Text Classification with Intra-layer and Inter-layer Graph Attention Networks

Anonymous ACL submission

Abstract

Text classification is a classical task in the field of natural language processing. Recently, graph neural networks (GNNs) have received considerable attention and made great breakthroughs on this task. However, current GNN-based methods neither fully utilize edge information nor obtain higher-order interactions of words. To address these problems, we propose the **I**ntra-layer and **I**nter-layer **G**raph **AT**tention networks (IIGAT) to obtain the higher-order interactions of word nodes and construct multi-dimensional edges between word nodes in the intra-layer GAT to enrich the semantic information of words. Extensive experiments on four benchmark datasets demonstrate the effectiveness of our methods on the text classification task.

1 Introduction

Text classification is one of the most fundamental tasks in natural language processing (NLP), focusing on analyzing the text corpus to get the correct label for the text. Text classification plays a vital role in many applications, such as sentiment classification (Tai et al., 2015), spam processing (Jindal and Liu, 2007) and question answering (Kalchbrenner et al., 2014).

The core of text classification is text representation learning, which focuses on obtaining a set of vector representations from text sequences. Inspired by the success of deep learning, the manual extraction of features in traditional models is gradually abandoned, and automatic means are widely used to learn text features. In particular, convolutional neural networks (CNNs) (Kim, 2014) and recurrent neural networks (RNNs) (Liu et al., 2016) leverage their powerful feature extraction capabilities to model consecutive words in texts. To overcome the computational cost associated with increased sentence length, transformers (Zhang and Zhang, 2020) compute the "influence" of each word in the text by parallelizing the input and achieve state-of-the-art results on applications with large-scale datasets.

However, the above methods ignore structural information between words in the text. In recent years, as graph neural networks (GNNs) (Scarselli et al., 2008) have gradually obtained more and more attention and made great progress in NLP, the research on text classification based on GNN has gained higher and higher popularity. GNN can build effective relational structures, maintain global structural information and successfully apply to graph-structured data. Yao et al. (2019) construct a text graph for the entire corpus based on word co-occurrence and document-word relations, jointly learn word and document embeddings, and finally classify the document nodes in the graph. Further, Huang et al. (2019) propose to build a single graph for each input document, and their model allows inductive learning to support online testing.

These graph-based methods have two main drawbacks. First, edge information is poorly utilized. The current GNN-based models employ the neighboring nodes of word nodes for aggregation (Kipf and Welling, 2016), and one-dimensional edge information (Hu et al., 2019) serves no complementary role in updating word embeddings. However, edge features in text graphs often contain much semantic information. Second, the number of layers of GNN is limited. On the one hand, GNN-based models generally build networks with only two layers (Zhang et al., 2020). In the shallow aggregation process, the central node tends to interact with its 1-hop neighbors and cannot obtain long-distance neighbor information. On the other hand, the traditional GNN-based models commonly generate over-smoothing after deepening the number of network layers (Li et al., 2019; Chen et al., 2020), resulting in features tending to be similar among nodes.

*Corresponding Author

To address these problems, we propose a new GNN model named Intra-layer and Inter-layer Graph Attention networks (IIGAT). We construct a single graph for each document, where nodes represent words, while edges represent multi-dimensional semantic relationships between words. We aggregate the node neighbor features in each layer through an intra-layer attention mechanism. The edge information is embedded into the nodes to update the word representations, which effectively uses edge features in the text graph. Meanwhile, we deepen the GNN layers to capture the higher-order interactions of words by adaptively selecting the effective node representations in each layer through an inter-layer attention mechanism.

In brief, our contributions are summarized as follows:

- We introduce multi-dimensional edge features to enhance the semantic information of the word nodes. The word representations are updated with edge and node information in each layer.
- We propose a model containing intra-layer and inter-layer attention mechanisms called IIGAT, in which word nodes can capture the interaction of higher-order words in a deep GNN.
- Extensive experiments indicate that our proposed IIGAT notably outperforms state-of-the-art text classification methods on several benchmark datasets.

