electronics

MDPI

Article

Short Text Classification Based on Hierarchical Heterogeneous Graph and LDA Fusion

Xinlan Xu [1], Bo Li [1,*], Yuhao Shen [1], Bing Luo [1], Chao Zhang [2,*] and Fei Hao [3]

[1] School of Computer and Software Engineering, Xihua University, Chengdu 610039, China; xinlanxu@stu.xhu.edu.cn (X.X.); shenyuhao@stu.shu.edu.cn (Y.S.); bingluo@mail.xhu.edu.cn (B.L.)
[2] Intelligent Policing Key Laboratory of Sichuan Province, Sichuan Police College, Luzhou 646000, China
[3] School of Computer Science, Shaanxi Normal University, Xi'an 710119, China; fhao@snnu.edu.cn
* Correspondence: libo@mail.xhu.edu.cn (B.L.); galoiszhang@scpolicec.edu.cn (C.Z.)

Abstract: The proliferation of short texts resulting from the rapid advancements of social networks, online communication, and e-commerce has created a pressing need for short text classification in various applications. This paper presents a novel approach for short text classification, which combines a hierarchical heterogeneous graph with latent Dirichlet allocation (LDA) fusion. Our method first models the short text dataset as a hierarchical heterogeneous graph, which incorporates more syntactic and semantic information through a word graph, parts-of-speech (POS) tag graph, and entity graph. We then connected the representation of these three feature maps to derive a comprehensive feature vector for the text. Finally, we used the LDA topic model to adjust the feature weight, enhancing the effectiveness of short text extension. Our experiments demonstrated that our proposed approach has a promising performance in English short text classification, while in Chinese short text classification, although slightly inferior to the LDA + TF-IDF method, it still achieved promising results.

Keywords: short text classification; LDA; hierarchical heterogeneous graph

Citation: Xu, X.; Li, B.; Shen, Y.; Luo, B.; Zhang, C.; Hao, F. Short Text Classification Based on Hierarchical Heterogeneous Graph and LDA Fusion. *Electronics* **2023**, *12*, 2560. https://doi.org/10.3390/electronics12122560

Academic Editor: George A. Papakostas

Received: 26 April 2023
Revised: 21 May 2023
Accepted: 3 June 2023
Published: 6 June 2023

1. Introduction

With the increase of Internet users and the rapid development of social networks, massive amounts of short text data have been generated in life, such as product descriptions, comment information, and microblogs. More and more short texts need to be classified. Maron [1] published the first paper on automatic text classification. However, traditional text classification methods are unsuitable for short text classification [2]. Therefore, how to extract hidden short text information and apply it to practical tasks such as topic tracking [3], sentiment analysis [4], industrial equipment inspection [5], and personalized user recommendation [6] is a hot topic in the field of natural language processing.

It is more challenging to classify short texts than long texts [7]. One reason is that short texts contain fewer terms, which results in the need for more contextual information on the text content. The other one is that many data lack labeling information due to the high cost of manual labeling. Therefore, understanding short texts requires auxiliary knowledge, for example concepts that can be found in common sense knowledge graphs [8,9], potential topics extracted from short text datasets [10], and entities in knowledge graphs [10]. Despite this, more than merely enriching auxiliary knowledge is needed to deal with the lack of labeled data [7,11]. In addition, training regular depth models requires large-scale label data [12].

So far, one of the most-effective methods for short text classification is graph neural networks. Among them, the best classification effect is the SHINE [13] model. It introduces more semantic and syntactic information by modeling short text datasets as hierarchical heterogeneous graphs of word-level component graphs. Besides, a dynamic short text

graph easily transfers labels between similar short texts. However, it could better solve problems such as sparse short text features.

In order to address the above problems, we propose the SHINE + LDA algorithm. It can improve the accuracy of short text classification by dealing with semantic and syntactic information missing in short texts. First, we constructed three feature graphs: word graph, POS tag graph, and entity graph. Then, we connected the representations of the feature graphs to obtain the feature vector of the text. Last but not least, we used the LDA topic model for feature weight adjustment after connecting the representations of the feature map to obtain the feature vector of the text. More specifically, the semantic calculation of word vectors gives higher weight to significant features in short texts. Adjusting the feature weight of short texts solves the problem of sparse text features. Extensive experimental results showed that our proposed approach has a promising performance in English short text classification, while in Chinese short text classification, although slightly inferior to the LDA + TF-IDF method, it still achieved promising results. The goal of the research work was to improve the accuracy of short text classification by dealing with semantic and syntactic information missing in short texts.

The potential applications of short text classification are as follows:

1. The short text classification technology is used to analyze the data of industrial equipment, which can better understand the running state of equipment and diagnose the fault.
2. The short text classification technology is used to classify and cluster large-scale news reports and social media topics and analyze and predict the development and trend of events.
3. According to the historical behaviors and interests of users, the short text library is classified and matched to recommend personalized goods or services for users.

The organizational structure of the remainder of the paper is as follows. The related work is given in Section 2. Our approach is presented in Section 3. Section 4 presents the experimental results of our proposed method, including a comparison with existing methods and its application to the classification of short texts in English and Chinese. Finally, Section 5 summarizes the content of the paper and points out the deficiencies of the current research and the direction for future research.

