Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 1787369, 9 pages
https://doi.org/10.1155/2022/1787369

Research Article

A Novel Keyword Generation Model Based on Topic-Aware and Title-Guide

Jialin Ma, Jieyi Cheng, and Yue Zhang

College of Computer Science, Huaiyin Institute of Technology, Huaian 223003, China

Correspondence should be addressed to Jieyi Cheng; 19225106@hyit.edu.cn

Received 15 March 2022; Revised 19 April 2022; Accepted 22 April 2022; Published 23 May 2022

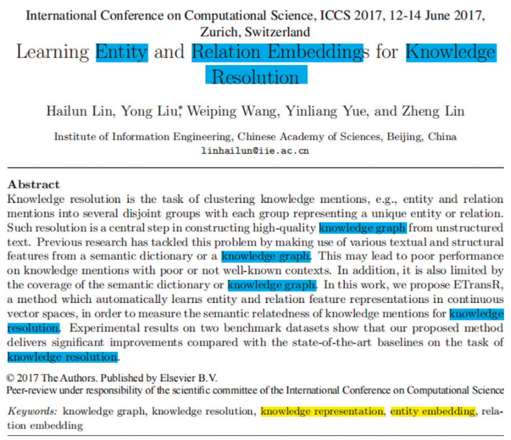
Academic Editor: Dalin Zhang

Keywords are usually one or more words or phrases that describe the subject information of the document. The traditional automatic keywords extraction methods cannot obtain the keywords which do not appear in the document, and the semantic information is not considered in the extraction process. In this paper, we introduce a novel Keyword Generation Model based on Topic-aware and Title-guide (KGM-TT). In the KGM-TT, the neural topic model is used to identify the latent topic words, and a hierarchical encoder technology with attention mechanism is able to encode the title and its content, respectively. The keywords are generated by the recurrent neural network with attention and replication mechanism in our model. This model can not only generate the keywords which do not appeared in the source document but also use the topic information and the highly summative word meaning in the title to assist the generation of keywords. The experimental results show that the *F*1 value of this model is about 10% higher than that of CopyRNN and CopyCNN.

1. Introduction

Keyword is the smallest unit to express the subject meaning of a document. It is widely used in tasks such as information retrieval, documents classification, and opinion mining [1]. With the rapid development of the Internet, the number of electronic documents is increasing exponentially. Various methods of automatically extracting and generating keywords have been proposed. The main step of traditional automatic keyword extraction methods (such as n-gram, TF-IDF, TextRank [2], SingleRank [3], and KEA [4]) is to identify candidate words and then rank all the candidates based on the importance computed by either unsupervised ranking approaches or supervised machine-learning approaches [5]. The keywords extracted by the above methods are words that have appeared in the source file, such as the keywords "knowledge graph," "knowledge resolution," and "relation embedding" in Figure 1. However, according to relevant research statistics, nearly half of the keywords given by the author in the scientific articles do not appeared in the source document [6], such as the keywords "knowledge representation" and "entity embedding" in Figure 1. Therefore, these kinds of extracting methods cannot generate keywords that do not appear in the source document, and they are difficult to mine the deep semantic information between context words.

To overcome the above drawbacks, some researchers had proposed a Sequence-to-Sequence model to solve the keyword generation task and had achieved some results, including seq2seq, CopyRNN [6], CorrRNN [7], and CopyCNN [8]. First, these methods treat the title and the main body content equally and just combine them as the source document input. Then, the encoder is used to maps each document word to a hidden state vector as the context representation. Finally, a decoder based on context representation is used to generate keywords from a predefined vocabulary [9]. However, these generation methods still have some flaws. They do not consider whether keywords exist in the document but directly extract them from the vocabulary, so they can generate keywords that do not appear in the source document. In fact, the title is the core subject condensed by the document author. However, they



International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland

Learning Entity and Relation Embeddings for Knowledge Resolution

Hailun Lin, Yong Liu*, Weiping Wang, Yinliang Yue, and Zheng Lin

Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
linhailun@iie.ac.cn

Abstract
Knowledge resolution is the task of clustering knowledge mentions, e.g., entity and relation mentions into several disjoint groups with each group representing a unique entity or relation. Such resolution is a central step in constructing high-quality knowledge graph from unstructured text. Previous research has tackled this problem by making use of various textual and structural features from a semantic dictionary or a knowledge graph. This may lead to poor performance on knowledge mentions with poor or not well-known contexts. In addition, it is also limited by the coverage of the semantic dictionary or knowledge graph. In this work, we propose ETransR, a method which automatically learns entity and relation feature representations in continuous vector spaces, in order to measure the semantic relatedness of knowledge mentions for knowledge resolution. Experimental results on two benchmark datasets show that our proposed method delivers significant improvements compared with the state-of-the-art baselines on the task of knowledge resolution.

© 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the International Conference on Computational Science

Keywords: knowledge graph, knowledge resolution, knowledge representation, entity embedding, relation embedding

Figure 1: An example of keyword generation.

ignore the guiding role of title on keyword generation and fail to give full play to the strong semantic information of document title.

The title is a highly purified and refined for the document, which can accurately reflect the nature, scope, and depth of the content of the document. Keywords are words that can express the characteristics of document subject content, and they play a similar and complementary role with title [10], so keywords and titles have similar semantic information. Taking an academic paper as an example, the title is usually straightforward, focused, concise, and directly expounds the research object, research method, and research purpose of the paper. Therefore, one or several main central words in the title can often be directly used as keywords [4]. As shown in Figure 1, "relation embedding" in keywords appears in the title but not in the body. In addition, the information in the title also helps to reflect which parts of the body of the document are important, such as including the same or related parts as the title.