2 Related work

2.1 Traditional Text Classification

Traditional text classification methods generally combine feature engineering and shallow classifiers. Text representation learning methods such as BOW (Zhang et al., 2010) and word2vec (Mikolov et al., 2013) feed the text embeddings obtained from feature engineering into a shallow classifier (Keller et al., 1985; Joachims, 1998; Breiman, 2001) for training. For similar purposes, the LightGBM (Ke et al., 2017) framework based on a decision tree algorithm (Brijain et al., 2014) has been proposed for a broad range of text classification tasks, showing extraordinary potential. However, traditional methods rely excessively on the empirical knowledge of experts and manual extraction of features, which consume massive resources and time. In contrast, our approach breaks this barrier by automatically learning the text representation in the raw datasets.

2.2 Deep Text classification

With the rapid development of deep learning, many deep learning models are used for text classification tasks. Based on its excellent performance on images (Krizhevsky et al., 2017), Kim (2014) applies convolution to text sequences and obtains the final vector representations by pooling operations. Liu et al. (2016) and Zhang et al. (2018) use memory units, learn historical information in training and capture long-term dependency types in text. Along with the proposed attention mechanism (Vaswani et al., 2017), different textual information is given different weights (Peters et al., 2018; You et al., 2019), highlighting the contribution of crucial information. Pre-trained models (Kenton and Toutanova, 2019; Croce et al., 2020; Zhang and Zhang, 2020) are introduced to learn global semantic information efficiently with unsupervised methods. Although the above approaches significantly improve the text classification task, they mainly model consecutive words and fail to consider the contextual information of text sequences. Our proposed model converts text sequences into graph structures, achieving semantic interactions among non-contiguous words.

2.3 Graph Neural Networks

GNNs (Scarselli et al., 2008) have been mainly dedicated to modeling non-Euclidean data. Research on applying GNN to text classification is divided into two main categories: transductive learning (Kipf and Welling, 2016) and inductive learning (Hamilton et al., 2017; Veličković et al., 2017). The former mainly learns a unique embedding for each word node based on the graph structure. The training requires the participation of all nodes in the graph, using unlabeled samples to solve the problem of insufficient training samples (Yao et al., 2019; Wang et al., 2021; Liu et al., 2020). The latter is mainly a way of dynamic modeling, where embedding is learned according to the change of node neighbors. When the graph structure in the test set changes, the corresponding node embedding will be adjusted (Huang et al., 2019; Zhang et al., 2020). However, both of these approaches use only two-layers GNN, which dramatically limits the perceptual field of neighbor aggregation (Wu et al., 2019; Chen et al., 2020) when labels are missing in the datasets or connections between nodes are sparse. In contrast, our approach deepens the layers of the GNN in the text classification task and

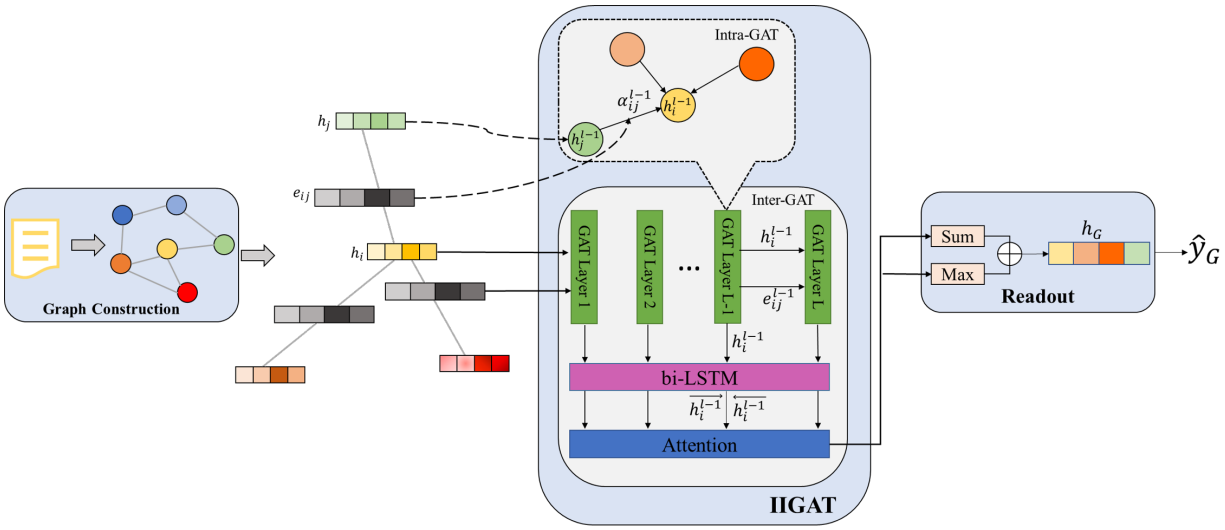


Figure 1: Illustration of the proposed IIGAT for text classification. We construct a graph for each document and input both node representations (e.g., h_i) and edge representations (e.g., e_{ij}) to IIGAT to obtain text representation for higher-order word aggregation.