2. Related Work

2.1. Short Text Classification

Short text classification is complicated [14]. Because of their restricted length, short texts lack context information and a strict syntactic structure. These are essential for text comprehension [8]. Therefore, it is crucial to extend short texts. Up to now, there are two kinds of short text expansion methods. They are external resource expansion and internal resource expansion. Expansion based on external resources refers to expanding short texts using an external corpus. For example, Chen [9] uses external resources to obtain rich information about short text expansion. Expansion based on internal resources refers to the expansion of short text by using the context of the text itself to construct the word set based on text content. For example, Paulo [15] proposed using word co-occurrences and word vectors to create a large pseudo-document in the original text for feature expansion. Nevertheless, enriching semantic information alone cannot compensate for the deficiency in labeled data, a common problem faced by short texts [11]. Consequently, the GNN-based method was born, which performs the node classification of semi-supervised short text classification. HGAT [10] utilizes the GNN with dual-level attention to jointly forward messages about topics, entities, and documents of corpus-level graph modeling, in which entities are words connected to knowledge graphs. FABG [16] first uses the bidirectional GRU (Bi-GRU) to learn the semantic information of the text. Then, it uses the complete attention mechanism to learn the weight of the current and previous outputs of the Bi-GRU in each step so that each step can obtain necessary information and ignore irrelevant information, which improves the effect of classification. BERT [17] is a two-way deep

language model based on Transformer, which can capture two-way context semantics. Zhang [18] proposed an ERNIE pre-training model based on BERT and knowledge map fusion, which uses multi-information entities in the knowledge map as external knowledge to improve language representation. Sitaula and Shahi [19] proposed a method of a multi-channel convolutional neural network using mixed features. Firstly, each tweet is represented by combining grammatical and semantic information, and then, a new multi-channel convolutional neural network is used to classify it, which integrates multiple CNNs and can capture multi-scale information. Sitaula et al. [20] used three different feature extraction methods to represent tweets. The three methods were fastText-based, domain-specific, and domain-agnostic. Three different convolutional neural networks were then used to implement the proposed features. Finally, three CNN models were integrated in an end-to-end way to achieve the final result. SHINE [13] proposes a GNN-based hierarchical heterogeneous graph consisting of subgraphs based on the word level. At present, the most-advanced method for short text classification is SHINE, as shown in Section 2.2.

2.2. SHINE Model

SHINE [13] is a new hierarchical heterogeneous graph representation learning method of short text classification, which can fully capture the sparse semantic relationship between short texts. It proposes a method to construct multiple heterogeneous graphs for text. Specifically, it proposes two different composition methods. They are word-level component graphs and short document graphs. The former describes the interaction between words, POS labels, and entities. The component graphs are easy to extract and transport extra semantic and syntactic information to compensate for the absence of context information. The latter is dynamically learned and optimized to encode the similarity between short texts, thereby enabling more efficient label propagation between similar short texts.

However, SHINE does not perform particularly well in solving problems such as sparse short text features. Therefore, we propose the SHINE + LDA model, which introduces the LDA thematic model on the basis of the SHINE model. It can make better use of semantic and syntactic information to extend short text and adjust the feature weight by the LDA topic model to improve the classification effect. The basic principle of LDA is described in Section 2.3.

2.3. LDA Topic Model

Blei [21] proposed the topic model algorithm LDA. The basic idea of this model is that each piece of text data is composed of multiple topics, and the probability of multiple words can represent each topic. The LDA model can effectively map the text to a low-dimensional topic vector. Then, the text feature is represented using the text topic's distribution vector. The LDA model is shown in Figure 1.

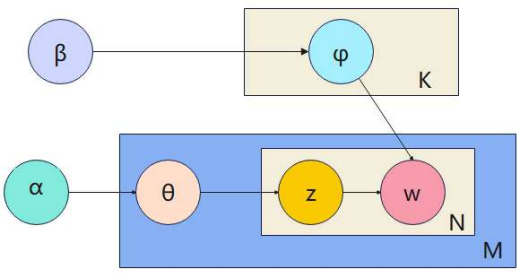

Figure 1. The LDA model.

The symbols in Figure 1 are defined as follows: M represents the total number of texts in the training corpus; N represents the total number of words in a text; K is the number of topics; θ is the distribution matrix of text topics; φ is the distribution matrix of topics

and words; α and β are the parameters of the Dirichlet distribution. $\theta_i \sim Dirichlet(\alpha)$, and $\varphi_k \sim Dirichlet((\beta)$, and W denotes the observable words in the text.

Suppose that D is the text set of the text dataset and Doc_m is the mth text in the available text D. The Doc_m is composed of word combinations $(W_{m1}, W_{m2}, W_{m3} \cdots \cdots W_{mn})$, where W_{mn} represents the nth word of the m^{th} document. The modeling process of the LDA model is as follows:

1. For text $Doc_m \in D$, the topic distribution θ_m is generated by $Dirichlet(\alpha)$.
2. For the nth word of the text W_{mn}, topic Z_{mn} of word W_{mn} is generated by the polynomial distribution $(Z_{mn} \sim Mult(\theta_m))$.
3. Determining the distribution probability matrix φ_m of topics and words is a distribution of $(\varphi_m \sim Dirichlet((\beta))$ with parameter β. At the same time, according to topic Z_{mn}, W_{mn} determines a term distribution φ_{zmn}.
4. According to the distribution of words φ_{zmn}, generate the selected topic term $W_{mn}(W_{mn} \sim Mult(\varphi_{zmn}))$.
5. Repeat Steps 1 to 4 to generate Doc_m for all words in the text. Repeat Steps 1 to 5 above for all text in the text set to generate the entire text set D.

