electronics

MDPI

Article

An Enhanced Method on Transformer-Based Model for ONE2SEQ Keyphrase Generation

Lingyun Shen [1,2] and Xiaoqiu Le [1,2,*]

[1] National Science Library, Chinese Academy of Sciences, Beijing 100190, China; shenlingyun@mail.las.ac.cn
[2] Department of Library, Information and Archives Management, School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China
* Correspondence: lexq@mail.las.ac.cn

Abstract: Keyphrase generation is a long-standing task in scientific literature retrieval. The Transformer-based model outperforms other baseline models in this challenge dramatically. In cross-domain keyphrase generation research, topic information plays a guiding role during generation, while in keyphrase generation of individual text, titles can replace topic roles and convey more semantic information. As a result, we proposed an enhanced model architecture named TAtrans. In this research, we investigate the advantages of title attention and sequence code representing phrase order in keyphrase sequence in improving Transformer-based keyphrase generation. We conduct experiments on five widely-used English datasets specifically designed for keyphrase generation. Our method achieves an F1 score in the top five, surpassing the Transformer-based model by 3.2% in KP20k. The results demonstrate that the proposed method outperforms all the previous models on prediction present keyphrases. To evaluate the performance of the proposed model in the Chinese dataset, we construct a new Chinese abstract dataset called CNKIL, which contains a total of 54,546 records. The F1 score of the top five for predicting present keyphrases on the CNKIL dataset exceeds 2.2% compared to the Transformer-based model. However, there is no significant improvement in the model's performance in predicting absent keyphrases.

Keywords: keyphrase generation; transformer; title attention; keyphrase order

Citation: Shen, L.; Le, X. An Enhanced Method on Transformer-Based Model for ONE2SEQ Keyphrase Generation. *Electronics* **2023**, *12*, 2968. https://doi.org/10.3390/electronics12132968

Academic Editors: Padma Iyenghar and Elke Pulvermüller

Received: 25 April 2023
Revised: 19 June 2023
Accepted: 26 June 2023
Published: 5 July 2023

1. Introduction

Keyphrase generation is a task to automatically produce a set of phrases or words which capture and represent the main idea of the given text. The main difference in keyphrase generation is that the term usually consists of several words as a semantic unit with richer semantic information. Keyphrase generation is an essential task in natural language understanding, such as text summarization [1–3], information retrieval [4], and digital library [5–7], especially in the scientific literature field. The primary advantage of keyphrase generation is its ability to facilitate a quick grasp of primary knowledge and the topic of long text or document.

The typical paradigm of keyphrase generation is called the ONE2ONE paradigm, which treats the target keyphrase as a discrete set rather than an ordered sequence. Each phrase sequence corresponding to the original text is divided into single phrases and paired with the text one by one. The decoder generates prediction target phrases using a greedy algorithm or beam search. However, these methods must generate keyphrases in an over-generated manner and cannot control the exact number of the phrase, as different texts may have different number of keyphrases. Yuan et al. [8] proposed the ONE2SEQ paradigm to generate keyphrases using an RNN-based model. They take the source text and its entire keyphrase sequence as a data pair in the ONE2SEQ paradigm. This training paradigm retains the inherent characteristic of the order information and accommodates the variable number of keyphrases in different texts or documents.

However, the methods mentioned above ignore the global effect of the title on the text information, and only treat the title as another part of the text. In the keyphrase generation task within an individual text, the title plays a crucial role in providing topic and summary information. Previous research has demonstrated that the title can offer more topic-related and comprehensive information about the text [9].

Furthermore, the utilization of a more efficient encoder-decoder architecture, such as Transformer layers, has been proven to enhance the generation performance [9–13]. Some researchers have explored the ONE2SEQ method using a Transformer-based model [14] and have suggested that Transformers outperform the RNN-based model. The primary reason behind this improvement is the utilization of self-attention and cross-attention mechanisms in Transformers, which enable more effective use of context information and long-distance dependency. Despite these developments, the challenge of keyphrase generation in the sequence-to-sequence paradigm remains insurmountable. The over-generate decoder tends to generate duplicated phrases when encountering the delimiter. Since the model's training mechanism involves predicting the next token, these models tend to predict the same token repeatedly when encountering identical delimiters.

Considering the problems mentioned above, this paper proposes an improved keyphrase generation model based on a Transformer encoder and decoder trained in the ONE2SEQ paradigm called TAtrans. Firstly, a title attention mechanism is used to guide the encoding of the text, and then a phrase order code is used to distinguish each keyphrase and follow the delimiter to avoid generating duplicated phrases. Additionally, an additional copy mechanism is incorporated to enhance keyphrase accuracy [15].

Our model is evaluated on five benchmark English datasets and one Chinese abstract-keyphrase dataset we collected. As common evaluation metrics in keyphrase generation research, we use the precision, recall, and F1 score to assess the performance of our method and other recently advanced models. The results indicate that our method outperforms the baseline models across all datasets.

In summary, this work makes the following contributions and innovations:

- The proposed model integrates title messengers in a cross-attention manner, which aligns with the structure of the Transformer model itself;
- The order embedding of the keyphrases in the sequence enables the model to capture order information between phrases, thereby enhancing the diversity of generated keyphrases;
- We have constructed a new Chinese abstract keyphrase dataset CNKIL, which contains a total of 54,546 abstracts and their corresponding keyphrase data.

In the remainder of this paper, we will first review the relevant research work in Section 2. Then Section 3 will elaborate on the proposed method and present the details of this paper's work. After that, Section 4 states the implementation details of the experiments. In Section 5, we report the results of the keyphrase prediction and ablation study and discuss the effectiveness of the proposed method through concrete examples. Finally, Section 6 concludes the paper and discusses this research's contributions, limitations, and future work.

2. Related Work

Keyphrase prediction for given text or document has two strategies. One is extraction's way, and another is the generation's way. The first one aims to directly extract present keyphrases that continuously appear in the input text. Early research based on statistical algorithms and machine learning methods is focused on keyphrase extraction. As deep learning prosperity in natural language processing, many dominant models try to achieve keyphrase generation, which can generate both present and absent keyphrases that do not match any contiguous subsequence of the input text [16].

2.1. Keyphrase Extraction

Keyphrase extraction is roughly classified as supervised and unsupervised methods, with the main difference being whether the training data is used to guide the model or algorithm in fitting the data.

In the early stage, the ideas of keyphrase extraction rely heavily on feature engineering and statistical algorithms, such as taking advantage of part-of Speech [17] or n-gram [18] to represent potential keyphrases, or a valid statistical model to rank the phrases appearing in the text. For example, TF-IDF [18–20] is a classical statistically effective method in keyphrase extraction. Another well-known strategy is graph-based algorithms, such as TextRank [21], PageRank [11], TopicRank [22], etc. [23,24]. These unsupervised methods may quickly start and apply in cross-domain without training costs like supervised methods, but their performance in practice cannot exceed supervised methods. Early supervised methods are mainly based on the Naïve Bayes algorithm [25] and Support Vector Machine (SVM) ranking [26]. They produce a ranking model for extracting keyphrases by fitting training data. Lately, some unsupervised methods have even outperformed traditional machine learning methods, such as Yake [27], integrated features ranking method into their developed system, and exceeded the supervised model KEA [18] on the benchmark. The pre-trained word-embedding methods used to represent the word or sentence semantic also further improved the accuracy of keyphrase extraction because of its rich semantic information within the embedding [28,29].

