ISSN: 2664-9640

Text multi-label recommendation method based on graph convolutional neural network

Jiayang Dai [a], Dong Zhou [b, *]

School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtang 411100, China.

[a] 504904397@qq.com, [b] dongzhou1979@hotmail.com

Abstract

With the continuous development of the global Internet, the amount of text information in the Internet has increased rapidly. Adding tags to massive information can facilitate the management of this information. Therefore, how to recommend tags for text information has become a current research hotspot. At present, most recommendation methods for text labels are based on semantic models, which recommend labels for text based on semantic information, but texts with the same label are not necessarily semantically similar, and recommending labels based only on semantics will cause certain deviations. In addition, when multi-label recommendation for text, there is often a certain correlation between labels, and most of the existing methods do not take this correlation into account. Therefore, we propose an encoder-decoder model based on graph convolutional neural network, which encodes the text content through an encoder based on GRUs, and constructs a heterogeneous graph composed of text vectors and words, and then uses The graph convolutional neural network extracts the relationship information between documents, and after obtaining the document vector containing semantic features and relationship information, it is input into the GRUs-based decoder model for multi-label recommendation. Experimental results show that our proposed method significantly outperforms methods based only on semantic models in terms of accuracy, recall and other indicators.

Keywords

Label recommendation, graph convolutional network, recurrent neural network, natural language processing, text content.

1. Introduction

With the development of the Internet and the advancement of the globalization process, the total amount of text information in the Internet is increasing rapidly, and how to efficiently manage and retrieve these data has become a concern of the industry [1]. Studies have shown that by adding tags to massive text information, the information can be organized and managed effectively [2]. Adding labels to a large amount of data manually is unacceptable in terms of efficiency and cost, so how to automatically generate labels for text has gradually attracted the attention of researchers.

Traditional label recommendation methods are usually based on collaborative filtering [3] and text content model [4-6]. The former recommends new labels based on the user's historical label records, while the latter recommends labels based on text content, such as TF-IDF (term frequency–inverse labeling). document frequency) [4], N-grams [5] and other methods by extracting text keywords as text labels; or using topic models such as LDA (Latent Dirichlet allocation) [6,7] to extract text topics information, and then recommend tags based on topic information between different texts. Traditional label recommendation methods usually rely

ISSN: 2664-9640

on artificial feature design, which is expensive and difficult to capture deeper implicit information.

With the development of neural networks and semantic models, label recommendation methods based on deep semantic models have gradually become mainstream methods. Compared with traditional label recommendation methods, deep neural networks can extract deeper semantic information from texts [8]. For example, Gong et al. [8] proposed a label recommendation method based on CNN (Convolutional Neural Networks), which calculates document embeddings and performs label recommendation through the CNN model; Li et al. [9] proposed a method based on LSTM (Long Short-Term Memory).) text label recommendation method, this method inputs the text into the LSTM model, uses the hidden vector output by the last LSTM unit as the text vector, and combines the LDA keyword to recommend the label through the attention mechanism; Adhikari et al. [10] proposed A BERT-based text label recommendation method is proposed, which models the text content through the BERT (Bidirectional Encoder Representations from Transformers) model, uses the CLS vector output by the BERT model as the text encoding and predicts the label. The calculation process of these methods is basically the same. First, the neural network is used to model the text content, extract the text semantic information, and then perform tag recommendation based on it.

The method of recommending tags only based on textual semantic information has certain defects: texts with the same tags are not necessarily semantically similar. For example, the two sentences "How to learn the spring framework?" and "Detailed explanation of the GC mechanism of JAVA" are completely different in semantics, but both can recommend the "JAVA" tag for them. Therefore, in addition to semantic-based methods, there are also methods to recommend labels for documents through other information, such as Yao et al. [11] proposed Text-GCN, which works by modeling the entire document collection as a document-word heterogeneity Graph structure, using GCN (Graph Convolutional Network) [12] to extract the structural information of the whole network to recommend labels. Since the set relationship of documents is implied in this heterogeneous graph structure, and the set relationship of documents can be used for text label recommendation, the experimental results show that Text-GCN has better performance in single-label recommendation tasks.

However, these methods also have certain defects. They often use less text semantic information, such as Text-GCN, which uses all 0 initialization for document nodes, one-hot encoding for words, and only relies on GCN to extract heterogeneous graph structure information. The document vector is generated without considering the word order of the text, so a lot of text content information is lost. Related research shows that the word order of text is crucial information for text content [9], so this method affects the effect of tag recommendation. In addition, when multi-label recommendation is made for text, there is usually a certain correlation between the labels. For example, a text about machine learning may have three labels of "python", "tensorflow" and "machine-learning" at the same time. While most of the existing multi-label recommendation methods do not consider this, many methods only use simple activation functions for multi-label recommendation, such as softmax.