In view of the fact that the existing extractive methods does not fully consider the semantics of context and the generative models ignore the guiding role of title, in this paper, we propose a Keyword Generation Model based on combination Topic-aware and Title-guide (KGM-TT). First, the neural topic model based on the variational autoencoder is used to generate the latent topic of the document and the hierarchical encoder with attention mechanism to encode the corpus; then, the circular neural network with attention copy mechanisms to decode the corpus coding and topic distribution for generating keywords. The KGM-TT model not only enriches semantic features by mining latent topics but also uses highly summative and valuable information in the title to jointly guide the generation of keywords.

2. Related Works

2.1. Keyword Extraction. At present, there are many keyword extraction methods, which are mainly divided into unsupervised methods and supervised methods [5]. For unsupervised extraction methods, Mihalcea and Tarau [2] proposed the TextRank-based graph ranking method to calculate the correlation between candidate keywords. Liu et al. [11] used the KeyCluster method of clustering to detect representative phrases from topic clusters. For supervised machine learning methods, the keyword extraction task is transformed into a binary classification problem [7]. Firoozeh et al. [4] introduced KEA method, which extracts candidate phrases from documents, then calculates several features such as TF-IDF value and location information of each candidate phrase, finally they used Naive Bayes model (NB) to judge whether the candidate words are keywords. Turney [12] adopted C4.5 decision tree training classifier. Wang and Peng [13] used Support Vector Machine (SVM) to extract keyword usage features, including word frequency and location information of words. Zhang [14] used Conditional Random Field (CRF) to extract keywords from documents.

The above extraction methods still cannot generate keywords that are not in the source documents. With the development of deep learning technology, a growing number of researches have been gradually evolving from keyword extraction to generative method [5]. Liu et al. [15] proposed to view the keywords extraction task from the perspective of machine translation. They modeled the keywords extraction process as a translation operation from document to keywords and took the keywords as the target output of translation. Since then, the mainstream keyword generation methods became based on Sequence-to-Sequence model (Seq2Seq) [4] such as what Meng et al. [6] introduced in a model based on CopyRNN to generate keywords in academic texts. They took the generation process as a Sequence-to-Sequence learning task framework, and employed a widely used encoder-decoder framework with attention and copy mechanisms. Chen et al. [7] proposed CorrRNN model to capture the correlation between multiple keywords. However, these Recurrent Neural Network (RNN) based models may suffer the low-efficiency issues. Because of the computation dependency between the current time step and the preceding time steps in RNN [5], Zhang et al. [8] proposed CopyCNN model based on convolutional neural network (CNN). It employs position embedding for obtaining a sense of order in the input sequence and adopts Gated Linear Units (GLU) as the nonlinearity function. In addition, some extraction methods also considered the influence of title, such as that Li et al. [16] introduced a graph-based ranking algorithm which sets the importance of words in the title to one and the others as zero, then propagates the influence of title phrases iteratively.

2.2. Topic Modeling. Keywords can represent the topics of a document. A document is usually composed of multiple semantic topics. Therefore, the extracted keywords should also cover all topics in the document. Li et al. [17] proposed the single topical PageRank (single TPR) method to calculate the probability of words under all topics of the document without decomposing into multiple PageRank models, which greatly reduces the amount of calculation. Parveen et al. [18] proposed the salience rank model to balance the topic specificity and corpus specificity of words and avoid extracting words that are not highly related to the topic. Our work is closely related to topic models. Topic-aware is to use topic model to find latent topics from document level word co-occurrence, and then join the model with keywords generation model. Starting from latent semantic analysis (LSA [19]), topic model is used to reveal the underlying semantic structure of document collection has been widely used in data mining, text processing, and information retrieval [3]. Among them, the typical topic models are probabilistic topic models (such as PLSA [20], LDA [21], and HDPs [22]). They provide an extensible basis for document modeling by referencing the latent topic of each variable in topic allocation [22]. With the wide application of deep learning technology in the field of natural language processing, some researchers have proposed a topic model based on neural network. This kind of method mainly uses neural network to reconstruct the text generation process of topic model and adds the sparse constraint of topic in the modeling to generate more expressive topic words [23]. We use a topic model based on variational autoencoder [24, 25] to infer latent topics, which is also conducive to the use of other neural models for end-to-end training without the need for specified model derivation. This model has been proved useful for citation recommendation [26] and conversation understanding [27]. Zeng et al. [28] proposed joint training topic model and short text classification model, but due to the diversity of keywords, this method cannot adapt to the scene of this work. Thus, this paper uses the neural topic model based on variational autoencoder to find the latent topics, which will also be trained together with the keyword generation model.

Sequence to sequence (seq2seq) model is a deep learning framework based on encoder and decoder. In 2014, Cho et al. [29] proposed seq2seq model based on cyclic neural network. This model uses the encoder and decoder to calculate the conditional probability of phrase pairs, which can improve the performance of machine translation. The Seq2seq can solve the problem of end-to-end sequence inequality and is suitable for a variety of natural language processing tasks. Using this model to generate keywords can generate keywords that have not appeared in the original document, which makes up for the shortcomings of the traditional keyword extraction model. The attention mechanism is also used in the coding process of this method, but it treats the title and the text alike.

3. Our Proposed Model

3.1. The Framework of KGM-TT. The traditional keyword extraction method cannot obtain the words that do not appear in the document. The existing generative methods ignore the guiding effect of document title on keywords and do not consider the subject semantic information of the document. The KGM-TT is a keyword generation model based on the fusion of topic aware and title guide, including neural topic model, title-guide hierarchical encoder, and topic-aware sequence decoder. The proposed KGM-TT integrates the semantic guidance of title words into the matching layer in the coding stage. The specific method is to use the attention mechanism to aggregate the title information for each word in the document content (Title-guide).