effectively addresses the problem of higher-order interactions of nodes.

3 Method

In this section, we depict our method in three key components: the graph construction, the IIGAT and the readout function. The overall architecture is shown in Figure 1.

3.1 Graph Construction

We build a single graph for each document, where vertices correspond to the unique words in the document and the edges correspond to the co-occurrence relationships between words. Formally, the constructed graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of edges. The co-occurrence relationship describes the frequency of co-occurrence between words in a fixed sliding window of the document, and edges are undirected in the graph.

Let $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ represents the feature matrix with the features of n word nodes, where d denotes the dimensionality of the features. We use the indices in the subscripts to denote the elements of a matrix or a tensor. Hence, $\boldsymbol{x}_i$ represents the representation of the i^{th} node. Similarly, let $\boldsymbol{E} \in \mathbb{R}^{n \times n \times t}$ represents the edge features between word nodes (Gong and Cheng, 2019) and $\boldsymbol{e}_{ij} \in \mathbb{R}^t$ represents the t-dimensional feature vector of the edge connecting the i^{th} and j^{th} word nodes. Specifically, $\mathbf{e}_{ij} = 0$ means no edge between the i^{th} and j^{th} word nodes.

3.2 IIGAT

To perform node representation learning in deepened GNN, we propose a new model called IIGAT in this section. Unlike the traditional GNN model, IIGAT contains both intra-layer and inter-layer attention to capture higher-order contextual information of words and avoid the performance degradation problem caused by gradient vanishing.

Intra-layer GAT. To reflect the different importance of neighbors of word nodes, we update the feature representations of nodes and edges within each layer by GAT.

Given a node i, its neighboring nodes can be represented by $\mathcal{N}_i$. In the previous GAT, the attention vector depends only on the node features and does not consider the weight of edges. In contrast, our proposed attention mechanism combines multi-dimensional edge features, and the node representation is updated by both neighboring nodes and the corresponding edges jointly. For a specific layer l, the aggregation process is shown in Figure 2, and the specific definitions are as follows:

$$\hat{\alpha}_{ij}^{l-1} = f\left(a^T\left[\boldsymbol{W_1} h_i^{l-1} \,||\, \boldsymbol{W_1} h_j^{l-1} \,||\, \boldsymbol{W_2} e_{ij}^{l-1}\right]\right), \tag{1}$$

$$\alpha_{ij}^{l-1} = \frac{\exp\left(\hat{\alpha}_{ij}^{l-1}\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(\hat{\alpha}_{ik}^{l-1}\right)}, \tag{2}$$

where f is the LeakyReLU activation function, $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ are shared matrices that augment the fea-

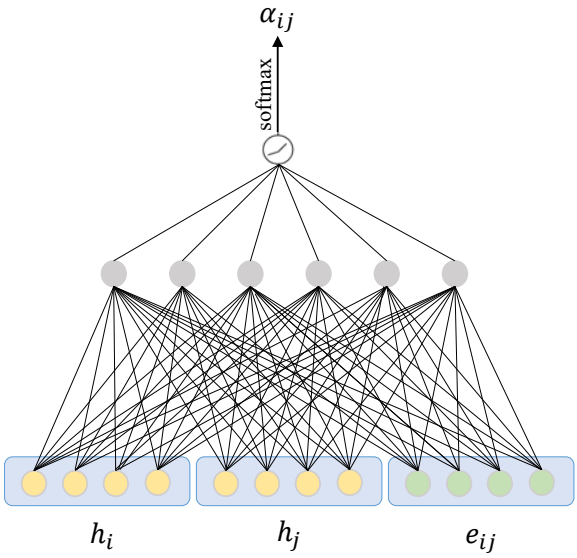


Figure 2: Illustration of the intra-layer GAT aggregation process.