To solve the above problems, we propose a GCN-based multi-label recommendation method. Specifically, we use a Gate Recurrent Unit (GRU) neural network as the content encoder of the document, and the hidden state output of the last GRU unit is used as the vector of the whole sentence. Then we construct a word-document heterogeneous graph, where the document nodes are encoded using the document encoding output from the content encoder and the word nodes are encoded using one-hot encoding, and each document node is not directly connected to each other but to the corresponding word node. The relationship between documents is then modeled by performing graph convolution operations on this heterogeneous graph, and its output, the text vector output from the graph convolution neural network is fed into a GRU-

ISSN: 2664-9640

based decoder, and finally the hidden state output from the decoder is used for label recommendation.

This approach extracts both the sequential information of the text and the structural information of the text collection, which is helpful to improve the label recommendation. At the same time, the GRU-based decoder uses the predicted label information when making multi-label recommendation, which makes a certain correlation between the recommended multiple labels. Our main contributions are as follows:

A textual multi-label recommendation model that considers both textual content and inter-textual relationship information is proposed, and the effectiveness of our approach is demonstrated experimentally on three datasets.

We verify the contribution of inter-label correlations to the recommendation effectiveness in multi-label recommendation tasks.

We investigate the impact of inter-textual relationship information on the textual label recommendation task.

2. Related work

According to the way to encode the text, current text labeling methods can be divided into traditional semantic-based label recommendation methods and deep semantic-based label recommendation methods.

2.1. Traditional semantic-based tag recommendation methods

Traditional semantic-based tag recommendation methods refer to the use of traditional methods to extract text content and perform tag recommendation, which usually rely on human feature design and have a low level of extracted features. For example, Zhang et al [13] designed a TF-IDF method for text classification, this method improves the traditional TF-IDF method using manually extracted feature words, etc. and synonyms to recommend tags for documents for target users.Krestel et al [6] used LDA for tag recommendation for text, this method first calculates the topic words of the text, and then based on different topic words and Mashal et al [14] designed a K-nearest neighbor based text tag recommendation method, which first extracts the subject words of the text and then clusters the documents using K-nearest neighbors based on the distribution between different document subject words to recommend suitable tags for the documents.

2.2. Deep semantic-based label recommendation method

Do not number your paper: All manuscripts must be in English, also the table and figure texts, otherwise we cannot publish your paper. Please keep a second copy of your manuscript in your office. When receiving the paper, we assume that the corresponding authors grant us the copyright to use the paper for the book or journal in question. Should authors use tables or figures from other Publications, they must ask the corresponding publishers to grant them the right to publish this material in their paper. Use italic for emphasizing a word or phrase. Do not use boldface typing or capital letters except for section headings (cf. remarks on section headings, below).

With the development of deep neural network technology, the deep semantic-based tag recommendation method gradually becomes the mainstream text tag recommendation method. Compared with traditional methods, neural networks can automatically extract semantic information from text and learn deep features that are not accessible by manual feature design.

