Article

RB-GAT: A Text Classification Model Based on RoBERTa-BiGRU with Graph ATtention Network

Shaoqing Lv *, Jungang Dong , Chichi Wang , Xuanhong Wang , Zhiqiang Bao

Posted Date: 25 April 2024

doi: 10.20944/preprints202404.1579.v1

Keywords: word embedding; RoBERTa; BiGRU; text classification; multi-head GAT

Article

RB-GAT: A Text Classification Model Based on RoBERTa-BiGRU with Graph Attention Network

Shaoqing Lv [1,2,*], Jungang Dong [1], Chichi Wang [1], Xuanhong Wang [1,2] and Zhiqiang Bao [1,2]

[1] School of Communication and Information Engineering, Xi'an University of Posts and Telecommunications, Xi'an, China; djg03192012@163.com (J.D.); dwangcc@163.com (C.W.); wxh@xupt.edu.cn (X.W.); baozhiqiang@xupt.edu.cn (Z.B.)

[2] Shaanxi Key Laboratory of Information Communication Network and Security, Xi'an University of Posts and Telecommunications, Xi'an, China

* Correspondence: Lvsq3601@xupt.edu.cn

Abstract: With the development of deep learning, several Graph Neural Networks (GNN)-based approaches have been utilized for text classification. However, GNNs encounter challenges in capturing contextual text information within a document sequence. To address this, a novel text classification model RB-GAT is proposed by combining RoBERTa-BiGRU embedding and a multi-head Graph ATtention Network (GAT). First, the pre-trained RoBERTa model is exploited to learn word and text embeddings in different contexts. Second, the Bidirectional Gated Recurrent Unit (BiGRU) is employed to capture long-term dependencies and bidirectional sentence information from the text context. Next, the multi-head graph attention network is applied to analyze this information, which serves as a node feature for the document. Finally, the classification results are generated through a Softmax layer. Experimental results on three benchmark datasets demonstrate that our method can achieve an accuracy of 71.48%, 98.45%, and 80.32% on Ohsumed, R8, and MR, which is superior to the existing nine text classification approaches.

Keywords: word embedding; RoBERTa; BiGRU; text classification; multi-head GAT

1. Introduction

As a crucial task in the realm of text mining, text classification [1] seeks to organize and summarize textual data by categorizing them into predefined groups, finding applications in diverse fields such as spam detection [2], sentiment analysis [3], and news classification [4]. Traditional approaches to text classification involve the amalgamation of feature engineering with shallow classification models [5], including K-nearest neighbors [6], Naive Bayes [7], and support vector machines [8]. However, these methods often require intricate feature engineering and fail to adequately account for the sequential structure of text data, thus impeding the model's capacity to comprehend semantic relationships between words. In recent years, the advent of sequence deep learning models [9] has heralded a paradigm shift in text classification. These models eliminate the reliance on manual feature design and rule-based systems, autonomously extracting rich semantic information from vast amounts of text data. Consequently, contemporary research in text classification predominantly revolves around data-driven deep learning models, prioritizing the extraction of meaningful patterns and representations directly from the data, which has demonstrated its effectiveness in processing sequential and grid-structured text data. The most widely adopted sequence deep learning techniques include the Convolutional Neural Network (CNN) [10] and the Recurrent Neural Network (RNN) [11].

However, when faced with text graphs containing multi-level relationships among documents, paragraphs, and words, encompassing diverse and rich information, sequence deep learning models may fall short and struggle to handle graph-structured data effectively [12]. On the other hand, Graph Neural Networks (GNNs) [13] demonstrate an excellent ability to process graph-structured data and capture intricate relationships between entities, finding wide applications in recommendation systems [14], drug discovery [15], and protein design [16]. Recently, GNNs have gained attention in text classification tasks, such as sentiment analysis [17] and document classification [18], due to their significance in capturing complex relationships within unstructured text graph data. By representing text data as a graph structure, GNNs can learn the representations of nodes and edges in the text graph, thereby modeling the relationships among the text data. However, these methods also exhibit certain limitations. For instance, existing models typically use one-hot encoding to initialize node features, resulting in high-dimensional and sparse feature matrices, potentially failing to effectively express text similarity. Furthermore, current methods overlook word relationships within the context of document sequences, neglect fine-grained word interactions, and struggle to handle emotional text data. As the field advances, addressing these challenges will be crucial to further enhancing the capabilities of GNNs for text classification tasks.

Meanwhile, several pre-trained word embedding models have emerged to improve the understanding of text relationships, such as BERT (Bidirectional Encoder Representations from Transformers) [19] and RoBERTa (Robustly Optimized BERT Approach) [20]. RoBERTa stands as an optimized iteration of the BERT language model, enhancing pre-training through the utilization of larger mini-batches and expanded datasets. Notably, RoBERTa adjusts key hyperparameters from BERT, which involves eliminating BERT's next-sentence pretraining objective and training with significantly larger mini-batches and learning rates. Thanks to its dynamic and contextualized representations, RoBERTa offers advantages in word embedding by capturing nuanced semantic meanings that adapt to surrounding text contexts, resulting in superior performance across various natural language processing tasks. Numerous methods have been introduced to handle text data and grasp both forward and backward dependencies within sequences, such as BiLSTM and BiGRU. BiGRU, a variant of the Recurrent Neural Network (RNN) architecture, aims to enhance the model's ability to extract information from past and future states in a sequence. The "bidirectional" characteristic of BiGRU signifies its capability to process data in both forward and backward directions. This enables it to comprehend the context from both preceding and succeeding points, effectively addressing long-term dependencies within the sequence. Such capability proves particularly valuable in text classification tasks, where the interpretation of words often hinges on both preceding and succeeding words.