ture dimension, and $||$ denotes a connection operation. The attention vector α_{ij}^{l-1} is normalized using the softmax function. In particular, the edge features are updated by a linear transformation whose parameters are the weight matrix $\boldsymbol{W^l}$:

$$e_{ij}^{l} = \boldsymbol{W^l} e_{ij}^{l-1}. \tag{3}$$

According to the obtained attention vector, GAT iteratively updates each node representation with the following formulas:

$$h_i^l = \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{l-1} \boldsymbol{W} h_j^{l-1}\right), \tag{4}$$

where h_i^l is the node representation obtained by updating node i at layer l, σ is the nonlinearity such as tanh, and $\boldsymbol{W}$ is an input transformation matrix.

As a result, the feature representation of each node can obtain more semantic information when jointly enhanced by neighboring nodes and edge features.

Inter-layer GAT. Since different-order neighbors of the central node produce different impacts in the text graph, we design an inter-layer attention mechanism to selectively aggregate node features of higher-order neighbors to avoid that deepening GAT produces the gradient vanishing problem.

Formally, we input the node features h_v^l resulting from the l-th layer to bi-directional LSTM (bi-LSTM) (Hochreiter and Schmidhuber, 1997) and obtain the forward hidden features $\overrightarrow{h_v^l}$ and backward hidden features $\overleftarrow{h_v^l}$, respectively. Bi-LSTM fully uses past and future information by flowing through the two-layer networks in forward and backward temporal order. Then, the most useful neighborhood range for each node v is determined by calculating the attention score for each layer of node features. Specifically, the calculation is as follows:

$$\hat{h}_v^l = \left[\overrightarrow{h_v^l} \,||\, \overleftarrow{h_v^l}\right], \tag{5}$$

$$\hat{\beta}_v^l = score\left(\hat{h}_v^l\right), \tag{6}$$

$$\beta_v^l = \text{softmax}\left(\hat{\beta}_v^l\right), \tag{7}$$

where $score$ is a learnable function that scores the node features of each layer and β_v^l denotes the weight of node features at each layer after normalization.

For the L layers GAT, the new node embedding h_v^{fianl} is updated as:

$$h_v^{final} = \textstyle\sum_{l=1}^{L} \beta_v^l \hat{h}_v^l, \tag{8}$$

where $\sum_{l=1}^{L}$ denotes the summation operation and h_v^{fianl} denotes the final word node representation obtained from the IIGAT model.

The proposed IIGAT model not only integrates multi-dimensional edge features to enrich the semantic information of nodes, but also effectively distinguishes high-order neighbors at different granularities, so that words can obtain sufficient knowledge to get the final text representation.

3.3 Readout Function

After the nodes aggregate all the higher-order neighbor information, we perform fusion learning on the final node representation to obtain the graph-level representation h_G of the document. We define the readout function as:

$$h_{\mathcal{G}} = \frac{1}{n}\sum_{v=1}^{n} h_v^{final} + \max_{v=1}^{n}\left(h_v^{final}\right). \tag{9}$$

We obtain the graph-level representation by averaging and maximizing together, which ensures that the keywords in the text can be useful.

The label is predicted by feeding the graph-level representation $h_{\mathcal{G}}$ into a softmax layer:

$$\hat{y} = \text{softmax}\left(\boldsymbol{W} h_{\mathcal{G}} + b\right), \tag{10}$$

where $\boldsymbol{W}$ is a trainable parameter matrix mapping the graph-level representation into an output space and b is a bias.

Finally, we use cross-entropy loss as the loss function of our text classification model. It is denoted as follows:

$$Loss = -\sum_{i \in \mathcal{D}} \sum_{i}^{C} y_i log(\hat{y}_i), \qquad (11)$$

where $\mathcal{D}$ is the set of document for training, C is the number of labels and y_i is the one-hot ground truth label of document.

4 Experiments

This section describes the experimental details and reports the experimental results. First, we introduce the datasets used for the experiments and the baseline models used for comparison.