A fundamental drawback of traditional unsupervised methods is only counting several features unable to capture the deep semantic information. In recent years, thanks to the development of deep learning, the neural network model for keyphrase extraction shows a stronger ability to identify the core information of a text. Constructing a multi-layer neural network can obtain more deep semantic information, such as the keyphrase extraction RNN-based model [30]. DivGraphPointer [31] uses Graph Convolutional Networks GCN [32] to construct a word graph ranking all phrases in the input text. This method exceeds the traditional TextRank [21] by 18% on the Inspec dataset. TANN [33] enhances the adversarial neural network with a topic feature, outperforming all keyphrase extraction baseline models. Apart from feature fusion, the attention mechanism has been proven as an effective way to further improve the neural model's performance on keyphrase prediction. Yingyi Zhang and Chengzhi Zhang [34] consolidated the attention layer with the RNN model to get a better result than a single RNN.

In addition, Haoran Ding and Xiao Luo et al. [35] exploit the combination of attention and pre-trained BERT model achieved a state-of-art competitive outcome among all unsupervised methods. The first two rows of Table 1 list the main keyphrase methods for keyphrase extraction.

2.2. Keyphrase Generation

Unlike keyphrase extraction, keyphrase generation can yield phrases that do not appear in the source text. The keyphrase generation model has benefited the encoder-decoder architecture and can produce new tokens that do not appear in the text. Thus, the majority of keyphrase generation models currently in use are based on this architecture. Meng et al. first proposed a generative sequence-to-sequence (seq2seq) model for keyphrase prediction named CopyRNN [36].

This model separates the text corresponding to keyphrases from multiple one-to-one (one text corresponds to one phrase, ONE2ONE) data pairs [36]. They feed the ONE2ONE data into the model and generate target phrases by the decoder with a beam search.

The obvious drawback of this model is that it can neither control the number of phrases nor reflect the semantic order between phrases. It can only generate as many phrases as possible through an exhaustive search. Therefore, they improved the ONE2ONE paradigm by directly using the text and the corresponding ground truth keyphrase sequence as training samples to become the ONE2SEQ diagram [8]. A Self-termination strategy is implemented by adding the phrase delimiter "<*peos*>" and ending token "<*eos*>".

After the transformer's success in NLP, Jiang et al. explored the keyphrase generation task of the transformer model to further improve the generation results [37]. However, the RNN or transformer architecture models both encoded the title the same as the text.

Based on Meng et al.'s research [36], Wang Y et al. [38] used a neural network model to guide the keyphrase generation process in a media dataset. Yue Wang et al. indicated that integrating topic information helps recognize keyphrases more accurately [38]. However, for a single abstract text, the topic is hard to identify, so Chen W et al. [39] proposed a title-guided method to instruct the keyphrase generation of a single text. Their methods both improved the CopyRNN by more than 3%.

As mentioned above, the title indicates the topic content of the text and has more semantic relevance with the keyphrase. For single text, the title implies the topic information and other more abundant information about the text body. Our paper aims to take advantage of title information to guide the keyphrase generation model based on transformer architecture. The third row in Table 1 lists the main methods and model architecture for generating keyphrases.

Table 1. Comparison of main methods for keyphrase prediction. The first line is the keyphrase extraction method based on statistical and machine learning models. The second row lists the keyphrase extraction methods by neural models, and the third row lists the main methods for keyphrase generation.

Prediction Type	Main Algorithm or Back-Bone Models	Name	Advantage & Disadvantage
Keyphrase extraction	Using word part of speech	feature engineering [17]	Unsupervised, simply but low accurate
	TF-IDF	TF-IDF [18–20]	Unsupervised, simply but low accurate
	graph-based algorithm	TextRank [21]	unsupervised, considered global information, not need extra feature
	graph-based algorithm	PageRank [11]	unsupervised, considered global information, not need extra feature
	graph-based algorithm	TopicRank [22]	unsupervised, considered global information, not need extra feature
	Naïve Bayes algorithm	[25]	unsupervised, considered global information, not need extra feature
	features ranking	Yake [27]	Integrated system, accessibility
	TF-IDF + Naïve Bayes	KEA [18]	supervised, improved than unsupervised methods
Keyphrase extraction	sentence embedding	Sentence Embeddings [29]	including more semantic information
	Pre-trained embedding	Embedrank [29]	including more semantic information
	RNN	RNN-base model [30]	including context information
	Graph Convolutional Networks GCN	DivGraphPointer [31]	combination with graph-based method and convolutional network
	GRU + human attention	human attention GRU [34]	including context information and focusing on key information
	BERT + attention	AttentionRank [35]	Using pre-trained language model, not need to train a model from scratch
Keyphrase generation	RNN + copy mechanisim	CopyRNN [36]	neglecting phrase sequence order
	RNN + one-to-sequrence training pardiam	CatSeq [8]	generating one phrase once
	GRU + Neural Topic Model	Topic-Aware Keyphrase generation model [38]	given more topic information
	transformer	One2Set [37]	generating unordered keyphrases
	GRU	CatSeqTG [39]	given more title information but neglect long distance dependency

The existing research on the keyphrase generative model has not targeted the improvement of the transformer architecture model, nor has it conducted in-depth research on the method of using one-to-sequence training data to train the model. This paper proposes

an enhanced method based on the Transformer-based model for ONE2SEQ keyphrase generation to address these two problems.

3. Proposed Approach

3.1. Problem Definition

The goal of this task is to predict the target phrase sequence given a source text, which consists of a title and a text body. We use $X = \{x_1, x_2, \ldots, x_N\}$ to denote source text, x_i is the i-th word of the text, N is the total number of words for X. $T_i = \{t_1, t_2, \ldots, t_M\}$ denotes the title, t_i is the i-th word of the title sequence, M is the total words number of T_i. Text corresponding ground truth keyphrase sequence is denoted by $Y = \{y^1; y^2; \ldots; y^Z\}$, y^j is the j-th keyphrase of X, Z denotes the number of keyphrases for X. For the j-th keyphrase $y^j = \left\{y_1^j, \ldots, y_k^j\right\}$, y_k^j is the k-th word of the phrase y^j, k is the number of words in the phrase y^j [39].

Different from the ONE2ONE training paradigm, we divide the phrase sequence into M individual phrases paired with text. We composed the training data pairs by (X, Y) based on the training paradigm of ONE2SEQ. This is conducive to preserving original information in terms of the order and length of the phrase sequence. The main purpose of this model is to generate a keyphrase sequence in a certain order.

3.2. The Framework of TAtrans

We first preprocess the original corpus to eliminate non-standard characters and uniform text formatting. If the text needs to be segmented, for example, when the text is Chinese, we use Jieba to segment the text. Our training data includes a segment of text and corresponding keyphrase sequences. This is a typical ONE2SEQ (one-to-sequence) training data construction method. Then the text is sent to the Transformer encoder to get the Initialization vector, and then through the title vector and attention guidance, the final embedding with title attention is obtained. This is embedded into the transformer decoder, and the model ultimately produces a predicted keyphrase sequence. The overall process of this work is shown in Figure 1.