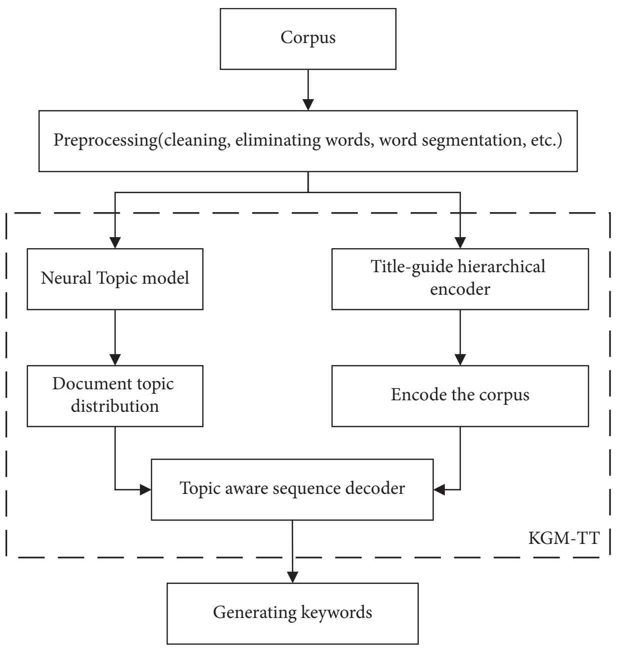

Corpus
Preprocessing(cleaning, eliminating words, word segmentation, etc.)
Neural Topic model
Title-guide hierarchical encoder
Document topic distribution
Encode the corpus
Topic aware sequence decoder
KGM-TT
Generating keywords

Figure 2: KGM-TT framework.

In addition, in the decoding process, using the results of the trained neural topic model, a decoder with topic aware ability is improved to generate keywords with topic sensitivity (Topic-aware).

The framework of KGM-TT model is shown in Figure 2. First, the corpus is preprocessed, and the document topic distribution is generated by neural topic model. Then, the corpus is encoded by the title-guide hierarchical encoder. Finally, the topic distribution and coding representation are input into the topic aware sequence decoder to decode and automatically generate keywords. The training process of the left neural topic model in the framework is used to obtain the topic distribution of the document. The right branch of the frame is the encoding process of the document word sequence. Different from reference [29], we use the title to guide the word coding in the document in the coding stage. In addition, in the decoding stage, the topic of document in the right branch is integrated. The measures of Title-guide and Topic-aware make our KGM-TT better than other models, such as references [6, 8, 9].

3.2. Neural Topic Model. The neural topic model (NVDM-GSM) is used in KGM-TT is based on variational autoencoder (VAE). As shown in Figure 3, it is composed of an encoder and a decoder to simulate the process of document reconstruction [24].

Specifically, the document $C = \{X_1, X_2, \cdots, X_L\}$ in corpus C as input, and process each document X into a BoW vector X_{bow}. X_{bow} is the V-dimensional vector on the vocabulary (V is the size of the vocabulary), input X_{bow} into the neural topic model and encode it into a continuous Gaussian variable Z (representing X's topic) by BoW encoder. Then the BoW decoder with Z as the condition reconstructs X and outputs a BoW vector X'_{bow}. The decoder simulates topic

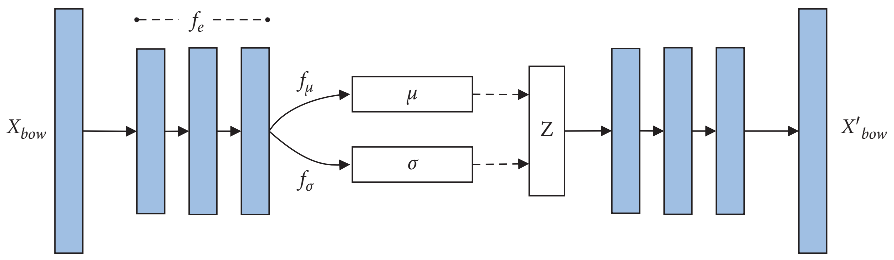


Figure 3: Neural topic model.

model's generation process. We then describe their division of labor.

BoW Encoder. Bow encoder is used to estimate a priori variables μ and σ, and σ is used to lead out topic representation Z. We adopt the following formula:

$$\mu = f_\mu\left(f_e\left(X_{\text{bow}}\right)\right),\ \log\sigma = f_\sigma\left(f_e\left(X_{\text{bow}}\right)\right). \tag{1}$$

where $f_*(\cdot)$ is a neural perceptron with an RuLU activated function.

BoW Decoder. Suppose that the attributive database C has K topics, and the topic vocabulary distribution of each topic in the vocabulary is represented by ϕ_K. For each document $X \in C$, there is a document subject distribution represented by θ (K-dimensional). Specific to the neural topic model, θ is constructed by *softmax* functions. The generation process of documents in the decoder is as follows:

(1) Draw latent topic variable $Z \sim N(\mu, \sigma^2)$

(2) Using softmax function construction topic mixture $\theta = \text{softmax}(W_\theta^T Z)$

(3) For each word, $w \in X$, $w \sim \text{softmax}(W_\phi^T \theta)$

Here $N(\mu, \sigma^2)$ represents the multidimensional Gaussian distribution, and σ^2 is the diagonal of the covariance matrix. W_θ is the matrix of $L * K$, L is the dimension of Z, and K is the number of topic. Here, it is used as the topic-word distributions $(\phi_1, \phi_2, \cdots, \phi_K)$. We adopt the topic mixture vector θ as the topic representation to guide keyword generation.

