Improving Low-Resource Keyphrase Generation through Unsupervised Title Phrase Generation

Byungha Kang, Youhyun Shin*
Department of Computer Science and Engineering, Incheon National University
{bhkang, yhshin}@inu.ac.kr

Abstract

This paper introduces a novel approach called title phrase generation (TPG) for unsupervised keyphrase generation (UKG), leveraging a pseudo label generated from a document title. Previous UKG method extracts all phrases from a corpus to build a phrase bank, then draws candidate absent keyphrases related to a document from the phrase bank to generate a pseudo label. However, we observed that when separating the document title from the document body, a significant number of phrases absent from the document body are included in the title. Based on this observation, we propose an effective method for generating pseudo labels using phrases mined from the document title. We initially train BART using these pseudo labels (TPG) and then perform supervised fine-tuning on a small amount of human-annotated data, which we term low-resource fine-tuning (LRFT). Experimental results on five benchmark datasets demonstrate that our method outperforms existing low-resource keyphrase generation approaches even with fewer labeled data, showing strength in generating absent keyphrases. Moreover, our model trained solely with TPG, without any labeled data, surpasses previous UKG method, highlighting the effectiveness of utilizing titles over a phrase bank. The code is available at https://github.com/kangnlp/low-resource-kpgen-through-TPG.

Keywords: keyphrase generation, unsupervised keyphrase generation, low-resource keyphrase generation, semi-supervised keyphrase generation

1. Introduction

Keyphrase generation (KG) is the task of generating a set of keyphrases representing the core content of a document. Unlike keyphrase extraction (KE), which can only extract *present* keyphrases that appear directly in a document, KG has the advantage of also generating *absent* keyphrases that do not appear in a document. The generated keyphrases can be used in text summarization (Zhang et al., 2004; Li et al., 2020), recommendation systems (Bai et al., 2018). In particular, absent keyphrases, which offer semantic expansion through synonyms and related terms, play a crucial role in enhancing retrieval performance by creating document index terms (Boudin et al., 2020).

KG can be broadly categorized into supervised and unsupervised methods. The supervised approach primarily leverages large datasets, such as the KP20k (Meng et al., 2017) with human-annotated data, and exhibits robust performance (Chen et al., 2020; Ye et al., 2021a,b; Xie et al., 2022). Recently, an unsupervised method that trains models with pseudo-labeled data in an environment without human-annotated data was proposed (Shen et al., 2022). Although candidate present keyphrases can be easily acquired through previously proposed unsupervised KE methods, obtaining candidate absent keyphrases remains challenging. Shen et al. (2022) extract noun phrases from all documents in the KP20k to build a *phrase bank*. They then retrieve candidate absent keyphrases related to a document from phrase bank to construct a pseudo label (also referred to as "silver" label).

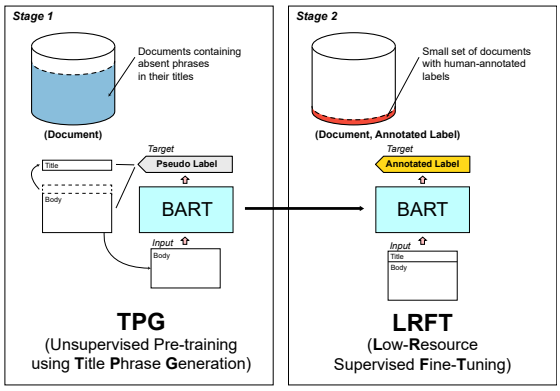

Figure 1: Overview of the proposed two-stage approach for low-resource keyphrase generation.

However, both the supervised and unsupervised methods have certain drawbacks. Supervised approaches, which depend on large datasets, cannot be applied to scenarios with limited or expensive annotated data. The unsupervised method also requires a large number of documents to build a phrase bank, which is essential for obtaining candidate absent keyphrases. To address the limitations of both methods, a *'low-resource'* semi-supervised approach has been proposed (Wu et al., 2022a), that involves unsupervised pre-training for representation learning and fine-tuning with a small set

*Corresponding author.

LREC-COLING 2024, pages 8853–8865
20-25 May, 2024. © 2024 ELRA Language Resource Association: CC BY-NC 4.0

Title: Stiffness analysis of parallelogram-type parallel manipulators using a strain energy method

Body: Stiffness analysis of a general PTPM using an algebraic method. Result comparison between the proposed method and a finite element analysis method. A new stiffness index relating the stiffness property to the wrench experienced in a task.

Gold Keyphrases: stiffness analysis ; parallelogram-type parallel manipulator ; strain energy method ; algebraic method ; stiffness index

Body-Present Keyphrases: stiffness analysis ; algebraic method ; stiffness index
Body-Absent Keyphrases: parallelogram-type parallel manipulator ; strain energy method

Figure 2: An example from KP20k dataset. Keyphrases present in the document body are colored blue, and absent ones are colored red. Two keyphrases absent in the document body are included in the title.

of labeled data.

In this study, we introduce a two-stage semi-supervised keyphrase generation method, as depicted in Figure 1. We propose an unsupervised pre-training method TPG to enhance absent keyphrase generation performance by utilizing each document's title. As will be discussed in more detail in Section 2, when the title and document body are treated separately instead of being concatenated, it is demonstrated that titles contain a wealth of absent phrases. Large datasets across various domains, such as KP20k (scientific papers), KPTimes (news), and StackExchange (questions), show that on average, 74% of the documents contain at least one absent phrase in their title.

We mine the noun phrases absent in the document body from the title and use them as candidate absent keyphrases. Additionally, phrases extracted from the document body are utilized as candidate present keyphrases, with some being replaced in the document body with <mask> tokens to serve as candidate absent keyphrases. Using candidate keyphrases, we construct a pseudo label for each document, then train the BART with these pseudo labels and subsequently fine-tune it with 5,000 (5k) and 20,000 (20k) labeled data samples, respectively. Our experiments on five scientific benchmark datasets demonstrate that our proposed method outperforms existing models in low-resource keyphrase generation settings. Thus, we experimentally demonstrate the effectiveness of TPG, which leverages phrases mined from each document's title, in enhancing keyphrase generation performance in low-resource scenarios.

The main contributions of this study are:

- Our analysis of large datasets such as KP20k, KPTimes, and StackExchange reveals that, on average, 74% of the documents contain at least one phrase in the title that is absent from the document body.
- Based on this observation, we propose TPG, a novel UKG approach that trains BART with a pseudo label, where phrases mined from the document's title serve as the primary candidate absent keyphrases.
- We experimentally demonstrate that TPG significantly contributes to the enhancement of absent keyphrase generation performance with fewer labeled data than existing low-resource keyphrase generation models.