Inspired by recent advancements and aiming to tackle the challenges in text classification tasks leveraging Graph Neural Networks (GNNs), this paper introduces a novel model called **R**oBERTa-**B**iGRU with **G**raph **AT**tention network (RB-GAT). RB-GAT initially constructs a heterogeneous graph of text words, utilizing techniques like TFIDF. Subsequently, the RoBERTa-BiGRU architecture is employed to generate embedding representations for both texts and words, effectively preserving long-term dependencies and bidirectional information within sentences. Finally, a multi-head two-layer Graph Attention Network (GAT) is employed to train the constructed heterogeneous graph of text words, refining the corresponding text and word representations.

RB-GAT integrates the graph attention mechanism to assess the significance of adjacent nodes in the message-passing process. This mechanism calculates weight values, allows the model to assign varying weights to different neighbor nodes, and captures intricate relationships between nodes. Moreover, the model incorporates a multi-head attention mechanism that operates based on a user-defined number of heads, which enables the model to extract text data information from diverse perspectives, enhancing the stability and robustness of the network.

The contributions of this paper are listed as follows:

Integrating Graph Neural Networks with RoBERTa-BiGRU Architecture: The RB-GAT model leads the way in combining GNNs with the RoBERTa-BiGRU framework, leveraging the distinct strengths of each to effectively navigate the intricacies of text classification. This pioneering approach enables the model to discern and explore the intricate relationships among words and documents within text graphs, as well as the sequential dependencies inherent in sentences. As a result, it offers a comprehensive understanding of textual data.

Utilization of a Multi-Head Graph Attention Network (GAT): RB-GAT elevates text and word representation refinement beyond conventional methods by embracing a multi-head, two-layer GAT.

This technique endows the model with a multi-head attention mechanism, allowing for a nuanced evaluation of the importance of neighboring nodes through variable weight assignments. This capability not only captures the subtle nuances of relationships but also significantly enhances the interpretability of the model.

Overcoming Prevalent Constraints in Text Classification Models: RB-GAT systematically tackles critical limitations present in existing text classification methodologies. Notably, it departs from the simplistic approach of one-hot encoding for initializing node features and rectifies the oversight of complex word interactions within document sequences. These improvements substantially enhance GNNs' ability to analyze and interpret emotionally and contextually rich text data, representing a significant advancement in text classification research.

2. Related Work

2.1. Sequence Deep Learning Based Text Classification Methods

Since 2010, sequence deep learning based text classification methods have steadily supplanted traditional machine learning techniques. Numerous effective models within the realm of sequence deep learning have been proposed and widely embraced for a myriad of text classification tasks. A notable example is the Recurrent Neural Network (RNN) [11], which treats words as temporal segments, capturing word relationships by calculating their similarities. RNNs acquire dependencies between words by memorizing preceding text and learning from subsequent text. However, RNNs encounter challenges, such as gradient explosion and vanishing gradients, attributed to long-term memory. To address these issues, the Long Short-term Memory Network (LSTM) [21] emerged as a variant of RNN, incorporating memory selection. LSTM mitigates problems like gradient explosion and vanishing gradients by utilizing forget, input, and output gates to regulate the flow of information in and out of its units. Researchers have extended LSTM for text sentiment classification. Xu et al. [22] proposed a bidirectional LSTM (BiLSTM) model, processing text sequences in both forward and backward directions using multiple LSTM networks to glean word context information. Additionally, Tai et al. [23] developed the Tree-LSTM model, extending LSTM to tree-structured networks, effectively acquiring rich semantic representations, and showcasing favorable performance in sentiment classification and sentence relationship prediction tasks. Furthermore, variants of LSTM, such as the Simple Recurrent Unit (SRU) [24], have been proposed. SRU mirrors LSTM in terms of model design but boasts faster computation speed for classification tasks, approximately 5 to 9 times swifter than LSTM, while achieving comparable or even superior classification performance. These models have sparked a revolution in text classification, enabling more precise and efficient analysis of textual data. However, these methods are limited in understanding global context because they encode only a limited amount of historical information at each timestep.