4.1 Datasets

We performed extensive experiments on four benchmark datasets for text classification, including Ohsumed, R8, R52 and Movie Review (MR). Specifically, the ohsumed dataset is a subset of the MEDLINE dataset and consists of 7400 medical documents divided into 23 categories of cardiovascular disease labels, with each document having one or more of these labels. R8 and R52 are two subsets of Reuters. R8 contains 7674 documents divided into 8 categories; R52 contains 9100 documents divided into 52 categories. MR is a movie review dataset containing 10,662 documents with two categories. Table 1 describes the statistical data of the used datasets and more detailed information can be found in (Yao et al., 2019).

Since we only considered single-label classification task, we first removed documents with two or more labels before the above datasets were trained. Then, we preprocessed the texts by cleaning them, mainly by removing the stop words defined in NLTK (Loper and Bird, 2002) and the low–frequency words (less than 5 times) in the above datasets (Rousseau et al., 2015; Blanco and Lioma, 2012). Since the text length of the MR dataset was too short, we had only performed word tokenization on it.

4.2 Baselines

We used the following kinds of models as baseline models for comparison with our model:

- **TF-IDF+LR** is a bag-of-words model that uses term frequency-inverse document frequency to calculate weights and then uses a logistic regression algorithm as a classifier.
- **TextCNN** (Kim, 2014) performs sentence-level classification tasks on pre-trained word vectors using CNNs.
- **LSTM** (Liu et al., 2016) proposes a RNN, which applies to text classification with pre-trained word embeddings.
- **RCNN** (Lai et al., 2015) applies a bidirectional recurrent structure in the model to capture the contextual information of words, and then adds a maximum pooling operation to automatically determine which words play a key role in text classification.
- **Graph-CNN** (Defferrard et al., 2016) introduces a graph CNN model, which operates convolutions over word embedding similarity graphs with the Chebyshev filter.
- **TextGCN** (Yao et al., 2019) suggests building a graph for the entire corpus and then learning word representations through graph convolutional networks.
- **TextLevel-GNN** (Huang et al., 2019) constructs a single graph for each sentence, sharing global parameters in GNN learning.
- **TensorGCN** (Liu et al., 2020) builds text graph tensor to describe semantic, syntactic and sequential contextual information.
- **HyperGAT** (Ding et al., 2020) puts forward the concept of hypergraphs to obtain higher expressiveness with less computation.
- **DADGNN** (Liu et al., 2021) presents to deepen GNN by decoupling two key stages (representation transformation and propagation).
- **NMGC** (Lei et al., 2021) designs a multi-hop neighbor information fusion strategy to aggregate different neighbor features from 1 hop to k hops.

4.3 Experimental Settings

For the datasets introduced above, we divided documents into a training set and a test set and randomly divide the data in the training set for training and validation with a ratio of 9:1, where the data in the validation set facilitates the adjustment of hyperparameters in the model. We utilized Adam optimizer with an initial learning rate of 10^{-3}, and L_2 weight decay is set to 10^{-4}. We set the dropout rate for

Table 1: The statistics of datasets.

Dataset	# Docs	# Train	# Test	# Words	# Categories	Avg.Len
MR	10,662	7,108	3,554	18,764	2	20.39
R8	7,674	5,485	2,189	7,688	8	65.72
R52	9,100	6,532	2,568	8,892	52	69.82
Ohsumed	7,400	3,357	4,043	14,157	23	135.82

Table 2: Test accuracy (%) comparison of different text classification models. The best results are bolded.

Model	MR	R8	R52	Ohsumed
TF-IDF + LR	74.59 ± 0.00	93.74 ± 0.00	86.95 ± 0.00	54.66 ± 0.00
TextCNN	77.75 ± 0.72	95.71 ± 0.52	87.59 ± 0.48	58.44 ± 1.06
LSTM	77.33 ± 0.89	96.09 ± 0.19	90.48 ± 0.86	51.10 ± 1.50
RCNN	77.68 ± 0.86	96.31 ± 0.33	90.54 ± 0.91	49.27 ± 1.07
Graph-CNN	76.65 ± 0.63	95.32 ± 0.26	92.94 ± 0.24	63.12 ± 0.55
TextGCN	76.74 ± 0.20	97.07 ± 0.10	93.56 ± 0.18	68.36 ± 0.56
TextLevel-GNN	75.47 ± 0.36	97.80 ± 0.20	94.60 ± 0.30	69.40 ± 0.60
TensorGCN	77.91± 0.07	98.04 ± 0.08	95.05 ± 0.11	70.11 ± 0.24
HyperGAT	78.32± 0.27	97.97 ± 0.23	94.98 ± 0.27	69.90 ± 0.34
DADGNN	78.64± 0.29	97.31 ± 0.09	94.35 ± 0.06	—
NMGC	76.21± 0.25	98.15 ± 0.16	95.16 ± 0.22	69.21 ± 0.17
IIGAT (ours)	**80.56 ± 0.32**	**98.29 ± 0.15**	**95.88 ± 0.26**	**71.56 ± 0.25**