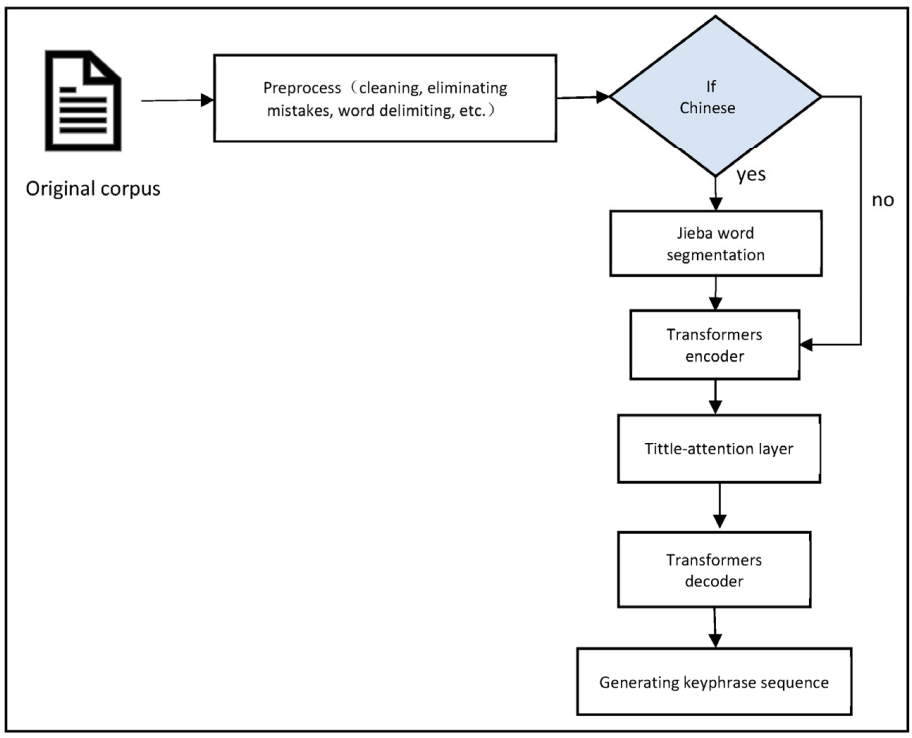

Figure 1. The flow chart of the proposed TAtrans generating framework.

3.3. Model Architecture

The model proposed in this paper, named TAtrans, builds on a Transformer-based encoder and decoder as the framework. In the encoder part, we randomly initialize word vectors and use a stack of transformers to learn the contextual embedding. Then, we added a title attention layer after the Transformers encoder layers to guide the contextual embedding of text. The decoder will generate the keyphrase sequence according to the generated token. Before feeding the generated token into the decoder, the position embedding and a fixed set of learned phrase order code embedding are taken as additional input of the decoder, which is used to alleviate generating duplicate words. The order code distinguishes the keyphrase and its following delimiter position in the target phrase sequence. An extra copy mechanism layer was added to the decoder. The overall architecture of our model is shown in Figure 2.

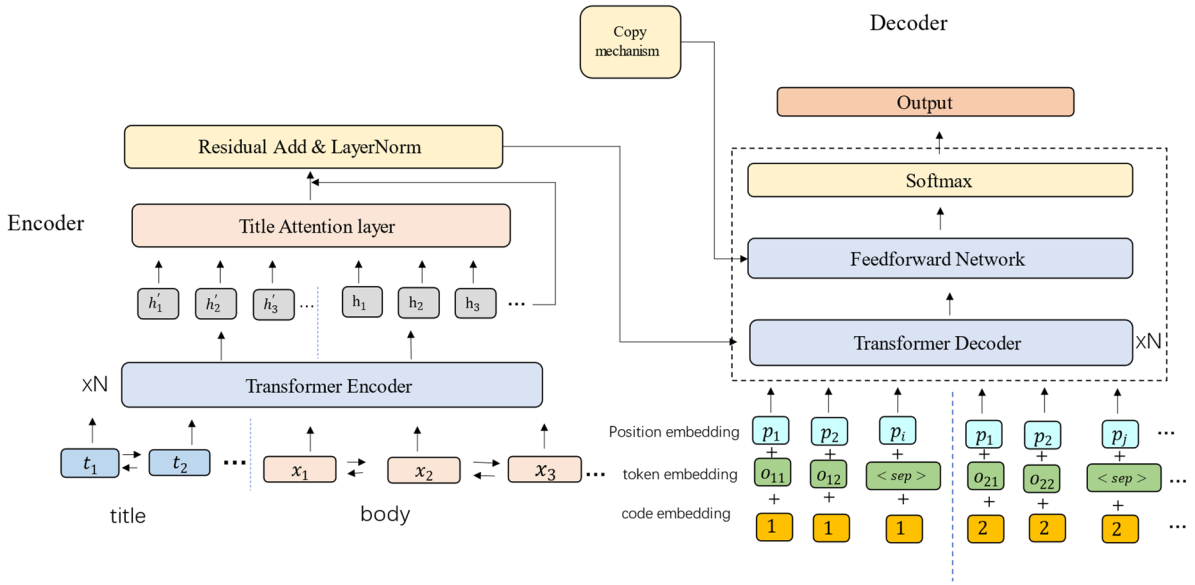

Figure 2. The architecture of the TAtrans model. Position embedding represents the place of the token in the whole keyphrase sequence. We use the number k to represent the *k*-th keyphrase in the ground truth keyphrase sequence. One phrase shares a sequence code.

3.4. Title Enhanced Encoder

3.4.1. Transformer Encoder Layer

At first, the text and its title are fed into a stack of N = 6 Transformer layers, and we get their original hidden from them. The hidden states of text and title are represented as follows:

$$H = Transformer(x_1, x_2 \ldots, x_N) = \{h_1, h_2 \ldots, h_N\} \quad (1)$$

$$T = Transformer(t_1, t_2 \ldots, t_M) = \{h_1^t, h_2^t, \ldots, h_M^t\} \quad (2)$$

where h_i denotes the hidden state of input token x_i of text. h_i^t denotes the hidden state of input token t_i of the title. The hidden state matrices are represented by H and T.

3.4.2. Title Attention

A general method of computing similarity is the inner product between two vectors, whose results reflect how relevant each word of text is to the title. We use $h_i^t \cdot h_j$ to represent the similarity between the i-th token's embedding of the title and the j-th token's embedding of the text body. The same as the cross-attention calculation [10], we regard T as K in cross-attention, H as Q in cross-attention, and V to calculate the similarity matrix of text and title:

$$score = h_i^t \cdot h_j \quad (3)$$

$$\alpha_{i,j} = \frac{\exp\left(h_i^t \cdot h_j\right)}{\sum_1^N \left(\exp\left(h_i^t \cdot h_{j'}\right)\right)} \tag{4}$$

where score is the product of the i-th token's embedding of the title and the j-th token's embedding of the text body output by the transformers layer. $i \in (0, M-1)$, $j \in (0, N-1)$. $\alpha_{i,j}$ is the attention weight of the i-th token of the title and the j-th token of the text. The j-th token of text similarity degree is

$$A = \{\alpha_1, \alpha_1, \ldots, \alpha_M\} \tag{5}$$

where $\alpha_i = \left\{\alpha_{i,j'}\right\}_{j'=1\ldots N}$. Finally, the title similarity matrix is formulated as $A \cdot T$.

Consequently, the hidden state of the text is added to the title similarity matrix above. We use the transformer optimizing method [10], followed by a residual connection and normalization layer, to avoid the embedding form rapidly growing. The encoder's output is as follows:

$$Output_{encoder} = LayerNorm(AT + H) \tag{6}$$

The *LayerNorm* layer is used to transform the embedding distribution between 0 and 1. The computation process of title attention is shown in Figure 3.