Unlike recurrent neural network models, which treat text data as time series data, Convolutional Neural Network (CNN) models take on a grid-like representation for text classification tasks [25]. While recurrent neural network (RNN) models demonstrate proficiency in understanding the semantic content within lengthy texts, CNN models specialize in identifying local features within the text, such as keywords or specific topics conveying particular emotions. Grefenstette et al. [26] introduced the Dynamic Convolutional Neural Network (DCNN) for text classification, utilizing temporal convolutions to capture variable-length phrases. Subsequently, Kim et al. [10] proposed a simpler CNN-based text classification model, employing a single convolutional layer on word vectors obtained from a neural language model. Liu et al. [27] extended the Kim-CNN model architecture by introducing a hidden layer to reduce the dimensionality of the vector representation. They also incorporated the maximum pooling method to obtain a more compact document representation, thereby enhancing the model's performance. Furthermore, Zhang et al. [28] proposed a character-level CNN model for text classification tasks that uses fixed-size characters as input and employs CNNs for feature extraction, achieving favorable results in various text classification scenarios. These CNN-based models provide alternative approaches to text classification by leveraging the grid-like structure of text data, demonstrating effectiveness in capturing local features and achieving competitive performance across various text classification tasks. However, the primary limitation of CNN-based models lies in their fixed context window size, which may hinder the capture of longer dependency patterns present in the text.

2.2. Graph Neural Network Based Text Classification Methods

Graph Neural Networks (GNNs) are specialized neural network models crafted for processing graph-structured data, capable of representing a variety of data types such as social networks and molecular structures. These models leverage the relationships between nodes in a graph to glean more meaningful data representations. The updating of node representations within a graph is facilitated through a messaging mechanism, wherein each node aggregates information from its neighbors and adjusts its vector representation accordingly. These resulting representations find application in diverse tasks, including node classification and graph classification. In the context of text classification tasks, GNNs typically construct a text graph structure to capture intricate relationships between nodes and effectively represent the text data. For instance, Yao et al. [29] introduced TextGCN, a text graph convolutional network based on Graph Convolutional Networks (GCN). TextGCN treats the text as a static heterogeneous graph, aggregating and updating node features using the GCN model. This approach demonstrates excellent performance on classified datasets, such as topics and news. Liu et al. [30] presented TensorGCN, a tensor graph convolutional network tailored for text classification tasks. The model constructs three text graph structures representing semantics, syntax, and word order, respectively. GCN is harnessed to learn from these structures, resulting in remarkable text classification performance. Defferrard et al. [31] introduced the Graph-CNN model, a significant advancement in adapting CNNs to process graph-structured data efficiently through spectral-based convolutional operations. To address the complexity and computational demands of traditional GCN models, Wu et al. [32] proposed a simplified approach known as the Simplified Graph Convolutional (SGC) network. This method eliminates nonlinearities between graph convolution layers and consolidates weight matrices between layers. SGC has demonstrated comparable or even superior performance on various benchmark datasets for text classification. Hu et al. [33] proposed a heterogeneous graph attention network, named HGAT, based on the Graph Attention Network (GAT). The HGAT model incorporates node-level and type-level two-tier attention mechanisms, proving effective for short text classification. Furthermore, Zhang et al. [34] introduced the TextING model to address the limitations of existing GNN models in capturing relationships between contextual words in documents and completing inductive learning of new words. In summary, GNNs offer a potent framework for processing graph-structured data and have been successfully applied to various text classification tasks, allowing for the modeling of rich relationships within textual data. However, these models primarily concentrate on extracting and utilizing the information embedded within the graph structure of the text. They often overlook effectively harnessing the semantic content of the text, especially the profound semantic insights available from various pre-trained models.

3. Method

3.1. Model Architecture

To achieve enhanced text classification performance, we propose a text classification model based on RoBERTa-BiGRU word embedding with a multi-head graph attention network (RB-GAT). As illustrated in Figure 1, the RB-GAT model comprises three main components: text graph construction, RoBERTa-BiGRU embedding, and the multi-head graph attention network. The first step involves constructing a document-word graph structure to capture contextual relationships within the text. Next, RoBERTa-BiGRU word embedding is utilized to obtain word embeddings with contextual information from the entire text. Subsequently, a multi-head GAT model is trained using the word embedding and the documents-word graph. This allows for a nuanced and comprehensive analysis of the graph data, enabling the model to focus on the most relevant parts of the text for classification. Finally, the processed features are passed through a Softmax layer for text classification, enabling the model to predict the most likely category for the given text. In the following sections, we will elaborate on each component in detail.