The text process simulated by the LDA model shows the joint probability distribution of variables as shown in Formula (1).

$$p(W, Z, \theta, \varphi \mid \alpha, \beta) = \prod_{m=1}^{|D|} p(W_{mn} \mid \varphi_{zmn}) \cdot p(Z_{mn} \mid \theta) \cdot p(\theta \mid \alpha) \cdot p(\varphi \mid \beta) \quad (1)$$

where W represents the observable words in the text, Z is the subject, θ is the distribution matrix of the text topic, φ is the distribution matrix of the topic and word, α and β are the parameters of the Dirichlet distribution, which is obtained by the grid search method, D represents the document aggregate of the text dataset, W_{mn} represents the nth word of the mth document, φ_{zmn} represents the lexical distribution of the nth term of the mth document, Z_{mn} represents the topic of the nth term of the mth document, and $p(x \mid y)$ is under the condition of the y occurrences and the event probability of x occurring.

The LDA model's probability distribution of hidden variables is complicated, so estimation methods are often used to calculate it. The estimation methods include variational Bayesian inference and the Gibbs sampling algorithm. Among them, the advantage of the Gibbs sampling algorithm [22] is its fast computation speed. Additionally, it is widely used because it is easy to understand and implement. This paper also used this method to process the text data.

3. Proposed Method

Currently, SHINE [13] performs better in short text classification. However, it still needs improvement in short text extension. It cannot avoid the problem of invalid expansion and reduces the effectiveness of short text expansion. Therefore, we propose a method to better compensate for the lack of context information in short text classification. To begin with, we modeled the short text dataset as a hierarchical heterogeneous graph consisting of a word-level graph. In addition, the GCN was used to obtain the node embedding of three feature graphs, and the feature vectors of the short texts were obtained by hierarchical pooling. Moreover, the LDA topic model was used to obtain the topic distribution vector of the short text. Furthermore, the feature vector of the short text and the topic distribution vector were weighted and fused to obtain a new feature vector. Finally, the GCN was used to learn the label probability distribution of the short text nodes. By combining words, topics, entities, and POS, we enriched the semantics of short texts, which significantly facilitates classification tasks.

3.1. Word-Level Graph [13]

A word-level graph can use both semantic and syntactic information to better understand the short text. For short texts, they often do not contain enough contextual

information to facilitate proper classification. Therefore, a word-level graph can be used to combine various information, such as words, entities, and POS, so as to enrich the semantics of short texts and further improve the effect of the classification task. The word-level graph is composed of the word graph, POS tag graph, and entity graph. The following sections describe how to build a word-level graph.

3.1.1. Node Embedding

$Z_T = \{N_T, M_T\}$ represents a word-level graph of type T, where N_T represents a set of nodes, $M_T \in R^{|N_T| \times |N_T|}$ represents the adjacency matrix, and the node feature is $X^I \in R^{|N_T| \times a_T}$. The node embeddings E_T are obtained using a two-layer graph convolutional network (GCN) [23].

$$E_T = \tilde{M}_T \cdot ReLu(\tilde{M}_T X_T V_T^1) V_T^2 \tag{2}$$

where $[ReLu(x)]_i = max([x]_i, 0)$, $\tilde{M}_T = B_T^{-\frac{1}{2}}(I + M_T)B_T^{-\frac{1}{2}}$ with $[B_T]_{ij} = \sum_j [M_T]_{ij}$ and V_T^1, V_T^2 are trainable parameters.

3.1.2. Graph Construction

According to the definition of a graph $G_T = \{N_T, M_T\}$, G_T represents a graph of type T, N_T represents a set of nodes, and $M_T \in R^{|N_T| \times |N_T|}$ represents the adjacency matrix. The construction of the graph needs to solve two problems:

1. How to define node N:
   This is defined according to the characteristics to be integrated. If it is merging POS information, N is the POS tag. If it is merging entity information, then N is the entity label.
2. How to build adjacency matrix:
   - Pointwise mutual information (PMI) [24]:
     PMI is used to measure the correlation between the two variables. The calculation formula of PMI is as follows.

$$PMI(x, y) = \log \frac{p(x, y)}{p(x)p(y)} = \log \frac{p(x \mid y)}{p(x)} = \log \frac{p(y \mid x)}{p(y)} \tag{3}$$

     where x and y are two variables. The larger the PMI, the higher the correlation between the two variables.
   - Cosine similarity:
     The cosine similarity is the cosine value of the angle between two vectors in the vector space, which is used to measure the similarity between two vectors. The calculation formula of the cosine similarity is as follows.

$$Similarity(\vec{x}, \vec{y}) = \cos\theta = \frac{\vec{x} \cdot \vec{y}}{|| \vec{x} || || \vec{y} ||} \tag{4}$$

     where $\vec{x}$ and $\vec{y}$ are two vectors. In the trigonometric function, the function value of the cosine function is $[-1, 1]$. Therefore, the cosine similarity range between the two vectors is $[-1, 1]$. When the angle between the two vectors is $0°$, the similarity is 1. When the angle is $180°$, the similarity is -1. The larger the cosine value, the more similar the two samples are.

