information

Article

Hierarchical Keyword Generation Method for Low-Resource Social Media Text

Xinyi Guan * and Shun Long

Department of Computer Science, Jinan University, Guangzhou 510632, China; tlongshun@jnu.edu.cn
* Correspondence: guanxy98@stu2020.jnu.edu.cn

Abstract: The exponential growth of social media text information presents a challenging issue in terms of retrieving valuable information efficiently. Utilizing deep learning models, we can automatically generate keywords that express core content and topics of social media text, thereby facilitating the retrieval of critical information. However, the performance of deep learning models is limited by the labeled text data in the social media domain. To address this problem, this paper presents a hierarchical keyword generation method for low-resource social media text. Specifically, the text segment is introduced as a hierarchical unit of social media text to construct a hierarchical model structure and design a text segment recovery task for self-supervised training of the model, which not only improves the ability of the model to extract features from social media text, but also reduces the dependence of the keyword generation model on the labeled data in the social media domain. Experimental results from publicly available social media datasets demonstrate that the proposed method can effectively improve the keyword generation performance even given limited social media labeled data. Further discussions demonstrate that the self-supervised training stage based on the text segment recovery task indeed benefits the model in adapting to the social media text domain.

Keywords: keyword generation; social media text; transfer learning; attention mechanism

Citation: Guan, X.; Long, S. Hierarchical Keyword Generation Method for Low-Resource Social Media Text. *Information* **2023**, *14*, 615. https://doi.org/10.3390/info14110615

Academic Editor: Ralf Krestel

Received: 24 September 2023
Revised: 13 November 2023
Accepted: 14 November 2023
Published: 15 November 2023

1. Introduction

Social media has emerged as a prominent platform for online social interaction and sharing opinions, which has led to an overwhelming volume of text information, leading to the problem of social media text information overload. The keyword generation technology in natural language processing (NLP) can automatically extract text features and generate the central words or phrases that best reflect the theme of the text, which not only helps us quickly acquire essential information and better understand the content, but also can be applied to various downstream NLP tasks, such as document classification [1,2], recommendation systems [3,4], information retrieval [5,6], text summarization [7,8], text classification [9], and knowledge graph [10]. Therefore, it is a vital method to alleviate the problem of text information overload.

Keyword generation research has primarily focused on traditional text, as the traditional text has clear text structure and keywords labeled by the author, which can form large-scale labeled datasets in the domain [11–13]. In contrast, the number of keyword generation task datasets of social media text is relatively small. As most of the keyword generation methods are based on deep learning models and heavily rely on massive and high-quality training datasets, the keyword generation task for social media text inevitably falls under the problem of insufficient training data.

In this paper, we propose a hierarchical keyword generation method for low-resource social media text (HKG), which leverages transfer learning technology to reduce the dependence of the keyword generation model on labeled data in the social media text domain. Specifically, a self-supervised text segment recovery task is added to the "Pre-training and Fine-tuning" process of the model training, and a hierarchical model structure is proposed based on the Transformer model [14] for the characteristics of social media text.

The hierarchical model encodes text semantics and hierarchical information by the attention mechanism that acts on text segments and the entire text, respectively, and then selects significant text segments to generate keywords through a selection block. The text segment recovery task first filters significant text segments through the hierarchical encoder and selection block of the model, then deletes text segments from the original text with a certain probability, and finally trains the model to recover the processed text to the original text, aiming to help the keyword generation model adapt to the target social media domain.

Comparative experiments and ablation experiments on public datasets demonstrate that HKG can effectively improve the low-resource keyword generation performance for social media text.

2. Related Work

Early research on keyword generation focused on extracting keywords from text, and it is a challenge to generate keywords that do not exist in the original text. In 2014, Cho et al. [15] proposed the sequence to sequence (Seq2Seq) model based on a recurrent neural network, which greatly promotes the development of natural language generation tasks and points the way to the development of stagnant keyword generation research.

The first to generate keywords based on the Seq2Seq model was the CopyRNN model proposed by Meng et al. [11], which uses a bidirectional gated recurrent unit network to generate keywords for academic text. In order to reflect the theme of the original text and avoid generating semantically repetitive keywords, Chen et al. [16] proposed a CorrRNN model that imposes constraints when generating keywords. Zhang et al. [17] applied attention, replication, and coverage mechanisms to the Seq2Seq model. Chen et al. [18] utilized the title of the text to guide the model while generating keywords, all of which improved the accuracy of keyword generation to a certain extent. Wang et al. [19] proposed TAKG, the first keyword generating model for social media text, utilizing corpus level latent topic representation to enrich the contextual representation of input obtained using the encoder. After that, Kim et al. [20] proposed a method for expanding missing related phrases from existing keywords to increase context, and Yang et al. [21] solved the problem of dispersed social media text information through graph convolutional networks.