ISSN: 2664-9640

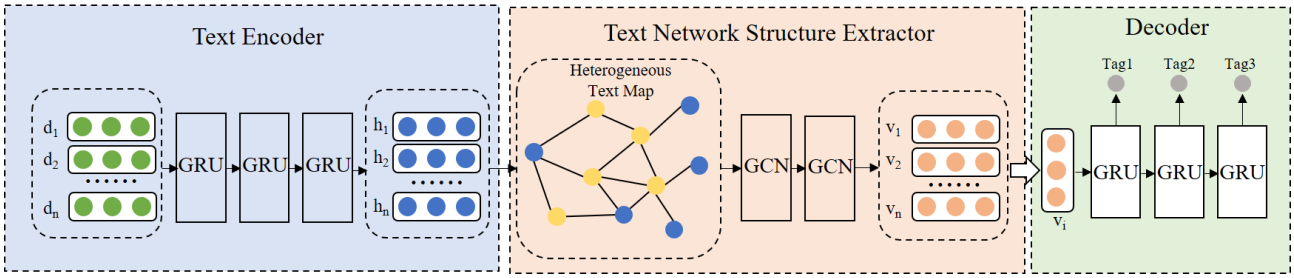

Figure 1: Overall framework

For example, Gong et al [15] proposed a text label recommendation model based on CNN and attention mechanism, which uses CNN to learn the embedding representation of a document and then uses the document representation to recommend labels for text. li et al [9] proposed a text label recommendation model based on LSTM, which combines text topics and text features for label recommendation through attention mechanism. tang et al [16] proposed a text label recommendation model based on LSTM, which combines text topics and text features for label recommendation through attention mechanism. et al [16] proposed a textual multi-label recommendation model based on multilayer GRU and attention mechanism, which considered three main pillars in multi-label recommendation task: sequential text encoding, label relevance, and label-text overlap, and achieved better results. In recent years, the emergence of pre-trained language models such as BERT has stimulated the development of deep semantic techniques. Adhikari et al. investigated the use of BERT for textual label recommendation tasks, and their results showed that BERT can significantly improve the effectiveness of label recommendation. In addition, there are also methods that use neural networks to capture non-semantic information for tag recommendation, for example, Yao et al [11] proposed a text coding model called Text-GCN, which models the overall structure of the document collection and the document content, and uses the overall structure information of the document collection for tag recommendation better than using the semantic information of a single document.

Most of the existing methods use only the content information of the text, such as semantic model, keywords, and subject terms; or use only the structural information of the text collection, such as Text-GCN and KNN(k-nearest neighbors), without considering combining both types of information, thus limiting the performance of the model.

3. Framework and Methodology

Figure 1 shows the general framework of our model, which consists of three main parts.

GRU-based text encoder: This part completes the encoding of the text, firstly, the text is converted into word vectors by pre-trained word2vec, and then it is fed into this text encoder, and the hidden state output of the last GRU unit is taken as the encoding vector of the text. In the figure, d1~dn refer to the input document and h1~hn the hidden state output by the encoder.

GCN-based text network structure extractor: After obtaining the encoding vectors of all documents, in order to extract the relationships between texts, we construct a heterogeneous graph of documents and words, where texts are connected to the word nodes they own, and word nodes are connected to documents and other words. The updated encoding vector contains both the content information of the text and the structure information of the whole text collection. In the figure, v1~vn refers to the document vector after combining the network structure information.

GRU-based decoder: After aggregating document information by GCN, we input the aggregated document vector into this decoder, and then input the state vector of each GRU unit into the

ISSN: 2664-9640

linear layer for multi-label recommendation. The vi in the figure refers to any one of the total document vectors, i.e., the decoder can only make label recommendation for one document at a time.

3.1. GRU-based text encoder

Since RNNs use a cyclic structure to read data, they are particularly suitable for modeling sequential information. To extract the sequential information of the text, we use a variant of RNN, GRU, as a text encoder, which is computed as follows. First, all words of the text are converted into corresponding word vectors by a pre-trained word vector model:

$$v_i = word2evc(w_i)$$

where w_i is the i-th word of the input document and v_i is the word vector corresponding to the i-th word.

Then the word vectors are sequentially input into the GRU model, and finally, the document state vector output by the encoder is used as the encoding vector for the whole document:

$$s, h = GRU(v)$$

where $v = [v_1, v_2, \dots, v_n]$, s is the GRU output state vector, h is the GRU output hidden vector, and we use v as the encoding vector for the whole document. The encoder can only compute the vectors of one document at a time, so it needs to compute the encoding vectors of all documents in turn before starting to build the document-word heterogeneous graph.

3.2. GCN-based Text Network Structure Extractor

After calculating the content encoding of each text, we model the whole text collection by a document-word heterogeneous graph, and then extract the structure information of the text collection by performing graph convolution operations on this heterogeneous graph.

Specifically, the heterogeneous graph in our model consists of two different nodes, document nodes and word nodes, where word nodes refer to all the words in the entire document collection. For document nodes, we use text vectors generated by text encoders as node values, and for word nodes, we use one-hot encoding.

For the edges in the graph, there are only document-word and word-word edges, i.e., documents are not directly connected to each other, they can only be connected by co-occurring words between each other. The weights between word nodes are computed using PMI (Point-wise Mutual Information) and the weights between words and documents are computed using TF-IDF. Specifically, for any two nodes i and j, the specific rules for calculating the weights between them are as follows.

$$\begin{cases} PMI(i,j) \; i,j \; is \; word，\; PMI(i,j) > 0 \\ TF - IDF_{i,j} \;\; is \; documnet，\; j \; is \; word \\ 1 \;\; i = j \\ 0 \;\; other \end{cases}$$