Next, we describe how it constructs the word graph, POS tag graph, and entity graph:

1. Word graph:
   Build a word graph [13] $G_W = \{N_W, M_W\}$, where G_W represents a graph of type W, N_W represents a set of nodes, and $M_W \in R^{|N_W| \times |N_W|}$ represents the adjacency matrix. Firstly, the short text is segmented into words using NLTK. Then, the word

set is represented as N, and the PMI between any two words in the entire dataset is calculated to construct M_W.

$$[M_W]_{ij} = max(PMI(n_w^i, n_w^j), 0) \tag{5}$$

where $n_w^i, n_w^j \in N_W$. Then, use Formula (2) to learn the node representation function E_W (W is the type of graph) based on this graph. This calculates the representation of each node, which is the embedding of each word. The initial representation uses one-hot encoding.

2. POS tag graph:
   Build a POS tag graph [13] $G_P = \{N_P, M_P\}$, where G_P represents a graph of type P, N_P represents a set of nodes, and $M_P \in R^{|N_P| \times |N_P|}$ represents the adjacency matrix. The construction and node representation of the graph are similar to the word graph. The only difference is that the words in the original dataset are replaced with POS tags, and the POS tag set provided by NLTK is used to obtain the POS tags for each word in the short text.
3. Entity graph:
   Build an entity graph [13] $G_e = \{N_e, M_e\}$, where G_e represents a graph of type e, N_e represents a set of nodes, and $M_e \in R^{|N_e| \times |N_e|}$ represents the adjacency matrix. The node set from the entity type is defined in the NELL knowledge base [25]. The adjacency matrix M_e is calculated as follows.

$$[M_e]_{ij} = max(Similarity(x_e^i, x_e^j), 0) \tag{6}$$

x_e is the embedding of the entity type. It is learned by the TransE [26] method, a classic knowledge graph embedding method. In general, the NELL knowledge base [25] is first used to identify the entities contained in each text. Then, use the TransE [26] method to learn the embedding of each entity. Finally, the cosine distance calculates the similarity between two entities to fill M_e.

Figure 2 shows that the word-level graph consists of three graphs. They are the word graph, entity graph, and POS tag graph.

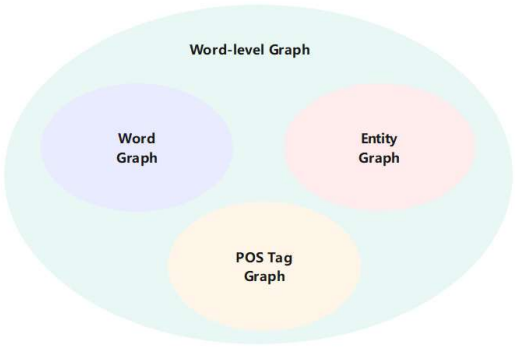

Figure 2. Word-level graph.

3.2. Short Text Graph

We dynamically learned the short text graph $G_S = \{N_S, M_S\}$, where N_S represents the text set and M_S represents the relation matrix between texts. To propagate labels more efficiently, we used hierarchical pooling to learn the similarity M_S between short texts from the word-level graph.

We used the above three feature maps to represent each node. Moreover, the similarity is calculated by the inner product of the two nodes.

$$\hat{x}_T^i = u(E_T^t S_T^i) \tag{7}$$

Here, the subscript T represents the type of feature graph, E_T is the node embeddings of the corresponding graph, u is the $L1$-norm, and $u(x) = \frac{x}{||x||_2}$.

For the word graph and POS tag graph, S_T represents the TF-IDF [27] value of each word or POS tag in text i. For the entity graph, when S_T equals 1, the entity in the graph exists in the current text i. When S_T is equal to 0, the entity in the graph does not exist in the current text. S_T characterizes the relationship between all nodes in the subgraph and the current node and uses this weight to weight the embedding of the nodes calculated by the subgraph. The weighted node's embedding matrix is used to represent text i. Finally, a text i connects the representations of the three feature graphs.

$$\hat{x}_s^i = \hat{x}_w^j \mid\mid \hat{x}_p^j \mid\mid \hat{x}_e^j \tag{8}$$

According to the generation principle of the LDA topic model described in Section 2.3, the LDA topic model is used to conduct the topic modeling for the short text, and the short text is represented as the topic distribution vector. Then, the feature vector of the short text and the topic distribution vector are weighted and fused to obtain a new feature vector x_s.

Use the inner product to represent the relation matrix M_S.

$$[M_S]_{ij} = \begin{cases} (x_s^i)^T x_s^j & \text{if } (x_s^i)^T x_s^j \geq \delta_s \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Among them, δ_s is the threshold for measuring text similarity and plays a sparse role. x_s is a short text feature learned by Formula (8).

Then, use the GCN to learn the label probability distribution of the document nodes.

$$\hat{H}_S = softmax(M_S \cdot ReLu(M_S X_S W_S^1) \cdot W_S^2) \tag{10}$$

where W_S^1 and W_S^2 are trainable parameters and can be trained using backpropagation algorithms, X_S is the short text embeddings, M_S is the adjacency matrix, $[softmax(x)]_i = \frac{exp([x]_i)}{\sum_j exp([x]_j)}$, and $ReLu(x) = max(0, x)$. Its classification losses are as follows.

$$\Gamma = -\sum_{i \in \Gamma_l} (f_s^i)^T \log(\hat{f}_s^i) \tag{11}$$

where Γ_l records the index of the labeled short text and f_s^i is a one-hot vector of all 0s.