In recent years, some research used transfer methods to reduce the dependence of keyword generation methods based on the Seq2Seq model on labeled data. Ye et al. [22] proposed two methods: the first uses unsupervised methods to "label" unlabeled data and mixes it with labeled data as training data. The second only trains the decoder, while the encoder is trained by other tasks. Wang et al. [23] proposed a topic-based adversarial neural network that uses adversarial transfer learning methods to learn transferable knowledge across domains and assist in keyword extraction in the target domain. The most widely used transfer method in the NLP is the "Pre-training and Fine-tuning" method. Guo et al. [24] proposed using a large corpus to pre-train the encoder, fine-tune it using specific datasets, and add multi-task training during the fine-tuning stage to improve the accuracy of the model. Sun et al. [25] proposed JointKPE, an open domain key phrase extraction architecture based on a pre-trained language model that can capture local and global information when extracting key phrases. Due to the lack of labeled data in the social media text domain, HKG also adopts the "Pre-training and Fine-tuning" method, but adds a self-supervised training stage based on the text segment recovery task, which helps the model adapt to the social media text domain.

In addition, HKG builds a hierarchical model structure for the sparse features of social media text. Some studies also propose the concept of hierarchical model structure. Ref. [26] proposes a hierarchical encoder for movie scripts to concatenate each scene's action, dialogue, and scene-level character embeddings into a single scene embedding using recurrent units. Ref. [27] obtained hierarchy-aware text representation for hierarchical text classification using BERT and Graphormer. Ref. [28] used the bidirectional Transformer encoder at the sentence-level and the document-level Transformer encoder to extend the

self-attention mechanism to long-form text modeling. However, the hierarchical structure of HKG is based on a novel fixed-length hierarchical unit—a text segment, which we propose for social media text, encoding the textual information with hierarchical information by masking the attention mechanism.

In conclusion, HKG is specifically designed for social media text that synthesizes two perspectives of model structure and model training method to generate keywords.

3. Method

3.1. Problem Formulation

The keyword generation task is defined as automatically generating one or more keywords that express the topic information of a given text. If the text is represented by a text sequence and its corresponding one or more keywords are characterized by a keyword sequence, then a set of text sequences and their keyword sequences of size N are formulated as $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,n}\}$ denotes the i-th text sequence containing n tokens and $\boldsymbol{y}_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,m}\}$ denotes the keyword sequence corresponding to $\boldsymbol{x}_i$ containing m tokens.

3.2. Architecture

Figure 1 shows the overall architecture of the HKG.

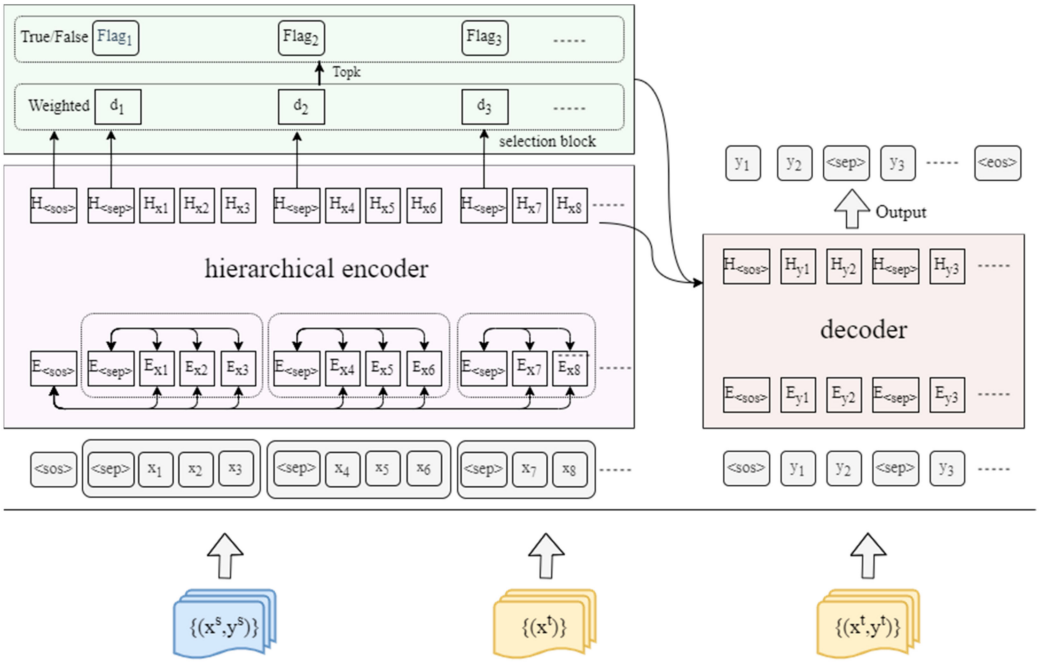

Figure 1. The overall architecture of the HKG.

The upper half is a hierarchical keyword generation model based on the Transformer, which is divided into a hierarchical encoder, a selection block, and a decoder. We define a text segment consisting of l tokens as a hierarchical unit of social media text. The hierarchical encoder encodes textual information with hierarchical information by applying the attention mechanism to the entire text and text segments. The selection block calculates the weights of text segments based on the text feature vectors obtained from the hierarchical encoder and then records significant text segments. The decoder generates keywords with text feature vectors and significant text segment records.

The lower half is the sample set used at different training stages of the model. The whole training stage is divided into the pre-training stage of the traditional text domain D_s, the self-supervised fine-tuning stage of the social media domain D_t, and the supervised fine-tuning stage of social media domain D_t. The training sets used are the labeled sample

set $\{(\boldsymbol{x}^s, \boldsymbol{y}^s)\}$ of D_s, the unlabeled sample set $\{(\boldsymbol{x}^t)\}$ of D_t, and the labeled sample set $\{(\boldsymbol{x}^t, \boldsymbol{y}^t)\}$ of D_t.