The formula for calculating PMI is as follows:

$$PMI(i,j) = log \frac{p(i,j)}{p(i)p(j)}$$

$$p(i,j) = \frac{\#S(i,j)}{\#S}$$

$$p(i) = \frac{\#S(i)}{\#S}$$

ISSN: 2664-9640

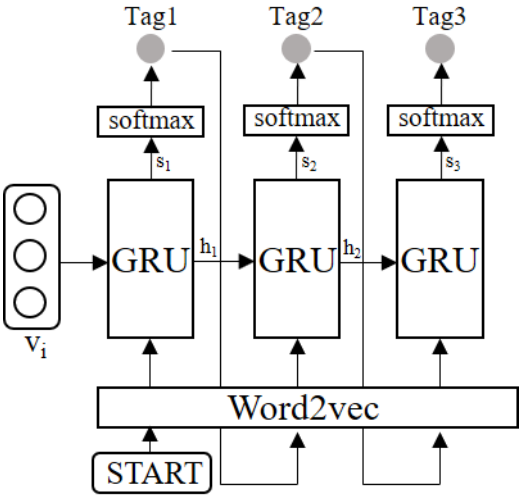

Figure 2: Decoder structure

where $\#S(i)$ is the total number of sliding windows containing word i in the whole corpus, $\#S(i,j)$ is the total number of sliding windows containing both word i and j, and $\#S$ is the total number of all sliding windows in the whole corpus.

A higher value of PMI means a higher correlation between two words, while a negative value of PMI means a low or no semantic correlation between two words. Therefore, the weight is set to 0 when PMI is negative.

After constructing the word-document heterogeneous graph, we update the vectors of each node by performing graph convolution calculation on this graph, and for the first layer, the graph convolution is calculated by the following formula.

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

where $\tilde{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, is the normalized adjacency matrix, X is the initial value of each node, W_0 is the weight matrix, and ρis the activation function. In the method of this thesis, $\rho(x) = ReLU(x)$.

For the other layers, the graph convolution is calculated as follows.

$$L^{(i+1)} = \rho(\tilde{A}L^{(i)}W_i)$$

where $L^{(i)}$ is the node state of the previous layer, W_i is the weight matrix of this layer, and $L^{(i+1)}$ is the updated node value of this layer.

In the graph convolution operation, each node only exchanges information with its own neighboring nodes, and a total of two layers of graph convolution computation are performed in order to enable individual document nodes to exchange information with each other. Compared with methods that rely only on document-word heterogeneous graphs to model documents, our method preserves the information of text order and can effectively improve the performance of the label recommendation task.

3.3. GRU-based text decoder

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Existing multi-label recommendation methods usually treat multi-label recommendation tasks as multiple single-label recommendation tasks, but in fact there is often some correlation

ISSN: 2664-9640

between multiple labels in multi-label recommendation tasks. Therefore, we use GRU as a decoder for multi-label recommendation, and after each recommended label, the

Decoder structure.

recommended label will be input to the next GRU unit for the next label recommendation, so that there is a certain correlation among the labels recommended by the model. For any GRU unit, its update formula is as follows.

$$h_i, s_i = GRU(x_i, h_{i-1})$$

where $h_{i-1}$is the hidden vector of the previous GRU unit, x_i is the external input accepted by the current GRU unit, h_i is the hidden vector of the output of the current GRU unit, which will be input to the next GRU unit, and s_i is the state vector of the output of the current GRU, which will be used to predict the current Tag. The structure of the whole decoder is shown in Figure 2.

The decoder accepts the text encoding vector and the vector corresponding to the START tag and outputs the hidden vector hi and the state vector s_i, where the hidden vector is sent to the next GRU and the state vector is sent to the softmax layer for tag prediction. When the labels are obtained, they are converted into word vectors and used as external inputs for the next GRU unit. The prediction is cycled until the upper limit of the set number of predictions is reached or the END label is predicted.

3.4. Loss function

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled.

Since we perform a multi-label recommendation task, we use cross-entropy as the loss function to train the model by minimizing the cross-entropy of the classification task. The specific formula of the loss function is as follows.

$$loss = -\sum_{d_i}\sum_{t_j} t_j \log p(t_j')$$

where d_i refers to the i-th document, t_j refers to the j-th label of the i-th document, t_j' is the currently predicted label, and $p(t_j')$is the probability of the currently predicted label.

4. Experiment

4.1. Experimental setting

4.1.1. Datasets

To ensure the validity of our approach, we test the method of this thesis on three different datasets, each consisting of a question and several tags related to the question, and the details of each dataset are shown in Table 1.

Ask Ubuntu (AU): a question and answer forum for Ubuntu developers

StackOverflow (SO): a well-known computer-related Q&A site.

Mathematics Stack Exchange (math): a question and answer site for mathematics learners and researchers.

To ensure computational efficiency, all data sets were processed in the following steps: first, we removed punctuation marks and converted them to all lowercase, then we split them into words and removed deactivated words, and then we removed low-frequency words that occurred less than 5. The reason for removing low-frequency words is that all three datasets are from developer Q&A sites, and there are a large number of code blocks, error messages,

ISSN: 2664-9640

abbreviations and other meaningless words in the original text. On the other hand, the large number of vocabulary nodes will make it impossible to construct GCN models.