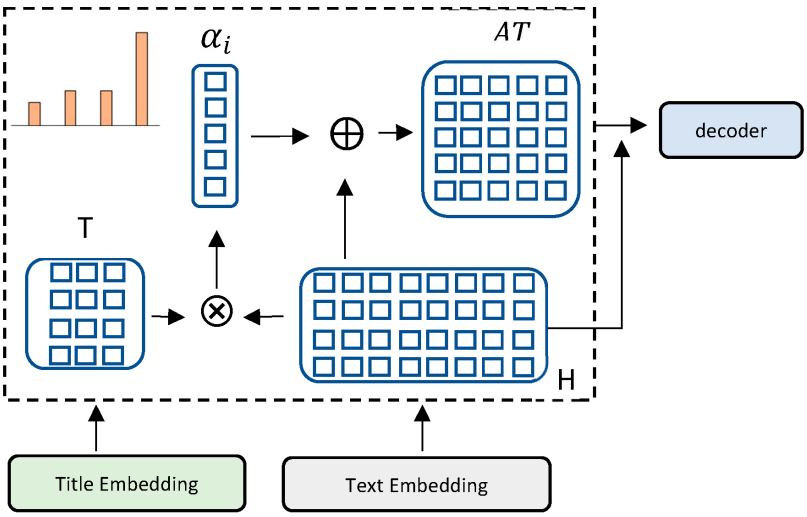

Figure 3. The computation process of title attention.

3.5. Keyphrase Sequence Code Embedding

In the decoder, we refer to Jiang Ye et al.'s [37] idea, which used a sequence code embedding to indicate the difference between generated phrases to avoid generating duplicate phrases. We add a set of phrase sequence code embedding as a part of the input. Each embedding represents the order of each phrase in a keyphrase sequence. The input of the decoder is as follows:

$$d_t^n d = e^{\omega}_{y_{t-1}^n} + e_t^p + c^n \tag{7}$$

where $e^{\omega}_{y_{t-1}^n}$ denotes the embedding of word y_{t-1}^n, e_t^p is the t-th sinusoid positional embedding, as in the Attention paper [10], and c^n is the n-th learned sequence code embedding. The decoder outputs the predictive distribution p_t^n, which is used to get the next word y_t^n. We set k as the max number k of the keyphrase sequence code. Intuitively, k is greater than the number of truth phrases.

Inspired by previous multiple relation extraction [40], there is an imbalance distribution of embeddings due to the delimiters in the sequence appearance repeatedly. We introduce a special token as the delimiter and set a threshold of 0, where those above the threshold are classified as phrases and those below the threshold are classi-

fied as delimiters. The decoder will predict the distribution of the next keyphrase token based on the joint input until encountering the terminal character "<*eos*>". We also employ a copy mechanism [41], which is generally adopted for many previous works of keyphrase generation.

3.6. Training

3.6.1. Sequence Code Training Loss

We use the Hungarian algorithm [42] to optimize the max matching between ground truth y_t^n at time step t and the prediction of the x-th phrase. Given the correspondence sequence code and the target phrase, we train the model under maximum likelihood loss, which is formulated as follows:

$$L(\theta) = -\sum_{n=1}^{N} \sum_{t=1}^{|y^n|} log \overline{p}_t^{\hat{\pi}(n)}(y_t^n) \tag{8}$$

where $\overline{p}_t^{\hat{\pi}(n)}(y_t^n)$ denotes the probability of word y_t^n, n denotes the index of x, $\hat{\pi}(n)$ is the parameter to make the correspondence between y_t^n and p^n calculated by the Hungarian algorithm lowest.

3.6.2. Joint Training Loss

In order to make the model have different capabilities over predicting present and absent keyphrases, a joint loss [8] was set to represent the present keyphrase and the absent keyphrase prediction, respectively,

$$L(\theta) = -\left(\gamma \sum_{n=1}^{\frac{N}{2}} \sum_{t=1}^{|y^n|} log \overline{p}_t^{\hat{\pi}^{pre}(n)}(y_t^n) + (1-\gamma) \sum_{n=\frac{N}{2}+1}^{N} \sum_{t=1}^{|y^n|} log \overline{p}_t^{\hat{\pi}^{abs}(n)}(y_t^n) \right) \tag{9}$$

γ is a hyperparameter that denotes the weight to limit the proportion of different phrase types.

4. Experiment Settings

4.1. Datasets

We conduct our experiments on five keyphrase benchmark datasets, which include KP20K, Inspec, Krapivin, NUS, and SemaEval-2010. In order to verify the cross-language ability of the model, 50K pieces of data of abstracts and corresponding keyphrases retrieved from CNKI were collected. The details of the datasets are as follows:

1. KP20k [36], which is a large-scale academic abstract dataset from ACM Digital Library, ScienceDirect, Wiley, Web of Science, etc., contains 528K records for training, 20K records for validation, and 20K records for testing;
2. Inspec [25], which contains 2000 abstracts with corresponding titles and keyphrases of journal papers from 1998 to 2002;
3. Krapivin [43] consists of 2304 scientific papers from the computer science domain published by the association for computing machinery (ACM). It has subsequently been verified by the reviewers;
4. NUS [44] also contains a scientific dataset consisting of 211 full papers with their keyphrases annotated by student volunteers;
5. SemEval-2010 [45] is an automatic keyphrase extraction task dataset collected from the ACM Digital Library, which contains 244 full papers with corresponding keyphrases;
6. CNKIL is a dataset collected by ourselves through retrieval of CNKI, involving information science and library science. We integrate a 20Krecord of library science abstracts from Xiatian's dataset [46] into ours to form the CNKIL dataset, which contains a total of 54,546 abstracts and corresponding keyphrase data.

In our experiment, we use KP20K raw data to train English models, including 512,462 samples for training, 20K for validation, and 20K for testing. Then we use CNKIBL

as the Chinse model training dataset, which includes 44,546 training data, 5000 verification data, and 5000 test data.

4.2. Baseline

We compare our method with the most advanced keyphrase generation models as baselines, as follows:

- CopyRNN [36] uses the ONEONE training mode RNN sequence-to-sequence model with copy mechanism [15];
- CatSeq [8] is an RNN-based model with a copy mechanism the same as CopRNN, but trained by ONE2SEQ paradigm;
- CatSeqTG [39], an extension of CatSeq [8] with additional title encoding and cross-attention;
- Transformer-based [37], a Transformer-based model with copy mechanism trained under ONE2SEQ paradigm;
- TAtrans (our method) is based on the Transformer-based encoder and decoder. The title attention and sequence code embedding methods are used to enhance the performance.

4.3. Implementation Details

Similar to the ONE2SEQ training paradigm proposed by Xingdi Yuan et al. [37] and Jiacheng Ye et al. [8], the data sample paired text and its keyphrase sequence. We use "<*eos*>" to split the title and abstract context and detach the embedding matrix after Transformer encoding layers, which is conducive to speeding up model training and sharing parameters. The original order of keyphrase sequences is sorted by authors. Our model encoder and decoder each include six-layer Transformers, the number of multi-head attention heads is eight, the learning rate is 0.0001, and the batch size is 12. During the decoding process, we use beam search decoding. We set λ to 0.5, word and sequence code embeddings to 512 dimensions, the maximum number of phrases to 20, and the batch size to 12. Dropout rate = 0.1. We set the maximum depth of beam search as six and the beam size as 200. The experiments were implemented on a Tesla V100 GPU. The English and Chinese models were tested three times, and the average value was reported.