3.3. Data Preprocessing

Before the text sequence $\boldsymbol{x}$ and the keyword sequence $\boldsymbol{y}$ are input into the model, the text sequence $\boldsymbol{x}$ needs to be divided into a text segments sequence $\boldsymbol{x}_{sep}$ consisting of multiple text segments of fixed length l using the separator "<sep>" and the keyword sequence $\boldsymbol{y}_{sep}$ is obtained by inserting the separator "<sep>" between tokens belonging to different keywords of the keyword sequence $\boldsymbol{y}$.

3.4. Hierarchical Encoder

As the model structure is based on the Transformer, the attention mechanism [14] is applied to encode input sequences and generate prediction sequences. The attention mechanism takes $\boldsymbol{K}$, $\boldsymbol{Q}$, and $\boldsymbol{V}$ as inputs, calculates attention weights through $\boldsymbol{Q}$ and $\boldsymbol{K}$, and then acts on $\boldsymbol{V}$ to obtain the weighted output vector $\boldsymbol{Z}$.

$$\boldsymbol{Z} = \boldsymbol{W}\boldsymbol{V} \tag{1}$$

$$\boldsymbol{W} = softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}} + \boldsymbol{M}\right) \tag{2}$$

where d_k is the dimension of $\boldsymbol{K}$, $\boldsymbol{W}$ is the attention weight matrix, $\boldsymbol{M}$ is the attention weight masking matrix, and $\boldsymbol{M}[i][j] = 0$ means that the vector at position i can obtain information about the vector at position j, otherwise $\boldsymbol{M}[i][j] = inf$.

The hierarchical encoder applies the attention mechanism, respectively, on the entire text and text segments to encode textual hierarchy and context information.

The embedding $\boldsymbol{E}_x$ of the input text segments sequence $\boldsymbol{x}_{sep}$ is obtained by adding the content embedding, position embedding, and segment embedding. The input $\boldsymbol{Q}_x$, $\boldsymbol{K}_x$, and $\boldsymbol{V}_x$ for the attention mechanism of the hierarchical encoder are obtained through matrix calculation using $\boldsymbol{E}_x$. For each text segment sequence $\boldsymbol{x}_{sep}$, we define the corresponding attention weight masking matrix $\boldsymbol{M}_x$.

$$\boldsymbol{M}_x[0][j] = \begin{cases} 0, \ if \ \boldsymbol{x}_{sep}[j] \neq \text{“}< SEP >\text{”} \\ -inf, \ if \ \boldsymbol{x}_{sep}[j] = \text{“}< SEP >\text{”} \end{cases} \tag{3}$$

$$\boldsymbol{M}_x[i][j] = \begin{cases} 0, \ if \ \left(\boldsymbol{x}_{sep}[i] \text{ in } \boldsymbol{SEG}_s\right) \ and \ \left(\boldsymbol{x}_{sep}[j] \text{ in } \boldsymbol{SEG}_s\right) \\ -inf, \ if \ \left(\boldsymbol{x}_{sep}[i] \text{ in } \boldsymbol{SEG}_s\right) \ and \ \left(\boldsymbol{x}_{sep}[j] \text{ not in } \boldsymbol{SEG}_s\right) \end{cases}, i = 1 \sim \left|\boldsymbol{x}_{sep}\right| \tag{4}$$

where $\boldsymbol{SEG}_s = \{< sep >, \boldsymbol{x}_1^s, , \boldsymbol{x}_2^s, \ldots, \boldsymbol{x}_l^s\}$ is the s-th text segment with length l in the $\boldsymbol{x}_{sep}$. Based on the segment that each token in the $\boldsymbol{x}_{sep}$ belongs to, we assign a value to $\boldsymbol{M}_x$ to mask the corresponding position.

After multiple calculations using the attention mechanism and other neural networks, the encoder outputs the text feature vector $\boldsymbol{H} = \left\{\boldsymbol{H}_{<sos>}, \boldsymbol{H}_{<sep>_1}, \boldsymbol{H}_{x_1}, \boldsymbol{H}_{x_2}, \ldots, \boldsymbol{H}_{<eos>}\right\}$, where $\boldsymbol{H}_{<sos>}$ represents the entire text, integrating the information of tokens in the $\boldsymbol{x}$, and $\boldsymbol{H}_{<sep>_s}$ represents the s-th text segment, integrating the information of tokens in the $\boldsymbol{SEG}_s$.

3.5. Selection Block

The selection block filters out significant text segments by calculating the difference between the feature vector representing the text segments and the feature vector representing the entire text.

Firstly, we calculate the importance of each text segment relative to the entire text to obtain the weighted vector $\boldsymbol{d}$.

$$d_i = dis\left(\boldsymbol{H}_{<sos>}, \boldsymbol{H}_{<sep>_i}\right) \tag{5}$$

$$\boldsymbol{d} = softmax(d_1, d_2, \ldots, d_{ns}) \tag{6}$$

where dis is a vector distance calculation function, d_i denotes the difference between $\boldsymbol{H}_{<sep>_i}$ and $\boldsymbol{H}_{<sos>}$, and $ns = \lfloor (|\boldsymbol{x}_{sep}| - 1)/(l+1) \rfloor$ denotes the number of text segments in the $\boldsymbol{x}_{sep}$.