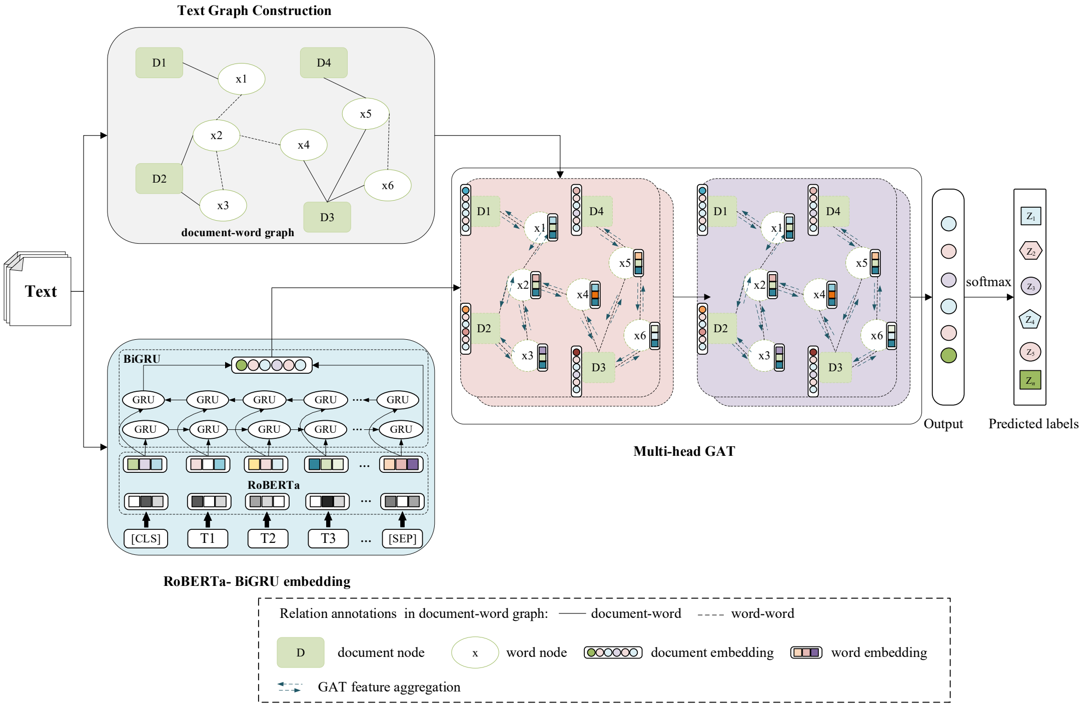


Figure 1. The overall framework of the RB-GAT model.

3.2. Text Graph Construction

To capture the intricate relationships and dependencies between words and documents, we constructed the text graph following a methodology akin to that employed in TextGCN [29]. We formed a heterogeneous graph G that encompasses both word and document nodes. Subsequently, edges were established between words and documents. Specifically, an edge was created if a word appeared in a document. The weight of this edge was typically determined by the term frequency-inverse document frequency (TF-IDF) of the word in that document. Additionally, edges were added between word nodes based on their co-occurrence within a specific context window within the corpus. The weight of these edges could be set as the point-wise mutual information (PMI) of the word pair, and if the PMI was less than 0, it was set to 0.

More specifically, the term frequency-inverse document frequency is calculated using the formula:

$$\mathrm{TF}-\mathrm{IDF}(w,d)=\mathrm{TF}(w,\,d)\times\log\left(\frac{N}{\mathrm{DF}(w)}\right) \tag{1}$$

where $\mathrm{TF}(w,\,d)$ is the frequency of word w in document d, N is the total number of documents, and $\mathrm{DF}(w)$ is the number of documents containing word w.

The point-wise mutual information is calculated using:

$$\mathrm{PMI}\left(w_i,w_j\right)=\log\left(\frac{p\left(w_i,w_j\right)}{p\left(w_i\right)p\left(w_j\right)}\right) \tag{2}$$

where $p(w_i,\,w_j)$ is the probability of words w_i and w_j co-occurring within a context window, and $p(w_i)$and $p(w_j)$ are the probabilities of words w_i and w_j occurring in the corpus.

Formally, the weight of edge between node i and node j in graph G is defined as:

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

3.3. RoBERTa-BiGRU Embedding

To better capture the intricate relationships between texts, our model leverages RoBERTa and BiGRU to obtain word embedding vectors. RoBERTa, like many advanced natural language processing (NLP) models, requires tokenization as a fundamental preprocessing step for text processing. Tokenization standardizes the format of input text, ensuring consistent processing across various text data. We employ Byte-Pair Encoding (BPE) for tokenizing the input text into tokens T_i, a common practice in Transformer-based models. Special tokens such as [CLS] for marking the beginning of text and [SEP] for segment separation are added to aid RoBERTa in understanding text boundaries and structure. Each token from the preprocessing step is mapped to a unique vector in a high-dimensional space. This mapping is achieved by passing the tokens through RoBERTa's embedding layer, which retrieves the initial embedding for each token. RoBERTa employs multiple layers of Transformer blocks to process these initial embeddings. Each block applies self-attention mechanisms, enabling the model to evaluate the significance of other tokens within the same text when representing a specific token. A given token's embedding can be extracted directly from a specific layer within the model or by aggregating embeddings across multiple layers to capture a richer representation. This results in each token's embedding encapsulating not only its own semantic information but also contextual nuances derived from the entire text.

These word embeddings $\mathbf{X} = \{x_1, x_2, \ldots, x_n\}$ for each document are then input into the BiGRU model. The BiGRU architecture effectively captures bidirectional contextual information by leveraging the strengths of Gated Recurrent Units (GRUs) to address issues related to long-term dependencies in sequential data. BiGRU integrates two GRU networks processing the sequence in opposite directions: one forward GRU that captures forward context (from the beginning to the end of a sequence) and one backward GRU that captures backward context (from the end to the beginning of a sequence). The final contextual embedding of the i-th text $\overrightarrow{h_i}$ is generated by concatenating its corresponding forward h_i^F and backward hidden states h_i^B

$$\overrightarrow{h_i} = [h_i^F \,||\, h_i^B] \tag{4}$$

where || denotes concatenation.

