Diversify Keyphrase Generation with Subtopic Content Modeling

Yanyan Ge[1,3](✉), Peng Yang[1,2,3], and Wenjun Li[1,3]

[1] School of Computer Science and Engineering, Southeast University, Nanjing, China
{geyanyan,pengyang}@seu.edu.cn
[2] School of Cyber Science and Engineering, Southeast University, Nanjing, China
[3] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing, China

Abstract. Keyphrase generation is a task of automatically creating keyphrases that reflect core information expressed in a given document. Keyphrase generation is actually a one-to-many problem, as it is possible predict phrases from different aspects of the document. In this paper, we explore the diversity of keyphrase generation and propose a novel model to improve the diversity of generated keyphrases by using the hierarchical structure of text content. Specifically, we relate hierarchical content with subtopics, which are modeled by a subgraph with the technique of graph clustering. In the generation stage, a multi-decoder is adopted to allow generating keyphrases in parallel, where each decoder corresponds to a subtopic. In addition, to take into account various means of expression, we introduce conditional variational autoencoder to enhance wording diversity. Experimental results on a public dataset confirms that our proposed method outperforms state-of-the-art methods on quantitative metrics and improves the keyphrase novelty.

Keywords: Keyphrase Generation · Seq2Seq · VAE · Graph Clustering

1 Introduction

Keyphrase is phrase or single word which reflect the core content and the main information of a document. Keyphrase generation (KG) belongs to natural language processing, which aims to effectively compress and refine the information of given text by using the neural network method to predict a set of phrases. Keyphrases can not only facilitate users to quickly obtain the central idea of the text and improve the speed of screening information, but also help users search and find the required documents.

There are two categories of keyphrase prediction methods: extractive and generative. Extractive methods usually intercept some word fragments directly from the source text based on simple statistical features or manually formulated rules. It focuses on how to extract the words or phrases containing key information as keyphrases, which means that the keyphrases obtained by extractive method already exist in the source text, called present keyphrases. Generative methods adopt the sequence-to-sequence (Seq2Seq) [7] model to generate word sequences after analyzing text. Compared with the extractive

© The Author(s) 2023

method, it can predict not only the present keyphrases, but also predict the keyphrases which do not exist in the source text. These words are collectively referred to as absent keyphrases.

The essence between document and keyphrase is a relationship of one-to-many. The core idea of a document usually needs to be summarized by several phrases. Therefore, generating more plentiful and different keyphrases is also very important. However, the standard encoder-decoder model which uses one-to-one to generate the target sequence. Therefore, when multiple different sequences are tried to be generated, the performance will be limited which cause the lack of diversity of the generated keyphrase.

Recently, most methods for diverse keyphrase generation through alternative search algorithms, such as beam search. In addition, there is another method to improve the diversity of generated keyphrases by introducing other mechanisms at decoding stage. For example, Chen et al. [5] proposed a framework of hierarchical decoding, which uses the exclusion mechanism to increase the difference between the first word of phrases to reduce the repetition rate of generated keyphrases. Ye et al. [13] proposed a method of one2set to enhance the diversity of generated keyphrases through K-step allocation mechanism of bipartite matching.

Although these methods have made some progress in improving the diversity of keyphrases, they are limited to diversification in the decoding steps. The contextual features of the decoding are used to promote diversity, while the content diversity caused by the hierarchy in the original text is rarely considered. For instance, when writing an article, people often discuss from different aspects around a certain central idea, such as causes, different clues, conclusions, etc., and finally check whether the substantive content deviates from the center. The behavior of people thinking about keyphrases can be regarded as the inverse process of the above behavior: When people get an article, they always read it roughly first. Then organize more relevant clues, such as words or phrases used to describe the same problem or thing. Finally summarize the selected parts and confirm the keyphrases. It can be inferred that the hierarchical organization of documents will also have an important impact on the generation of diversified keyphrases. On the other hand, the change of language expression also has an impact on the diversity of keyphrases. The same content will produce different keyphrases because of different wording. In summary, we believe that the keyphrases diversity needs to comprehensively consider two aspects: the diversity of document content and wording expression.