Then, we select the most significant k text segments based on $\boldsymbol{d}$, and record whether a text segment is significant using the text segment flag vector $\boldsymbol{f} = (flag_1, flag_2, \cdots, flag_i)$.

$$flag_i = \begin{cases} True, \ d_i \ in \ topk(\boldsymbol{d}) \\ False, \ d_i \ not \ in \ topk(\boldsymbol{d}) \end{cases} \tag{7}$$

where $topk(\boldsymbol{d})$ represents the top k elements in $\boldsymbol{d}$ sorted from largest to smallest. The hyperparameter k is associated with the length l of a text segment, and we conduct experiments on the optimal combinations of k and l in Section 4.4.3.

3.6. Decoder

The decoder utilizes significant text segment feature vectors to generate corresponding keywords, enabling the keyword generation model to focus on more important information. The embedding $\boldsymbol{E}_y$ of the input keyword sequence $\boldsymbol{y}_{sep}$ is obtained by adding the content embedding, position embedding, and keyword embedding, and then encoding to the weighted output vector $\boldsymbol{Z}_y$.

The attention mechanism of the decoder while generating the keywords is computed with input $\boldsymbol{Q}_z$ obtained from the $\boldsymbol{Z}_y$ by matrix computation, and $\boldsymbol{K}_h$ and $\boldsymbol{V}_h$ obtained from the $\boldsymbol{H}$ bymatrix computation. For each $\boldsymbol{H}$, we define the corresponding attention weight masking matrix $\boldsymbol{M}_h$ based on the text segment flag vector $\boldsymbol{f}$.

$$\begin{gathered} \boldsymbol{M}_h[i : i+l+1] = \begin{cases} 0, if \ \boldsymbol{f}[i] \ is \ True \\ -inf, if \ \boldsymbol{f}[i] \ is \ False \end{cases}, \\ i = 1, 1+1*(l+1), 1+2*(l+1), \ldots, 1+\lceil (|\boldsymbol{x}_{sep}| - 1)/(l+1) \rceil * (l+1) \end{gathered} \tag{8}$$

where i is the positional index of the character "<sep>" in the $\boldsymbol{x}_{sep}$.

The decoder outputs the predicted distribution $\hat{\boldsymbol{y}}$ on the vocabulary of keywords, $\hat{\boldsymbol{y}} \in \mathbb{R}^{|\boldsymbol{y}_{sep}| \times d_o}$, and d_o is the size of output vocabulary.

3.7. Training

In order to achieve better keyword generation results and avoid the need for the model to learn knowledge from scratch in the target domain, we train the hierarchical keyword generation model to learn in the traditional text domain with large-scale labeled data, and then migrate the knowledge to the low-resource social media text domain. At the fine-tuning stage of the model training, we design a self-supervised text segment recovery task, which is derived from the method of generating keywords in low-resource domains proposed by Wu et al. [29], which proves that based on filtering out the essential phrases in the text, destroying the text and then recovering it can effectively improve the training performance of keyword generation model. The model training based on the text segment recovery task is shown in Algorithm 1.

First, the text sequence $\boldsymbol{x}^t$ is preprocessed to the text segment sequence $\boldsymbol{x}^t_{sep}$. Then, the $\boldsymbol{x}^t_{sep}$ is input into the hierarchical encoder to obtain the text feature vector $\boldsymbol{H}$, and the $\boldsymbol{H}$ is used to define the text segment flag vector $\boldsymbol{f}$ by calculating the importance of each text segment relative to the entire text in the selection block. Next, we process $\boldsymbol{x}^t_{sep}$ based on $\boldsymbol{f}$ to obtain a corrupted text sequence $\boldsymbol{x}^t_{del}$. Specifically, if the text segment flag corresponding to a text segment ***SEG*** is *True*, the significant text segment is deleted from $\boldsymbol{x}^t_{sep}$ with the probability p_1, otherwise it is deleted with the probability of p_2. Finally, the $\boldsymbol{x}^t_{del}$ is input to the hierarchical encoder and the $\boldsymbol{x}^t_{sep}$ is input to the decoder, guiding the model to recover the $\boldsymbol{x}^t_{del}$ into $\boldsymbol{x}^t_{sep}$, which helps the hierarchical keyword generation model to be better adapted

to the social media text domain, and lays the foundation for the subsequent supervised fine-tuning of the labeled sample set.