The output of the BiGRU model comprises bidirectional contextual embeddings for each word in the input sequence. These embeddings are enriched with comprehensive contextual information, rendering them well-suited for downstream text classification tasks.

3.4. Multi-Head GAT Model

We apply GAT to process the document-word graph G and node attribute $\boldsymbol{h} = \{\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_1}\}$. Each node's features $\overrightarrow{h_i}$ are linearly transformed using a shared weight matrix $\mathbf{W}$. The attention coefficients between each pair of nodes are computed to determine how much focus should be given to neighboring nodes' features. The attention coefficient e_{ij} from node i to node j in G is calculated as follows:

$$e_{ij} = \text{LeakyReLU}\left(\vec{a}\left[\mathbf{W}\overrightarrow{h_i} \,||\, \mathbf{W}\overrightarrow{h_j}\right]\right) \tag{5}$$

where $\vec{a}$ is a learnable weight vector, LeakyReLU is a non-linear activation function and || denotes concatenation.

The raw attention coefficients are normalized using the softmax function to make coefficients easily comparable across different nodes:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k\in\mathcal{N}_i}\exp(e_{ik})} \tag{6}$$

where N_i denotes the neighbors of node *i*, including *i* itself if self-loops are added.

The node features are updated by aggregating neighbors' features weighted by the normalized attention coefficients:

$$\overrightarrow{h_i'} = \sigma\left(\sum_{j\in\mathcal{N}_i} \boldsymbol{\alpha_{ij}}\mathbf{W}\overrightarrow{h_j}\right) \tag{7}$$

where σ denotes the ReLU non-linear activation function.

To stabilize the learning process of the model, we employ a multi-headed attention mechanism. The hidden states of the nodes are calculated using *K* independent attention mechanisms through Equation (7), and then the *K* outputs are concatenated together as the input of the next layer.

$$\overrightarrow{h_i'} = ||_{k=1}^{K}\sigma\left(\sum_{j\in\mathcal{N}_i} \boldsymbol{\alpha_{ij}}^{\boldsymbol{k}}\mathbf{W}^{\boldsymbol{k}}\overrightarrow{h_j}\right) \tag{8}$$

The process is repeated for the second layer, taking $\overrightarrow{h_i'}$ (the output of the first layer) as the input. This involves applying a new weight matrix and recalculating the attention coefficients and aggregated features to produce the final output features for each node $\overrightarrow{h_i''}$. The embeddings of nodes (words/documents) in the second layer have the same dimensionality as the label set, and predictions are obtained by applying softmax:

$$z_i = \frac{e^{\overrightarrow{h_i''}}}{\sum_{k=1}^{C} e^{\overrightarrow{h_k''}}} \tag{9}$$

The objective of training is to minimize the cross-entropy loss between the true and predicted labels. The loss function is defined as:

$$loss = -\sum_{i=1}^{N} y_i \log z_i \tag{10}$$

where *N* is the number of documents, and y_i is the actual label of the document *i*, and z_i is the predicted label of the document.

4. Experiments

This section offers a detailed insight into the experimental methodology, starting with a description of the datasets employed. It then proceeds to elaborate on the experimental procedures and the evaluation criteria utilized in the study. Subsequently, a comprehensive analysis of the results obtained from the comparative assessment of various models is presented.

4.1. Datasets

We ran our experiments on three widely used text classification benchmark datasets: Ohsumed, R8, and MR [29]. Specific parameters are shown in Table 1.

Table 1. Datasets.

Datasets	#Docs	#Training set	#Test set	#Classes
Ohsumed	7,400	3,357	4,043	23
R8	7,674	5,485	2,189	8
MR	10,662	7,108	3,554	2

The Ohsumed dataset, http://disi.unitn.it/moschitti/corpora.htm: It is sourced from the MEDLINE database, which is a significant bibliographic database of medical literature maintained by the National Library of Medicine. It consists of 7,400 single-label documents, evenly distributed across 23 different disease categories. For this study, 3,357 documents are used as the training set and 4,043 documents as the test set.

The R8 dataset, https://www.cs.umb.edu/~smimarog/textmining/datasets/: It is a subset of the Reuters dataset. It comprises 7,674 documents, equally divided into eight different categories. In this paper, 5,485 documents are utilized as the training set and 2,189 documents as the test set.

The MR dataset, http://www.cs.cornell.edu/people/pabo/movie-review-data/: It is a collection of movie reviews designed for binary emotion classification. It consists of 10,662 documents, with 5,331 being positive reviews and 5,331 being negative reviews. In our experiments, 7,108 documents of this dataset are employed as the training set and 3,554 documents as the test set.

4.2. Implementation Details

This study adopts Python 3.7 as the programming language and utilizes Torch 1.10.2 as the deep learning framework. The GPU employed is the NVIDIA GeForce RTX 2080ti. A heterogeneous graph of the original text is constructed following the approach described in TextGCN [29]. The text sequence length is set to 128, with any exceeding length truncated. In accordance with the strategy delineated in the proposal [29], a random subset comprising 10% of the documents from the training dataset was selected to establish the validation set. For model training, a batch size of 32 is utilized, and the training is performed for 100 iterations. The loss function employed is cross-entropy, and the optimizer used is Adam [35]. The learning rate for RoBERTa and BiGRU is set to 0.00001, the GAT module adopts a learning rate of 0.001, and a dropout rate of 0.5 is applied.