4.4. Evaluation Metrics

Based on previous studies to evaluate the keyphrase generation quality [8,36], we use precision, recall, and F1 score to evaluate the prediction accuracy of the proposed method. We calculate these indicators based on the top k generated results, which are defined as follows:

$$P@k = \frac{\left[\hat{Y}_{:k} \cap Y\right]}{\left|\hat{Y}_{:k}\right|} \tag{10}$$

$$R@k = \frac{\left[\hat{Y}_{:k} \cap Y\right]}{\left|Y_k\right|} \tag{11}$$

$$F_1@k = \frac{2 * P@k * R@k}{P@k + R@k} \tag{12}$$

where k is a pre-defined constant (usually five or 10), P is precision defined as the ratio of correct keyphrase prediction and all keyphrase prediction, $\hat{Y}_{:k}$ is the ground truth of the keyphrase set, and Y is the set of keyphrases predicted by the model. $P@k$ is the precision of top k. $R@k$ is the recall of top k, which is defined as the ratio of correct keyphrases prediction and keyphrase ground truth. F_1 is the F_1 score, which indicates the harmonic mean of the precision rate and recall rate, and is often smaller than one.

Based on the above definition, we choose five, 10, and M for constant k as the cutoffs of the keyphrase number, where M is the number of the keyphrase prediction. They are the standard selections for most of the existing keyphrase generation studies. In the experiment

section, we report the F1@5, F1@10, and F1@ M to compare the performance of our model with other representative previous models.

To explore the order of the keyphrase generated by the model, we adapt the MAP (mean average precision) metric for evaluation. MAP refers to the mean average precision (AP) of all texts in a dataset. It considers the order of keyphrase sequence prediction. Before calculating MAP, it is necessary to calculate AP first, which refers to the average precision of the k-th keyphrase prediction of a text. The AP's equation is as follows:

$$AP = \frac{\sum_k^n P@k \times rel@k}{n} \tag{13}$$

where n is the number of keyphrases ground truth of text I, $P@k$ is the same as the above text. $rel@k$ is an indicator function that equals 1 if the predicting phrase belongs to ground truth and equals 0 otherwise. Then, the MAP quation of a dataset is as follows:

$$MAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{14}$$

where AP_i refers to the AP of the i-th text, and N is the number of texts in the dataset. We take $MAP@M$ as the result of the evaluation order.

5. Results and Analysis

5.1. Present Keyphrase Prediction

We report the prediction results of our models compared to existing keyphrase generation methods on various datasets, including the five benchmark datasets described in Section 4.1. We use F1@5 and F1@ M scores to assess the models' performance of present keyphrase prediction. Table 2 shows details of these results.

Table 2. The F1 scores, including F1@5 and F1@M rediction results for present keyphrases of proposed model and compared models mentioned in 4.2 over five abstract-keyphrase datasets. The best results of each dataset are bold.

Model	Inspec		Krapivin		NUS		SemEval		KP20k	
	F1@5	F1@M	F1@5	F1@M	F1@5	F1@ M	F1@5	F1@M	F1@5	F1@M
CopyRNN [36]	**29.2**	25.4	**30.2**	27.6	34.2	30.9	29.1	24.8	32.8	26.1
CatSeq [8]	22.5	25.7	26.9	34.6	32.3	37.9	24.2	27.6	29.1	35.7
CatSeqTG [39]	22.9	27.0	28.2	35.7	32.5	38.2	24.4	28.6	29.1	35.9
Transformer-based [37]	23.7	28.3	27.4	36.6	**38.2**	39.6	28.7	29.0	32.2	37.6
TAtrans (ours)	28.8	**31.7**	28.4	**37.8**	35.1	**45.0**	**33.8**	**30.1**	**35.4**	**39.2**

An obvious conclusion is that the Transformer-based model has a significant improvement over all RNN-based models (CopyRNN [36], CatSeq [8], CatSeqTG [39]), which confirms the Transformer architecture's generative model is more effective than other baseline models on keyphrase generation tasks. After adding the title attention and sequential sequence code, the results on these five English datasets have further improved compared with the above. Table 2 illustrates that the proposed model has a significant improvement on most of the datasets. For instance, in the Inspec dataset, our model improves F1@ M by 3.4% over the Transformer-based model, and 6.3% over CopyRNN. After adding title attention, all indicators of the model are higher than the Transformer-based model, which shows that the title attention mechanism combined with sequence code embedding is an effective approach to improve keyphrase prediction.

In contrast to F1@5 and F1@M of our model, TAtrans performs better on longer keyphrase sequence generation. This result may be attributed to the utilization of the ONE2SEQ training mechanism. From training data, the model could pick up additional keyphrase sequence information.

5.2. Absent Phrases Prediction

In this section, we evaluate the performance of the model in predicting absent keyphrases. As in the previous section, we use F1@5 and F1@M to evaluate the performance of our method and baseline methods in predicting absent keyphrases on different datasets. Table 3 reports the details of these results.

Table 3. The absent keyphrase results of five compared keyphrase generation models on multiple datasets (we round to one decimal place). The best results are bold.

Method	Kp20K		Inspec		Krapivin		NUS		SemEval	
	F1@5	**F1@M**	**F1@5**	**F1@M**	**F1@5**	**F1@M**	**F1@5**	**F1@M**	**F1@5**	**F1@M**
CopyRNN [36]	1.5	3.2	0.4	0.8	1.8	3.6	1.6	2.8	**1.6**	**2.8**
CatSeq [8]	1.5	3.2	0.4	0.8	1.8	3.6	1.6	2.8	**1.6**	**2.8**
CatSeqTG [39]	1.5	3.2	0.5	1.1	1.8	3.4	1.8	3.6	1.1	1.8
Transformer-based [37]	2.2	4.6	1.1	1.9	3.5	**6.3**	2.6	4.4	1.4	1.8
TAtrans (ours)	**3.4**	**4.5**	1.4	**2.6**	**4.6**	6.2	**3.9**	**5.3**	**1.6**	2.6

As the result above show, our model is also the best at predicting absent keyphrases. Our method outperforms the Transformer-based method by 1.2 times in terms of the top five absent phrases prediction, which indicates that our model can understand the core meaning of the text more accurately and generate absent keyphrases more efficiently.

5.3. Prediction on Chinese Dataset

To verify our model cross-language compacity, we implemented our model on the Chinese dataset CNKIBL and compared it with the Transformer-based method. We use F1@5, F1@10, and F1@ M to evaluate our model performance on the Chinese dataset. Before training, we utilized Jieba as the tokenizer to segment the Chinese text. Then the rest of the process is the same as in English. Table 4 displays the overall results of generating present keyphrases on the CNKIBL dataset:

Table 4. Overall results of F1@5, F1@10, and F1@ M on the CNKIBL dataset. The best results are bold.

Metric	**Transformer-Based [37]**	**TAtrans (Ours)**
F1@5	33.5	35.7
F1@10	34.9	36.6
F1@M	36.1	**37.5**

From the Table 4, it can be concluded that our method has shown improvements in the F1@5, F1@10, and F1@M metrics compared to the baseline Transformer method. Our method achieves a 1.4% increase in F1@M score over the Transformer-based method. Moreover, the results of F1@5 of the model outperform those of the baseline model on the Inspec and SemEval English datasets.

As seen in Figure 4, our method has improved results on both the English and Chinese datasets. This confirms that the model can complete cross-lingual keyphrase prediction tasks effectively.

5.4. Prediction Order Analysis

In addition to measuring the present and absent keyphrase prediction separately, we utilize MAP as an additional metric to evaluate the overall quality of the keyphrase sequences generated by models. MAP takes the order information of the complete keyphrase sequence into account while assessing the accuracy of the k-th keyphrase prediction. The main reason is because the higher the correct keyphrase position generated, the greater the AP value according to Equation (13). The suggested model's comparative results with CopyRNN [36] and the Transformer-based method on five datasets are presented in Table 5.