Figure 3 shows the natural language processing techniques from short text datasets to build a heterogeneous corpus-level graph. Figure 4 shows our proposed SHINE + LDA architecture, which aggregates hierarchically on the word-level component graph to obtain a short text graph. First, it constructs three feature maps: word graph, POS tag graph, and entity graph. Then, it connect the representations of these three feature maps to obtain the feature vectors of the text. In the end, the LDA topic model is used to adjust the feature weight and update the weight of the short text.

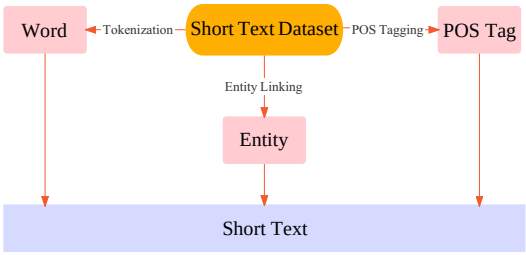

Figure 3. Graph construction.

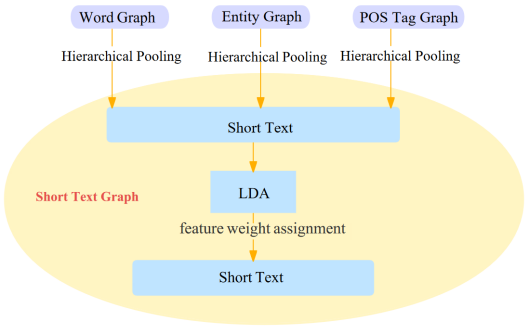

Figure 4. Framework of SHINE + LDA.

The algorithm for SHINE + LDA is shown in Algorithm 1.

Algorithm 1 Pseudocode for our algorithm.

Input: short text dataset S, word graph $G_W = \{N_W, M_W\}$, POS tag graph $G_P = \{N_P, M_P\}$, entity graph $G_e = \{N_e, M_e\}$ (see Section 3.1), sample-specific aggregation vector S_T^i, where $T \in \{W, P, e\}$;
for $i = 1, 2, \cdots, I$ **do**
  for $T \in \{W, P, e\}$ **do**
    obtain the node embedding E_T of G_T through Formula (2);
  end for
  obtain the short text feature X_S by layer pooling on G_T through Formula (8);
  update the feature weight of the short text through the LDA topic model (see Section 2.3);
  obtain a short document embedding from G_S, and perform class prediction through Formula (10);
  optimize the model parameters relative to Formula (11) by back propagation;
end for

4. Experiments

Each result was averaged over five runs as the final result.

4.1. Experimental Environment

The server was the EMC PowerEdge R740; the CPU was 5220R 2.2G, 24C/48T (24 core 48 processes) * 2; the GPU was NVIDIA GRX 3090; the memory was 384G; the operating system was Ubuntu 18.04.6, CUDA 11.4.0, Python 3.7, and Pytorch 1.2.

4.2. Datasets

We selected five short text datasets for the experiments. They were Twitter, Snippets, TagMyNews, MR, and Headlines Today.

Twitter: The dataset was provided by NLTK4, a vast corpus of tweets and replies [10].

Snippets: The dataset was published by Phan et al. [11] and is a web search fragment returned by a Google search.

TagMyNews: The dataset is an English news headline from a Simple Syndication (RSS) feed published by Hu et al. [10].

MR: a dataset of movie reviews [28].

Headlines Today: a dataset of daily news (The data come from today's headline client. To download the dataset, please visit the website: https://github.com/BenDerPan/toutiao-text-classfication-dataset, accessed on 14 May 2018). We randomly selected 15,425 pieces of data.

Table 1 shows the feature information of the datasets. It includes the number of datasets, the average length in words, the number of categories, the number of training sets, and the proportion of training sets in parentheses.

Table 1. Characteristics of datasets.

Name	Quantity	Average Length in Words	Classes	Train (Ratio)
Twitter	10,000	3.5	2	40 (0.40%)
Snippets	12,340	14.5	8	160 (1.30%)
TagMyNews	32,549	5.1	7	140 (0.43%)
MR	10,662	7.6	2	40 (0.38%)
Headlines Today	15,425	19.2	15	300 (1.94%)

The following is the preprocessing we performed on all datasets. We marked each sentence and deleted low-frequency words, non-English characters, and stop words less than five times in the corpus [29]. We randomly selected 40 tagged short texts for each dataset in each class. Moreover, half was used as a training set, the other half as a validation set [10]. After Kipf and Welling [23], all the remaining short texts were used as test sets and unlabeled texts in training.

4.3. Hyperparameter Setting

We set our parameters according to the parameters in the comparison experiment. We set the entity embedding dimension d_e to 100. For all datasets, we set the sliding window size of the PMI of G_W and G_P to 5, set the embedding size of all GCN layers used to 200, and set the threshold δ_S of G_S to 2.7. We set the number of topics $k = 15$ for the MR, TagMyNews, Headlines Today, and Twitter datasets in LDA. We set $k = 20$ for Snippets. For all datasets, each text was assigned to the first $P = 2$ topics with the highest probability. We implemented this method in PyTorch and used Adam [30] to train the model at a maximum of 1000 epochs at a learning rate of 5×10^{-3}. We stopped training in advance if the verification loss was not reduced for ten consecutive periods. The dropout rate was set to 0.5.