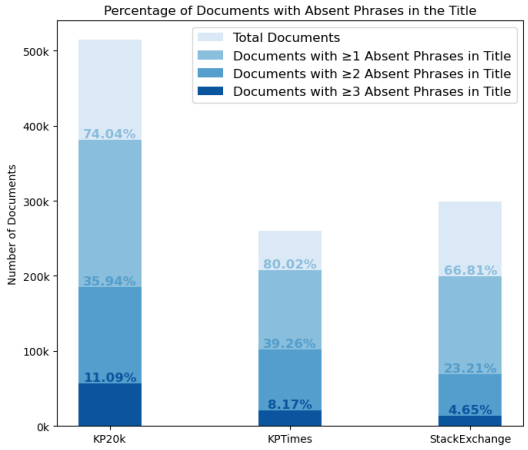

Figure 3: Percentage of documents that contain absent phrases in their titles in each of the three different types of datasets.

2. Absent Phrases in Titles

The title captures the essence of the document in concise sentences. Because writers intend to convey the main content effectively through the title to grab the readers' attention, the title can be seen as a brief summary of the document. Therefore, instead of treating a title as equivalent to the document body by concatenating it, as in previous studies, it may be more beneficial for keyphrase generation (KG) performance to utilize titles differently.

In this context, Ye and Wang (2018) demonstrated an enhancement in KG performance by introducing the auxiliary task of generating a document's title from its body. However, rather than generating the title in its entirety, introducing a task that generates a pseudo keyphrases set from phrases extracted from the title may be more effective. This is because of its closer alignment with the ultimate objective task compared with simply generating the title. By treating titles as separate metadata, certain phrases present in the title may become absent phrases because they do not appear in the document body. This implies that titles can be leveraged as a resource to mine for candidate absent keyphrases. As shown in Figure 2, three

Dataset	KP20k	KPTimes	StackExchange
Type	Paper (Abstract)	News	Question
Total Docs	514,154	259,923	298,965
Docs (*Abs* in Title ≥ 1)	380,676	207,984	199,736
Docs (*Abs* in Title ≥ 2)	184,792	102,047	69,403
Docs (*Abs* in Title ≥ 3)	57,009	21,227	13,895
Docs (*Abs* in Title ≥ 4)	13,923	2,034	2,258
Docs (*Abs* in Title ≥ 5)	3,252	108	347
Avg. number of *Pres* in Title	1.8	1.3	1.5
Avg. number of *Abs* in Title	1.3	1.3	1.0

Table 1: Statistics on phrases in the titles for each of the three different types of datasets (train set).

correct keyphrases are contained in the title, of which '*parallelogram-type parallel manipulator*' and '*strain energy method*' are absent from the document body.

To recognize the potential of titles as resources for mining candidate absent keyphrases, we analyzed phrases contained in the titles of various large datasets of different types, including the KP20k of scientific papers (Meng et al., 2017), the KPTimes of news articles (Gallina et al., 2019), and the StackExchange of questions (Yuan et al., 2020). After separating the document body and titles, a morphological analysis of each word in the title was performed using the tokenizer provided by NLTK[1], and the noun phrases were extracted. Following the method suggested by Meng et al. (2017), we tokenize the extracted noun phrases and the document body, and then stem each word using the NLTK's PorterStemmer to classify phrases as present or absent.

As observed in Figure 3, approximately 74% of all document titles contained at least one absent phrase[2]. Approximately 33% of the document titles incorporated two or more absent phrases, and approximately 8% of the titles had three or more absent phrases. More detailed information on phrases from the dataset-specific titles can be found in Table 1.

These analytical results highlight the capability of titles to serve as a valuable resource for mining not just high-quality candidate present keyphrases but also candidate absent keyphrases. This perspective can be particularly helpful in scenarios lacking annotated data. Most of the domains contained readily available titles. By simply separating the title from the document body, one can extract candidate present keyphrases as well as candidate absent keyphrases to create pseudo labels.

Therefore, we propose a straightforward unsupervised pre-training method called TPG, which utilizes pseudo labels. During the TPG phase, the model input does not include the title. Instead, we extract phrases that are present and absent in the document body from the title and consider these phrases as the highest-ranked candidate keyphrases. Next, we extract phrases from the document body. Some of these are augmented as candidate present keyphrase, whereas others are augmented as candidate absent keyphrase by replacing them with <mask> tokens in the document body. The pre-trained language model, BART is then trained on these pseudo labels. After the TPG pre-training, the model is further fine-tuned on a smaller set of labeled data. More details are provided in the subsequent sections.

[1] https://www.nltk.org/

[2] Here, an 'absent phrase' refers a phrase that is not found within the document body but is included in the title.

3. Methodology

The proposed method consists of a two-stage process: 1) unsupervised pre-training using titles, termed as **TPG**, and 2) low-resource supervised fine-tuning, referred to as **LRFT**. In this study, we utilize BART (Lewis et al., 2020), a Transformer encoder-decoder based pre-trained language model. We train the model using the One2Seq paradigm, in which the present and absent keyphrases is generated as a single ranked sequence (Yuan et al., 2020). The learning objective is to produce a sequence set of keyphrases, termed the target text Y, which is ranked by importance when given an input text $X = x_1, \ldots, x_{|x|}$ (where $|x|$ represents the total number of tokens in the input document). Y is composed of a sequence of present keyphrases $Y^P = [y^p_1, \ldots, y^p_{|Y_P|}]$ (where $|Y_P|$ is the number of present keyphrases) and absent keyphrases $Y^A = [y^a_1, \ldots, y^a_{|Y_A|}]$ (where $|Y_A|$ is the number of absent keyphrases). In the **TPG**, X does not include the title, and Y is a pseudo label constructed from phrases extracted from the title and X. In **LRFT**, X includes the title, and Y utilizes the annotated keyphrases from the dataset.

3.1. Unsupervised Pre-training

As shown in Table 1, 380,676 documents in the KP20k training set contain at least one absent phrase in their titles. To ensure the inclusion of at least one absent phrase in the pseudo labels, we conduct unsupervised pre-training using these 380,676 documents.