3.3. Title-Guide Hierarchical Encoder.

As shown in Figure 4, the title-guide hierarchical encoder module is composed of sequence encoding layer, matching layer, and merging layer. The sequence encoding layer reads the title input and main body input and learns their context representation, respectively. The matching layer matches the relevant title information for each word of the document, reflecting the important words in the document. The merging layer merges the aggregated title information into each word to generate a title-guide context representation.

3.3.1. Sequence Encoding Layer.

The vector table maps each word in the text and title to a dense vector with a fixed size d_e. Two bidirectional Gate Recurrence Units (GRU) [29] are used to encode the context and title, respectively, and the context information is combined into the representation of each word. The specific formula is as follows:

$$\begin{aligned}
\overrightarrow{u}_i &= GRU_{11}\left(x_i, \overrightarrow{u}_{i-1}\right),\\
\overleftarrow{u}_i &= GRU_{12}\left(x_i, \overleftarrow{u}_{i+1}\right),\\
\overrightarrow{v}_j &= GRU_{21}\left(t_j, \overrightarrow{v}_{j-1}\right),\\
\overleftarrow{v}_j &= GRU_{22}\left(t_j, \overleftarrow{v}_{j+1}\right),\\
u_i &= \left[\overrightarrow{u}_i; \overleftarrow{u}_i\right],\\
v_j &= \left[\overrightarrow{v}_j; \overleftarrow{v}_j\right],
\end{aligned} \tag{2}$$

where $i = 1, 2, \cdots, L_x$, $j = 1, 2, \cdots, L_t$, x_i is the vector of the i-th word of the document in the corpus, t_j is the vector of the j-th word of the document title. u_i and v_j is the context vector of the i-th word and the j-th title word, $\overrightarrow{u}_i, \overleftarrow{u}_i, \overrightarrow{v}_j, \overleftarrow{v}_j$ is the hidden vector of $d/2$ dimension, where d is the hidden layer dimension of bidirectional GRU. $\longrightarrow$ indicates the coding direction to the right and $\leftarrow$ indicates the coding direction to the left.

3.3.2. Matching Layer.

The attention-based matching layer is engaged to aggregate the relevant information from the title for each word within the context. The aggregation operation $c_i = \text{attn}(u_i, [v_1, v_2, \cdots, v_{L_t}]; W_1)$ is as follows:

$$c_i = \sum_{j=1}^{L_t} \alpha_{i,j} v_j,\ \alpha_{i,j} = \frac{\exp\left(s_{i,j}\right)}{\sum_{k=1}^{L_t} \exp\left(s_{i,k}\right)},\ s_{i,j} = \left(u_i\right)^T W_1 v_j. \tag{3}$$

where c_i the aggregated information vector of the i-th word of document x, $s_{i,j}$ is the unnormalized attention between u_i and v_j, $\alpha_{i,j}$ is the normalized attention between u_i and v_j.

Documents contain title and main body, so the matching layer is also composed of two parts. One part is title-to-title self-matching, in order to better contact the context information of each title word. Because the title contains a lot of high summary information, this part is used to strengthen the importance of the title itself and plays a vital role in capturing the important information of the document. The other part is the matching from main body to title. Each word in the text aggregates the information of the title according to semantic relevance. This part is used to highlight the words highly related to the title, and use the title information to reflect the importance of the words in the

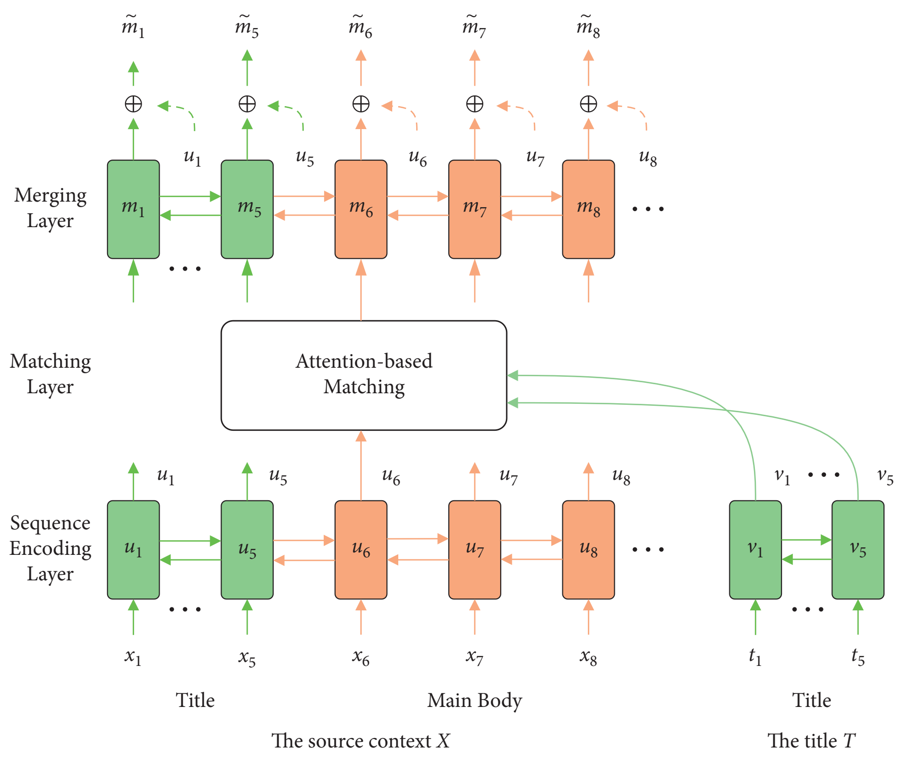


FIGURE 4: The title-guide hierarchical encoder module.

text. Compared with the previous Sequence-to-Sequence methods, this matching layer makes full use of the summary information contained in the title.