4.3. Experimental Metrics

In our framework, we employ accuracy and macro F1 as the primary metrics for evaluation. For each document category $\mathbf{Y}_i$, the methodologies for computing accuracy and F1 score are detailed in Equations (11)–(14).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

$$Macro\ F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{14}$$

where TP represents the count of instances within class $\mathbf{Y}_i$ accurately classified as $\mathbf{Y}_i$, whereas FP denotes the instances from classes distinct from $\mathbf{Y}_i$ yet classified as $\mathbf{Y}_i$. Conversely, TN refers to instances not belonging to $\mathbf{Y}_i$ and correctly classified into categories other than $\mathbf{Y}_i$, and FN embodies the instances of $\mathbf{Y}_i$ that have been erroneously classified into classes other than $\mathbf{Y}_i$.

4.4. Experimental Results and Analysis

4.4.1. Accuracy of Different Algorithms

In the comparative experiment, three main types of models were employed: 1) word embedding-based models, such as PV-DBOW and FastText; 2) sequence deep learning-based models utilizing CNN and BiLSTM; 3) Graph neural network-based models, primarily including TextGCN, TensorGCN, Graph-CNN, SGC, and TextING. An extensive experimental evaluation was conducted using benchmark datasets.

The results outlined in Table 2 illustrate that our proposed RB-GAT model outperforms all baseline methods, including word embedding-based models, sequence deep learning-based models, and graph-based approaches. Specifically, word embedding-based models like FastText and PV-DBOW, which rely on word embeddings to capture semantic similarities, serve as fundamental approaches. While effective to some extent, our analysis indicates their limitations in fully capturing the complexities and contextual nuances present in diverse datasets, as evidenced by their performance. Sequence deep learning models such as CNN and BiLSTM, which leverage the sequential nature of text, demonstrate improved performance, particularly in capturing context and long-range dependencies within text sequences. These findings underscore the significance of sequence modeling in text classification tasks.

Table 2. Test accuracy (%) comparison with baselines on benchmark datasets.

Model	Ohsumed	R8	MR
FastText	57.70	96.13	75.14
PV-DBOW	46.65	85.87	61.09
CNN	58.44	95.71	77.75
BiLSTM	49.27	96.31	77.68
TextGCN	68.36	97.07	76.74
SGC	68.53	97.23	75.91
Graph-CNN	63.86	96.99	77.22
TextING	70.42	98.04	79.82
TensorGCN	70.11	98.04	77.91
RB-GAT	71.48	98.45	80.32

Graph-based models, including our RB-GAT, represent the forefront of leveraging relational information embedded in text. These models exhibit superior performance, with RB-GAT achieving unprecedented accuracy rates: 71.48% on Ohsumed, 98.45% on R8, and 80.32% on MR. This performance not only demonstrates RB-GAT's effectiveness in diverse linguistic contexts but also highlights its innovative integration of GAT's attention mechanism with the powerful language understanding capabilities of RoBERTa-BiGRU. RB-GAT's leading edge can be attributed to its dual capacity to effectively model relational data through GAT and to deeply understand textual nuances via RoBERTa-BiGRU. This synergy enables RB-GAT to outperform both traditional models and contemporary graph-based approaches, positioning it as a significant advancement in text classification research.

Furthermore, as detailed in Table 3, RB-GAT outperforms all considered models across the three datasets, with Macro F1 scores of 67.90% on Ohsumed, 94.84% on R8, and 76.17% on MR. This superior performance underscores the algorithm's ability to effectively leverage the syntactic and semantic relationships within texts, attributed to the sophisticated integration of GAT's attention mechanism with the contextual understanding capabilities of RoBERTa-BiGRU. Specifically, the Ohsumed dataset, with its medical terminologies and complex relationships, was notably well-handled by RB-GAT, suggesting its potential in domains requiring deep contextual understanding. On the R8 and MR datasets, RB-GAT's performance further validates its generalizability and efficacy in capturing diverse textual phenomena.

Table 3. Test Micro F1 Score (%) comparison with baselines on benchmark datasets.

Model	Ohsumed	R8	MR
FastText	54.88	90.64	76.22
PV-DBOW	43.07	81.31	57.81
CNN	53.16	88.76	75.60
BiLSTM	48.66	88.55	75.26
TextGCN	61.45	92.88	75.58
SGC	65.34	93.50	71.90