3.1.1. Mining Phrases from Title

We tokenize the title and perform POS tagging for each word using NLTK. Noun phrases are defined using the regular expression $NP = < NN.*|JJ > * < NN.* >$ and extracted using NLTK's RegexpParser. Then, we tokenize the extracted noun phrases and the document and stem each word using the NLTK's PorterStemmer to classify them as present or absent phrases. Phrases

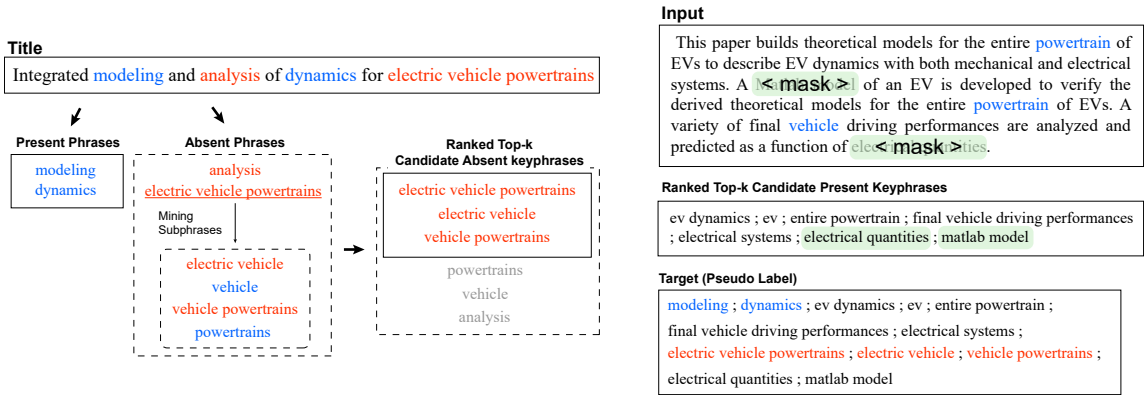

Figure 4: (Left) Process of mining phrases from the title. (Right) Masking the input document and constructing the pseudo label.

mined from the title are defined as Y^P_{title} for present phrases and Y^A_{title} for absent phrases.

According to our observations, the Y^A_{title} contains only the longest noun phrases within the title. However, many subphrases within the extracted noun phrases are absent from the document body. As shown in Figure 4, the initially extracted absent phrase "electric vehicle powertrains" includes absent phrases "electric vehicle" and "vehicle powertrains". To provide more signals for the model to generate phrases that do not appear in the input, we augment the Y^A_{title} by adding the absent subphrases of each phrase in Y^A_{title}. However, this could lead to an excessive increase in Y^A_{title}. Therefore, we rank Y^A_{title} based on the embedding similarity score $Score(X, p)$ between the document's embedding $E(X)$ and the embedding $E(p)$ of the phrases in Y^A_{title}. Only the top-10 phrases that are absent are included in the Y^A_{title}.

$$Score(X, p) = E(X) \cdot E(p)$$

For the embeddings, we use SBERT[3] and employ the dot-product for the similarity score.

3.1.2. Constructing Pseudo Label

We not only employ both present and absent phrases extracted from the title, but also use phrases extracted from the document body to construct a pseudo label.

To extract candidate keyphrases from the document body, we extract noun phrases using the same method as for the title, but do not mine subphrases. Initially, all phrases extracted from the document body are considered candidate present keyphrases and defined as Y^P_{doc}. We rank Y^P_{doc} in the same way as when ranking Y^A_{title}, using the similarity score between the embedding of the entire document body $E(X)$ and the embedding $E(p)$ of a phrase in Y^P_{doc}. Only the top-10 phrases are included in Y^P_{doc}.

Next, the phrases ranked from 6th to 10th in Y^P_{doc} are replaced with the <mask> token for training. The phrases masked in the document body become candidate absent keyphrases, thus we can define $Y^A_{doc} = Y^P_{doc}[5:10]$. The Y^P_{doc} is also redefined as $Y^P_{doc} = Y^P_{doc}[:5]$.

Consequently, we now have a total of four phrases lists: $Y^P_{title}, Y^A_{title}, Y^P_{doc}, Y^A_{doc}$. We regard phrases extracted from the title as high-level phrases with a higher probability of being keyphrases than those extracted from the document body. Therefore, they are placed before the phrases extracted from the document body. Therefore, the final pseudo label is constructed as $Y = < Y^P_{title}, Y^P_{doc}, Y^A_{title}, Y^A_{doc} >$. We refer to this strategy of placing the present keyphrases at the front and the absent keyphrases at the end as Pres-Abs. Figure 4 illustrates the process of constructing a pseudo label.

3.1.3. Training with Masked Document

In this study, we utilize BART (Lewis et al., 2020), a Transformer encoder-decoder based pre-trained language model. BART is pre-trained through 'text-infilling' by sampling spans from the document following a poisson distribution with a mean length of $\lambda = 3$ and then replacing these spans with <mask> tokens for subsequent restoration. The pretrained BART model has an inherent propensity to restore appropriate text at positions where the <mask> tokens are present in the input document. Therefore, we can expect tasks such as replacing the present phrase in the document with a <mask> token and having a model to generate masked phrases or restoring the original text to be beneficial for the absent keyphrase generation task.

Wu et al. (2022a) demonstrated that pre-training by masking salient spans in a document and

[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

then restoring or generating the masked portions could enhance the performance of low-resource keyphrase generation. Inspired by this, we employ a masking technique to augment the candidate absent keyphrases in Y. Because phrases are generally extracted from a document body in numbers exceeding 10, we mask the phrases ranked below the top-5 but within the top-10 in the ranked candidate present keyphrases with `<mask>` tokens wherever they appear in the document body. These replaced phrases are utilized as candidate absent keyphrases. Finally, we train the BART using the masked document body as the input text and the pseudo label as the target text.

3.2. Supervised Fine-tuning

When trained using the TPG approach, the model can generate absent keyphrases on its own, without any supervised fine-tuning. As discussed in Section 4.5.3, the TPG-trained BART outperforms the recently proposed UKG model. However, the performance of UKG models in keyphrase generation can be poor in actual applications.

A practical solution to this challenge is to perform supervised fine-tuning using small, cost-effective, curated labeled data. Considering the challenges of limited resources, we randomly sampled 5,000 and 20,000 documents from the KP20k dataset for supervised fine-tuning. In this phase, the title is concatenated with the document body as part of the input X, and Y is composed in the Pres-Abs order using the annotated keyphrases, just as we organized the pseudo labels during the TPG.

