Fast and Constrained Absent Keyphrase Generation by Prompt-Based Learning

Huanqin Wu[1], **Baijiaxin Ma**[2], **Wei Liu**[1], **Tao Chen**[1], **Dan Nie**[1]

[1] Tencent AI Platform Department, China
[2] Peking University
{huanqinwu, thinkweeliu, vitochen, kathynie}@tencent.com, mabaijiaxin@stu.pku.edu.cn

Abstract

Generating absent keyphrases, which do not appear in the input document, is challenging in the keyphrase prediction task. Most previous works treat the problem as an autoregressive sequence-to-sequence generation task, which demonstrates promising results for generating grammatically correct and fluent absent keyphrases. However, such an end-to-end process with a complete data-driven manner is unconstrained, which is prone to generate keyphrases inconsistent with the input document. In addition, the existing autoregressive decoding method makes the generation of keyphrases must be done from left to right, leading to slow speed during inference. In this paper, we propose a constrained absent keyphrase generation method in a prompt-based learning fashion. Specifically, the prompt will be created firstly based on the keywords, which are defined as the overlapping words between absent keyphrase and document. Then, a mask-predict decoder is used to complete the absent keyphrase on the constraint of prompt. Experiments on keyphrase generation benchmarks have demonstrated the effectiveness of our approach. In addition, we evaluate the performance of constrained absent keyphrases generation from an information retrieval perspective. The result shows that our approach can generate more consistent keyphrases, which can improve document retrieval performance. What's more, with a non-autoregressive decoding manner, our model can speed up the absent keyphrase generation by 8.67× compared with the autoregressive method.

Introduction

Keyphrase prediction task aims to obtain a set of keyphrases, which are several phrases that highlight core topics or information of a document. As a basic NLP task, keyphrase prediction is essential for numerous downstream tasks such as information retrieval (Kim et al. 2013), document clustering (Hulth and Megyesi 2006), and summarization (Wang and Cardie 2013; Pasunuru and Bansal 2018).

Keyphrases of a document can be categorized into the *present keyphrase* that appears continuously in the document and *absent keyphrase*, which doesn't appear in the document. Early works mostly focus on the keyphrase extraction (Witten et al. 2005; Nguyen and Kan 2007; Medelyan, Frank, and Witten 2009; Lopez and Romary 2010; Zhang et al. 2016; Alzaidy, Caragea, and Giles 2019; Sun et al. 2020). These methods aim to extract text spans or phrases from the document, which show promising results on present keyphrase prediction. However, such extractive methods cannot handle the absent keyphrase, which is also significant and requires comprehensive understanding of the document.

Title: Interactive visualization of volumetric data with WebGL in real-time
Document: This article presents and discusses the implementation of a direct volume rendering system for the Web, which articulates a large portion of the rendering task in the client machine. By placing the rendering emphasis in the local client, our system takes advantage of its power, while at the same time eliminates processing from unreliable bottlenecks (e.g. network). The system developed articulates in efficient manner the capabilities of the recently released WebGL standard, which makes available the accelerated graphic pipeline (formerly unusable). The dependency on specially customized hardware is eliminated, and yet efficient rendering rates are achieved. The Web increasingly competes against desktop applications in many scenarios, but the graphical demands of some of the applications (e.g. interactive scientific visualization by volume rendering), have impeded their successful settlement in Web scenarios. Performance, scalability, accuracy, security are some of the many challenges that must be solved before visual Web applications popularize. In this publication we discuss both performance and scalability of the volume rendering by WebGL ray casting in two different but challenging application domains: medical imaging and radar meteorology.
Absent keyphrases: real time visualization, weather radar volume
Unconstrained generation result: virtual reality

Figure 1: Example of a document and its expected absent keyphrases. The overlapping words between the input document and absent keyphrases are marked with red. We define such words as keywords in this paper. Our approach treats keywords as constrained signals for generating the absent keyphrase.

To address this issue, several generative methods (Meng et al. 2017; Chen et al. 2018; Ye and Wang 2018; Wang et al. 2019; Chen et al. 2019b; Chan et al. 2019; Zhao and Zhang 2019; Chen et al. 2020; Yuan et al. 2020; Ahmad et al. 2021) have been proposed. Such generative methods mainly adopt the sequence-to-sequence (Seq2Seq) model to predict a target sequence, which is a concatenation of present and absent keyphrases. Therefore, the generative approach can predict both kinds of keyphrases. However, such an end-to-end generation manner is prone to generate some absent keyphrases inconsistent with the source documents. As shown in Figure 1, although "virtual reality" is grammatically correct and fluent, such phrase is irrelevant with the input document.

Intending to mitigate the issue mentioned above, we explore to constrain the generation process for obtaining more consistent absent keyphrase. We are inspired by the observation that some absent keyphrases overlap with the document. As shown in Figure 1, the overlapping words usually point out valuable content in the keyphrase and the document, which can provide significant clues for the generation. For example, the overlapping word "visualization" reflects the theme of the original document, which is also highly related to the absent keyphrase "real time visualization". This phenomenon is common in the keyphrase prediction task. We treat such overlapping words as keywords for the document and the keyphrase. Existing work has not explored the effectiveness of such keywords for keyphrase generation, and our work focuses on applying them as constrained signals for absent keyphrase generation.