Similarly, since there is also a large amount of meaningless content in the tags of the text that appears only a few times, high frequency tags are intercepted according to the frequency of tags and the total number of tags, and the number of tags shown in Table 1 is the value after interception.

Table 1: Three Scheme comparing

Dataset name	Dataset size	Number of words	Number of nodes	Number of tags
AU	65653	18508	71 030	600
SO	57984	22730	69 117	200
math	64398	26214	78 200	717

4.1.2. Experiment grouping

The section headings are in boldface capital and lowercase letters. Second level headings are typed as part of the succeeding paragraph (like the subsection heading of this paragraph). All manuscripts must be in English, also the table and figure texts, otherwise we cannot publish your paper. Please keep a second copy of your manuscript in your office. When receiving the paper, we assume that the corresponding authors grant us the copyright to use the paper for the book or journal in question. When receiving the paper, we assume that the corresponding authors grant us the copyright to use the paper for the book or journal in question. When receiving the paper, we assume that the corresponding authors grant us the copyright to use.

Our method first encodes the text by GRU, then extracts the document collection structure information by GCN, and finally recommends tags by GRU decoder. In order to study the performance difference of other RNN models in our method, we additionally set up experimental groupings using LSTM as encoder and decoder. According to the different encoder-decoder models used, these two groups are named **GRU-GCN** and **LSTM-GCN**, respectively.

In addition, we set up three ablation experimental subgroups to study the enhancement mechanism of the method in this thesis, which are named and implemented as follows:

Remove text encoder (no-encoder): this subgroup does not use the GRU-based text encoder to pre-encode the text, but is similar to Text-GCN, which directly initializes the text nodes with all-0, computes the document encoding using GCN only, and then feeds it into the decoder for label recommendation.

Remove GCN (no-GCN): this grouping does not use GCN to extract text collection structure information, and uses only the encoder and decoder for label recommendation.

Remove decoder (no-decoder): This grouping does not use the decoder based on GRUs, and inputs the text encoding directly into the softmax layer for multi-label recommendation after obtaining the text encoding.

4.2. Baseline

In order to compare with our proposed method and verify its effectiveness, the following methods are set as baselines in this thesis:

Maxide[17]: this method is a traditional multi-tag recommendation method, where the feature matrix is filled by matrix operations thus recommending tags for the text.

ISSN: 2664-9640

Tagspace[8]: this method models the text based on a CNN model, obtains a text vector and then calculates its relevance to all tags by a tag score function, and then performs ranking and recommendation.

LSTM[9]: this method models the text using an LSTM model and then inputs the text vector into the softmax layer to calculate the probability of all candidate tags.

TLSTM[9]: this method combines the hidden vector output from LSTM and the topic words generated by the LDA model in order to compute the document vector through the attention mechanism, and then inputs it into the softmax layer for multi-label recommendation.

Text-GCN[11]: this method extracts the structure of the whole text collection by GCN to calculate the encoding vector of the text, and then inputs it into the softmax layer for multi-label recommendation.

4.3. Hyperparameter setting

For the word vector model, we use word2vec based on pre-training using the wiki corpus, and the embedding dimension of the word vector is set to 300. for the GRU text encoder, the maximum input text length is set to 50, the embedding dimension is set to 256, and the dropout rate is 0.2. for the GRUs decoder, the maximum prediction length is set to 5, and the dropout rate is 0.2.

For GCN, we use two layers of convolution, the embedding size of the first convolution layer is 300, the window size is 20, and the dropout rate is 0.5. This setting refers to the settings of GCN and Text-GCN, and the reason for using two layers of convolution is that in each graph convolution operation, each node can only exchange information with its direct neighbor nodes, and it needs to perform two graph convolution operations for the document nodes to pass information between each other. And related research shows that setting more convolution layers has no positive effect on the performance improvement of the model, so we use two layers of convolution.

The optimizer uses adam, and the initial learning rate is set to 0.02. To ensure the efficiency of training, we use early stopping. in addition, to avoid the decoder from falling into the local optimal solution, we use beam search in the testing phase to improve the results, and for beam search, we set the beam width size to 3. in addition, to In addition, to ensure the reliability of the results, we randomly disrupt the data before each training, take 20% of the data as the test, and take 10% of the remaining 80% of the data as the validation set.