3.3.3. Merging Layer. The source contextual vector u_i and the aggregated information vector c_i are used as the input to the information merging layer. The specific formula is as follows:

$$\begin{aligned}\overrightarrow{m}_i &= GRU_{31}\left([u_i; c_i], \overrightarrow{m}_{i-1}\right),\\ \overleftarrow{m}_i &= GRU_{32}\left([u_i; c_i], \overleftarrow{m}_{i+1}\right),\\ \tilde{m}_i &= \lambda u_i + (1-\lambda)\left[\overrightarrow{m}_i, \overleftarrow{m}_i\right],\\ M &= \left[\tilde{m}_1, \tilde{m}_2, \cdots, \tilde{m}_{L_x}\right],\end{aligned} \tag{4}$$

where u_i is a residual connection, $\lambda \in (0, 1)$ is the corresponding hyperparameter. Finally, the title-guide context representation $[\tilde{m}_1, \tilde{m}_2, \cdots, \tilde{m}_{L_x}]$ is obtained and stored as M for subsequent decoding process.

3.4. Topic-Aware Sequence Decoder. As shown in Figure 5, the topic aware sequence decoder is conditional on encoding representation M and latent topic θ, the generation process of the following keyword Y is defined:

$$\Pr(Y|X) = \prod_{j=1}^{|y|} \Pr\left(y_j | Y_{\langle j}, M, \theta\right). \tag{5}$$

where $Y_{\langle j} = \langle y_1, y_2, \cdots, y_{j-1}\rangle$, $\Pr(y_j | Y_{\langle j}, M, \theta)$, denoted as p_j, is a word distribution over vocabulary, reflecting how likely a word to fill in the j-th slot in target keyword.

The sequence decoder adopts a unidirectional GRU. On the general state update, the j-th hidden state s_j is further designed to add the latent topic θ of the document X:

$$s_j = f_{GRU}\left(\left[z_j; \theta\right], s_{j-1}\right), \tag{6}$$

where z_j is the input of the j-th decoder, s_j is the hidden state at the j-th time, s_{j-1} is the previous hidden state, and $[;]$ indicates the connection operation.

The decoder decodes sequence M and obtains key information through the attention mechanism. When predicting the j-th word in the keyword, the attention weight on $w_i \in X_{seq}$ is defined as $\alpha_{ij}^{'}$:

$$\begin{aligned}\alpha_{ij}^{'} &= \frac{\exp\left(f_\alpha\left(\tilde{m}_i, s_j, \theta\right)\right)}{\sum_i^{|X|} \exp\left(f_\alpha\left(\tilde{m}_{i\prime}, s_j, \theta\right)\right)},\\ f_\alpha\left(\tilde{m}_i, s_j, \theta\right) &= v_\alpha^T \tanh\left(W_\alpha\left[\tilde{m}_i; s_j; \theta\right] + b_\alpha\right),\end{aligned} \tag{7}$$

where $v_\alpha, W_\alpha, b_\alpha$ are trainable parameters and $f_\alpha(\cdot)$ indicates the semantic relationship between the i-th word and the j-th target word to be predicted. This relationship is also calibrated with the input latent topic information θ to explore and highlight the topic words. Thus, the topic sensitive context vector c_j is obtained:

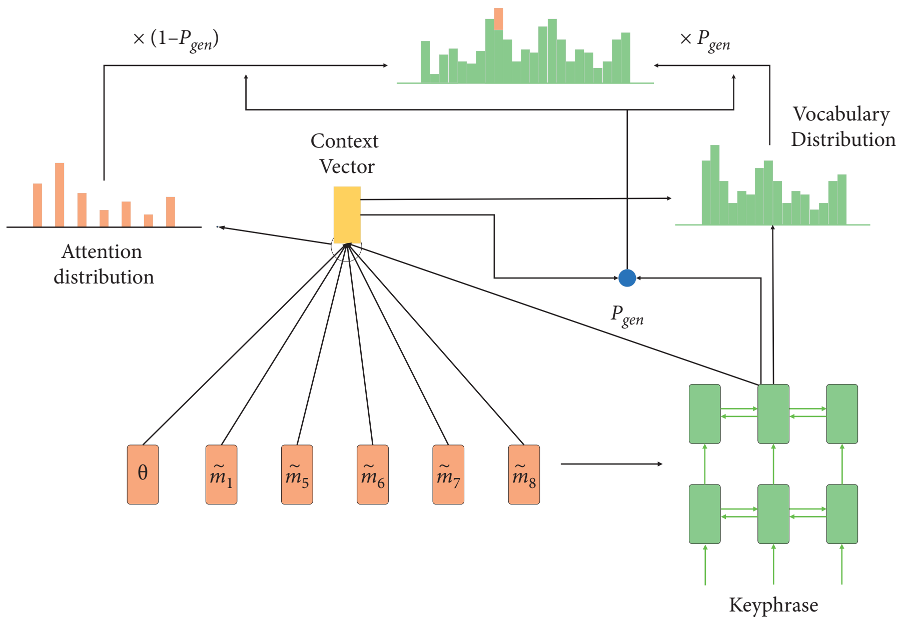


Figure 5: The title-guide hierarchical decoder module.

$$c_j = \sum_{i=1}^{|X|} \alpha'_{ij} \tilde{m}_i. \tag{8}$$

In addition, under the condition of c_j, the j th word is generated on the vocabulary according to the following formula:

$$p_{\text{gen}} = \text{softmax}\left(W_{\text{gen}}\left[s_j; c_j\right] + b_{\text{gen}}\right). \tag{9}$$

A copy mechanism See et al. [30] is added here to extract keywords from the source document. Specifically, $\lambda_j \in [0,1]$ is used as a soft switch to decide whether to copy a word directly from the original text as the j-th target word.

$$\lambda_j = \text{sigmoid}\left(W_\lambda\left[z_j; s_j; c_j; \theta\right] + b_\lambda\right), \tag{10}$$

where W_λ, b_λ are trainable parameters and topic information θ is also injected here to guide the switch decision.