Algorithm 1 Self-supervised training based on the text segment recovery task

Input: Unlabeled sample set in the social media text domain $\{(\boldsymbol{x}_i^t)\}$
Hierarchical keyword generation model M

for $k = 1 \sim epoch$ **do**
  for $i = 1 \sim N$ **do**
    $\boldsymbol{x}_i^t$—>$\boldsymbol{x}_{i\_SEP}^t$
    $\boldsymbol{x}_{i\_SEP}^t$—>$M.encoder$—>$\boldsymbol{H}$
    $\boldsymbol{H}$—>$M.Selection\ block$—>$\boldsymbol{f}$
    for $(j, \boldsymbol{SEG})$ **in** $\boldsymbol{x}_{i\_SEP}^t$ **do**
      random probability p
      if f_j **is** $True$ **do**
        if $p < p_1$ **then** $\boldsymbol{x}_{i\_del}^t = \boldsymbol{x}_{i_{SEP}}^t - \boldsymbol{SEG}$
      else do
        if $p < p_2$, **then** $\boldsymbol{x}_{i\_del}^t = \boldsymbol{x}_{i_{SEP}}^t - \boldsymbol{SEG}$
    end
    $\left(\boldsymbol{x}_{i\_del}^t, \boldsymbol{x}_{i\_SEP}^t\right)$—>$M$—>$\hat{\boldsymbol{y}}_i^t$
    $loss = -\sum_{i=1}^{n} \boldsymbol{x}_{i\_SEP}^t \log \hat{\boldsymbol{y}}_i^t$
    loss backward and optimizer step
  end
end

4. Experiments

4.1. Datasets

For the traditional text domain dataset, we use the "News2016zh" dataset provided by Xu et al. [30], which includes approximately 2.5 million Chinese news articles. For the social media text domain dataset, we use the "Weibo" dataset of approximately 40K Chinese posts provided by Wang et al. [19]. Since the "News2016zh" dataset contains a lot of noisy data, we use the jieba [31] tokenizer to process the samples, and then filter the samples using a stopword list [32], removing meaningless numbers, URLs, and special symbols. The samples in the "Weibo" dataset have already been basically processed using its collectors, so we only use the same stopword list [32] to filter the samples, and finally retain the samples with a text length of at least five. In addition, if the difference between the two datasets for transfer learning is too significant, it will negatively impact the experimental results. So, we exclude the samples that are too long in the "News2016zh" dataset according to the statistics of the "Weibo" dataset.

The statistics of the datasets are shown in Table 1.

Table 1. The statistics of the datasets.

Dataset		News2016zh	Weibo
Size		147,456	38,505
Text length	max	65	101
	min	5	5
	avg	21.39	20.70
Keyword length	max	29	36
	min	1	1
	avg	8.45	2.76
Number of keywords	max	12	9
	min	1	1
	avg	1.45	1.06

At the pre-training stage, we used the "News2016zh" dataset with a training set size of 131,072 and a validation set size of 16,384. At the fine-tuning and test stages, we used the "Weibo" dataset. For self-supervised fine-tuning, the training set size is 24,208 and the validation set size is 3297. For supervised fine-tuning, the maximum training set size is 10,000 and the maximum validation set size is 1000. The test set size is 1300.

4.2. Evaluation

We used the Rouge, which is commonly used in text generation tasks, and the metric F1@1, which is commonly used for evaluating keyword generation method, as the metrics for model evaluation. F1@K [19] compares the first K keywords generated with the ground-truth keywords to compute the F1 score.

In order to validate the performance of the HKG, several keyword generation models are selected as baselines.

- TF-IDF [33]: A commonly used unsupervised keyword extraction method that evaluates the importance of words or phrases in a text prediction database based on their frequency.
- TextRank [34]: A graph-based sorting algorithm that constructs a graph network based on the semantic relationships between words in text.
- TAKG [19]: A Seq2Seq model for social media text keyword generation tasks, which utilizes the topic information of the text to assist in keyword generation tasks [19].
- Transformer [14]: a Seq2Seq model based on a multi-head self-attention mechanism that performs well on several text generation-like tasks.
- BART [35]: A large-scale pre-trained language model based on the Transformer model, which can greatly improve the performance of downstream text generation tasks.

TF-IDF and TextRank are unsupervised methods used to compare the performance of keyword extraction and keyword generation methods, which we implement using jieba [31]. TAKG is specifically designed for social media text, and Transformer is the basic architecture of the HKG model, used to compare the performance when solving the problem of dispersed information for social media text, and to observe the advantages of HKG's transfer learning-based model training strategy. For TAKG, we use the model released by its authors [36], and for Transformer, we implement the model using the transformer module of the torch package. BART is a model based on transfer learning and the Transformer structure like HKG for comparing keyword generation capabilities when labeled data is limited. We fine-tune a pre-trained Chinese BART model to generate keywords [37]. Because [26–28] are models specifically designed for other NLP tasks, HKG will not be compared with them.

4.3. Implementation Details

For all keyword generation models based on the Transformer structure, we set the embedding layer dimension to 512, the number of self-attention heads to 8, the number of encoder and decoder layers to 6, the encoder and decoder layer dimensions to 512, and the feedforward neural network dimension to 2048. Based on the statistics of datasets, the maximum length of the text sequence input into the model is set to 64 and the maximum length of the keyword sequence is set to 24, with the excess being truncated. We use the Gelu function as the activation function, and the Euclidean distance calculation formula as the vector distance calculation function dis.

We trained all models using the Adam optimizer, with a gradient clipping threshold of 0.2, and initialized model parameters using Xavier. At the pre-training stage, the batch size was 128 and the learning rate was 2×10^{-4}. At the self-supervised fine-tuning stage, the batch size was 32 and the learning rate was 5×10^{-5}. At the supervised fine-tuning stage, the batch size was 64 and the learning rate was 1×10^{-4}. We used Python 3.8.8, Anaconda3, and Pytorch1.10.2, and all models were trained on NVIDIA GeForce RTX 2080 Ti.

