give mean results.

fewer memory consumption. Our model has fewer edges compared with previous methods, and this also show the advantages of our proposed model.

3.6 Ablation Study

To further analyze our model, we perform ablation studies and Table 4 shows the results.

In (1), we fix the weights of edges and initialize them with point-wise mutual information (PMI), and the size of sliding windows is set to 20, which is the same as (Yao et al., 2019). Removing the trainable edges makes the model perform worse on all data sets, which demonstrates the effectiveness of trainable edges. In our opinion, the main reason is that trainable edges can better model the relations between words compared with fixed edges.

In (2), we change the max-reduction by mean-reduction. In the original model, the node gets its new representation from received messages by obtaining the maximum value alone each dimension. From Table 4, we can see that the $\max$ reduction can achieve better results. The node reduction function is similar to the pooling operation on CNN. Reduction by $\max$ highlights features that are highly discriminating and provides non-linearity, which helps to achieve better results.

In (3), we remove the pre-trained word embeddings from nodes and initialize all the nodes with random vectors. Compared with the original model, the performances are slightly decreased without pre-trained word embeddings. Therefore, we believe that the pre-trained word embeddings have a particular effect on improving the performance of our model.

4 Related Work

In this section, we will introduce the related works about GNN and text classification in detail.

4.1 Graph Neural Networks

Graph Neural Networks (GNN) has got extensive attention recently (Zhou et al., 2018; Zhang et al., 2018b; Wu et al., 2019). GNN can model non-Euclidean data, while traditional neural networks can only model regular grid data. While many tasks in reality such as knowledge graphs (Hamaguchi et al., 2017), social networks (Hamilton et al., 2017) and many other research areas (Khalil et al., 2017) are with data in the form of trees or graphs. So GNN are proposed (Scarselli et al., 2009) to apply deep learning techniques to data in graph domain.

4.2 Text Classification

Text classification is a classic problem of natural language processing and has a wide range of applications in reality. Traditional text classification like bag-of-words (Zhang et al., 2010), n-gram (Wang and Manning, 2012) and Topic Model (Wallach, 2006) mainly focus on feature engineering and algorithms. With the development of deep learning techniques, more and more deep learning models are applied for text classification. Kim (2014); Liu et al. (2016) applied CNN and RNN into text classification and achieved results which are much better than traditional models.

With the development of GNN, some graph-based classification models are gradually emerging (Hamilton et al., 2017; Veličković et al., 2017; Peng et al., 2018). Yao et al. (2019) proposed Text-GCN and achieved state-of-the-art results on several mainstream datasets. However, Text-GCN has the disadvantages of high memory consumption and lack of support online training. The model presents in this paper solves the mentioned problems in Text-GCN and achieves better results.

5 Conclusion

In this paper, we proposed a new graph based text classification model, which uses text level graphs instead of a single graph for the whole corpus. Experimental results show that our model achieves state-of-the-art performance and has a significant advantage in memory consumption.

Acknowledgments

Our work is supported by the National Key Research and Development Program of China under Grant No.2017YFB1002101 and National Natural Science Foundation of China under Grant No.61433015 and No.61572049. The corresponding author of this paper is Houfeng Wang.

References

Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer.

Joost Bastings, Ivan Titov, Wilker Aziz, Diego Marcheggiani, and Khalil Simaan. 2017. Graph convolutional encoders for syntax-aware neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1957–1967.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. 2016. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510.

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 preprint arXiv:1806.01261*.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. 2014. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*.

Michaël 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.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, pages 2224–2232.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1263–1272.

Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. 2017. Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. *arXiv preprint arXiv:1706.05674*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034.

Mikael Henaff, Joan Bruna, and Yann LeCun. 2015. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*.

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

Nitin Jindal and Bing Liu. 2007. Review spam detection. In *Proceedings of the 16th international conference on World Wide Web*, pages 1189–1190.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. 2017. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems*, pages 6348–6358.

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.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2873–2879.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, pages 1063–1072. International World Wide Web Conferences Steering Committee.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

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.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Hanna M Wallach. 2006. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM.

Sida Wang and Christopher D Manning. 2012. 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*, pages 90–94. Association for Computational Linguistics.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2019. A comprehensive survey on graph neural networks.

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.

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.

Yue Zhang, Qi Liu, and Linfeng Song. 2018a. Sentence-state lstm for text representation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 317–327.

Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2018b. Deep learning on graphs: A survey.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2018. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*.