Finally, the distribution p_j of the j-th target word can be predicted by using the following formula:

$$p_j = \lambda_j \cdot p_{\text{gen}} + \left(1 - p_{\text{gen}}\right) \cdot \sum_{i=1}^{|X|} \alpha'_{ij}, \tag{11}$$

where attention scores $\left\{\alpha'_{ij}\right\}_{i=1}^{|X|}$ serve as the extractive distribution over the source input.

3.5. Jointly Learning Topics and Keywords. Our model KGM-TT is an end-to-end learning of latent topic model and keyword generation. First, the objective functions of the two modules are defined, respectively.

For the neural topic model, the objective function is defined based on negative lower bound of negative variation, and its loss function is as follows:

$$L_{GSM} = D_{KL}\left(p(Z)q(Z|X)\right) - \mathrm{E}_{q(Z|X)}\left[p(X|Z)\right]. \tag{12}$$

where the first term is the Kullback–Leibler divergence loss and the second term reflects the reconstruction loss. $p(Z)$ represents a priori distribution, $q(Z|X)$ and $p(X|Z)$ represent the process of BoW encoder and BoW decoder, respectively.

For the keyword generation model, the minimum cross entropy loss function is used for training on all training sets:

$$L_{KG} = -\sum_{n=1}^{N} \log\left(\Pr\left(Y_n | X_n, \theta_n\right)\right), \tag{13}$$

where N is the number of instances of the training set and θ_n is X_n 's latent topics induced from GSM.

Finally, the linear combination of L_{GSM} and L_{KG} is used to define the training objectives of the whole framework:

$$L_{KG-CTATG} = L_{GSM} + L_{KG}, \tag{14}$$

where γ hyperparameters balance the effects of neural topic model and generative model.

The two models are trained together and their parameters are updated at the same time. After the training, the beam search is used to generate the ranking list of keywords.

4. Experiment and Analysis

4.1. Experimental Data. The corpus CNKI is crawled by our own crawler from China's largest scientific and technological publications databases (https://www.cnki.net). It has 18,000 papers in total and includes 5,6589 words. These documents are papers published from 2000 to 2020, including the title, abstract content, keywords, and publication time of the paper. In order to reflect its good portability and achieve good results in large-scale data sets, the largest publicly available keyword generation data set KP20k built by Meng et al. [8] in 2017 is selected for testing and evaluation in our experiment. The KP20k consists of a large number of high-quality scientific publications from different fields of computer science. The details of CNKI and KP20k are shown in Table 1.

4.2. Parameter Setting. For the neural topic model, it is implemented according to the design of Zeng et al. [28] and the number of topics K is set to 50. For the hierarchical encoder, the vocabulary V is defined as 50,000 words with the highest frequency of use. Set the embedding dimension d_e to 100, the number of hidden nodes to 256, the damping coefficient λ to 0.5, and the word embedding is randomly initialized and evenly distributed in $[-0.1, 0.1]$. Using the optimization model of Adam et al. [31], the batch size is 64, the initial learning rate is set to 10^{-4}, the gradient cutting is set to 1, and the launch rate is set to 0.1. The convergence speed of neural topic model is much slower than that of generative model. Therefore, before joint training, train the neural topic model for 100 iterations and the generative model for 1 iteration. Empirically set $\gamma = 1.0$ to balance the loss of neural subject model and generation model, and iteratively update the parameters in each module. Then update their combination in turn. In the test, set the maximum depth of beam search to 6 and the beam size to 200.

4.3. Evaluating Indicator. We use Precision (P) to measure the accuracy of the model, Recall (R) to measure the integrity of the model, and F1-measure to evaluate the performance of keyword extraction methods. T_o represents the keyword set provided by the dataset itself, T_e represents the keyword set extracted by the model, and $T_o \cap T_e$ represents the correctly extracted keyword set. Precision, Recall, and F1-measure are defined as follows:

$$\begin{aligned} p &= \frac{\left|T_o \cap T_e\right|}{\left|T_o\right|}, \\ R &= \frac{\left|T_o \cap T_e\right|}{\left|T_e\right|}, \\ F1 - \text{Measure} &= \frac{2 \times P \times R}{P + R}. \end{aligned} \tag{15}$$

4.4. Experimental Results and Analysis. For the keyword prediction existing in the source text, two unsupervised models, including TF-IDF and TextRank, and a supervised model KEA are used as the traditional extraction method. In addition, the following keyword generation models are considered: Sequence-to-Sequence (Seq2Seq) model without copy mechanism, sequence-to-sequence model with copy mechanism CopyRNN and CopyCNN. For the prediction of keywords that do not exist in the source text, since the traditional extraction methods cannot generate such keywords, the baseline models are CopyRNN and CopyCNN. For all baseline models, select the same parameter settings as Meng et al. [6].