3.3. Beam Search Decoding

We employ the beam search for decoding. Beam search enables the generation of more diverse keyphrases. Beam search in KG differs from its application in typical natural language generation (NLG) tasks. In conventional NLG tasks, only the top beam's output is considered, whereas in KG, all specified beam outputs are considered (Meng et al., 2021). Following Shen et al. (2022), we set the beam size to 20.

4. Experiments

4.1. Datasets

Unsupervised Pre-training Dataset. For the unsupervised pre-training, TPG, we construct masked document-pseudo label pairs, as mentioned in Section 3.1, using 380,676 documents from the KP20k dataset that contain one or more absent phrases in the title. Of these, 350,000 documents are used for the training set, and the remaining 30,676 are used for the validation set.

Dataset	KP20k	Inspec	Krapivin	NUS	SemEval
Docs	20,000	500	400	211	100
Avg. Doc Length	178	129	183	219	235
Avg. keys	5.28	9.83	5.85	11.65	14.66
Avg. Present-Keys	3.32	7.23	3.25	6.34	6.25
Avg. Absent-Keys	1.96	2.59	2.59	5.31	8.41
Avg. Key Length	2.04	2.48	2.21	2.22	2.38

Table 2: Statistics of the test datasets.

Low-resource Fine-tuning Dataset. To compare the performances based on the degree of limited resources, we create two separate training sets by randomly sampling 5,000 and 20,000 documents from the KP20k dataset. In both cases, 2,000 randomly sampled documents are used as the validation set.

Testing dataset. For evaluation, we employ the test dataset of KP20k as well as four scientific datasets: Inspec (Hulth, 2003), Krapivin (Krapivin and Marchese, 2009), NUS (Nguyen and Kan, 2007), and SemEval2010 (Kim et al., 2010). The statistics of each dataset used for evaluation is detailed in Table 2.

Following Wu et al. (2022a), we preprocess all datasets by lowering the text and replacing integers with `[digit]`. During the unsupervised pre-training phase (TPG), only the body text are used as the input text, excluding the title. In the low-resource fine-tuning (LRFT) stage and during inference, both the title and body text are concatenated using `[sep]` to form the input text.

4.2. Baselines

In this study, to verify the effectiveness in a low-resource setting, we compare the performances of models that use only a small amount of labeled data for training.

One-stage methods. The one-stage method trains directly using a small labeled dataset without any separate unsupervised pre-training. We set **ExHiRD-h** (Chen et al., 2021), **One2Set** (Ye et al., 2021b), and **Transformer** with weights initialized randomly without pre-training as our baselines. These models were trained using only 20,000 labeled documents, D_{kp}, from the KP20k training set.

Two-stage methods. Wu et al. (2022a) demonstrated that representations trained by either masking (M) or deleting (D) salient spans (SS) from the document, and subsequently predicting (P) SS or restoring (R) the noisy document, are beneficial for the keyphrase generation task. They constructed the D_{aux} from the KP20k training set, excluding the 20,000 examples used for D_{kp}. The model was pre-trained on D_{aux} using the proposed masking/deletion and prediction/restoration method, and then fine-tuned on the D_{kp}.

Model	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
ExHiRD-h (20k)	0.35	0.57	0.26	0.41	0.65	0.98	0.46	0.57	0.43	0.56
One2Set (20k)	0.54	0.98	0.10	0.15	0.71	1.32	0.69	1.01	0.66	0.94
Transformer (20k)	1.16	1.90	0.48	0.71	1.30	1.86	1.50	2.02	1.17	1.44
BART (20k)	0.93	1.87	0.89	1.58	1.37	2.52	1.06	1.70	0.87	1.24
BART+SSP-M (20k)	1.39	2.78	0.93	1.70	2.24	4.34	1.77	2.92	1.66	2.31
BART+SSP-D (20k)	1.35	2.73	0.91	1.63	2.19	4.06	1.86	2.79	1.28	1.78
BART+SSR-M (20k)	1.95	3.42	1.04	1.73	2.41	3.87	2.16	3.12	1.85	2.39
BART+SSR-D (20k)	1.95	3.76	1.22	2.07	2.55	4.63	3.11	5.31	2.15	2.89
BART+TPG (5k)	2.55	3.79	2.29	3.46	3.16	4.40	3.80	4.89	3.02	3.46
BART+TPG (20k)	**2.97**	**4.33**	**2.77**	**3.74**	**3.83**	**5.24**	**4.71**	**6.53**	**3.08**	**3.56**

Table 3: Results on *Absent* keyphrase generation. 5k and 20k denote fine-tuning on 5,000 and 20,000 human-annotated data, respectively. The best performance is **bold** and the second-best is underlined.

Model	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
ExHiRD-h (20k)	24.01	29.92	22.41	25.21	22.83	29.32	28.26	33.75	22.23	26.71
One2Set (20k)	15.76	23.84	10.46	14.21	15.23	23.24	20.61	28.22	15.11	20.48
Transformer (20k)	11.06	18.04	6.63	9.91	10.05	17.12	14.51	20.72	8.77	12.13
BART (20k)	26.97	31.54	28.54	33.93	26.62	31.12	33.88	33.08	26.33	30.12
BART+SSP-M (20k)	28.04	32.30	27.39	32.25	27.51	33.59	34.35	39.21	24.49	27.72
BART+SSP-D (20k)	28.29	32.63	27.29	32.84	27.46	32.49	33.44	38.05	26.04	29.47
BART+SSR-M (20k)	25.83	33.00	22.57	28.09	23.18	30.01	31.13	36.86	22.60	27.28
BART+SSR-D (20k)	28.82	**35.43**	24.35	30.17	27.08	**34.30**	34.34	**40.49**	23.69	29.04
BART+TPG (5k)	29.70	28.58	**38.48**	**41.33**	**30.42**	29.46	34.87	36.15	**28.76**	**31.24**
BART+TPG (20k)	**30.49**	29.02	34.44	39.06	29.40	28.66	**35.82**	36.78	28.68	30.69

Table 4: Results on *Present* keyphrase generation. 5k and 20k denote fine-tuning on 5,000 and 20,000 human-annotated data, respectively. The best performance is **bold** and the second-best is underlined.