Based on the above observations, we propose a novel model which integrates text hierarchy to decompose the keyphrase generation task into encoding, subtopic selection and decoding. After encoding the text semantics, the model clusters the document into several parts with different contents based on the hierarchical information of the document. Before decoding, at first, the decoder selects a subtopic and then generates the target sequence. So as to improve the diversity of keyphrases from document content selection. According to the local characteristics of the graph structure, we regard a subtopic of the text as a specific local area in the syntactic graph, and transforms the segmentation of the document content into a subgraph segmentation task. Specifically, the syntactic graph is divided by using clustering method, and the whole syntactic graph is transformed into multiple subgraphs. Each subgraph corresponds to a specific subtopic. In the decoding process, multiple decoders in the decoder group decode different subtopics in parallel

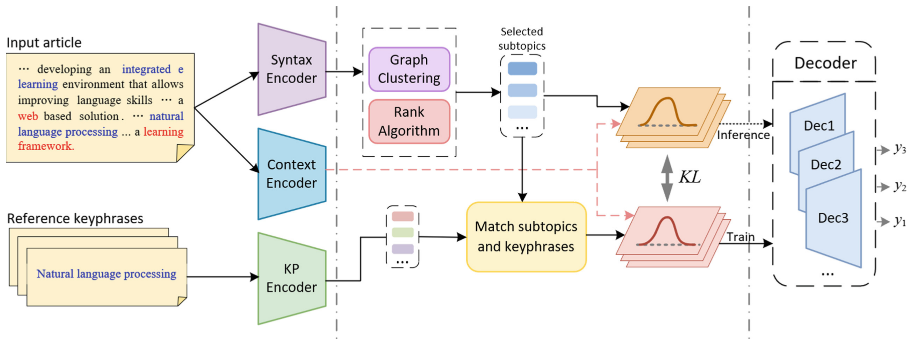
Input article
··· developing an integrated e learning environment that allows improving language skills ··· a web based solution. ··· natural language processing ... a learning framework.
Reference keyphrases
Natural language processing
Syntax Encoder
Context Encoder
KP Encoder
Graph Clustering
Rank Algorithm
Selected subtopics
Match subtopics and keyphrases
KL
Inference
Train
Decoder
Dec1
Dec2
Dec3
y_3
y_2
y_1

Fig. 1. An illustration of our proposed model for keyphrase generation.

to generate different keyphrases. In addition, in order to further improve the diversified expression ability of the model, the Variational Autoencoder (VAE) [8] is combined with the framework of standard encoder-decoder in this paper. Through the distributed latent representation of subtopic, model can capture the flexibility of wording and output vivid and diversified keyphrases. Experimental results demonstrate that our model can generate keyphrases that are more preferable in terms of wording diversity, and meanwhile outperforms baselines on quantitative metrics.

2 Model Architecture

The overview of our proposed model is shown in Fig. 1. The model consists of four modules: text representation encoder, subtopic generation module, sample latent variable network and a multi-decoder. The text encoder includes context encoder, syntax encoder and keyphrase encoder. The encoder performs encoding of the context and structural features of the input text and encoding of the keyphrases (training stage). Subtopic generation module completes the syntactic graph clustering and provides different subtopics for the decoder group. The variational network produces the latent variable. The multi-decoder composed of a group of decoders with the attention mechanism. The decoders in the decoder group generate keyphrases in parallel conditioned on a specific subtopic.

2.1 Encoder

Given a document X which contains m words, $X = (w_1, w_2, \ldots, w_m)$, we first map each word into low-dimensional real-valued vector. Then, the word embeddings are fed through a Bi-directional Gated Recurrent Unit (BiGRU) to obtain the contextual representation of each word.

In this work, we construct the document-level graph based on syntactic dependency tree using Stanford Dependency Parser [12], and employ a multi-layer graph convolution network (GCN) [9] to obtain the dependency features. For graph convolutional networks at L layer, nodes obtain information of L-order neighbor nodes, thus there is structural information in the feature vector representation of nodes. Intuitively, each word in the text contributes to the content of the text, but each word plays a different role and contributes to a different degree. For example, words with emotional color contribute more than article. The readout function is:

$$\boldsymbol{h}_g = Mean\left(\boldsymbol{H}^G\right) + Max\left(\boldsymbol{H}^G\right) \tag{1}$$

where $\boldsymbol{H}^g$ is the output of L-layer GCNs.