4.4. Evaluation metrics

We use a total of three metrics to validate the effectiveness of the proposed method, namely accuracy, recall and F1. where accuracy denotes the proportion of target tags in the recommended tags and is given by the following equation.

$$\text{Precision@k} = \frac{|\widehat{tag}_{ki} \cap tag_i|}{\widehat{tag}_{ki}}$$

Recall represents the proportion of the recommended tags in the target tags, and its formula is as follows:

$$\text{Recall@k} = \frac{|\widehat{tag}_{ki} \cap tag_i|}{tag_i}$$

Table 2: Three Scheme comparing

Data set	model	@1			@3			@5		
		P	R	F1	P	R	F1	P	R	F1
AU	Maxide	0.117	0.049	0.069	0.095	0.124	0.108	0.079	0.168	0.107
	Tagspace	0.084	0.03	0.044	0.072	0.079	0.075	0.064	0.118	0.083

ISSN: 2664-9640

	LSTM	0.339	0.170	0.226	0.211	0.307	0.250	0.165	0.412	0.236
	TLSTM	0.433	0.183	0.257	0.284	0.399	0.332	0.221	0.427	0.291
	Text-GCN	0.368	0.179	0.241	0.233	0.351	0.280	0.193	0.433	0.267
	LSTM-GCN	0.413	0.215	0.283	0.268	0.377	0.313	0.201	0.431	0.274
	GRU-GCN	0.421	0.220	0.289	0.266	0.390	0.316	0.233	0.447	0.306
SO	Maxide	0.120	0.050	0.071	0.101	0.097	0.099	0.074	0.153	0.100
	Tagspace	0.134	0.061	0.084	0.077	0.103	0.088	0.069	0.159	0.096
	LSTM	0.541	0.311	0.395	0.301	0.537	0.386	0.207	0.634	0.312
	TLSTM	0.604	0.311	0.411	0.364	0.558	0.441	0.251	0.642	0.361
	Text-GCN	0.567	0.345	0.429	0.323	0.574	0.413	0.281	0.693	0.400
	LSTM-GCN	0.594	0.367	0.454	0.344	0.608	0.439	0.251	0.657	0.363
	GRU-GCN	0.611	0.371	0.462	0.348	0.611	0.443	0.243	0.688	0.359
Math	Maxide	0.113	0.049	0.071	0.096	0.12	0.107	0.079	0.168	0.107
	Tagspace	0.154	0.061	0.084	0.109	0.139	0.122	0.09	0.185	0.121
	LSTM	0.546	0.27	0.395	0.311	0.414	0.355	0.211	0.539	0.303
	TLSTM	0.589	0.281	0.411	0.331	0.459	0.385	0.234	0.561	0.330
	Text-GCN	0.551	0.307	0.429	0.327	0.476	0.388	0.228	0.566	0.325
	LSTM-GCN	0.559	0.344	0.426	0.312	0.499	0.384	0.237	0.610	0.341
	GRU-GCN	0.587	0.331	0.462	0.336	0.511	0.405	0.240	0.591	0.341

where $\widehat{tag}_{ki}$ denotes the top-k tags recommended by the model for the i-th article, and tag_i denotes the set of real tags of the article.

The F1 value, on the other hand, integrates the accuracy and recall, which is calculated as follows.

$$\mathrm{F1} = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

5. Results and Analysis

5.1. Experimental results

Table 2 shows the main experimental results, demonstrating the performance comparison between our proposed method and its variants with each baseline, where P refers to Precision and R refers to Recall.

5.2. Analysis of the results

As can be seen from Table 2, our proposed method works better than the traditional non-neural methods, methods using only text semantic information and methods using only text structural information. The best variant GRU-GCN improves the accuracy by 0.109 on average, the recall by 0.124 on average, and the F1 by 0.113 on average on the AU dataset; it improves the accuracy by 0.133 on average, the recall by 0.208 on average, and the F1 by 0.149 on average on the SO dataset; and it improves the Math dataset by 0.123 accuracy. The experimental results show that our proposed method can effectively combine the semantic information of the text and the structural information of the text collection to improve the effectiveness of text label recommendation.

ISSN: 2664-9640

Table 3: Ablation Experiment Results

Data set	model	@1			@3			@5		
		P	R	F1	P	R	F1	P	R	F1
AU	no-encoder	0.387	0.207	0.270	0.387	0.207	0.270	0.387	0.207	0.270
	no-GCN	0.213	0.106	0.142	0.213	0.106	0.142	0.213	0.106	0.142
	no-decoder	0.357	0.143	0.204	0.357	0.143	0.204	0.357	0.143	0.204
	LSTM-GCN	0.413	0.215	0.283	0.413	0.215	0.283	0.413	0.215	0.283
	GRU-GCN	0.421	0.220	0.289	0.421	0.220	0.289	0.421	0.220	0.289
SO	no-encoder	0.576	0.367	0.448	0.576	0.367	0.448	0.576	0.367	0.448
	no-GCN	0.312	0.204	0.247	0.312	0.204	0.247	0.312	0.204	0.247
	no-decoder	0.534	0.297	0.382	0.534	0.297	0.382	0.534	0.297	0.382
	LSTM-GCN	0.594	0.367	0.454	0.594	0.367	0.454	0.594	0.367	0.454
	GRU-GCN	0.611	0.371	0.462	0.611	0.371	0.462	0.611	0.371	0.462
Math	no-encoder	0.541	0.317	0.400	0.541	0.317	0.400	0.541	0.317	0.400
	no-GCN	0.382	0.205	0.267	0.382	0.205	0.267	0.382	0.205	0.267
	no-decoder	0.521	0.334	0.407	0.521	0.334	0.407	0.521	0.334	0.407
	LSTM-GCN	0.559	0.344	0.426	0.559	0.344	0.426	0.559	0.344	0.426
	GRU-GCN	0.587	0.331	0.462	0.587	0.331	0.462	0.587	0.331	0.462