- **BART+SSP-M** & **BART+SSP-D**: Pre-training involves either masking (M) or deleting (D) salient spans (SS) and then predicting the SS. Subsequently, fine-tuning is performed on D_{kp}.
- **BART+SSR-M** & **BART+SSR-D**: Pre-training involves either masking (M) or deleting (D) salient spans (SS), and then restoring the original document. Subsequently, fine-tuning is performed on D_{kp}.

The experimental results of all the baselines adopted in this study refer to the performance reported by Wu et al. (2022a).

4.3. Evaluation

Following Wu et al. (2022a), we use the F1@5 and F1@M (Yuan et al., 2020) metrics to evaluate the prediction performance for both present and absent keyphrases. F1@5 computes the F1 score by considering only the top 5 predicted keyphrases in comparison with the ground truth, whereas F1@M assesses the F1 score based on the entire set of keyphrases predicted by the model. We report the macro-average F1@5 and F1@M performance scores. We conduct experiments three times on randomly sampled 5k and 20k labeled datasets, and report their average performance.

4.4. Implementation Details

We use 'bart-base'[4] as a base model. We add `[digit]` and `[sep]` as special tokens and train the model.

For the TPG pre-training phase, we employ a batch size of 32, a learning rate of 2e-4, and a warm-up ratio of 0.1. The model is trained for 10 epochs, saving a checkpoint after each epoch. From these, we select the checkpoint with the lowest validation loss for the LRFT phase. For the LRFT, we use a batch size of 16, a learning rate of 1e-5, and a warm-up ratio of 0.1, and training for 10 epochs. We save checkpoints after each epoch and pick the model with the lowest validation loss for evaluation. All experiments were conducted using a single NVIDIA A100-PCIE-40GB GPU.

4.5. Results

4.5.1. Main Results

Table 3 shows the results of the *Absent* Keyphrase generation experiments after pre-training with TPG and then fine-tuning with a small labeled dataset. The performance of the proposed model surpasses that of existing models. Remarkably, BART+TPG

[4] https://huggingface.co/facebook/bart-base

(5k) fine-tuned with 5,000 labeled data points achieves higher performance than the baselines fine-tuned with the 20,000 labeled data points. BART+TPG (20k), fine-tuned with the same 20,000 labeled data as the baselines, achieves state-of-the-art (SOTA) performance across all benchmark datasets. This demonstrates that the TPG stage effectively aligns the model, enhancing its capability to generate absent keyphrases more effectively.

A noteworthy observation is that our proposed TPG approach, even though it is pretrained on fewer documents than the baselines, achieves superior performance. BART+SSP-M & BART+SSP-D and BART+SSR-M & BART+SSR-D are pretrained on nearly all documents in the KP20k training set (about 500,000), excluding 20,000. However, our TPG approach involves the pre-training of only around 350,000 documents that contain at least one absent phrase in their titles. This indicates that our TPG approach requires less data, time, and cost for pre-training, yet demonstrates a higher performance, even when fine-tuned with a smaller amount of labeled data. This emphasizes that our TPG approach is more efficient than previous methods, providing substantial improvements in absent KG performance in low-resource settings. Furthermore, it proves effective even in extremely low-resource situations, with as few as 5,000 labeled data points available.

Table 4 shows the *Present* keyphrase generation performance in a low-resource context. The BART trained with TPG consistently achieves state-of-the-art (SOTA) performance in F1@5 across all datasets. Remarkably, the BART+TPG (5k), even when trained with just 5,000 data points, exceeds the performance of existing models fine-tuned with 20,000 data points across all datasets. In some cases, it outperforms BART+TPG (20k) fine-tuned with 20,000 data points. This highlights that TPG itself ensures robust keyphrase generation performance and provides effective representations in highly resource-constrained settings. However, in terms of F1@M, our models do not always outperform the two-stage baselines on the KP20k, Krapivin, and NUS datasets. This may be owing to the limitations of the unsupervised keyphrase extraction (UKE) method used to extract candidate present keyphrases. Nevertheless, by exploring more advanced UKE techniques, there is a promising potential to further enhance the performance in the present keyphrase generation, pointing to a direction for future work.

4.5.2. Title Generation vs. Title Phrase Generation

We conduct a comparative experiment to determine the effectiveness of two pre-training tasks for low-resource KG: title generation (TG), which involves generating the title itself, and title phrase generation (TPG), which involves generating a pseudo label constructed from mined phrases within the title. TG is similar to that proposed by Ye and Wang (2018), but it differs in that we use only 350,000 documents from KP20k that contain at least one absent phrase in the title for training. For a fair comparison with TPG, we first train solely with TG before proceeding to fine-tuning. After pre-training with both TG and TPG, we fine-tune using 5,000 labeled data points to compare the keyphrase generation performance.

Model	Present		Absent	
	F1@5	F@M	F1@5	F@M
BART	28.15	30.03	1.65	1.89
BART+TG	30.86	33.03	2.55	3.09
BART+TPG	**32.45**	**33.35**	**2.96**	**4.00**

Table 5: Average performance across 5 datasets: BART fine-tuned on 5k samples without pre-training, with TG pre-training, and with TPG pre-training.

Table 5 presents the average performance across the five benchmark datasets based on different pre-training methods, and it is evident that TG also contributes to low-resource keyphrase generation. However, TPG achieves a higher average performance for both the present and absent keyphrase generations. This suggests that the task of generating a pseudo label based on the title is more suitable in extremely resource-limited situations than generating the entire title itself.

4.5.3. TPG in Zero-Shot Settings: Phrase Bank vs. Title

The previous UKG method, AutoKeyGen (Shen et al., 2022) utilizes a phrase bank built by extracting noun phrases from all documents in the KP20k as a resource to obtain candidate absent keyphrases. However, we have demonstrated that the title can serve as a valuable resource for deriving candidate absent keyphrases. To compare the two strategies for acquiring candidate absent keyphrases, we evaluate the keyphrase prediction performance of two unsupervised models, AutoKeyGen and TPG, both trained solely with pseudo labels and without any fine-tuning the annotated data.

We omit the subphrase mining process introduced in Section 3.1.1, when TPG is used solely in an unsupervised scenario, because while subphrases extracted from the title's absent phrases can provide useful signals to learn useful representations in the intermediary steps of KG, they can lead the model to learn to generate redundant phrases in a fully unsupervised scenario.