4.4. Evaluation Metrics

We evaluated the classification results with the more commonly used evaluation index accuracy (ACC) and F1 value [31]:

1. Accuracy: Accuracy (ACC) refers to the number of samples in which the results are correctly predicted as a percentage of the total results [32].

$$ACC = \frac{TP + TN}{TP + FP + FN + TN} \tag{12}$$

   Among them, TP indicates that the prediction is a positive class and the actual number of texts in the positive class. FP indicates that the prediction is a positive class, which is the number of texts in the negative class. TN indicates that the prediction is negative, and it is the number of negative texts. FN means that the prediction is a negative class, which is the number of texts in the positive class.
2. F1 value: The F1 value is the harmonic average of the precision and recall rate. Because the precision and recall rate have difficulty in fully reflecting the classification results, the F1 value comprehensive evaluation index was introduced [33].

$$F1 = \frac{2 * P * R}{P + R} \tag{13}$$

   where $P = \frac{TP}{TP+FP}$ and $R = \frac{TP}{TP+FN}$.

4.5. Compared Methods

LDA + SVM is a short text classification method based on the topic model and support vector machine. First, LDA is used to model the topic of the text, and then, the topic features are input into SVM for classification. It can capture the underlying topic information in

the text, but it requires much preprocessing and hyperparameter adjustment and may not handle lexical diversity and context information well. HGAT is a short text classification method based on a graph attention network. This method also represents text as nodes and forms a graph and then uses the graph attention network for classification. It can better capture the relationships between texts and assign different weights to each node, but it is also computationally complex for large datasets. SHINE is a novel hierarchical heterogeneous graph representation learning method, which can effectively learn from hierarchical graphs modeling different short text data. SHINE can better utilize the interaction between nodes of the same type and capture the similarities between short texts. However, it still needs improvement in short text extension. SHINE + LDA adds the LDA topic model to solve the problem of sparse features in short text extension.

The proposed SHINE + LDA method was compared with the above methods.

According to the results in Table 2, although the difference between the SHINE model and the SHINE + LDA model on the Twitter and Snippets datasets is small, the SHINE + LDA model had better ACC and F1 indicators on the four datasets than the other five models. This indicated that the SHINE + LDA model can make better use of semantic and syntactic information to extend short texts and adjust the feature weights through the LDA topic model to improve the classification effect.

Table 2. Training results compared with some English short text classification methods.

Model	Twitter		Snippets		TagMyNews		MR	
	ACC	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**
SHINE	72.54%	72.19%	82.39%	81.62%	62.50%	56.21%	64.58%	63.89%
HGAT	63.21%	62.48%	82.36%	74.44%	61.72%	53.81%	62.75%	62.36%
LDA + SVM	54.34%	53.97%	62.54%	56.40%	40.40%	30.40%	54.40%	48.39%
SHINE + LDA	73.17%	73.17%	84.45%	84.45%	72.75%	72.71%	73.00%	69.24%

4.6. Model Sensitivity on Twitter

Figure 5 depicts the effect of changes in the threshold δ_S on short text classification performance. First, the performance improved with the δ_S increase. When reaching a specific value, the performance decreased with the δ_S increase. When reducing to a certain extent, the performance increased with the δ_S increase. When increasing to a specific value, the performance decreased sharply with the δ_S increase. When δ_S was too small, G_S may become sparse, resulting in lower performance. However, when δ_S was too large, G_S may lose its proper functionality, reducing the performance of short text classification. Figure 6 depicts the impact of the GCN embedding size on short text classification performance. With the increase of GCN embedding size, the performance also improved. When increasing to a certain value, the performance decreased with the increase of the GCN embedding size. When reducing to a certain value, the performance increased with the increase of the GCN embedding size. When reaching a certain value, the performance decreased again with the increase of the GCN embedding size. If the GCN embedding size is too small, it may not capture enough neighbor node information, resulting in performance degradation. Conversely, if the GCN embedding size is too large, it may result in overfitting or increased computational complexity, which also affects performance.

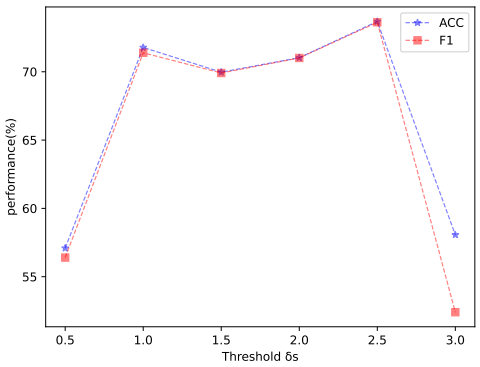

Figure 5. Varying δ_S.

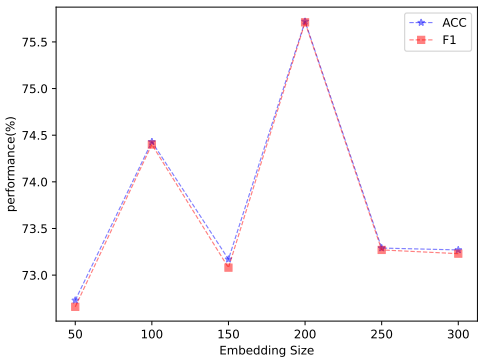

Figure 6. Varying the embedding size of GCN.

4.7. Application to Chinese

The difference between English and Chinese is that there is no space between words in Chinese to indicate the boundary of words. Therefore, Chinese word segmentation is the basic module and the primary link in various natural language processing systems. So far, there are numerous Chinese word segmentation methods. This article uses the jieba segmentation.