the training process as 0.5 and the epoch number as 400. The output dimension of the linear mapping W was fixed to 64 for all hidden layers. For baseline models, we used default parameter setups as in their published papers. In addition, after the text was cleaned, we initialized the word features using 300-dimensional pre-trained GloVe word embeddings (Pennington et al., 2014) and pre-processed the edge features using random initialization.

4.4 Test Performance

We first perform comprehensive experiments to evaluate the performance of the baselines and our model, and the results are shown in Table 2 in the form of accuracy rates. It is not difficult to observe from the results that our model outperforms all baselines on four datasets, demonstrating superior capabilities in text classification applications. We can analyze and explore this result in depth below:

- **Traditional Models**. The TF-IDF+LR model integrates all words in the corpus in a bag-of-words format, ignoring grammatical and sequential elements. In datasets like Ohsumed with long text and an excessive number of words, contextual information cannot be captured, resulting in abysmal model performance.

- **Deep Learning Models**. Our model also outperforms deep learning models based on CNNs and RNNs. This is since CNN-based models use convolutional operations to model the text locally and fail to obtain global information about the text. Meanwhile, the RNN-based model captures the long-term dependency of words, but brings many parameters to be computed, leading to overfitting. Our graph-based approach allows word nodes to learn more accurate word representations based on the different collocations of neighboring nodes.

- **Graph-based Models**. Early graph-based models such as TextGCN and TextLevel-GNN use co-occurrence relations of words to extract semantic information in text sequences to construct text graphs. The graph-based model is significantly higher than the other models in the Ohsumed dataset, indicating the critical role of semantic information in text classification. However, since the MR dataset aims to address the sentiment classification task, the sequence-based model shows

Table 3: Test classification accuracy of models in inter-layer GAT with attention (w/ A) and without attention (w/o A) by varying the number of model layers.

Model	MR	R8	R52	Ohsumed
Inter-layer GAT (2 layers) w/o A	79.32	97.60	94.30	69.20
Inter-layer GAT (2 layers) w/ A	78.96	97.65	94.34	69.25
Inter-layer GAT (4 layers) w/o A	77.84	96.76	92.09	68.98
Inter-layer GAT (4 layers) w/ A	79.82	97.85	94.56	70.86
Inter-layer GAT (6 layers) w/o A	77.79	96.50	91.89	68.32
Inter-layer GAT (6 layers) w/ A	**80.06**	**98.29**	**95.88**	**71.12**

Table 4: Comparison of classification accuracy with different edge weights.

Model	MR	R8	R52	Ohsumed
(1) 0/1 edge	75.64	90.12	87.35	62.25
(2) PPMI edge w/o T	76.35	96.25	93.32	67.62
(3) PPMI edge w/ T	77.24	68.61	97.13	94.05
Original	**80.56**	**98.29**	**95.88**	**71.23**

a more robust classification performance than the text graph that considers only semantic information. Later, HyperGAT and TensorGCN proposed the concepts of hypergraph and graph tensor, respectively, to incorporate sequence, semantic and syntactic information into text graphs with appropriate aggregation strategies. Although they fail to consider higher-order interactions of words, the results show that they improve the performance of graph-based models in sentiment classification. Recently, DADGNN and NMGC effectively handle the word higher-order interactions in the text by solving the overfitting problem in graph convolutional networks. However, their models are limited by the characteristics of graph convolutional networks and cannot tackle inductive tasks. In summary, the above results reveal that our model can show more stable and better performance in all graph-based models.

4.5 Ablation Study

In this subsection, we performed ablation studies to explore the role of different modules in the IIGAT model.