Instead of incorporating the constrained signals of keywords with implicit representation in the model, we propose a novel prompt-based learning method, which can achieve constrained generation by explicitly affecting the final output keyphrase. We argue that there are at least two advantages of constrained absent keyphrase generation with a prompt-based learning manner. Firstly, the training and inference process of prompt-based learning is closer to the pre-training process, which can fully utilize the language knowledge in the pre-training model for keyphrase prediction. What's more, we can achieve non-autoregressive inference under the prompt-based fashion, which can significantly speed up the generation process. Specifically, we first design a prompt construction process based on keywords, then a mask-predict method is applied to predict final results based on the constraint of prompt.

Experiments conducted on the widely used public datasets show that our method can outperform mainstream generative models. Moreover, we conduct evaluation of the absent keyphrase from an information retrieval perspective, which further shows that our approach can generate more consistent keyphrases and improve document retrieval performance compared with unconstrained method. We also observe large margin improvement when ground truth keywords are used as constraints in the absent keyphrase generation, which further confirms our assumption . The contributions of this paper can be summarized as follows:

- To the best of our knowledge, it's the first attempt to explore constrained and non-autoregressive generation for absent keyphrase prediction.
- We propose a novel prompt-based learning method for constrained absent keyphrase generation to mitigate the inconsistent generation problem.
- Our approach can speed up the absent keyphrase generation by 8.67 times compared with the autoregressive generation method.
- We evaluate the consistency between generative absent keyphrase and document from an information retrieval perspective.

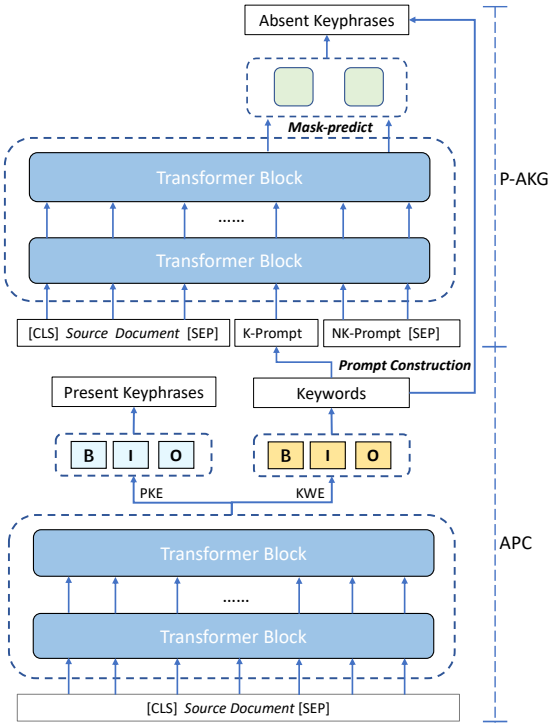

Figure 2: Overview of our approach. The Transformer model used in APC and P-AKG share the same parameters.

Approach

Overview

To realize constrained absent keyphrase generation, we first create prompt, which can provide significant clues and constraints for absent keyphrase generation. The keyword is a necessary prerequisite of automatic prompt construction for constrained generation. We take the overlapping words between the input document and absent keyphrases as the ground-truth keywords in training. To acquire the keywords for testing, we train a **K**ey**W**ords **E**xtractor (KWE) by multi-task learning with the keyphrase prediction task. In addition, we apply a mask-predict method to fill the slot in prompt for obtaining the whole absent keyphrase.

Although we aim to optimize the absent keyphrase generation, our approach can also predict present keyphrases by treating the **P**resent **K**eyphrase **E**xtraction (PKE) as a sequence labeling task. Specifically, we simply add a labeling head on the model to solve it. Figure 2 gives an overview of our approach. The process of our constrained absent keyphrase generation method can be summarized as follows:

- **Automatic Prompt Construction (APC)**. APC aims to obtain several prompts, which contain essential constraints for absent keyphrase generation. We firstly use an extractor to extract keywords, which is a necessary prerequisite for constructing such prompt. Then the prompt will be constructed automatically for the constrained generation.

- **Prompt-based Absent Keyphrase Generation (P-AKG)**. For a given document and corresponding prompt, P-AKG intends to fill the slot in the prompt based on the input document under mask-predict fashion.

Model Architecture

We use a Transformer-based model with prefix LM architecture as the backbone for jointly learning to extract and perform prompt-based generation. The prefix LM (Dong et al. 2019; Raffel et al. 2020) architecture is similar to an encoder-decoder model with parameters shared across the encoder and decoder. It utilizes special self-attention mask to perform conditioned generation. In detail, the prefix LM applies fully-visible masking to build the self-attention of source sequence and causal masking for predicting the target sequence. The hidden state of source and target sequence will be used for extraction and prompt-based constrained generation task.