For predicting the keywords existing in the source text, the keyword prediction ability of the above model on CNKI and KP20k datasets is compared. The F1-measure values of the top 5 and top 10 predictions of each model are shown in Table 2.

Table 1: The number of experimental documents and its division.

Corpus	Training	Verification	Verification	Total
CNKI	12,600	3,600	1,800	1,8000
KP20k	530,809	20,000	20,000	570,809

Table 2: Present keyword predicting results.

	Model	CNKI		KP20k	
		F1@5	F1@10	F1@5	F1@10
Extractive models	TF-IDF	0.233	0.311	0.104	0.126
	TextRank	0.324	0.302	0.176	0.147
	KEA	0.336	0.328	0.180	0.163
Generative models	Seq2Seq	0.417	0.392	0.243	0.216
	CopyRNN	0.486	0.424	0.321	0.260
	CopyCNN	0.518	0.473	0.349	0.283
	KGM-TT	0.602	0.569	0.392	0.334

It can be found from the table that all the generation models are better than the traditional baseline method. In addition, it can be noted that the model proposed in this paper has significant advantages in both datasets. For example, on the KP20k dataset, our model improves 15.2% (F1@10 score) than the best generative model CopyCNN. Generally speaking, the recall rate (R) of the top 10 and top 50 predictions is engaged as an indicator to measure how many absent keywords are correctly predicted. The experimental results are shown in Figure 6.

On the two datasets, we observed that our model is always better than the previous sequence to sequence model, such as our model is on the kp20k dataset, R@10 score is 10.2% higher than CopyCNN model. In general, the results show that our model can generate keywords better than the baseline model and capture the latent semantic information in the context content.

The quality judgment of keywords is controversial. Therefore, in addition to the above objective evaluations, we also invited five artificial experts to evaluate the prediction results of the models. Each expert was given the task of judging 100 papers. They need to judge the right and wrong results of the three generative keyword models: CopyRNN, CopyCNN, and our KGM-TT. For fairness, hide the models name in the expert evaluation process. We conducted the experiment on Chinese corpus CNKI and selected five keywords for evaluation. The experimental results are shown in Table 3.

According to the judgment results of artificial experts in Table 3, the F1 index of model CopyRNN and model CopyCNN is basically close. In contrast, the KGM-TT is ahead, about 5 percentage points higher than the other two models. The specific reason is the KGM-TT use the attention mechanism to aggregate the title information for each word in the document content and use a decoder with topic aware ability. It is improved to generate keywords with topic sensitivity.

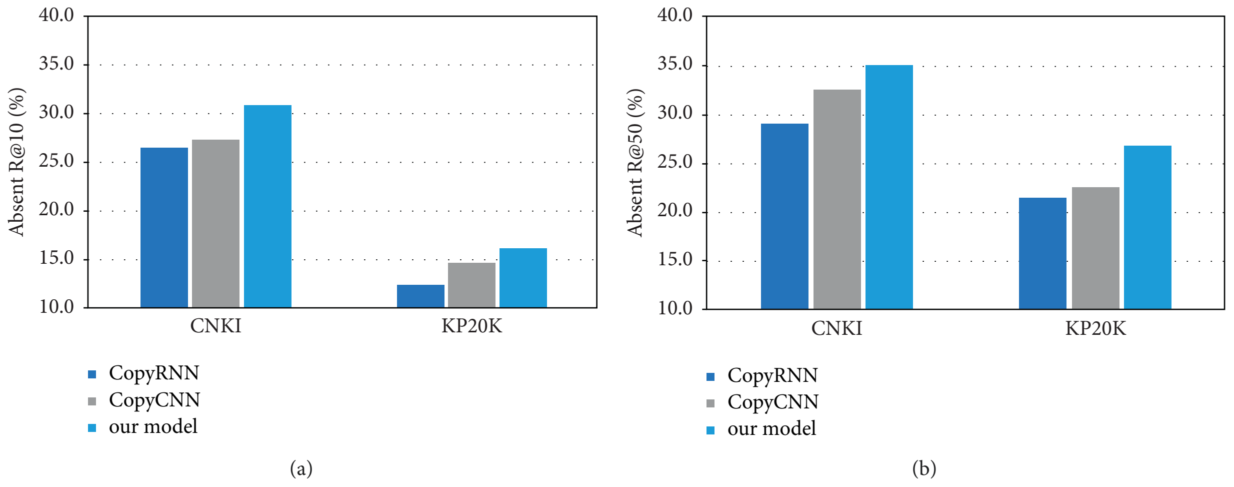


Figure 6: Absent keyword predicting results.

Table 3: Expert evaluation results.

	CopyRNN	CopyCNN	KGM-TT
F1@5	0.529	0.531	0.587

5. Conclusion

The traditional keyword extraction method and cannot obtain the words that do not appear in the document. The existing generative methods ignore the guiding effect of document title on keywords and do not consider the subject semantic information of the document. We present a keyword generation model based on combination topic aware and title-guide (KGM-TT). In the process of encoding document words, it is guided by the title semantics. In the decoding process, a topic aware decoder is used. Therefore, the keywords generated by the KGM-TT are more subject-sensitive and higher quality. The experimental results show that the KGM-TT model proposed in this paper can make better use of the strong semantic information of the latent topic and title of the document. The KGM-TT is superior to other methods in keyword prediction, and can generate keywords with high accuracy. However, there are a large number of synonyms in the words generated by the keyword generation model. This is a problem that needs to be solved in our research in the future.

Data Availability

The data generated and analyzed during this research are available from the corresponding author on request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] G. Berend, "Opinion expression mining by exploiting keyphrase extraction," in *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pp. 1162–1170, Chiang Mai, Thailand, November 2011.