4.5.1 The multi-dimensional edge features in Intra-layer GAT

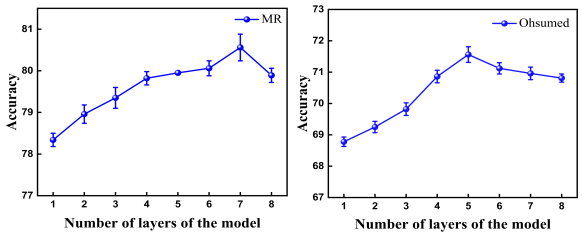
We verify the effectiveness of multi-dimensional edge features in the intra-layer graph attention network and Table 4 shows the results.

In case (1), we do not attribute any semantic information to the edges in the text graph and simply give attention to the existence of connections between words using 0 or 1 weights (the existence of a connection is assigned a value of 1). It can be seen that the model classification performance deteriorates when the edge weights only represent the connection information because it fails to reflect the semantic relationship between words.

In case (2) and (3), we both use positive pointwise mutual information (PPMI) to calculate the weights between two words. The difference is whether they are trainable (w/o T denotes not trainable, w/ T denotes trainable). From the results, the fixed edge weights do not perform as well as trainable edges in the model, indicating the effectiveness of updating the edge weights in model training. However, neither of them is as effective as the multi-dimensional edge features in our model. We conclude that multi-dimensional edge features can carry more semantic information and be better integrated into word embeddings in combination with word features.

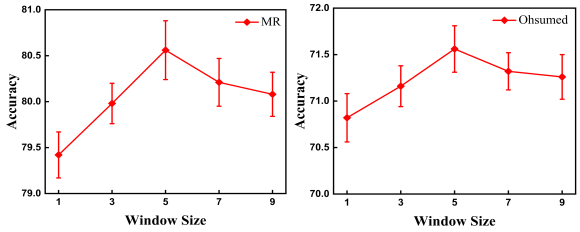
4.5.2 The Attention Mechanism in Inter-layer GAT

As can be seen from the Table 3, we discuss the effect of the attention mechanism in the inter-layer GAT on the model performance. The results show that the inter-layer attention mechanism has little improvement on the model performance when the model is stacked with two layers. However, when the model stacks more than two layers, eliminating the inter-layer attention mechanism gradually decreases model performance as the number of network layers deepens. The main reason is that if the deep GNN cannot effectively distinguish the importance of nodes in each layer, the final node representation aggregates the nodes in all layers



(a) MR (b) Ohsumed

Figure 3: Test accuracy (%) with different layers on the MR dataset and Ohsumed dataset.



(a) MR (b) Ohsumed

Figure 4: Test accuracy (%) with different sliding window sizes on the MR dataset and Ohsumed dataset.

with equal status. The deeper the networks are, the more frequently the nodes interact. The representation obtained for each node in the text graph may tend to be consistent, damaging the classification accuracy. Moreover, the data can show that when nodes aggregate 4-hop neighbors or 6-hops, the final node representation can adaptively select node features in each layer through an inter-layer attention mechanism, demonstrating the improvement of model performance by higher-order word interactions.

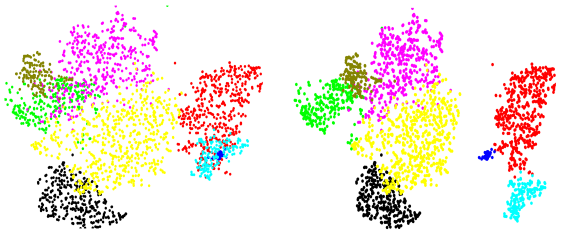
4.6 Parameter Sensitivity

Figure 3 shows the performance of our model after deepening the number of network layers. The stacking of network layers allows the semantic information of the central node to be propagated indirectly from more neighbors than just the 1-hop neighbors. We found that the experimental results of the model gradually improved as the number of network layers increased. The long text dataset Ohsumed achieves the best performance when the network is stacked with five layers, compared to the short text dataset MR where the performance degrades when the network is stacked with seven layers. It illustrates that the short text needs to aggregate more higher-order neighbors to enrich the semantic information of words due to the sparsely available labels.