4.7.1. Jieba Segmentation

Jieba word segmentation is based primarily on a statistical dictionary to construct a prefix dictionary. Furthermore, it uses the prefix dictionary to segment the input sentence to obtain all possible segmentations. In addition, it constructs a directed acyclic graph according to the segmentation position. Finally, the maximum probability path is calculated using a dynamic programming algorithm, and the word segmentation is performed according to the path. For unregistered words, jieba uses the HMM model based on Chinese character composition and the Viterbi algorithm for derivation.

4.7.2. Experimental Result

We applied this method to the Headlines Today dataset to test its performance on short text classification for Chinese. The proposed SHINE + LDA method was compared with the following methods.

According to the results in Table 3, it can be seen that the SHINE + LDA model was better than the LDA + SVM and LDA + KNN models on the ACC and F1 indexes, but slightly worse than the LDA + TF-IDF model. This showed that the SHINE + LDA model can make better use of semantic and syntactic information to extend short texts and adjust the feature weights through the LDA topic model to improve classification. However, there were still some defects in this model, so it was necessary to strengthen the collection ability of text content information. All in all, the results showed that the SHINE + LDA model still had room for improvement on the performance of Chinese short text classification, but it had a certain feasibility and practicability.

Table 3. Training results compared with some Chinese short text classification methods.

Model	ACC	F1
LDA + SVM	61.2%	59.4%
LDA + KNN	60.1%	57.7%
LDA + TF-IDF	86.8%	87.1%
SHINE + LDA	84.2%	83.9%

5. Conclusions

In this paper, we proposed a new hierarchical heterogeneous graph representation learning method for short text classification, which is particularly useful to make up for the lack of context information. In particular, it can effectively learn from hierarchical graphs modeled from different perspectives on short text datasets. The word-level component graph is used to understand short text from the perspective of semantics and syntax, and the dynamically learned short text graph allows efficient and effective label propagation in similar short texts. It can also assign higher weights to essential features in short texts using LDA for word vector semantic computation. Extensive experimental results showed that this method was always superior to the most-progressive method on four benchmark datasets. Moreover, its application to short text classification for Chinese also achieved good results.

However, this method also had some limitations. On the one hand, because this method required constructing multiple heterogeneous graphs and processing them with algorithms such as the GCN and LDA topic model, its computational complexity was high. This increased the time and space overhead of training and testing the model, which may limit its use on large-scale datasets. On the other hand, this method used a hierarchical heterogeneous graph and LDA topic model to fuse various information. However, this fusion method may ignore some important information of the original text and may have problems such as model overfitting because the fusion method is not flexible enough. Therefore, this fusion approach may have limitations in more complex scenarios.

The future work of this paper can be summarized as follows:

1. Improved algorithm efficiency and accuracy:
   Future studies could continue to optimize the algorithm and improve its classification efficiency and accuracy.
2. Explore other text feature fusion methods:
   The algorithm fuses the hierarchical heterogeneous graph and LDA topic model to process text features, and future studies can explore the fusion methods of other text features.

Author Contributions: Methodology, B.L. (Bo Li); Formal analysis, B.L. (Bing Luo); Resources, C.Z.; Data curation, Y.S.; Writing—original draft, X.X.; Writing—review & editing, F.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Science and Technology Program of Sichuan Province, China (Grant No. 2023YFG0264) and the Opening Project of Intelligent Policing Key Laboratory of Sichuan Province (Grant Nos. ZNJW2022KFMS004, ZNJW2022KFQN002, ZNJW2023KFQN007).

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

Abbreviations

The following abbreviations are used in this paper:

LDA	Latent Dirichlet allocation
POS	Parts-of-speech
PMI	Pointwise mutual information
ACC	Accuracy