Compared with the variant using LSTM, the experimental group using GRU as encoder and decoder has better results, and the F1 of GRU-GCN is on average 0.014, 0.003, and 0.019 higher than LSTM-GCN on the three datasets. this may be due to the long input text in the label recommendation task, and the multi-loop input makes the model vulnerable to gradient disappearance/gradient explosion impact. Compared with LSTM, GRU simplifies the model structure, reduces the model parameters, and is less prone to overfitting in inputs of the same length, resulting in better results.

Our proposed method works best on the SO dataset, which may be due to the shorter average length of the SO dataset, which makes it easier to read the complete text for semantic extraction when the decoder input length is fixed, and therefore better; on the contrary, the AU dataset has the longest average length, which makes it more difficult for the model to read the complete text and thus lose the semantic data, which in turn hurts the model performance.

This is because the average number of tags in the three corpora is 2 to 3. When only one tag is recommended, the model can easily recommend one of the tags and thus has a high accuracy, but cannot recommend all of them, which leads to a low recall and thus reduces F1. On the contrary, when 5 tags are recommended, the model can easily recommend all of them, but also has a large number of invalid recommendations, which in turn reduces F1. which reduces the precision and thus leads to a lower F1. This suggests that the nature of the dataset affects the choice of the prediction window size.

5.3. Ablation experiment

5.3.1. Experimental results

We set up an ablation experiment to verify the effectiveness of the method in this paper, and the results are shown in Table 3.

ISSN: 2664-9640

5.3.2. Result Analysis

When the encoder is removed, the encoding method of the model is similar to Text-GCN, i.e., only the structural information of the text collection is extracted by GCN to generate the document vector, and the encoded vector is obtained and then fed into the decoder for label recommendation. The results show that the performance degradation of the experimental group on the single-label recommendation task is small, with F1 degradation of 0.019, 0.014 and 0.023 on the three datasets, respectively. but significant performance degradation is observed in both 3-label prediction and 5-label prediction.F1 decreases by 0.074, 0.092 and 0.082 on the three datasets for a recommendation window size of 3, respectively. The decline is more pronounced for recommendation window size 5, with F1 decreasing by 0.090, 0.123, and 0.126 for the three datasets, respectively.

When the GCN is removed, the structure of the model is similar to that of the seq2seq model. From the experimental results, it can be seen that the performance of the model decreases significantly when the text structure information is extracted without GCN. when the recommendation window is 1, the model performance decreases relatively little, and F1 decreases by 0.147, 0.215 and 0.156 on the three data sets, respectively. when the label prediction window is 3 and 5, the model performance decreases more significantly, and F1 decreases by The performance of the model decreases even more when the recommendation window size is 5, with F1 decreasing by 0.165, 0.172 and 0.119 for the three datasets, and the performance decreases the most among the three sets of ablation experiments, which indicates that the extracted text set structure information plays an important role in the text label recommendation task. In addition, after removing the GCN, the model performance degradation is larger for recommending 1 tag and 3 tags, and smaller for recommending 5 tags. According to the experimental results in the previous subsection, GCN works better in single-label classification tasks, while the recommendation tasks with multiple labels rely more on semantic information, so the impact on multi-label recommendation tasks is smaller when the module is removed.

When the decoder was removed and the sotfmax layer was used directly for multi-label recommendation, the performance of the model also showed significant degradation, where the performance of single-label recommendation was less affected, with F1 dropping by 0.085, 0.080 and 0.016 for the three datasets when the recommendation window size was 1, while the performance of multi-label recommendation dropped more, with F1 dropping by 0.107, 0.108 and 0.067 for the three datasets when the recommendation window size was 3. When the recommendation window size is 3, F1 decreases by 0.107, 0.108 and 0.067 for the three datasets, respectively; when the recommendation window size is 5, F1 decreases by 0.097, 0.114 and 0.146 for the three datasets, respectively. This is due to the fact that the mechanism of softmax is only suitable for the single-label recommendation task. When using softmax for label recommendation, only one label can be selected as a positive sample for training, and the prediction is selected according to the probability of each label from highest to lowest. This makes the model work properly for single-label recommendation tasks, while for multi-label recommendation tasks, it is difficult for the model to learn features of multiple labels at the same time and to make effective multi-label recommendations.