In this paper, the original document is used as source sequence. Instead of treating absent keyphrase generation as a sequence generation task, we manual design textual prompt with some unfilled slots as target sequence. Then the prefix LM language model is used to parallel fill the slots for completing the whole keyphrase. Source sequence and prompt will be concatenated as final input sequence X with `[SEP]` and then fed into the prefix LM. Specifically, we concatenate X with `[CLS]` and `[SEP]` tokens as the input sequence:

$$I = \{[\text{CLS}]\, X\, [\text{SEP}]\} \tag{1}$$

Afterwards, we feed input sequence I into prefix LM and obtain output hidden state $H = [h_1, \ldots, h_n]$. Such hidden states will be used for extraction and generation task in our approach.

Automatic Prompt Construction

To obtain the prompt automatically, we first perform keyword extraction on the source document and then conduct the prompt based on the keywords. Figure 3 gives an example of the prompt construction process.

Keyword Extraction In this paper, we use the keyword as the prerequisite for constructing the prompt. Public datasets for the keyphrase prediction task do not provide the keywords for the input document. Therefore, as mentioned above, we roughly regard the overlapping words (stop-words are excluded) between the input document and the absent keyphrase as the ground-truth keywords. Then we train a keyword extraction model for constrained absent keyphrase generation.

For keyword extraction task, the output layer is a softmax classifier over the hidden state $H_d = [h_1, \ldots, h_d]$ for each word in the document. The classifier predicts the probability of each the word in the document being a keyword in BIO format:

$$\mathbf{y}_i^k = \text{softmax}(\mathbf{W}^k h_i + \mathbf{b}^k) \tag{2}$$

where W^k, b^k are trainable parameters.

Keyword based Prompt Construction For a given document and corresponding keywords, we aim to construct the prompt for constrained absent keyphrase generation. Specifically, we define two types of prompt for constrained generation in this paper:

- **K-Prompt**: For a keyword kw in the document, we apply "phrase of kw is `[MASK] [MASK]` kw `[MASK] [MASK]`" as prompt. The final absent keyphrase can be obtained by combining the kw and mask-predict results. Such prompt can ensure that the kw appears in the final output keyphrase. We only take the top K keywords for constructing the prompt to reduce noise in keyword extraction.
- **NK-Prompt**: There are also some absent keyphrases without keywords in the document. We denote the prompt as "other phrases are `[MASK] [MASK] [MASK] [MASK]`" for such scenario. The predicted result on each `[MASK]` position will be combined as the absent keyphrase. Such prompt can provide the constraint about generating the keyphrase non-overlapping with input document.

It should be noted that the number of `[MASK]` tokens in K-Prompt and NK-Prompt is hyper-parameters when constructing the prompt. These two type prompts will be concatenated as the final prompt for the generation.

Prompt based Absent Keyphrase Generation

Prompt-based generation treats the keyphrase prediction task as a masked language modeling problem. The model directly generates word on the `[MASK]` position in the prompt. The generator takes hidden state of `[MASK]` position $H_m = [h_{m1}, \ldots, h_{mn}]$ as input and predicts the word of each `[MASK]` position:

$$\mathbf{y}_i^a = \text{softmax}(\mathbf{W}^a h_{mi} + \mathbf{b}^a) \tag{3}$$

where W^a, b^a are trainable parameters.

As shown in Figure 3, if there is no need to predict a token on the `[MASK]` position, the ground truth of such position is set to a special token `[NULL]` during training. We will discard the `[NULL]` token during inference.

Present Keyphrase Extraction

Although our approach aims to optimize the absent keyphrase generation, it can also predict present keyphrases by simply adding a labeling head on the model. Specifically, the hidden state of source document $H_d = [h_1, \ldots, h_d]$ is used as input of present keyphrase extractor. The extractor then predicts the probability of each word being a constituent of a present keyphrase in BIO format:

$$\mathbf{y}_i^p = \text{softmax}(\mathbf{W}^p h_i + \mathbf{b}^p) \tag{4}$$

where W^p, b^p are trainable parameters.

Multi-Task Training

As shown in Figure 4, we apply multi-task training for our approach. To train three tasks at the same time, source document and prompt are connected with `[SEP]` and used as the

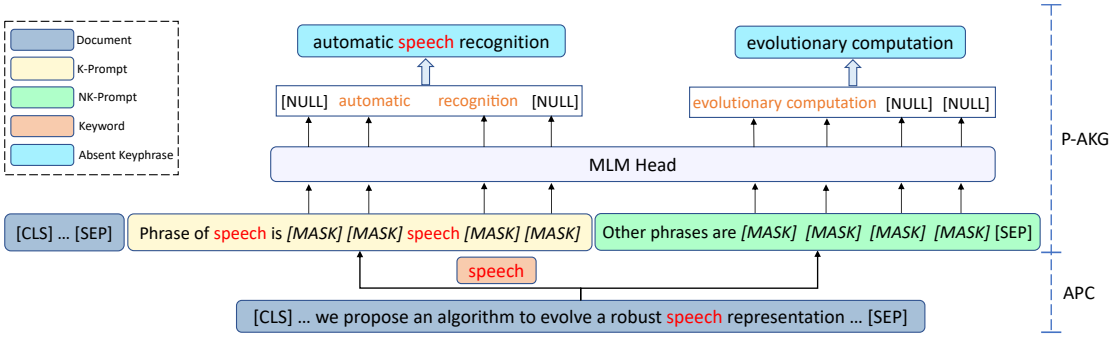

Figure 3: Example of prompt construction and constrained generation process.