We explore whether training on documents with more absent phrases in their titles improves absent keyphrase generation, even if it reduces the

Model	KP20k			Inspec			Krapivin			NUS			SemEval		
	@5	@10	@$\mathcal{O}$	@5	@10	@$\mathcal{O}$	@5	@10	@$\mathcal{O}$	@5	@10	@$\mathcal{O}$	@5	@10	@$\mathcal{O}$
AutoKeyGen	**23.4**	**24.6**	**23.8**	30.3	34.5	33.1	17.1	15.5	15.8	21.8	23.3	23.7	18.7	24.0	22.7
TPG (*Abs* ≥ 1)	21.6	17.4	16.1	38.9	**41.7**	**40.8**	**22.6**	**20.7**	**19.6**	**26.6**	**25.9**	24.8	**27.3**	**26.4**	**26.2**
TPG (*Abs* ≥ 2)	20.2	16.4	16.7	**39.8**	39.9	40.3	21.3	18.6	18.9	26.3	24.5	**24.9**	25.3	24.6	24.8
TPG (*Abs* ≥ 3)	17.9	13.7	16.1	39.2	35.2	37.6	20.6	15.6	17.9	23.2	20.2	22.9	24.5	20.4	22.2

Table 6: Present keyphrase generation performance in a zero-shot setting.

Model	KP20k		Inspec		Krapivin		NUS		SemEval	
	R@10	R@20	R@10	R@20	R@10	R@20	R@10	R@20	R@10	R@20
AutoKeyGen	2.3	2.5	1.7	2.1	**3.3**	**5.4**	2.4	3.2	1.0	1.1
TPG (*Abs* ≥ 1)	1.6	1.6	2.1	2.1	1.7	1.7	3.4	3.4	**1.4**	1.4
TPG (*Abs* ≥ 2)	**2.5**	**2.7**	2.8	3.2	2.7	2.9	**3.4**	**3.6**	1.1	**1.5**
TPG (*Abs* ≥ 3)	2.0	2.4	**3.6**	**3.8**	2.4	2.9	1.4	2.7	1.1	1.3

Table 7: Absent keyphrase generation performance in a zero-shot setting.

total number of documents used for training. As analyzed in Table 1, we conduct experiments on TPG trained on documents that contain at least one absent phrase in the title (*Abs* ≥ 1, 380k), on documents that contain at least two absent phrases (*Abs* ≥ 2, 185k), and on documents that contain at least three absent phrases (*Abs* ≥ 3, 57k), respectively.

Following Shen et al. (2022), we use F1@5, F1@10, and F1@$\mathcal{O}$[5] to evaluate the present keyphrase prediction, and R@10 and R@20 to evaluate absent keyphrase prediction as evaluation metrics. In TPG, only the body text, excluding the title, is used as input for the model, so it does not learn about the `[sep]` token. Therefore, for inference, we concatenate the title and body text with a space instead of using the `[sep]` token.

Tables 6 and 7 show the performance of TPG in predicting present and absent keyphrases, respectively, in the zero-shot setting. As shown in Table 6, the performance of TPG in predicting the present keyphrases is proportional to the number of documents used for training. Furthermore, TPG trained on a relatively small number of documents TPG (*Abs* ≥ 3, 57k) outperforms AutoKeyGen trained on the entire KP20k training set (514k).

Regarding absent keyphrase prediction shown in Table 7, TPG trained to generate at least 2 candidate absent keyphrases, even when trained on a smaller number of documents (*Abs* ≥ 2, 185k), demonstrates superior performance compared to TPG trained to produce at least 1 candidate absent keyphrase (*Abs* ≥ 1, 380k) and surpasses the performance of AutoKeyGen on all datasets except Krapivin. On the other hand, a slight performance drop is observed for TPG trained on around 57k documents (*Abs* ≥ 3). The experimental results show that TPG, utilizing only titles without a Phrase Bank, can achieve better keyphrase generation performance much more efficiently compared to AutoKeyGen.

[5] $\mathcal{O}$ denotes the number of ground truth keyphrases.

Model	**Present**		**Absent**	
	F1@5	F@M	F1@5	F@M
BART+TPG (Random)	**34.03**	**35.16**	2.13	3.26
BART+TPG (Abs-Pres)	33.36	34.58	1.52	2.51
BART+TPG (Pres-Abs)	32.45	33.35	**2.96**	**4.00**

Table 8: Average performance across the five benchmark datasets based on different fine-tuning ordering strategies.

4.5.4. Impact of Fine-tuning Ordering Strategy

We compare the impact of keyphrase ordering during fine-tuning with limited labeled data (5k) on the KG performance. We explored three ordering strategies: 'Random', which randomly shuffles a set containing both present and absent keyphrases; 'Abs-Pres', positioning absent keyphrases first, followed by the present ones; and 'Pres-Abs', which does the opposite, listing present keyphrases before absent ones.

As shown in Table 8, Pres-Abs ordering, which aligns with the method used for creating pseudo labels in TPG, showed the best results for generating absent keyphrases. Interestingly, the Abs-Pres order performed better than Pres-Abs in generating present keyphrases. This suggests that the model tends to predict keyphrases positioned towards the latter part more accurately. One possible explanation is that predicting the initial keyphrases is inherently more challenging. Keyphrases positioned later may benefit from the context provided by the keyphrases generated earlier, leading to more accurate predictions. However, Random ordering exhibits the highest performance in predicting present keyphrases, and also outperforms the Abs-Pres ordering in predicting absent keyphrases. This indicates that Random ordering can provide a relatively balanced performance in predicting both present and absent keyphrases.

5. Related Work

Unsupervised Keyphrase Extraction. Unsupervised keyphrase extraction (UKE) methods extract noun phrases from a document and rank them based on various criteria to assign importance scores. Various UKE approaches have been introduced, including statistical methods (Jones, 2004; Campos et al., 2018), graph-based methods (Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Bougouin et al., 2013; Florescu and Caragea, 2017), and embedding-based methods (Bennani-Smires et al., 2018; Sun et al., 2020). Since the advent of transformer-based pre-trained language models (PLM), such as BERT (Devlin et al., 2019), embedding-based UKE approaches utilizing contextual embedding have demonstrated remarkable performance (Ding and Luo, 2021; Liang et al., 2021; Zhang et al., 2022; Song et al., 2023). Recently, methods leveraging the self-attention maps of PLMs (Kang and Shin, 2023) and utilizing designed prompts (Kong et al., 2023) have been proposed.