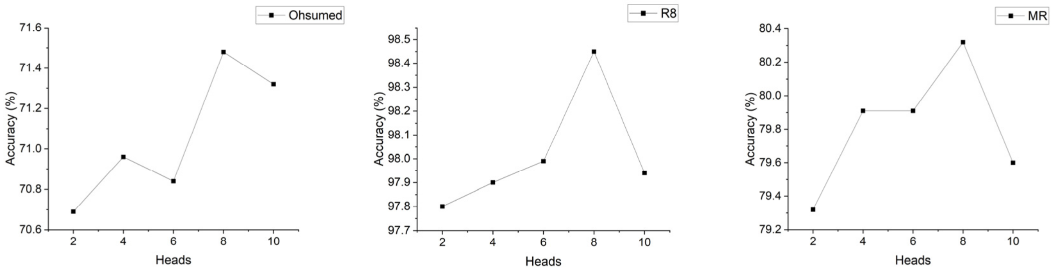
doi:10.20944/preprints202404.1579.v1

Graph-CNN	59.49	92.90	74.04
TextING	66.51	93.85	75.41
TensorGCN	66.78	93.99	73.52
RB-GAT	67.90	94.84	76.17

4.4.2. Comparison of the Accuracy of Models with Different Head Sizes

In our work, we utilize a multi-head graph attention network, wherein the number of heads functions as a hyperparameter. Each head corresponds to an independent attention model that influences the number of distinct attention mechanisms employed to compute node embeddings.

Figure 2 illustrates the classification accuracy of the RB-GAT model on the Ohsumed, R8, and MR datasets with varying head numbers. It is evident from the figure that the model's accuracy varies depending on the chosen head numbers. Notably, on the Ohsumed, R8, and MR datasets, the RB-GAT model achieves its highest accuracy when the number of heads is set to 8, resulting in accuracies of 71.48%,98.45%, and 89.32%, respectively. The overall trend observed across the three datasets is an initial increase followed by a subsequent decrease in accuracy. Specifically, for the Ohsumed dataset, an initial increase in the number of heads from 2 to 8 results in a gradual improvement in accuracy, peaking at 71.48% with 8 heads. This indicates an optimal density of attention mechanisms for capturing the intricate relationships within biomedical literature categorized in Ohsumed. Conversely, a subsequent increase to 10 heads leads to a slight decrease in performance, suggesting a threshold beyond which additional heads may introduce noise or redundant information processing. Similar trends are observed for the R8 and MR datasets, where accuracy incrementally rises with the number of heads, reaching the highest accuracy of 98.45% and 80.32% with 8 heads, respectively.



(a) (b) (c)

Figure 2. Comparison of classification accuracy of models with different head sizes for dataset: (a) Ohsumed, (b) R8, and (c) MR.

In general, increasing the number of heads enhances the model's expressive capability, enabling it to capture more intricate relationships among graph nodes. Additionally, a higher number of heads allows the model to focus on multiple aspects of the input graph concurrently, surpassing the limitations of a single attention mechanism. However, it is important to consider that augmenting the number of heads also increases the model's complexity, leading to potential challenges during training and a higher risk of overfitting.

5. Conclusion

In this paper, we introduce RB-GAT, a novel text classification model based on graph neural networks (GNNs) that combines the strengths of RoBERTa-BiGRU and a graph attention network. Our model is built upon the foundation of constructing a heterogeneous graph, which captures intricate relationships between texts and words, utilizing advanced embedding techniques to represent these entities effectively. The integration of RoBERTa-BiGRU enables our model to preserve long-term dependencies and bidirectional information within texts, crucial for understanding the semantic richness of language. Additionally, the application of a multi-head, two-layer Graph

Attention Network allows RB-GAT to dynamically assess and weigh the importance of adjacent nodes in the graph, facilitating a deeper understanding of relationships and interactions within textual data.

The empirical evaluation of RB-GAT, performed on three well-established text classification datasets, highlights its superior performance compared to existing sequential deep learning and other GNN-based methods. These findings underscore the robustness, stability, and effectiveness of our model in tackling the challenges of text classification tasks. By providing a holistic solution that addresses both the representational and relational aspects of text data, RB-GAT establishes a new standard for future research in this field. We anticipate that the principles and methodologies presented in this paper will stimulate further advancements in text mining and related disciplines, pushing the boundaries of what can be achieved.