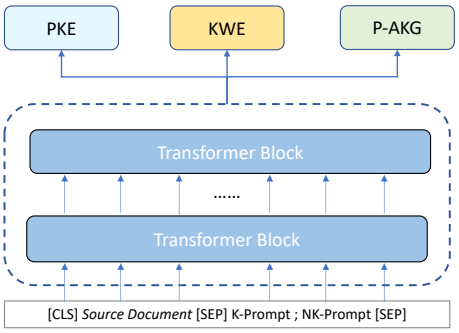

Figure 4: Training process of our approach.

input of prefix LM model, the contextualized representation for the source document is used for the present keyphrase and the keyword extraction, and the hidden state of the `[MASK]` is used to masked language modeling training. The objection of PKE is formulated as:

$$\mathcal{L}_{PKE} = -\frac{1}{M}\sum_{i=1}^{M}\sum_{c=1}^{C} w_p \hat{\mathbf{y}}_i^{(c,p)} \log\left(\mathbf{y}_i^{(c,p)}\right) \quad (5)$$

Similarly, the objection of KWE is formulated as:

$$\mathcal{L}_{KWE} = -\frac{1}{M}\sum_{i=1}^{M}\sum_{c=1}^{C} w_{kw} \hat{\mathbf{y}}_i^{(c,k)} \log\left(\mathbf{y}_i^{(c,k)}\right) \quad (6)$$

where M refers to the length of the document, C refers to the number of the PKE and KWE label. w_p and w_{kw} refer to the loss weight for the PKE and KWE positive label. $\hat{\mathbf{y}}_i^p$ and $\hat{\mathbf{y}}_i^k$ refer to the gold label of present keyphrase and keyword.

For the P-AKG task, training objection is to maximize the likelihood of masked tokens, which is formulated as:

$$\mathcal{L}_{AKG} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{V_s} \hat{\mathbf{y}}_i^{(j,a)} \log\left(\mathbf{y}_i^{(j,a)}\right) \quad (7)$$

where N refers to the number of masked tokens, V_s refers to the size of the vocabulary. $\hat{\mathbf{y}}_i^a$ refers to the ground-truth word.

Finally, the overall loss of our model is formulated as:

$$\mathcal{L} = \lambda_p \mathcal{L}_{PKE} + \lambda_k \mathcal{L}_{KWE} + \lambda_a \mathcal{L}_{AKG} + \delta \quad (8)$$

where λ_p, λ_k, λ_a, and δ are the weights for three tasks, these weights are learned during training, which are used for multi-task loss adjustment proposed by Kendall, Gal, and Cipolla (2018).

Experiments

Dataset

We follow the setup widely used in the keyphrase prediction task, which is training the model on the `KP20K` (Meng et al. 2017) dataset, and giving an evaluation on three more benchmark datasets: `NUS` (Nguyen and Kan 2007), `INSPEC` (Hulth 2003) and `SEMEVAL` (Kim et al. 2010). We use the data from `KP20K` validation set as validation data and apply them to identify optimal checkpoints for testing. We follow the pre-process, post-process setting of Meng et al. (2017, 2019); Yuan et al. (2020)[1]. In addition, we apply `ACM-CR` (Boudin and Gallina 2021)[2] to build retrieval tasks for evaluating the consistency between generative absent keyphrase and document. More details of the dataset are provided in Appendix.

Evaluation Metrics

We follow previous works, which use $F_1@5$ and $F_1@M$ to evaluate the performance of the model. Specifically, we use the partition of present and absent provided by Meng et al. (2017) and calculate $F_1@5$ and $F_1@M$ (use all predicted keyphrases for F_1 calculation) after stemming and removing duplicates. Following previous works, when compute $F1@5$ and the number of prediction keyphrases is less than five, we randomly append incorrect keyphrases until it obtains five predictions, which aim to avoid $F_1@5$ become the same with $F_1@M$ when the prediction number is less than five. As recommended in Färber and Jatowt (2020), we evaluate the performance of retrieval by recall@10 retrieved results for context-aware citation recommendation.

[1]We follow the official GitHub repository to prepare datasets which are available on https://github.com/memray/OpenNMT-kpg-release.

[2]https://github.com/boudinfl/redefining-absent-keyphrases/blob/main/data/acm-cr/acm-cr.v1.tar.gz

Experimental Setup

Setting We reuse most hyper-parameters from the pre-trained prefix LM[3]. The weight of positive label w_p in PKE is set to 5.0. The weight of positive label w_{kw} in KWE is set to 10.0. Follow the pre-trained prefix LM, our model is implemented using PyTorch. The learning rate is 1e-5 and the proportion of warmup steps is 0.1. We set the batch size to 200 and the maximum length to 384. The number of `[MASK]` tokens in each K-Prompt is 4 (two `[MASK]` tokens on each side of the keyword) and the number of `[MASK]` tokens in NK-Prompt is 8. We set the K of the top keywords as 6. We shuffle the order of keywords for constructing K-Prompt to obtain three training samples for each document. We train our model on the training set for 50 epochs. It takes about 50 minutes per epoch to train the model on 4 Nvidia Tesla V100 GPU cards with mixed-precision training.