References

1. Maron, M.E. Automatic Indexing: An Experimental Inquiry. *J. ACM* **1961**, *8*, 404–417. [CrossRef]
2. Vo, D.T.; Ock, C.Y. Learning to classify short text from scientific documents using topic models with various types of knowledge. *Expert Syst. Appl.* **2015**, *42*, 1684–1698. [CrossRef]
3. Du, Y.; Yi, Y.; Li, X.; Chen, X.; Fan, Y.; Su, F. Extracting and tracking hot topics of micro-blogs based on improved Latent Dirichlet Allocation. *Eng. Appl. Artif. Intell.* **2020**, *87*, 103279. [CrossRef]
4. Kilimci, Z.H.; Omurca, S.I. Extended Feature Spaces Based Classifier Ensembles for Sentiment Analysis of Short Texts. *Inf. Technol. Control* **2018**, *47*, 457–470. [CrossRef]
5. Zhu, L.; Tian, N.; Li, W.; Yang, J. A Text Classification Algorithm for Power Equipment Defects Based on Random Forest. *Int. J. Reliab. Qual. Saf. Eng.* **2022**, *29*, 2240001. [CrossRef]
6. Chen, H.Y. Personalized recommendation system of e-commerce based on big data analysis. *J. Interdiscip. Math.* **2018**, *21*, 1243–1247. [CrossRef]
7. Peng, H.; Li, J.; He, Y.; Liu, Y.; Bao, M.; Wang, L.; Song, Y.; Yang, Q. Large-Scale Hierarchical Text Classification with Recursively Regularized Deep Graph-CNN. In Proceedings of the 2018 World Wide Web Conference, Lyon, France, 23–27 April 2018.
8. Wang, J.; Wang, Z.; Zhang, D.; Yan, J. Combining Knowledge with Deep Convolutional Neural Networks for Short Text Classification. In Proceedings of the IJCAI, Melbourne, Australia, 19–25 August 2017.
9. Chen, J.; Hu, Y.; Liu, J.; Xiao, Y.; Jiang, H. Deep Short Text Classification with Knowledge Powered Attention. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'19/IAAI'19/EAAI'19), Honolulu, HI, USA, 27 January–1 February 2019; AAAI Press: Washington, DC, USA, 2019. [CrossRef]
10. Yang, T.; Hu, L.; Shi, C.; Ji, H.; Li, X.; Nie, L. HGAT: Heterogeneous Graph Attention Networks for Semi-Supervised Short Text Classification. *ACM Trans. Inf. Syst.* **2021**, *39*, 1–29. [CrossRef]
11. Phan, X.H.; Nguyen, L.M.; Horiguchi, S. Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-Scale Data Collections. In Proceedings of the 17th International Conference on World Wide Web (WWW'08), Beijing, China, 21–25 April 2008; pp. 91–100. [CrossRef]
12. Liu, P.; Qiu, X.; Huang, X. Recurrent Neural Network for Text Classification with Multi-Task Learning. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16), New York, NY, USA, 9–15 July 2016; AAAI Press: Washington, DC, USA, 2016; pp. 2873–2879.
13. Wang, Y.; Wang, S.; Yao, Q.; Dou, D. Hierarchical Heterogeneous Graph Representation Learning for Short Text Classification. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online, Punta Cana, Dominican Republic, 16–20 November 2021; pp. 3091–3101. [CrossRef]
14. Li, Q.; Peng, H.; Li, J.; Xia, C.; Yang, R.; Sun, L.; Yu, P.S.; He, L. A Survey on Text Classification: From Shallow to Deep Learning. *arXiv* **2020**, arXiv:2008.00364.
15. Bicalho, P.V.; Pita, M.; Pedrosa, G.; Lacerda, A.M.; Pappa, G.L. A general framework to expand short text for topic modeling. *Inf. Sci.* **2017**, *393*, 66–81. [CrossRef]
16. Tang, Q.; Li, J.; Chen, J.; Lu, H.; Du, Y.; Yang, K. Full Attention-Based Bi-GRU Neural Network for News Text Classification. In Proceedings of the 2019 IEEE 5th International Conference on Computer and Communications (ICCC), Chengdu, China, 6–9 December 2019; pp. 1970–1974.
17. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186. [CrossRef]
18. Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; Liu, Q. ERNIE: Enhanced Language Representation with Informative Entities. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1441–1451. [CrossRef]
19. Sitaula, C.; Shahi, T.B. Multi-channel CNN to classify nepali covid-19 related tweets using hybrid features. *arXiv* **2022**, arXiv:2203.10286.
20. Sitaula, C.; Basnet, A.; Mainali, A.; Shahi, T.B. Deep Learning-Based Methods for Sentiment Analysis on Nepali COVID-19-Related Tweets. *Comput. Intell. Neurosci.* **2021**, *2021*, 2158184. [CrossRef] [PubMed]
21. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
22. Porteous, I.R.; Newman, D.; Ihler, A.T.; Asuncion, A.U.; Smyth, P.; Welling, M. Fast collapsed gibbs sampling for latent dirichlet allocation. In Proceedings of the KDD, Las Legas, NV, USA, 24–27 August 2008.
23. Kipf, T.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2017**, arXiv:1609.02907.
24. Church, K.W.; Hanks, P. Word Association Norms, Mutual Information, and Lexicography. In Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 26–29 June 1989; pp. 76–83. [CrossRef]
25. Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E.R.; Mitchell, T.M. Toward an Architecture for Never-Ending Language Learning. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI'10), Atlanta, Georgia, 11–15 July 2010; AAAI Press: Washington, DC, USA, 2010; pp. 1306–1313.

26. Bordes, A.; Usunier, N.; Garcia-Durán, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-Relational Data. In Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13), Red Hook, NY, USA, 5–10 December 2013; Volume 2, pp. 2787–2795.
27. Aggarwal, C.C.; Zhai, C. A Survey of Text Classification Algorithms. In *Mining Text Data*; Springer: Boston, MA, USA, 2012.
28. Pang, B.; Lee, L. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In Proceedings of the ACL, Ann Arbor, MI, USA, 25–30 June 2005.
29. Yao, L.; Mao, C.; Luo, Y. Graph Convolutional Networks for Text Classification. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'19/IAAI'19/EAAI'19), Honolulu, HI, USA, 27 January–1 February 2019; AAAI Press: Washington, DC, USA, 2019. [CrossRef]
30. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2015**, arXiv:1412.6980.
31. Linmei, H.; Yang, T.; Shi, C.; Ji, H.; Li, X. 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), Hong Kong, China, 3–7 November 2019; pp. 4821–4830. [CrossRef]
32. Ikonomakis, E.; Kotsiantis, S.; Tampakas, V. Text classification: A recent overview. In Proceedings of the 9th WSEAS International Conference on Data Networks, Communications, Computers (DNCOCO'10), Faro, Portugal, 3–5 November 2005; p. 125.
33. Yang, Y.; Liu, X. A Re-Examination of Text Categorization Methods. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), Berkeley, CA, USA, 15–19 August 1999; pp. 42–49. [CrossRef]