4.4. Results and Analysis

4.4.1. Comparative Experiments

As shown in Table 2, we compared our proposed method HKG with the baselines on the "Weibo" dataset, and HKG outperformed the prior best model BART in all metrics by 0.44 point (Rouge-1), 1.12 points (Rouge-2), 1.03 points (Rouge-L), and 0.81 point (F1@1), respectively.

Table 2. Performance comparison of various models/methods.

Model	Rouge-1	Rouge-2	Rouge-L	F1@1
TF-IDF	2.36	0.64	2.12	2.36
TextRank	1.72	0.27	1.14	1.72
TAKG	-	-	-	21.57
Transformer	27.72	15.73	27.63	21.79
BART	32.34	20.70	30.09	26.36
HKG	32.78	21.82	31.12	27.20

Among all the models/methods, the two unsupervised methods, TF-IDF and TextRank, perform the worst because they cannot generate absent keywords. The two Seq2Seq-based models, TAKG and Transformer, perform better than the unsupervised methods. However, they need to be trained from scratch, and their performance on the same magnitude of the training dataset is worse than that of BART and HKG using transfer learning technology. In addition, HKG filters text information, which can help the model pay more attention to important information when generating keywords. Compared with BART, which is also based on transfer learning technology and the Transformer model, there is a slight improvement for HKG on Rouge-1, and HKG performs better on Rouge-2, Rouge-L, and F1@1, as the adequately trained models all perform well in generating individual tokens that correspond to keywords, while the HKG method is better at generating fluent keyword sequences and complete keywords. Table 3 further compares the performance of BART and HKG with different resource richness.

Table 3. Performance comparison of BART and HKG with different resource richness. ('0', '1 K', and '10 K' represent the size of the social media text domain labeled dataset.).

Model	Rouge-1			Rouge-2			Rouge-L			F1@1		
	0	1 K	10 K	0	1 K	10 K	0	1 K	10 K	0	1 K	10 K
BART	30.9	30.9	32.3	16.2	16.4	20.7	27.4	27.7	30.1	19.0	19.2	26.4
HKG	31.1	31.2	32.8	17.9	18.0	21.8	29.3	29.5	31.1	22.7	23.1	27.2

It can be seen that both HKG and BART perform closely on Rouge-1 with different sizes of labeled data training sets as Rouge-1 focuses on evaluating the model's ability to generate individual tokens of keywords, which pre-trained models have basically acquired. When fine-tuning the keyword generation models using only a 1K-sized labeled data training set, the BART model performs worse, especially on the Rouge-2, Rouge-L, and F1@1 metrics, due to the lack of textual fluency and coherence of the generated keyword sequences, and the better performance of the HKG should be attributed to the fact that it has learned the hierarchical information of text when generating keywords. Also, when using a 10K-sized labeled data training set, it can be seen that the HKG method basically performs better. In conclusion, HKG is more suitable for the low-resource social media text domain.

4.4.2. Ablation Experiments

The training process of the keyword generation model can be divided into a pre-training stage, a self-supervised fine-tuning stage, and a supervised fine-tuning stage.

Table 4 shows the results of model performance at different training stages. Although there is a significant difference between the traditional text dataset "News2016zh" and the social media text dataset "Weibo", the model with only supervised pre-training still has acceptable performance as the "News2016zh" has a rich corpus. The model with supervised fine-tuning performs worst as the social media text dataset size is relatively small. As for the self-supervised fine-tuning stage, it is evident that adding this training stage to the "Pre-training and Fine-tuning" process with the text segment recovery task can effectively help the keyword generation model adapt to the target social media text domain.

Table 4. Performance of different model training stages. ("√" indicates that the model was trained at this stage and "×" indicates that the model was not trained at this stage.).

Training Stage			Rouge-1	Rouge-2	Rouge-L	F1@1
Pre-Training	Self-Supervised Fine-Tuning	Supervised Fine-Tuning				
√	×	×	30.77	13.29	19.87	15.38
×	×	√	14.60	8.34	14.24	12.87
√	×	√	31.24	20.27	29.91	25.21
√	√	√	32.78	21.82	31.12	27.20

4.4.3. Model Hyperparameter Experiments

We use several settings of the length l of text segments and the number k of significant text segments to explore the effect on the hierarchical keyword generation model's performance, as shown in Figure 2. From the result, the length of the text segment l has a more significant impact on HKG's performance. When l is set too small, less information is contained in each text segment, and the critical information is likely to be filtered out while selecting text segments, which may lead to a lack of fluency and coherence in the generated keyword sequences. When l is set too long, the model's performance shows little difference in filtering different numbers of significant text segments. When $l = 6$, the information contained in a text segment is relatively complete and coherent, so when $k = 2$, most of the critical information in the text sequence is retained, and the redundant irrelevant information is filtered, obtaining the best keyword generation performance.