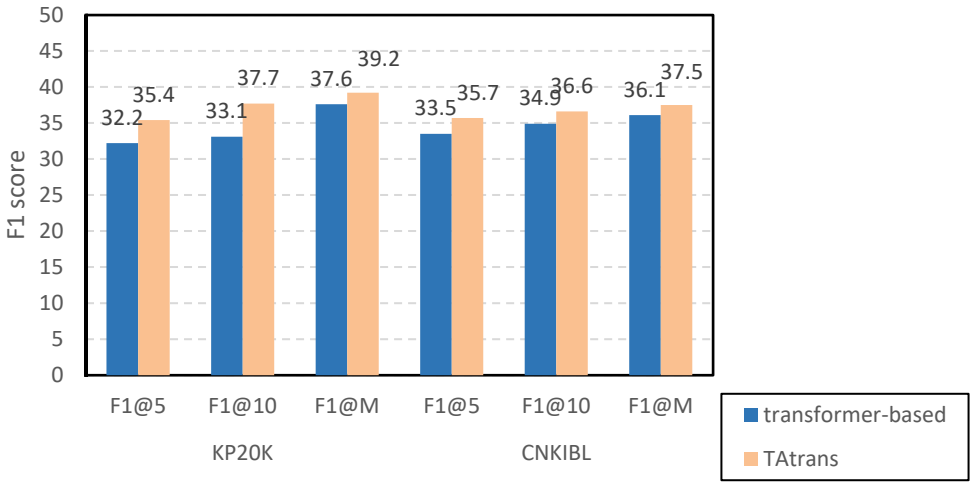

Figure 4. The results of F1@5, F1@10, and F1@M of Transformer-based and TAtrans on KP20K and CNKIBL.

Table 5. The MAP results comparison of CopyRNN, Transformer-based [37], and TAtrans for keyphrase generation. The best results of each dataset are bold.

Model	MAP				
	Inspec	**Krapivin**	**NUS**	**SemEval**	**KP20k**
CopyRNN [36]	0.17	0.22	0.21	0.13	0.27
Transformer-based [37]	0.2	0.33	0.34	0.2	0.35
TAtrans (ours)	**0.33**	**0.64**	**0.47**	**0.32**	**0.42**

As shown in Table 5, the model we proposed performs the best over all five datasets. This suggests that, in compared to other models, the keyphrases generated by our model have a significant sequential advantage. The results of the Transformer-based model over all five datasets were also better than those of CopyRNN, indicating that self-attention has improved the model's keyphrase recognition capabilities. Compared to the Transformer-based model [37], our model has improved by at least 0.07 (on KP20K) and at most 0.29 (on Krapivin). This result proves that adding keyphrase sequence code embedding for each phrase before inputting the decoder can accelerate generating the right keyphrase. On the other hand, Krapivin's data quality is better than KP20K, since it is calibrated by viewers, whereas KP20K has many errors in both the abstract and target keyphrase. Therefore, it further implies that the training data used to feed the model has an essential impact on its prediction performance.

5.5. Ablation Study

We conducted an ablation study experiment to validate the effectiveness of the main components of the TAtrans model. The results of the ablation study experiment performance on both present keyphrase generation and absent keyphrase generation on the KP20K dataset are reported in Table 6.

As shown in Table 6, after removing the title attention component and only retaining the maintain parts of the model (the w/o Title Attention (our model), the present keyphrase generation F1@M scores decrease more than 1.5%, and the absent keyphrase generation performance also becomes obviously worse. We next investigated the effect of removing sequence code embedding from the proposed model architecture. According to Table 6, it can be observed that the F1@5 and F1@M decrease more than 1.8% on both present and absent keyphrase generation, which demonstrates the effectiveness of the sequence code embedding in generated keyphrases.

Table 6. The results (%) of the ablation study of the TAtrans model (proposed in this paper) compared with two baselines on the KP20k dataset. w/o means eliminating a component from the model.

Model	Present		Absent	
	F1@5	**F1@M**	**F1@5**	**F1@M**
CatSeq [8]	29.1	35.7	1.6	2.8
Transformer-based [37]	32.2	37.6	2.2	4.6
w/o Title Attention (our model)	34.1	37.8	3.32	4.4
w/o Sequence Code Embedding (our model)	33.7	37.3	3.22	4.31
TAtrans (our model)	**35.4**	**39.2**	**3.4**	4.5

Further investigation revealed that when comparing the removed component's model with the baseline model (Transformer-based [37] and CatSeq [8]), retaining title attention or sequence code embedding alone is also superior to the Transformer-based model, suggesting that each of these two components has a distinct improvement effect on the keyphrase generation. The two components combined further improve the model's capacity to produce keyphrases significantly.

5.6. Case Study

5.6.1. Title Attention Analysis

In order to verify the effectiveness of title attention in predicting keyphrases, we selected a section of text from the KP20k dataset titled "*A Prime Dual Approximation Algorithm for the Asymmetric Prime Collecting TSP*" and fed it into the Transformer-based model and the TAtrans model proposed in this article for prediction. The Transformer-based prediction keyphrase is "*primal algorithm*", and this article can accurately predict the '*primal-dual algorithm*'. We visualized the attention of the encoder in the Transformer-based model and the TAtrans model proposed in this article, but the results are shown in Figure 5.

According to Figure 5, it can be observed that the text "*present a primary manual log(n) approximation algorithm for the Asian prize collecting*" overlooks the significance of the word '*dual*' in the initial model, despite its importance in the title "*A primary dual approximation algorithm for the Asian prize collecting TSP*". However, after the title attention output, the attention weight of '*dual*' increased, enabling the model to capture the phrase '*primary-dual*'. This highlights the effectiveness of title attention in adjusting the model's attention weight for keyphrases, ultimately enhancing the accuracy of the model's predictions.

5.6.2. Error Analysis

We further analyze the effectiveness of the keyphrase sequence produced by TAtrans. An example prediction is illustrated in Figure 6. In comparison, the baseline Transformer-based model repeatedly generated the word "design" when generating the keyphrase sequence for an abstract text titled "*Blotto Game Based Low Complexity Fair Multiuser Subcarrier Allocation for Uplink Odma Networks*". However, the proposed model, TAtrans, successfully generated distinct phrases. This occurrence can be attributed to autoregressive models, which rely on previous tokens to predict the next one, often resulting in the repetition of generating tokens after encountering the same terms. To address this issue and ensure the distinction between delimiter tokens, we incorporate additional embedding features for each delimiter sign. This approach highlights the effectiveness of keyphrase sequence code embedding in diversifying the generated phrases.