Baselines We compare various generative models that which can predict absent keyphrases under the Seq2Seq generation framework:

- Pure generative models. CatSeq (Yuan et al. 2020) is a classic setting of the pure generative model, which predicts present and absent keyphrases in a Seq2Seq manner. We report the performance of CatSeq and various improved models on it, including CatSeqCorr (Chen et al. 2018), catSeqTG (Chen et al. 2019b), and CatSeqD (Yuan et al. 2020). ExHiRD-h (Chen et al. 2020), a recently released model, is also included for comparing.
- SEG-Net (Ahmad et al. 2021). A joint model contains a selector that selects the salient sentences in a document and an extractor-generator that jointly extracts and generates keyphrases from the selected sentences.
- UniKeyphrase[4] (Wu et al. 2021). A unified present keyphrase extraction and absent keyphrase generation framework based on the pre-trained prefix LM model.

Result and Analysis

Present and Absent Keyphrase Prediction The present and absent keyphrase prediction performance of all methods are shown in Table 1. Although our approach aims to optimize the absent keyphrase generation, we can also find the performance of present keyphrase prediction of our method is comparable with the strong baseline models. What's more, the performance of absent keyphrase generation of our approach can outperform most generative baseline, which demonstrates the effectiveness of our constrained generation method.

Upper Bound Performance We explore the upper bound performance of our constrained generation model. Specifically, we directly use the ground-truth keyword for the inference of the absent keyphrase. In such setting, the error of keyword extraction is eliminated. The results are shown in Table 1. Although the improvement is impressive, it is based on the assumption that we know the keyword of input document in advance. Actually, obtaining a golden set of keywords may be difficult even for human. We believe the result holds out a promising prospect for the further development of keyphrase prediction task.

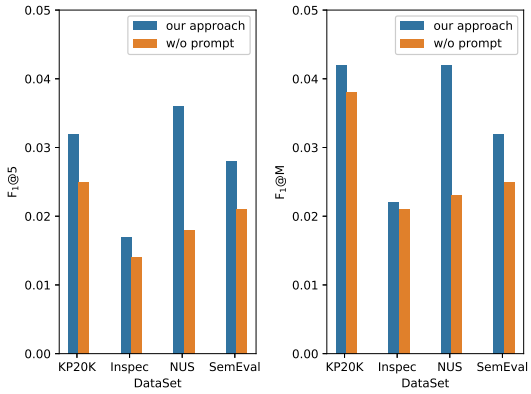

Figure 5: Comparison of the performance for our prompt-based approach and vanilla mask-predict (without prompt) on absent keyphrase generation.

Performance of Retrieval by Indexing Constrained Generative Absent Keyphrase To further verify our approach can generate more consistent absent keyphrase compare with baseline method, we directly apply the absent keyphrase for the downstream information retrieval task. Following Boudin and Gallina (2021), we evaluate the performance of the retrieval task under various indexing configurations, including adding previous unconstrained and our constrained generative absent keyphrase. The unconstrained absent keyphrases are produced by UniKeyphrase, a strong Seq2Seq model for absent keyphrase generation based on the pre-training LM. We use the implementation available in Boudin and Gallina (2021)[5] for building the retrieval system. Table 2 presents the results of retrieval models, which demonstrates that our constrained generation method can produce a more consistent absent keyphrase than the unconstrained method for improving document retrieval performance.

Effectiveness of Prompt To confirm the effectiveness of prompt construction in the constrained generation. We compare the performance for prompt-based generation and vanilla mask-predict (without prompt like "phrase of keyword"). Figure 5 shows the performance of absent keyphrase generation on various datasets. As shown in Figure 5, we can find that our model with the prompt-based generation achieves better performance than those without prompt, proving that the prompt is essential for providing task-specific guidance and constraint for absent keyphrase generation.

[3]We use the official provided pre-trained model, which is available on https://unilm.blob.core.windows.net/ckpt/unilm1-base-cased.bin.

[4]We use the publicly implementation available in https://github.com/thinkwee/UniKeyphrase.

[5]https://github.com/boudinfl/redefining-absent-keyphrases.