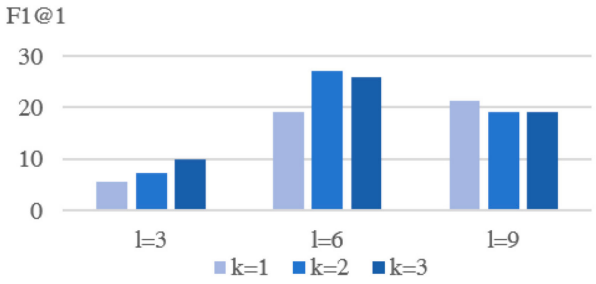

Figure 2. Performance comparison (F1@1 scores) of different text segment lengths and number of important text segments settings.

4.4.4. Case Study on the Real Social Media Platform

On Sina Weibo, a popular social media platform in China, if a Weibo post is labeled with a suitable topic tag, it will be easier to classify and search. According to the keywords generated from Weibo post content, corresponding topic tags can be attached to Weibo posts, which affects the popularity of Weibo, as shown in Table 5.

Table 5. A case of generated topic tags for a Weibo post.

Weibo Post:	马航客机失联接近两星期，中国未有停止搜索行动。中国国防部新闻发言人表示，军方将根据马方提供的最新信息及前一阶段搜寻情况继续保持足够的搜寻兵力，配合卫星和雷达扩大搜寻范围，加大搜救力度。 (The Malaysia passenger plane lost contact for nearly two weeks, and China has not stopped its search operations. A spokesperson for the Ministry of National Defense of China states that the military will continue to maintain sufficient search forces based on the latest information provided by the Malaysian side and the previous stage of search, cooperate with satellites and radar to expand the search range, and increase search and rescue efforts.)
BART:	#马航客机失联# (#Malaysia Passenger Plane Lost Contact#)
HKG:	#马航飞机失联# (#Malaysia Airplane Lost Contact#)

In this case, for the same Weibo, the keyword generated using the BART directly corresponds to the noun "客机 (passenger plane)" in the Weibo, while the HKG uses the more widely expressed noun "飞机 (airplane)". However, the topic tag "#马航客机失联# (#Malaysia Passenger Plane Lost Contact#)" has 223 thousand discussions, and "#马航客机失联# (#Malaysia Passenger Plane Lost Contact#)" has 3 million 823 thousand discussions, which means the public will choose the more widely expressed noun "飞机 (airplane)" when discussing the event. The topic tags generated using HKG can bring more popularity to Weibo posts.

5. Conclusions

This paper focuses on generating keywords for low-resource social media text based on transfer learning technology. While fine-tuning, a self-supervised text segment recovery task is designed to help the keyword generation model adapt to the target social media text domain, and a hierarchical model structure is proposed based on the characteristics of social media text. Experiments on actual social media datasets have demonstrated that our proposed method can effectively improve the keyword generation performance of low-resource social media text. As a next step, we will further improve our keyword generation method, taking into account the theme, hot words, and contextual information of social media text.

Author Contributions: Methodology, writing—original draft preparation, X.G.; writing—review and editing, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Guangdong Basic and Applied Basic Research Foundation (2023A1515012833) (JB), the Science and Technology Program of Guangzhou (202201010544) (JB), the National Natural Science Foundation of China (61901192) (JB), Guangdong Provincial Key Laboratory of Traditional Chinese Medicine Informatization (2021B1212040007) (YL), and Science and Technology Projects in Guangzhou (2023B03J1297) (HW).

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [https://github.com/yuewang-cuhk/TAKG], accessed on 1 September 2023.

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

References

1. Hammouda, K.M.; Matute, D.N.; Kamel, M.S. Corephrase: Keyphrase extraction for document clustering. In Proceedings of the Machine Learning and Data Mining in Pattern Recognition: 4th International Conference (MLDM), Leipzig, Germany, 9–11 July 2005; pp. 265–274.
2. Zhang, C.; Yang, Q.; Zhang, J.; Gou, L.; Fan, H. Topic Mining and Future Trend Exploration in Digital Economy Research. *Information* **2023**, *14*, 432. [CrossRef]
3. Wu, X.; Bolivar, A. Keyword extraction for contextual advertisement. In Proceedings of the 17th International Conference on World Wide Web, Beijing, China, 21–25 April 2008; pp. 1195–1196.
4. Dave, K.S.; Varma, V. Pattern based keyword extraction for contextual advertising. In Proceedings of the 19th ACM international conference on Information and knowledge management, Toronto, Canada, 26–30 October 2010; pp. 1885–1888.