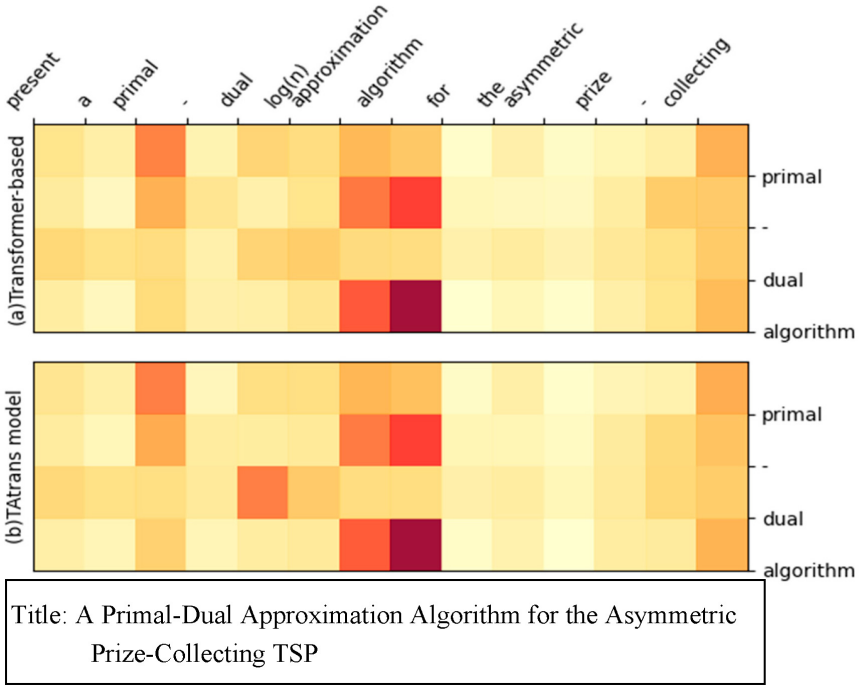

Figure 5. The text titled "*A primary dual approximation algorithm for the Asymmetric Prime Collecting TSP*" visualizes the attention of the keyphrase "*primary-dual algorithm*". (**a**) Attention visualization of Transformer-based. (**b**) Attention visualization of TAtrans proposed by us.

Tittle: Blotto Game Based Low Complexity Fair Multiuser Subcarrier Allocation For Uplink Ofdma Networks.
Abstract: This article presents a subcarrier allocation scheme based on a blotto game (sabg) for orthogonal frequency division multiple access (ofdma) networks where correlation between adjacent subcarriers is considered . in the proposed game , users simultaneously compete for subcarriers using a limited budget . in order to win as many good subcarriers as possible in this game , users are required to wisely allocate their budget . efficient power and budget allocation strategies are derived for users for obtaining optimal throughput . by manipulating the total budget available for each user , competitive fairness can be enforced for the sabg . in addition , the conditions to ensure the existence and uniqueness of nash equilibrium (ne) for the sabg are also established . an low complexity algorithm that ensures convergence to ne is proposed . simulation results show that the proposed low complexity sabg can allocate resources fairly and efficiently for both uncorrelated and correlated fading channels .
Keyphrase Sequence:
Gold: design; distributed behavior; subcarrier allocation**Transformer-based:** design; design; design; distributed behavior; behavior
TAtrans(ours): design; distributed behavior; subcarrier allocation;

Figure 6. A keyphrase prediction example by Transformer-based [37] and TAtrans model. The gold is the keyphrase ground truth. The red font displays significant errors generated by Transformer-based model. The blue font displays that our model correctly predicts the following keyphrases.

6. Conclusions

This paper proposed a Transformer-based framework keyphrase generation model incorporating title attention and adding a sequence order embedding in the decoder. We conducted experiments on multiple datasets, including English and Chinese. The results show that our method outperforms the existing seq2seq model and further improves upon

the Transformer-based approach. The additional keyphrase sequence code embeddings help the model produce more diversified phrases and achieve the best MAP result when considering the order information of the keyphrase sequence comparison with the CopyRNN and Transformer-based model. The main reason is that this method enables the model to distinguish delimiters embeddings between phrases so that it prevents generating duplicate words. To evaluate the cross-language capability of our model, we collected a Chinese dataset named CNKIBL, which contains scientific and technological literature.

The proposed method has made various improvements compared to previous methods in generating public datasets with five keyphrases datasets. In the Inspect dataset, the TAtrans model improves F1@M by 3.4% over the Transformer-based model, and 6.3% over CopyRNN [36]. In addition, we use the MAP metric to consider the quality of the model in generating keyphrase sequences. The experimental results show that the TAtrans model outperformed both CopyRNN [36] and Transformer-based models. In the Krapivin dataset, the proposed model improved by up to 0.29 compared to Transformer-based models. This result indicates that the TAtrans model has the ability to capture the order information of a specific keyphrase sequence. The experimental results on CNKIBL demonstrate that our model's performance also outperforms recently proposed generation models in Chinese datasets. Notably, its performance on Chinses literature text exceeds the transformer-based model by 2.2%, highlighting its cross-language keyphrase generation ability.

However, the proposed method still has certain weakness, especially when we incorporate the enhanced information in keyphrase generation instead of transforming in-depth the architecture of the transformer-based model. This may make the model unable to adapt to other types of NLP tasks easily. Moreover, our model's performance in absent keyphrase generation has not significantly improved. This limitation may be attributed to the inference procedure and the requirement for more effective decoding strategies. In future work, it is necessary to investigate diverse approaches for distinguishing delimiters between phrases in the keyphrase sequence.

Author Contributions: Conceptualization, L.S. and X.L.; Data curation, L.S.; Formal analysis, L.S.; Investigation, L.S.; Methodology, L.S.; Project administration, L.S. and X.L.; Resources, L.S.; Software, L.S.; Supervision, X.L.; Validation, L.S.; Visualization, L.S.; Writing—original draft, L.S.; Writing—review & editing, L.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: In this study, we use five public English dataset provided by the https://huggingface.co website, which can be searched in this website (accessed on 24 April 2023). The code implementation of our project and the Chinese dataset used in this paper were accessed at https://github.com/sherry-robot/TAtrans, accessed on 24 April 2023.

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

References

1. Wang, L.; Cardie, C. Domain-Independent Abstract Generation for Focused Meeting Summarization. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; pp. 1395–1405.
2. Çano, E.; Bojar, O. Keyphrase Generation: A Text Summarization Struggle. *arXiv* **2019**, arXiv:1904.00110.
3. Swaminathan, A.; Zhang, H.; Mahata, D.; Gosangi, R.; Shah, R.R.; Stent, A. A Preliminary Exploration of GANs for Keyphrase Generation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Online, 16–20 November 2020; pp. 8021–8030.
4. Zhai, C.; Lafferty, J. A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. In *ACM SIGIR Forum*; ACM: New York, NY, USA, 2017; Volume 51, pp. 268–276. [CrossRef]
5. Zhai, C. Fast Statistical Parsing of Noun Phrases for Document Indexing. *arXiv* **1997**, arXiv:cmp-lg/9702009v1.
6. Lu, K.; Kipp, M.E.I. Understanding the retrieval effectiveness of collaborative tags and author keywords in different retrieval environments: An experimental study on medical collections. *J. Assoc. Inf. Sci. Technol.* **2014**, *65*, 483–500. [CrossRef]
7. Gutwin, C.; Paynter, G.; Witten, I.; Nevill-Manning, C.; Frank, E. Improving browsing in digital libraries with keyphrase indexes. *Decis. Support Syst.* **1999**, *27*, 81–104. [CrossRef]