Author Contributions: Conceptualization, Shaoqing Lv and Chichi Wang; Data curation, Shaoqing Lv; Formal analysis, Shaoqing Lv and Xuanhong Wang; Funding acquisition, Shaoqing Lv; Investigation, Xuanhong Wang and Zhiqiang Bao; Methodology, Shaoqing Lv; Project administration, Zhiqiang Bao; Resources, Zhiqiang Bao; Software, Jungang Dong; Supervision, Shaoqing Lv; Validation, Jungang Dong and Chichi Wang; Visualization, Jungang Dong; Writing – original draft, Chichi Wang; Writing – review & editing, Shaoqing Lv. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Natural Science Basic Research Program of Shaanxi (Program No. 2022JQ-675).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Kowsari, K.; Jafari Meimandi, K.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text classification algorithms: A survey. *Information* **2019**, *10*, 150.
2. Ahmed, H.; Traore, I.; Saad, S. Detecting opinion spams and fake news using text classification. *Security and Privacy* **2018**, *1*, e9.
3. Melville, P.; Gryc, W.; Lawrence, R.D. Sentiment analysis of blogs by combining lexical knowledge with text classification. In Proceedings of the Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009; pp. 1275-1284.
4. Barberá, P.; Boydstun, A.E.; Linn, S.; McMahon, R.; Nagler, J. Automated text classification of news articles: A practical guide. *Political Analysis* **2021**, *29*, 19-42.
5. Chowdhary, K.; Chowdhary, K. Natural language processing. *Fundamentals of artificial intelligence* **2020**, 603-649.
6. Shah, K.; Patel, H.; Sanghvi, D.; Shah, M. A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augmented Human Research* **2020**, *5*, 1-16.
7. Liu, P.; Zhao, H.-h.; Teng, J.-y.; Yang, Y.-y.; Liu, Y.-f.; Zhu, Z.-w. Parallel naive Bayes algorithm for large-scale Chinese text classification based on spark. *Journal of Central South University* **2019**, *26*, 1-12.
8. Kalcheva, N.; Karova, M.; Penev, I. Comparison of the accuracy of SVM kemel functions in text classification. In Proceedings of the 2020 International Conference on Biomedical Innovations and Applications (BIA), 2020; pp. 141-145.
9. Minaee, S.; Kalchbrenner, N.; Cambria, E.; Nikzad, N.; Chenaghlu, M.; Gao, J. Deep learning--based text classification: a comprehensive review. *ACM computing surveys (CSUR)* **2021**, *54*, 1-40.
10. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 2014; pp. 1746–1751.
11. Liu, P.; Qiu, X.; Huang, X. Recurrent neural network for text classification with multi-task learning. In Proceedings of the Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016; pp. 2873-2879.
12. Wu, L.; Chen, Y.; Shen, K.; Guo, X.; Gao, H.; Li, S.; Pei, J.; Long, B. Graph neural networks for natural language processing: A survey. *Foundations and Trends® in Machine Learning* **2023**, *16*, 119-328.

13. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI open* **2020**, *1*, 57-81.
14. Wu, S.; Sun, F.; Zhang, W.; Xie, X.; Cui, B. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys* **2022**, *55*, 1-37.
15. Xiong, J.; Xiong, Z.; Chen, K.; Jiang, H.; Zheng, M. Graph neural networks for automated de novo drug design. *Drug discovery today* **2021**, *26*, 1382-1393.
16. Strokach, A.; Becerra, D.; Corbi-Verge, C.; Perez-Riba, A.; Kim, P.M. Fast and flexible protein design using deep graph neural networks. *Cell systems* **2020**, *11*, 402-411.
17. Liao, W.; Zeng, B.; Liu, J.; Wei, P.; Cheng, X.; Zhang, W. Multi-level graph neural network for text sentiment analysis. *Computers & Electrical Engineering* **2021**, *92*, 107096.
18. Xie, Q.; Huang, J.; Du, P.; Peng, M.; Nie, J.-Y. Graph topic neural network for document representation. In Proceedings of the Proceedings of the Web Conference 2021, 2021; pp. 3055-3065.
19. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* **2018**.
20. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* **2019**.
21. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation* **2019**, *31*, 1235-1270.
22. Xu, G.; Meng, Y.; Qiu, X.; Yu, Z.; Wu, X. Sentiment analysis of comment texts based on BiLSTM. *Ieee Access* **2019**, *7*, 51522-51532.
23. Tai, K.S.; Socher, R.; Manning, C.D. Improved semantic representations from tree-structured long short-term memory networks. In Proceedings of the Proceedings of the Annual Meeting of the Association for Computational Linguistics, 2015; pp. 1556-1566.
24. Lei, T.; Zhang, Y.; Wang, S.I.; Dai, H.; Artzi, Y. Simple recurrent units for highly parallelizable recurrence. *arXiv preprint arXiv:1709.02755* **2017**.
25. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **1998**, *86*, 2278-2324.
26. Grefenstette, E.; Blunsom, P. A convolutional neural network for modelling sentences. In Proceedings of the The 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland, 2014; pp. 655-665.
27. Liu, J.; Chang, W.-C.; Wu, Y.; Yang, Y. Deep learning for extreme multi-label text classification. In Proceedings of the Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval, 2017; pp. 115-124.
28. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. *Advances in neural information processing systems* **2015**, *28*.
29. Yao, L.; Mao, C.; Luo, Y. Graph convolutional networks for text classification. In Proceedings of the Proceedings of the AAAI conference on artificial intelligence, 2019; pp. 7370-7377.
30. Liu, X.; You, X.; Zhang, X.; Wu, J.; Lv, P. Tensor graph convolutional networks for text classification. In Proceedings of the Proceedings of the AAAI conference on artificial intelligence, 2020; pp. 8409-8416.
31. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems* **2016**, *29*.
32. Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; Weinberger, K. Simplifying graph convolutional networks. In Proceedings of the International conference on machine learning, 2019; pp. 6861-6871.
33. Hu, L.; Yang, T.; Shi, C.; Ji, H.; Li, X. Heterogeneous graph attention networks for semi-supervised short text classification. In Proceedings of the Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP), 2019; pp. 4821-4830.
34. Zhang, Y.; Yu, X.; Cui, Z.; Wu, S.; Wen, Z.; Wang, L. Every document owns its structure: Inductive text classification via graph neural networks. In Proceedings of the Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020; pp. 334-339.
35. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.