Supervised Keyphrase Generation. Starting with CopyRNN (Meng et al., 2017), various methods have been proposed to generate both present and absent keyphrases using encoder-decoder generative models. Attention mechanisms (Chen et al., 2018; Zhao and Zhang, 2019), reinforcement learning (Chan et al., 2019), generative adversarial networks (GAN) (Swaminathan et al., 2020), and hierarchical modeling of phrases and words (Chen et al., 2020) have been employed in keyphrase generation. Methods utilizing pretrained seq2seq PLMs, such as BART, have also been proposed (Kulkarni et al., 2022; Zhao et al., 2022). Wu et al. (2022b) analyzed various PLMs in the KG through an empirical study. Ye et al. (2021b) introduced a novel paradigm called One2Set, which generates an unordered set of keyphrases in parallel. Subsequently, Xie et al. (2022) proposed the WR-SetTrans model to refine the calibration errors inherent in the One2Set approach.

Unsupervised Keyphrase Generation. AutoKeyGen (Shen et al., 2022) is the first to approach the keyphrase generation task in an unsupervised manner without relying on any human-annotated data. AutoKeyGen obtains candidate present keyphrases using the UKE method and then retrieves candidate absent keyphrases from a phrase bank that stores all phrases extracted from all documents in a corpus. These candidate keyphrases are used to create a pseudo label for each document, to train a Seq2Seq model.

Semi-supervised Keyphrase Generation. Supervised methods demonstrate commendable performance but have the disadvantage of being heavily reliant on large labeled datasets. Unsupervised methods that do not require labeled data often exhibit inferior performance in absent keyphrase generation. In response to this, semi-supervised approaches have been suggested for pre-training on tasks that do not require annotated data and subsequently fine-tuning the model with a small amount of labeled data. Ye and Wang (2018) proposes a multi-task semi-supervised method that utilizes synthetic keyphrases constructed using the UKE method and generates titles alongside the keyphrases. KPDrop (Ray Chowdhury et al., 2022) replaces some candidates extracted from the recent UKE method (Liang et al., 2021) in documents with the <mask> token and then fine-tunes using 5,000 labeled data samples. Wu et al. (2022a) introduce representation learning by masking salient spans in documents and training on tasks to predict these spans or restore the original document, thereby enhancing low-resource keyphrase generation.

6. Conclusion

In this paper, we introduce TPG (Title Phrase Generation), a straightforward unsupervised pretraining task aimed at improving keyphrase generation performance in low-resource scenarios. Based on the observation that document titles contain phrases not present in the document body, we present a novel perspective that leveraging the titles as a resource for constructing pseudo labels. We first train BART through TPG, where each document body serves as the source text and the pseudo label derived from the document's title serves as the target text. We then fine-tune this model on a small amount of labeled data. Experimental results on five benchmark datasets demonstrate that our model outperforms previous low-resource supervised keyphrase generation (KG) models, notably achieving state-of-the-art results in bsent keyphrase generation. Furthermore, our proposed method surpasses existing approaches even with fewer labeled data, underscoring its effectiveness in extremely resource-constrained scenarios.

7. Acknowledgements

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ICAN(ICT Challenge and Advanced Network of HRD) support program(IITP-2024-RS-2023-00260175) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation).

8. Bibliographical References

2000. The nlm indexing initiative. In *Proceedings of the AMIA Symposium*, page 17. American Medical Informatics Association.

Haoli Bai, Zhuangbin Chen, Michael R. Lyu, Irwin King, and Zenglin Xu. 2018. Neural relational topic models for scientific article analysis. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 27–36, New York, NY, USA. Association for Computing Machinery.

Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. 2018. Simple unsupervised keyphrase extraction using sentence embeddings. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 221–229, Brussels, Belgium. Association for Computational Linguistics.

Florian Boudin, Ygor Gallina, and Akiko Aizawa. 2020. Keyphrase generation for scientific document retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1118–1126, Online. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan. Asian Federation of Natural Language Processing.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. 2018. Yake! collection-independent automatic keyword extractor. In *Advances in Information Retrieval - 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings*, volume 10772 of *Lecture Notes in Computer Science*, pages 806–810. Springer.

Hou Pong Chan, Wang Chen, Lu Wang, and Irwin King. 2019. Neural keyphrase generation via reinforcement learning with adaptive rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2163–2174, Florence, Italy. Association for Computational Linguistics.

Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4057–4066, Brussels, Belgium. Association for Computational Linguistics.

Wang Chen, Hou Pong Chan, Piji Li, and Irwin King. 2020. Exclusive hierarchical decoding for deep keyphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1095–1105, Online. Association for Computational Linguistics.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R. Lyu. 2019. 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*, AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Wang Chen, Piji Li, and Irwin King. 2021. A training-free and reference-free summarization evaluation metric via centrality-weighted relevance and self-referenced redundancy. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 404–414, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Haoran Ding and Xiao Luo. 2021. AttentionRank: Unsupervised keyphrase extraction using self and cross attentions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1919–1928, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lam Do, Pritom Saha Akash, and Kevin Chen-Chuan Chang. 2023. Unsupervised open-domain keyphrase generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10614–10627, Toronto, Canada. Association for Computational Linguistics.

Corina Florescu and Cornelia Caragea. 2017. PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, Vancouver, Canada. Association for Computational Linguistics.

Ygor Gallina, Florian Boudin, and Beatrice Daille. 2019. KPTimes: A large-scale dataset for keyphrase generation on news documents. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 130–135, Tokyo, Japan. Association for Computational Linguistics.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, page 216–223, USA. Association for Computational Linguistics.

Karen Spärck Jones. 2004. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 60(5):493–502.

Byungha Kang and Youhyun Shin. 2023. SAMRank: Unsupervised keyphrase extraction using self-attention map in BERT and GPT-2. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10188–10201, Singapore. Association for Computational Linguistics.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, Uppsala, Sweden. Association for Computational Linguistics.

Aobo Kong, Shiwan Zhao, Hao Chen, Qicheng Li, Yong Qin, Ruiqi Sun, and Xiaoyan Bai. 2023. PromptRank: Unsupervised keyphrase extraction using prompt. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9788–9801, Toronto, Canada. Association for Computational Linguistics.

Mikalai Krapivin and Maurizio Marchese. 2009. Large dataset for keyphrase extraction.

Mayank Kulkarni, Debanjan Mahata, Ravneet Arora, and Rajarshi Bhowmik. 2022. Learning rich representation of keyphrases from text. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 891–906, Seattle, United States. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. 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*, pages 7871–7880, Online. Association for Computational Linguistics.