[2] R. Mihalcea and P. Tarau, "Textrank: bringing order into text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 404–411, Barcelona, Spain, July 2004.

[3] L. Sterckx, T. Demeester, J. Deleu, and C. Develder, "Topical word importance for fast keyphrase extraction," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 121-122, Florence, Italy, May 2015.

[4] N. Firoozeh, A. Nazarenko, F. Alizon, and B. Daille, "Keyword extraction: issues and methods," *Natural Language Engineering*, vol. 26, no. 3, pp. 259–291, 2020.

[5] H. Shaohu, Z. Yingyi, and Z. Chengzhi, "Review of keyword extraction studies," *Data Analysis and Knowledge Discovery*, vol. 5, no. 3, pp. 45–59, 2020.

[6] R. Meng, S. Zhao, S. Han et al., "Deep keyword generation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 582–592, Vancouver, Canada, July 2017.

[7] J. Chen, X. Zhang, Y. Wu et al., "Keyword generation with correlation constraints," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4057–4066, Brussels, Belgium, October 2018.

[8] Y. Zhang, Y. Fang, and W. D. Xiao, "Deep keyword generation with a convolutional sequence to sequence model," in *Proceedings of the IEEE 4th International Conference on Systems and Informatics (ICSAI)*, pp. 1477–1485, Hangzhou, China, November 2017.

[9] W. Chen, Y. Gao, J. Zhang, I. King, and M. R. Lyu, "Title-guide encoding for keyword generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 6268–6275, Honolulu, Hawaii, USA, January 2019.

[10] L. Ding, Z. Zhang, H. Liu, and J. G. Li, "Automatic keyphrase extraction from scientific Chinese medical abstracts based on character-level sequence labeling," *Journal of Data and Information Science*, vol. 6, no. 3, pp. 35–57, 2021.

[11] Z. Liu, P. Li, and Y. Zheng, "Clustering to find exemplar terms for keyword extraction," in *Proceedings of the 2009 conference on empirical methods in natural language processing*, pp. 257–266, Singapore, August 2009.

[12] P. D. Turney, Z. Liu, P. Li, Y. Zheng, and M. Sun, “Learning algorithms for keyword extraction,” *Information Retrieval*, vol. 2, no. 4, pp. 303–336, 2000.

[13] J. Wang and H. Peng, “Keywords extraction from web document by the least squares support vector machine,” in *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 293–296, Washington, USA, September 2005.

[14] C. Zhang, “Automatic keyword extraction from documents using conditional random fields,” *Journal of Computational Information Systems*, vol. 4, no. 3, pp. 1169–1180, 2008.

[15] Z. Liu, X. Chen, and Y. Zheng, “Automatic keyword extraction by bridging vocabulary gap,” in *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pp. 135–144, Uppsala, Switzerland, June 2011.

[16] D. Li, S. Li, W. Li, and C. Zhang, “A semi-supervised key phrase extraction approach: learning from title phrases through a document semantic network,” in *Proceedings of the ACL 2010 conference short papers*, pp. 296–300, Uppsala, Switzerland, July 2010.

[17] D. Parveen, H. M. Ramsl, M. Strub et al., “Topical coherence for graph-based extractive summarization,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1949–1954, Lisbon, Portugal, September 2015.

[18] N. Teneva and W. Cheng, “Salience rank: efficient keyphrase extraction with topic modelling,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 530–535, Beijing, China, July 2017.

[19] T. K. Landauer, P. W. Foltz, and D. Laham, “An introduction to latent semantic analysis,” *Discourse Processes*, vol. 25, no. 2-3, pp. 259–284, 1998.

[20] T. Hofmann, “Probabilistic latent semantic indexing,” in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50–57, Berkeley, CA, USA, August 1999.

[21] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of Machine Learning Research*, vol. 2, no. 3, pp. 993–1022, 2003.

[22] Y. W. Teh, M. I. Jordan, and M. J. Beal, D. M. Blei, Hierarchical dirichlet processes,” *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1566–1581, 2006.

[23] J. J. Huang, P. W. Li, and M. Peng, “Review of deep learning-based topic model,” *Chinese Journal of Computers*, vol. 43, no. 05, pp. 827–855, 2020.

[24] Y. Miao, E. Grefenstette, P. Blunsom et al., “Discovering discrete latent topics with neural variational inference,” in *Proceedings of the International Conference on Machine Learning (PMLR)*, pp. 2410–2419, Sydney, Australia, August 2017.

[25] A. Srivastava and C. Sutton, “Autoencoding variational inference for topic models,” pp. 1703–1712, 2017, https://arxiv.org/.

[26] H. Bai, Z. Chen, and M. R. Lyu, “Neural relational topic models for scientific article analysis,” in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 27–36, Sheffield, UK, 2018.

[27] J. Zeng, J. Li, Y. He et al., “What you say and how you say it: joint modeling of topics and discourse in microblog conversations,” *Transactions of the Association for Computational Linguistics*, vol. 7, no. 6239, pp. 267–281, 2019.

[28] J. Zeng, J. Li, and Y. Song, “Topic memory networks for short text classification,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3120–3131, Brussels, Belgium, October 2018.

[29] K. Cho, B. Van Merriënboer, C. Gulcehre et al., “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” 2014, https://arxiv.org/pdf/1406.1078.

[30] A. See, P. J. Liu, C. D. Manning et al., “Get to the point: summarization with pointer-generator networks,” 2017, https://arxiv.org/pdf/1704.04368.

[31] D. P. Kingma and J. Ba, “Adam: a method for stochastic optimization,” 2014, https://www.arxiv.org/abs/1412.6980v1.