Figure 4 presents the test accuracies of the IIGAT with different sliding window sizes on the MR dataset and Ohsumed dataset. Sliding window size affects the density of neighboring nodes. A sliding window that is too small could not yield useful co-occurrence information between words, while a sliding window that is too large may produce interfering data that impacts word embedding. From the results, it can be seen that the model performance is optimal for both MR dataset and Ohsumed dataset when the sliding window size is 5.

4.7 Document Visualization

To intuitively probe the effectiveness of our model in deep GNNs, we use the t-SNE tool (Van der Maaten and Hinton, 2008) to visualize the document embedding for comparison. Figure 5 shows the visualization results of IIGAT stacking with two and six layers on the R8 dataset. We observe that IIGAT stacks with six layers can learn more distinguishable document embeddings than stacks with two layers. It proves that our model can enhance document embeddings by allowing words to gain more semantic information.



(a) IIGAT, two layers (b) IIGAT, six layers

Figure 5: The t-SNE visualization of document embeddings in R8.

5 Conclusion

In this paper, we propose a novel text classification method based on graph neural networks called IIGAT. It constructs a single graph for each document and adds multi-dimensional edge features between word nodes that are embedded in the word representations during aggregation. In addition, we capture the higher-order interactions of words by combining intra-layer and inter-layer attention mechanisms. The results of the extensive experiments demonstrate that the proposed model is superior to the state-of-the-art methods.

References

Roi Blanco and Christina Lioma. 2012. Graph-based term weighting for information retrieval. *Information retrieval*, 15(1):54–92.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Mr Brijain, R Patel, MR Kushik, and K Rana. 2014. A survey on decision tree algorithm for classification.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, pages 1725–1735. PMLR.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. Gan-bert: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 2114–2119.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Kaize Ding, Jianling Wang, Jundong Li, Dingcheng Li, and Huan Liu. 2020. Be more with less: Hypergraph attention networks for inductive text classification. *arXiv preprint arXiv:2011.00387*.

Liyu Gong and Qiang Cheng. 2019. Exploiting edge features for graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9211–9219.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

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

Linmei Hu, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4821–4830.

Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2019. Text level graph neural network for text classification. *arXiv preprint arXiv:1910.02356*.

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

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

N Kalchbrenner, E Grefenstette, and Philip Blunsom. 2014. A convolutional neural network for modelling sentences. In *52nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

James M Keller, Michael R Gray, and James A Givens. 1985. A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, (4):580–585.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Fangyuan Lei, Xun Liu, Zhengming Li, Qingyun Dai, and Senhong Wang. 2021. Multihop neighbor information fusion graph convolutional network for text classification. *Mathematical Problems in Engineering*, 2021.

Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. 2019. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9267–9276.

P. Liu, X. Qiu, and X. Huang. 2016. Recurrent neural network for text classification with multi-task learning. *AAAI Press*.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8409–8416.

Yonghao Liu, Renchu Guan, Fausto Giunchiglia, Yanchun Liang, and Xiaoyue Feng. 2021. Deep attention diffusion graph neural networks for text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8142–8152.

Edward Loper and Steven Bird. 2002. NLTK: the natural language toolkit. *CoRR*, cs.CL/0205028.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Jeffrey Pennington, Richard Socher, and Christopher D 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.

Matthew Peters, M. Neumann, M. Iyyer, M. Gardner, and L. Zettlemoyer. 2018. Deep contextualized word representations.

François Rousseau, Emmanouil Kiagias, and Michalis Vazirgiannis. 2015. Text categorization as a graph classification problem. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1702–1712.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80.

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

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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

Xiao Wang, Nian Liu, Hui Han, and Chuan Shi. 2021. Self-supervised heterogeneous graph neural network with co-contrastive learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1726–1736.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR.

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.

Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 32.

Haopeng Zhang and Jiawei Zhang. 2020. Text graph transformer for document classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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):43–52.

Yue Zhang, Qi Liu, and Linfeng Song. 2018. Sentence-state lstm for text representation. In *ACL (1)*.

Yufeng Zhang, Xueli Yu, Zeyu Cui, Shu Wu, Zhongzhen Wen, and Liang Wang. 2020. Every document owns its structure: Inductive text classification via graph neural networks. *arXiv preprint arXiv:2004.13826*.