8. Yuan, X.; Wang, T.; Meng, R.; Thaker, K.; Brusilovsky, P.; He, D.; Trischler, A. One Size Does Not Fit All: Generating and Evaluating Variable Number of Keyphrases. *arXiv* **2018**, arXiv:1810.05241.
9. Diao, S.; Song, Y.; Zhang, T.J.A. Keyphrase Generation with Cross-Document Attention. *arXiv* **2020**, arXiv:2004.09800.
10. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I.J.A.E.-P. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762.
11. Brin, S.; Page, L. The anatomy of a large-scale hypertextual Web search engine. *Comput. Netw. ISDN Syst.* **1998**, *30*, 107–117. [CrossRef]
12. Khandelwal, U.; Clark, K.; Jurafsky, D.; Kaiser, L.J.A. Sample Efficient Text Summarization Using a Single Pre-Trained Transformer. *arXiv* **2019**, arXiv:1905.08836.
13. Liu, Y.; Lapata, M.J.A.E.-P. Hierarchical Transformers for Multi-Document Summarization. *arXiv* **2019**, arXiv:1905.13164.
14. Glazkova, A.; Morozov, D.J.A.E.-P. Applying Transformer-based Text Summarization for Keyphrase Generation. *arXiv* **2022**, arXiv:2209.03791. [CrossRef]
15. Gu, J.; Lu, Z.; Li, H.; Li, V.O.K. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. *arXiv* **2016**, arXiv:1603.06393.
16. Xie, B.; Song, J.; Shao, L.; Wu, S.; Wei, X.; Yang, B.; Lin, H.; Xie, J.; Su, J. From statistical methods to deep learning, automatic keyphrase prediction: A survey. *Inf. Process. Manag.* **2023**, *60*, 103382. [CrossRef]
17. Barker, K.; Cornacchia, N. Using Noun Phrase Heads to Extract Document Keyphrases. In Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence, Quebec, QC, Canada, 14–17 May 2000; pp. 40–52.
18. Witten, I.H.; Paynter, G.W.; Frank, E.; Gutwin, C.; Nevill-Manning, C.G. KEA: Practical automatic keyphrase extraction. In Proceedings of the Fourth ACM Conference on Digital Libraries, Berkeley, CA, USA, 11–14 August 1999; pp. 254–255.
19. Frank, E.; Paynter, G.W.; Witten, I.H.; Gutwin, C.; Nevill-Manning, C.G. Domain-Specific Keyphrase Extraction. In Proceedings of the International Joint Conference on Artificial Intelligence, Bremen, Germany, 31 October–5 November 2005.
20. Won, M.; Martins, B.; Raimundo, F. Automatic Extraction of Relevant Keyphrases for the Study of Issue Competition. In Proceedings of the Computational Linguistics and Intelligent Text Processing: 20th International Conference, CICLing 2019, La Rochelle, France, 7–13 April 2019; pp. 648–669.
21. 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.
22. Bougouin, A.; Boudin, F.; Daille, B. TopicRank: Graph-Based Topic Ranking for Keyphrase Extraction. In Proceedings of the International Joint Conference on Natural Language Processing, Nagoya, Japan, 14–18 October 2013; pp. 543–551.
23. Herings, P.J.-J.; van der Laan, G.; Talman, D.J.M.T.E. Measuring the Power of Nodes in Digraphs. 2001. Available online: https://papers.tinbergen.nl/01096.pdf (accessed on 25 June 2023).
24. Joshi, R.; Balachandran, V.; Saldanha, E.; Glenski, M.; Volkova, S.; Tsvetkov, Y.J.A.E.-P. Unsupervised Keyphrase Extraction via Interpretable Neural Networks. *arXiv* **2022**, arXiv:2203.07640.
25. Hulth, A. Improved automatic keyword extraction given more linguistic knowledge. In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, 11 July 2003; pp. 216–223. Available online: https://aclanthology.org/W03-1028/ (accessed on 25 June 2023).
26. Jiang, X.; Hu, Y.; Li, H. A Ranking Approach to Keyphrase Extraction. In Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Boston, MA, USA, 19–23 July 2009; pp. 756–757. [CrossRef]
27. Campos, R.; Mangaravite, V.; Pasquali, A.; Jorge, A.M.; Nunes, C.; Jatowt, A. YAKE! Collection-Independent Automatic Keyword Extractor. In Proceedings of the European Conference on Information Retrieval, Grenoble, France, 26–29 March 2018.
28. Pagliardini, M.; Gupta, P.; Jaggi, M. Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features. *arXiv* **2017**, arXiv:1703.02507.
29. Bennani-Smires, K.; Musat, C.; Hossmann, A.; Baeriswyl, M.; Jaggi, M. Simple Unsupervised Keyphrase Extraction using Sentence Embeddings. *arXiv* **2018**, arXiv:1801.04470.
30. Chen, J.; Zhang, X.; Wu, Y.; Yan, Z.; Li, Z. Keyphrase Generation with Correlation Constraints. *arXiv* **2018**, arXiv:1808.07185.
31. Sun, Z.; Tang, J.; Du, P.; Deng, Z.H.; Nie, J.Y. DivGraphPointer: A Graph Pointer Network for Extracting Diverse Keyphrases. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019.
32. Kipf, T.N.; Welling, M.J.A.E.-P. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907.
33. Wang, Y.; Liu, Q.; Qin, C.; Xu, T.; Wang, Y.; Chen, E.; Xiong, H. Exploiting Topic-Based Adversarial Neural Network for Cross-Domain Keyphrase Extraction. In Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM), Singapore, 17–20 November 2018; pp. 597–606.
34. Zhang, Y.; Zhang, C. Using Human Attention to Extract Keyphrase from Microblog Post. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July 28–2 August 2019; pp. 5867–5872.
35. Ding, H.; Luo, X. AttentionRank: Unsupervised Keyphrase Extraction using Self and Cross Attentions. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online and Punta Cana, Dominican Republic, 7–11 November 2021; pp. 1919–1928.
36. Meng, R.; Zhao, S.; Han, S.; He, D.; Brusilovsky, P.; Chi, Y. Deep Keyphrase Generation. *arXiv* **2017**, arXiv:1704.06879.

37. Ye, J.; Gui, T.; Luo, Y.; Xu, Y.; Zhang, Q.J.A.E.-P. One2Set: Generating Diverse Keyphrases as a Set. *arXiv* **2021**, arXiv:2105.11134.
38. Wang, Y.; Li, J.; Chan, H.P.; King, I.; Lyu, M.R.; Shi, S. Topic-Aware Neural Keyphrase Generation for Social Media Language. *arXiv* **2019**, arXiv:1906.03889, 2516–2526.
39. Chen, W.; Gao, Y.; Zhang, J.; King, I.; Lyu, M.R. Title-guided encoding for keyphrase generation. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 6268–6275.
40. Zhang, N.; Chen, X.; Xie, X.; Deng, S.; Tan, C.; Chen, M.; Huang, F.; Si, L.; Chen, H.J.A.E.-P. Document-level Relation Extraction as Semantic Segmentation. *arXiv* **2021**, arXiv:2106.03618.
41. See, A.; Liu, P.J.; Manning, C.D.J.A.E.-P. Get To The Point: Summarization with Pointer-Generator Networks. *arXiv* **2017**, arXiv:1704.04368.
42. Kuhn, H.W.J.N.R.L. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *52*, 83–97. [CrossRef]
43. Krapivin, M.; Marchese, M. Large Dataset for Keyphrase Extraction. 2009. Available online: https://huggingface.co/datasets/midas/krapivin (accessed on 25 June 2023).
44. Nguyen, T.D.; Kan, M.-Y. Keyphrase Extraction in Scientific Publications. In Proceedings of the International Conference on Asian Digital Libraries. In Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers: 10th International Conference on Asian Digital Libraries, ICADL 2007, Hanoi, Vietnam, 10–13 December 2007; Proceedings 10. Springer: Berlin/Heidelberg, Germany, 2007; pp. 317–326.
45. Kim, S.N.; Medelyan, O.; Kan, M.-Y.; Baldwin, T. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. In Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, 15–16 July 2010; pp. 21–26.
46. Tian, X. Extracting Key-phrases from Chinese Scholarly Papers. *Data Anal. Knowl. Discov.* **2020**, *4*, 76–86. [CrossRef]