6. Conclusion

We proposed a text multi-label recommendation method based on graph convolutional neural network by using a GRU-based text encoder and a GCN-based text collection structure extractor to encode both the semantic information of text and the structure of the entire text collection, and a GRU-based decoder for multi-label recommendation. Compared with the methods using only text semantic information and only text collection structure, our method can extract richer

ISSN: 2664-9640

text information in layers and can effectively improve the effectiveness of text multi-label recommendation. Experiments on three datasets show that our proposed method improves F1 by 0.067~0.237 compared to the methods using only text semantic information or only text collection structure information. proving the effectiveness of our method.

Our method relies on the text set structure information extracted by GCN, and due to the limitation of the computational method of GCN, the whole network needs to be re-computed when the nodes of the graph change. In addition, GCN requires the whole complete graph to be computed, i.e., all document node vectors need to be computed before starting to build the heterogeneous graph, and the performance of the model is greatly affected when the data size is large, which limits the application of our method, so our next work will try to use techniques such as SageGCN to improve the computational volume problem of the whole heterogeneous graph.

References

[1] Internet world statistics on: http://www.. Internetworldstats.com.

[2] Q Xie, Y Zhu, F Xiong, et al: Interactive resource recommendation with optimization by tag association and significance analysis, Neurocomputing, vol. 391(2020), p. 210-219.

[3] Mishne G: Autotag: A collaborative approach to automated tag assignment for weblog posts, Proceedings of the 15th International Conference on World Wide Web(New York, 2006), vol. 1, p. 953–954,.

[4] Pujari M, Kanawati R: Tag recommendation by link prediction based on supervised machine learning, Proceedings of the 6th International Conference on Weblogs and Social Media(Dublin , 2012), vol. 1, p. 547-550.

[5] Martins E F, Belém F M, Almeida J M, et al: On cold start for associative tag recommendation, Journal of the Association for Information Science and Technology, vol. 67(2016)No.1: p. 83-105.

[6] Krestel R, Fankhauser P, Nejdl W: Latent dirichlet allocation for tag recommendation, Proceedings of the 3rd ACM Conference on Recommender Systems(New York, 2006), vol. 1, p. 61-68.

[7] Ramage D, Hall D, Nallapati R, Manning C D: Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora, Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics(Singapore, 2009), vol. 1, p. 248-256.

[8] Weston J, Chopra S, Adams K: #Tagspace: Semantic embeddings from hashtags, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics(Qatar, 2014), vol. 1, p. 1822-1827.

[9] Li Y, Liu T, Jiang J, L Zhang: Hashtag recommendation with topical attention-based LSTM, Proceedings of the 26th International Conference on Computational Linguistics. Association for Computational Linguistics(Osaka, 2016), vol.1 pp. 3019-3029.

[10] Adhikari, A., Ram, A., Tang, R., et al: Docbert: Bert for document classification, arXiv preprint arXiv:1904.08398, 2019.

[11] Yao, L., Mao, C., & Luo, Y: Graph convolutional networks for text classification, In Proceedings of the AAAI conference on artificial intelligence(Honolulu, July 2019), Vol. 33, No. 01, p. 7370-7377.

[12] Kipf, T. N., & Welling, M. , "Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

[13] Yun-tao, Z., Ling, G., & Yong-cheng, W.: An improved TF-IDF approach for text classification, Journal of Zhejiang University-Science A, vol. 6(2005) No.1, p. 49-55,.

[14] Mashal I, Alsaryrah O, Chung T Y: Testing and evaluating recommendation algorithms in internet of things, Journal of Ambient Intelligence and Humanized Computing, vol. 7(2016) No.6, p.889-900.

ISSN: 2664-9640

[15] Gong Y, Zhang Q: Hashtag recommendation using attention-based convolutional neural network," Proceedings of the 25th International Joint Conference on Artificial Intelligence(New York, July 2016),vol.1, p. 2782-2788.

[16] Tang, S., Yao, Y., Zhang, S., et al: An integral tag recommendation model for textual content, In Proceedings of the AAAI Conference on Artificial Intelligence(Honolulu, July 2019), Vol. 33, No. 01, p. 5109-5116.

[17] Xu M, Jin R, Zhou Z: Speedup matrix completion with side information: Application to multi-label learning, Proceedings of the 26th International Conference on Neural Information Processing Systems(Lake Tahoe , 2013),vol.1, p. 2301-2309.