5. Jones, S.; Staveley, M.S. Phrasier: A system for interactive document retrieval using keyphrases. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, CA, USA, 15–19 August 1999; pp. 160–167.
6. Boudin, F.; Gallina, Y.; Aizawa, A. Keyphrase generation for scientific document retrieval. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), Online, 5–10 July 2020; pp. 1118–1126.
7. Zhang, Y.; Zincir-Heywood, N.; Milios, E. World wide web site summarization. *Web Intell. Agent Syst. Int. J.* **2004**, *2*, 39–53.
8. Banbhrani, S.K.; Xu, B.; Liu, H.; Lin, H. SC-Political ResNet: Hashtag Recommendation from Tweets Using Hybrid Optimization-Based Deep Residual Network. *Information* **2021**, *12*, 389. [CrossRef]
9. Berend, G. Opinion Expression Mining by Exploiting Keyphrase Extraction. In Proceedings of the 5th International Joint Conference on Natural Language Processing, Chiang Mai, Thailand, 8–13 November 2011; pp. 1162–1170.
10. Wang, H.; Wang, Y. EREC: Enhanced Language Representations with Event Chains. *Information* **2022**, *13*, 582. [CrossRef]
11. Meng, R.; Zhao, S.; Han, S.; He, D.; Brusilovsky, P.; Chi, Y. Deep Keyphrase Generation. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 582–592.
12. Gallina, Y.; Boudin, F.; Daille, B. KPTimes: A Large-Scale Dataset for Keyphrase Generation on News Documents. In Proceedings of the 12th International Conference on Natural Language Generation (INLG), Tokyo, Japan, 29 October–1 November 2019; pp. 130–135.
13. Li, Y.; Zhang, Y.; Zhao, Z. CSL: A Large-scale Chinese Scientific Literature Dataset. In Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 3917–3923.
14. Vaswani, A.; Shazeer, N.; Parmar, N. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17), Red Hook, NY, USA, 4 December 2017; pp. 6000–6010.
15. Cho, K.; van Merrienboer, B.; Gülçehre, Ç. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
16. Chen, J.; Zhang, X.; Wu, Y. Keyphrase Generation with Correlation Constraints. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 13 November 2018; pp. 4057–4066.
17. Zhang, Y.; Xiao, W. Keyphrase Generation Based on Deep Seq2Seq Model. *IEEE Access* **2018**, *6*, 46047–46057. [CrossRef]
18. Chen, W.; Gao, Y.; Zhang, J. Title-Guided Encoding for Keyphrase Generation. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 6268–6275.
19. Wang, Y.; Li, J.; Chan, H.P. Topic-Aware Neural Keyphrase Generation for Social Media Language. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 17 July 2019; pp. 2516–2526.
20. Kim, J.; Jeong, M.; Choi, S. Structure-augmented keyphrase generation. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Virtual, 7–11 November 2021; pp. 2657–2667.
21. Yang, P.; Ge, Y.; Yao, Y. GCN-based document representation for keyphrase generation enhanced by maximizing mutual information. *Knowl. Based Syst.* **2022**, *243*, 108488.
22. Ye, H.; Wang, L. Semi-Supervised Learning for Neural Keyphrase Generation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 13 November 2018; pp. 4142–4153.
23. Wang, Y.; Liu, Q.; Qin, C. Exploiting Topic-Based Adversarial Neural Network for Cross-Domain Keyphrase Extraction. In Proceedings of the 2018 IEEE International Conference on Data Mining, Sentosa, Singapore, 17–20 November 2018; pp. 597–606.
24. Guo, L.; Sun, H.; Qi, Q. Keyword Extraction Algorithm Based on Pre-training and Multi-task Training. In Proceedings of the Sixth International Congress on Information and Communication Technology, Singapore, 10–11 November 2022; pp. 723–734.
25. Sun, S.; Liu, Z.; Xiong, C. Capturing Global Informativeness in Open Domain Keyphrase Extraction. In Proceedings of the Natural Language Processing and Chinese Computing: 10th CCF International Conference (NLPCC), Qingdao, China, 13–17 October 2021; pp. 275–287.
26. Bhat, G.; Saluja, A.; Dye, M.; Florjanczyk, J. Hierarchical Encoders for Modeling and Interpreting Screenplays. In Proceedings of the Third Workshop on Narrative Understanding, Online, 22 March 2021; pp. 1–12.
27. Wang, Z.; Wang, P.; Huang, L.; Sun, X.; Wang, H. Incorporating Hierarchy into Text Encoder: A Contrastive Learning Approach for Hierarchical Text Classification. In Proceedings of the Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022.
28. Sakhrani, H.; Parekh, S.; Ratadiya, P. Transformer-based Hierarchical Encoder for Document Classification. In Proceedings of the 2021 International Conference on Data Mining Workshops (ICDMW), IEEE, Auckland, New Zealand, 7–10 December 2021; pp. 852–858.
29. Wu, D.; Ahmad, W.U.; Dev, S. Representation Learning for Resource-Constrained Keyphrase Generation. In Proceedings of the Findings of the Association for Computational Linguistics (EMNLP), Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 700–716.
30. NLP Chinese Corpus: Large Scale Chinese Corpus for NLP. Available online: https://zenodo.org/records/3402023 (accessed on 7 September 2019).
31. Fxsjy, Jieba. Available online: https://github.com/fxsjy/jieba (accessed on 20 January 2020).
32. Goto456, Stopwords. Available online: https://github.com/goto456/stopwords (accessed on 12 May 2023).
33. Salton, G.; Yang, C.S.; Yu, C.T. A Theory of Term Importance in Automatic Text Analysis. *J. Am. Soc. Inf. Sci.* **1975**, *26*, 33–44.

34. Mihalcea, R.; Tarau, P. TextRank: Bringing Order into Text. In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 404–411.
35. Lewis, M.; Liu, Y.; Goyal, N. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 19 June 2020; pp. 7871–7880.
36. Yuewang-Cuhk, TAKG. Available online: https://github.com/yuewang-cuhk/TAKG (accessed on 5 August 2019).
37. Fnlp, Bart-Base-Chinese. Available online: https://huggingface.co/fnlp/bart-base-chinese (accessed on 30 December 2022).