Task	Model	Inspec		NUS		SemEval		KP20k	
		F_1@M	F_1@5	F_1@M	F_1@5	F_1@M	F_1@5	F_1@M	F_1@5
Present Keyphrase	CatSeq	0.262	0.225	0.397	0.323	0.283	0.242	0.367	0.291
	CatSeqD	0.263	0.219	0.394	0.321	0.274	0.233	0.363	0.285
	CatSeqCorr	0.269	0.227	0.390	0.319	0.290	0.246	0.365	0.289
	CatSeqTG	0.270	0.229	0.393	0.325	0.290	0.246	0.366	0.292
	ExHiRD-h	0.291	0.253	–	–	0.335	0.284	0.374	0.311
	SEG-Net	0.265	0.216	**0.461**	0.396	0.332	0.283	**0.379**	0.311
	UniKeyphrase	0.288	**0.260**	0.443	**0.415**	0.322	0.302	0.352	0.347
	Our approach	**0.294**	**0.260**	0.439	0.412	**0.356**	**0.329**	0.355	**0.351**
Absent Keyphrase	CatSeq	0.008	0.004	0.028	0.016	0.028	0.020	0.032	0.015
	CatSeqD	0.011	0.006	0.024	0.015	0.024	0.016	0.031	0.015
	CatSeqCorr	0.009	0.005	0.024	0.014	0.026	0.018	0.032	0.015
	CatSeqTG	0.011	0.005	0.018	0.011	0.027	0.019	0.032	0.015
	ExHiRD-h	0.022	0.011	–	–	0.025	0.017	0.032	0.016
	SEG-Net	0.015	0.009	0.036	0.021	0.030	0.021	0.036	0.018
	UniKeyphrase	**0.022**	0.012	0.037	0.026	0.029	0.021	**0.058**	**0.032**
	Our approach	**0.022**	**0.017**	**0.042**	**0.036**	**0.032**	**0.028**	0.042	**0.032**
	Our approach with GK	*0.176*	*0.080*	*0.086*	*0.061*	*0.062*	*0.051*	*0.156*	*0.067*

Table 1: Results of keyphrase prediction on benchmarks. "Our model with GK" means directly using the **G**round-truth **K**eyword for absent keyphrase generation. The bold-faced values indicate the best performances across the board.

Index	BM25	BM25 + RM3
Title & Abstract	35.64	34.09
Title & Abstract & UCG-AK	36.17	33.97
Title & Abstract & CG-AK	**36.59**	**36.02**

Table 2: Retrieval performance (Recall@10) of BM25 and BM25+RM3 using various indexing configurations on the ACM-CR dataset. "UCG-AK" and "CG-AK" mean **U**nconstrained and **C**onstrained **G**enerative **A**bsent **K**eyphrase.

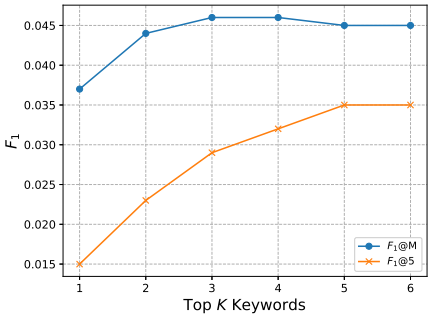

Figure 6: Performance of absent keyphrase generation with different top K keywords when constructing K-Prompt.

Analysis of Top K Keywords In this section, we study the influence of top K Keywords on the absent keyphrase prediction performance. Figure 6 shows the performance of absent keyphrase generation on validation data under different top K keywords. We can find that taking more keywords could achieve better scores on F_1@5 for absent keyphrase generation, which proves the effectiveness of prompt constrained generation. It is worth noting that when the number of keywords is larger than 4, the absent keyphrase performance drops slightly on F_1@M. We suppose that when the number of keywords becomes too much, the error of keyword extraction will affect the performance of absent keyphrase generation.

Model	Params	Time(ms)	Speedup
UniKeyphrase	110M	915.79	/
Proposed model	110M	105.75	8.67×

Table 3: The comparison of the average time about predicting one document with the proposed model and UniKeyphrase, a typical autoregressive generation method.

Inference Speed One of the core contributions of our framework is that the generation process can be significantly accelerated with the proposed prompt-based non-autoregressive mechanism. We evaluate the inference speed of our proposed method with UniKeyphrase, which is an autoregressive generative method similar to our model. For a fair comparison, we use the same device (NVIDIA V100) to evaluate the inference time on the KP20K test set. We set the batch size to 1 both for our method and baseline. As shown in Table 3, our proposed two-stage inference method can transform the keyphrase prediction into a non-autoregressive form and increase the speed of generation effectively (up to 8.67×) compare with the autoregressive model.

Case Study To further demonstrate the effectiveness of the constrained generation method, we give a case of predictions on the KP20k dataset. For fairness, we compare our approach with UniKeyphrase, which is a Seq2Seq generation method based on a pre-trained prefix LM model without constraint. In this case, red are denoted as keyword in the input document. As shown in Figure 7, the unconstrained result generated by UniKeyphrase is "software engineering," which is inconsistent with the input document. In contrast

Title: Automatically generated CSP specification
Document: Two possibilities of automated CSP (Communicating Sequential Processes) support are introduced in [11] and [10] using either behavioral diagrams or application source code. While in the first approach a tool generates CSP specification from behavioral diagrams, based on UML Composite States diagram, in the second approach an application source code is translated directly into CSP specification using a compiler. This paper reviews tools related to both techniques.
Golden absent keyphrases: formal specification, model, grammar
Unconstrained generation result: software engineering
Keyword extraction result: specification
Constrained generation result: formal specification

Figure 7: Case of absent keyphrase generation by our approach and unconstrained generation result. The keyword is marked with red.

to the unconstrained result, our approach first extracts the keyword "specification" from the document, and then generate the final keyphrase "formal specification" constrained by "specification."