Haoran Li, Junnan Zhu, Jiajun Zhang, Chengqing Zong, and Xiaodong He. 2020. Keywords-guided abstractive sentence summarization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8196–8203.

Xinnian Liang, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2021. Unsupervised keyphrase extraction by jointly modeling local and global context. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 155–164, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rui Meng, Xingdi Yuan, Tong Wang, Sanqiang Zhao, Adam Trischler, and Daqing He. 2021. An empirical study on neural keyphrase generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4985–5007, Online. Association for Computational Linguistics.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Vancouver, Canada. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Proceedings of the 10th International Conference on Asian Digital Libraries: Looking Back 10 Years and Forging New Frontiers*, ICADL'07, page 317–326, Berlin, Heidelberg. Springer-Verlag.

Jishnu Ray Chowdhury, Seo Yeon Park, Tuhin Kundu, and Cornelia Caragea. 2022. KPDROP: Improving absent keyphrase generation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4853–4870, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Alexander Schutz. 2008. Keyphrase extraction from single documents in the open domain exploiting linguistic and statistical methods.

Xianjie Shen, Yinghan Wang, Rui Meng, and Jingbo Shang. 2022. Unsupervised deep keyphrase generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11303–11311.

Mingyang Song, Huafeng Liu, and Liping Jing. 2023. HyperRank: Hyperbolic ranking model for unsupervised keyphrase extraction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16070–16080, Singapore. Association for Computational Linguistics.

Yi Sun, Hangping Qiu, Yu Zheng, Zhongwei Wang, and Chaoran Zhang. 2020. Sifrank: A new baseline for unsupervised keyphrase extraction based on pre-trained language model. *IEEE Access*, 8:10896–10906.

Avinash Swaminathan, Haimin Zhang, Debanjan Mahata, Rakesh Gosangi, Rajiv Ratn Shah, and Amanda Stent. 2020. A preliminary exploration of GANs for keyphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8021–8030, Online. Association for Computational Linguistics.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, pages 855–860. AAAI Press.

Di Wu, Wasi Ahmad, Sunipa Dev, and Kai-Wei Chang. 2022a. Representation learning for resource-constrained keyphrase generation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 700–716, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Di Wu, Wasi Uddin Ahmad, and Kai-Wei Chang. 2022b. Pre-trained language models for keyphrase generation: A thorough empirical study. *arXiv preprint arXiv:2212.10233*.

Binbin Xie, Xiangpeng Wei, Baosong Yang, Huan Lin, Jun Xie, Xiaoli Wang, Min Zhang, and Jinsong Su. 2022. WR-One2Set: Towards well-calibrated keyphrase generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7283–7293, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hai Ye and Lu Wang. 2018. Semi-supervised learning for neural keyphrase generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4142–4153, Brussels, Belgium. Association for Computational Linguistics.

Jiacheng Ye, Ruijian Cai, Tao Gui, and Qi Zhang. 2021a. Heterogeneous graph neural networks for keyphrase generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2705–2715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021b. One2Set: Generating diverse keyphrases as a set. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4598–4608, Online. Association for Computational Linguistics.

Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. 2020. One size does not fit all: Generating and evaluating variable number of keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7961–7975, Online. Association for Computational Linguistics.

Linhan Zhang, Qian Chen, Wen Wang, Chong Deng, ShiLiang Zhang, Bing Li, Wei Wang, and Xin Cao. 2022. MDERank: A masked document embedding rank approach for unsupervised keyphrase extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 396–409, Dublin, Ireland. Association for Computational Linguistics.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2004. World wide web site summarization. *Web Intelli. and Agent Sys.*, 2(1):39–53.

Yuxiang Zhang, Yaocheng Chang, Xiaoqing Liu, Sujatha Das Gollapalli, Xiaoli Li, and Chunjing Xiao. 2017. Mike: Keyphrase extraction by integrating multidimensional information. In *Proceedings of the 2017 ACM on Conference on*

Information and Knowledge Management, CIKM '17, page 1349–1358, New York, NY, USA. Association for Computing Machinery.

Guangzhen Zhao, Guoshun Yin, Peng Yang, and Yu Yao. 2022. Keyphrase generation via soft and hard semantic corrections. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7757–7768, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jing Zhao and Yuxiang Zhang. 2019. Incorporating linguistic constraints into keyphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5224–5233, Florence, Italy. Association for Computational Linguistics.

Absent keyphrase generation						
Model	PubMed		StackExchange		KPTimes	
	F1@5	F@M	F1@5	F@M	F1@5	F@M
BART (5k)	0.52	0.46	0.92	0.96	0.50	0.54
BART+TPG (5k)	**0.74**	**0.94**	**1.37**	**1.65**	**0.51**	**0.58**
Present keyphrase generation						
Model	PubMed		StackExchange		KPTimes	
	F1@5	F@M	F1@5	F@M	F1@5	F@M
BART (5k)	18.0	18.1	15.8	17.5	**14.9**	**15.7**
BART+TPG (5k)	**22.3**	**20.7**	**15.9**	**18.0**	12.6	13.0

Table 9: Results on absent and present keyphrase generation for the PubMed, StackExchange, and KPTimes datasets

9. Appendix A. Experiments on Additional Datasets

We conduct additional experiments to demonstrate the effectiveness of TPG not only on scientific paper abstracts but also on texts of varying lengths and types. We perform additional experiments on the PubMed (Schutz, 2008), comprising full-text scientific papers in the biomedical domain, the StackExchange (Yuan et al., 2020), which consists of questions, and the KPTimes (Gallina et al., 2019), which consists of news articles. We experiment with two models, BART (5k), which underwent only LRFT, and BART+TPG (5k), which underwent both TPG and LRFT. We apply the same preprocessing as in the main experiments. Since the texts in all three datasets are lengthy, we input only up to 512 tokens to the models for inference.

Table 9 displays the results of the absent and present keyphrase prediction experiments on the three datasets. The results show that across all datasets, BART+TPG (5k) outperforms BART (5k) in the absent keyphrase prediction. Moreover, for the present keyphrase prediction, BART+TPG (5k) also demonstrates improved performance, except on the KPTimes dataset. On the news-based KPTimes dataset, the performance gain of BART+TPG (5k) is marginal or even underperforms BART. This outcome could be attributed to our use of only scientific paper abstracts for TPG. Although this study focuses on scientific paper abstracts, in future work, we expect to build an open-domain keyphrase generation model by incorporating corpora of various types and domains for TPG.