Related Work

Keyphrase Extraction and Generation

Keyphrase extraction aims to select phrases in the document directly. Two-step extraction is a typically extractive method. The method firstly identifies a set of candidate phrases from the document by heuristics. Then, the candidate keyphrases are sorted and ranked to get predicted results (Hulth 2003; Nguyen and Kan 2007; Medelyan, Frank, and Witten 2009; Lopez and Romary 2010; Mihalcea and Tarau 2004; Wan and Xiao 2008). Other extractive methods mainly apply sequence labeling model. The documents are fed to an encoder then the model learns to predict the likelihood of each word being a keyphrase (Zhang et al. 2016; Alzaidy, Caragea, and Giles 2019; Sun et al. 2020). However, the extractive model cannot handle the absent keyphrase.

Keyphrase generation addresses the above issue in a sequence-to-sequence generation manner. Meng et al. (2017) first propose CopyRNN, a Seq2Seq framework with attention and copy mechanism. A semi-supervised method is explored by Ye and Wang (2018). Chen et al. (2018) investigate a review mechanism to reduce duplicates. Chen et al. (2019b) focus on leveraging the title information to improve keyphrases generation. The deeper topics of the document are exploited by Wang et al. (2019). Zhao and Zhang (2019) utilize linguistic constraints to prevent the model from generating overlapped phrases. Chan et al. (2019) and Swaminathan et al. (2020) introduce a reinforcement learning approach for keyphrase generation. Chen et al. (2020) propose an exclusive hierarchical decoding framework to generate keyphrases. Yuan et al. (2020) introduce a new model to generate multiple keyphrases as delimiter-separated sequences. Zhao et al. (2021) propose to deal with present and absent keyphrases generation separately with different mechanisms. Huang et al. (2021) present an AdaGM method to increase the discreteness of the keyphrase generation. Ye et al. (2021) propose an one2set method for generating diverse keyphrases as a set. There are also some works (Chen et al. 2019a; Ahmad et al. 2021; Wu et al. 2021) focus on jointly learning extraction and generation for keyphrase prediction. In contrast to these methods, our approach can be non-autoregressive and constrained to generate absent keyphrases.

Prompt based Text Generation.

Recently, prompting methods have been applied to text generation tasks based on pre-trained LMs. Raffel et al. (2019) explored the ability of prompt models to perform generation tasks such as text summarization and machine translation using prompts. Brown et al. (2020) introduce in-context learning for text generation, creating a prompt with manual templates. Schick and Schütze (2020) focus on fixed-prompt LM tuning for few-shot text summarization with manually crafted templates. Li and Liang (2021) investigate fixed-prompt LM tuning for text summarization and data-to-text generation in few-shot settings. Dou et al. (2021) employ the prompt-based LM tuning strategy on the text summarization task. However, the prompt-based keyphrase generation method hasn't been explored.

Conclusion and Future Work

This paper focuses on constrained generating absent keyphrases. In detail, we propose an absent keyphrase generation solution based on a prompt-based learning fashion. Specifically, keywords are first extracted for automatic prompt construction. Then, we use a mask-predict-based approach to generate the final absent keyphrase constrained by prompt. Experiments on keyphrase generation benchmarks and evaluation from an information retrieval perspective have demonstrated the effectiveness of the proposed model. In addition, the non-autoregressive decoding process can speed up the absent keyphrase generation compared with the sequence-to-sequence generation method. The prompt used in this paper is constructed by a manually defined template, while such a process may fail to discover optimal prompts. In the future, we will explore automating the template design process.

References

Ahmad, W.; Bai, X.; Lee, S.; and Chang, K.-W. 2021. Select, Extract and Generate: Neural Keyphrase Generation with Layer-wise Coverage Attention. 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)*, 1389–1404. Association for Computational Linguistics.

Alzaidy, R.; Caragea, C.; and Giles, C. L. 2019. Bi-LSTM-CRF sequence labeling for keyphrase extraction from scholarly documents. In *The world wide web conference*, 2551–2557.

Boudin, F.; and Gallina, Y. 2021. Redefining Absent Keyphrases and their Effect on Retrieval Effectiveness. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4185–4193.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chan, H. P.; Chen, W.; Wang, L.; and King, I. 2019. Neural Keyphrase Generation via Reinforcement Learning with Adaptive Rewards. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2163–2174. Florence, Italy: Association for Computational Linguistics.

Chen, J.; Zhang, X.; Wu, Y.; Yan, Z.; and Li, Z. 2018. Keyphrase Generation with Correlation Constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4057–4066.

Chen, W.; Chan, H. P.; Li, P.; Bing, L.; and King, I. 2019a. An Integrated Approach for Keyphrase Generation via Exploring the Power of Retrieval and Extraction. 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)*, 2846–2856.

Chen, W.; Chan, H. P.; Li, P.; and King, I. 2020. Exclusive Hierarchical Decoding for Deep Keyphrase Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1095–1105.

Chen, W.; Gao, Y.; Zhang, J.; King, I.; and Lyu, M. R. 2019b. Title-Guided Encoding for Keyphrase Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6268–6275.

Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; and Hon, H.-W. 2019. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, 13063–13075.

Dou, Z.-Y.; Liu, P.; Hayashi, H.; Jiang, Z.; and Neubig, G. 2021. GSum: A General Framework for Guided Neural Abstractive Summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4830–4842.

Färber, M.; and Jatowt, A. 2020. Citation recommendation: approaches and datasets. *International Journal on Digital Libraries*, 21(4): 375–405.

Huang, X.; Xu, T.; Jiao, L.; Zu, Y.; and Zhang, Y. 2021. Adaptive Beam Search Decoding for Discrete Keyphrase Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 13082–13089.

Hulth, A. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 216–223.

Hulth, A.; and Megyesi, B. 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 537–544.

Kendall, A.; Gal, Y.; and Cipolla, R. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7482–7491.

Kim, S. N.; Medelyan, O.; Kan, M.-Y.; and Baldwin, T. 2010. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, 21–26.

Kim, Y.; Kim, M.; Cattle, A.; Otmakhova, J.; Park, S.; and Shin, H. 2013. Applying graph-based keyword extraction to document retrieval. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 864–868.

Li, X. L.; and Liang, P. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Lopez, P.; and Romary, L. 2010. HUMB: Automatic key term extraction from scientific articles in GROBID. In *Proceedings of the 5th international workshop on semantic evaluation*, 248–251.

Medelyan, O.; Frank, E.; and Witten, I. H. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, 1318–1327.

Meng, R.; Yuan, X.; Wang, T.; Brusilovsky, P.; Trischler, A.; and He, D. 2019. Does Order Matter? An Empirical Study on Generating Multiple Keyphrases as a Sequence. *arXiv preprint arXiv:1909.03590*.

Meng, R.; Zhao, S.; Han, S.; He, D.; Brusilovsky, P.; and Chi, Y. 2017. Deep Keyphrase Generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 582–592.

Mihalcea, R.; and Tarau, P. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 404–411.

Nguyen, T. D.; and Kan, M.-Y. 2007. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, 317–326. Springer.

Pasunuru, R.; and Bansal, M. 2018. Multi-Reward Reinforced Summarization with Saliency and Entailment. In

Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), 646–653.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140): 1–67.

Schick, T.; and Schütze, H. 2020. Few-shot text generation with pattern-exploiting training. *arXiv preprint arXiv:2012.11926*.

Sun, S.; Xiong, C.; Liu, Z.; Liu, Z.; and Bao, J. 2020. Joint Keyphrase Chunking and Salience Ranking with BERT. *arXiv preprint arXiv:2004.13639*.

Swaminathan, A.; Zhang, H.; Mahata, D.; Gosangi, R.; Shah, R. R.; and Stent, A. 2020. A Preliminary Exploration of GANs for Keyphrase Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 8021–8030. Online: Association for Computational Linguistics.

Wan, X.; and Xiao, J. 2008. Single Document Keyphrase Extraction Using Neighborhood Knowledge. In *AAAI*, volume 8, 855–860.

Wang, L.; and Cardie, C. 2013. Domain-independent abstract generation for focused meeting summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1395–1405.

Wang, Y.; Li, J.; Chan, H. P.; King, I.; Lyu, M. R.; and Shi, S. 2019. Topic-Aware Neural Keyphrase Generation for Social Media Language. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2516–2526.

Witten, I. H.; Paynter, G. W.; Frank, E.; Gutwin, C.; and Nevill-Manning, C. G. 2005. Kea: Practical automated keyphrase extraction. In *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*, 129–152. IGI global.

Wu, H.; Liu, W.; Li, L.; Nie, D.; Chen, T.; Zhang, F.; and Wang, D. 2021. UniKeyphrase: A Unified Extraction and Generation Framework for Keyphrase Prediction. *arXiv preprint arXiv:2106.04847*.

Ye, H.; and Wang, L. 2018. Semi-Supervised Learning for Neural Keyphrase Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4142–4153.

Ye, J.; Gui, T.; Luo, Y.; Xu, Y.; and Zhang, Q. 2021. ONE2SET: Generating Diverse Keyphrases as a Set. *arXiv preprint arXiv:2105.11134*.

Yuan, X.; Wang, T.; Meng, R.; Thaker, K.; Brusilovsky, P.; He, D.; and Trischler, A. 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*, 7961–7975.

Zhang, Q.; Wang, Y.; Gong, Y.; and Huang, X.-J. 2016. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, 836–845.

Zhao, J.; Bao, J.; Wang, Y.; Wu, Y.; He, X.; and Zhou, B. 2021. SGG: Learning to Select, Guide, and Generate for Keyphrase Generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 5717–5726.

Zhao, J.; and Zhang, Y. 2019. Incorporating linguistic constraints into keyphrase generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 5224–5233.