2.2 Subtopic Selection

In order to improve the diversity of keyphrase from the level of text content, we use the topic generation network to provide several subtopics covering different aspects for a given document, and decomposition the diversity implementation from the decoding stage. The text graph is firstly clustered according to similarity. By this step, the whole text graph is divided into subgraphs focusing on different topics, which can better capture the relevant information of specific subtopics. Specifically, for each word in document, we calculate the probability that it belongs to each subgraph:

$$\boldsymbol{q}^{ass} = softmax\left(\boldsymbol{W}_a \boldsymbol{h}_i^G + \boldsymbol{b}_a\right) \tag{2}$$

where $\boldsymbol{h}_i^G$ denotes node's representation from GCN, $\boldsymbol{W}_a$ and $\boldsymbol{b}_a$ are trainable parameters.

Then, the hidden state of the *i-th* subgraph is obtained through weighted sum of nodes:

$$\boldsymbol{h}_i^g = \boldsymbol{Q}_i^{assT} \boldsymbol{H}^G \tag{3}$$

Finally, calculating the cosine similarity score between document representation and subtopic's hidden, and reordering the subtopics.

$$\alpha_i = score\left(\boldsymbol{h}_i^g, \boldsymbol{c}\right) \tag{4}$$

At training time, we apply a MLP to match subtopics with targets.

2.3 Prior and Posterior

Most of the existing models use deterministic network, and minimize the cross entropy with ground truth at the word level in the training stage. However, cross-entropy-based training objectives require that the generated phrase and reference phrase must match strictly at the word level, which limits the diversity of keyphrases. Different from deterministic models, VAEs are based on probability distribution, and sample latent variable as the condition to reconstruct the data similar to the input but not exactly the same. This increases the flexibility and variability of the generated data. We introduce latent variable to model the changing semantic information of expression, and improve the flexibility of model expression through sampling, thus improving the diversity of keyphrases.

Consistent with previous work, we assume that the latent variable follows an isotropic Gaussian distribution. In order to compute the prior on the subtopic z,

$p(z|\boldsymbol{c},\boldsymbol{g}) \sim \mathcal{N}(\boldsymbol{\mu}',\boldsymbol{\sigma}'^2\boldsymbol{I})$, we concatenate the document representation $\boldsymbol{c}$, subtopic state $\boldsymbol{g}$ and keyphrase hidden state $\boldsymbol{y}$, and perform affine transformations. Similarly, to compute the approximate posterior $q_\phi(z|\boldsymbol{c},\boldsymbol{g},\boldsymbol{y}) \sim \mathcal{N}(\boldsymbol{\mu},\boldsymbol{\sigma}^2\boldsymbol{I})$, we linearly project $\boldsymbol{c}$ and $\boldsymbol{g}$.

$$\begin{bmatrix} \boldsymbol{\mu} \\ \log(\boldsymbol{\sigma}^2) \end{bmatrix} = MLP_q([\boldsymbol{c};\boldsymbol{g};\boldsymbol{y}]) \tag{5}$$

$$\begin{bmatrix} \boldsymbol{\mu}' \\ \log\left(\boldsymbol{\sigma}'^2\right) \end{bmatrix} = MLP_p([\boldsymbol{c};\boldsymbol{g}]) \tag{6}$$

2.4 Multi-decoder

Unlike previous work where a single decoder is used to generate a target sequence, we propose a novel multi-decoder model to generate keyphrases. We use k separate decoders to generate different keyphrases, and k is the number of subtopics. Ideally, multiple decoders should not focus on the same subtopic to generate keyphrases. For each decoder, we adopt one-layer GRU with copying mechanism. To ensure that the decoder starts with the context of the subtopic, we use document representation and the latent variable $\boldsymbol{z}$ of subtopic to compute the initial hidden state of the decoder:

$$\boldsymbol{s}_0 = \boldsymbol{W}_s[\boldsymbol{z};\boldsymbol{c}] + \boldsymbol{b}_s \tag{7}$$

where $\boldsymbol{W}_s$ and $\boldsymbol{b}_s$ being learning parameters, $\boldsymbol{z}$ is either obtained by the posterior network at training time or sampled from the prior network during testing. Meanwhile, the re-parametrization trick [8] is used to sample $\boldsymbol{z}$.

At each decoding step t, we calculate two distributions: vocabulary distribution and word attention score. Combining a soft select matching score to obtain the final predicting word distribution.

2.5 Training

We adopt the evidence lower bound (ELBO) objective to train the model, and the formulation of ELBO as:

$$\mathcal{L}_{loss} = \mathbb{E}_{z\sim q_\phi(z|y,g,c)}\left[logp_\theta(y|z,g,c)\right] - D_{KL}[q_\phi(z|y,g,c)||p_\theta(z|g,c)] \tag{8}$$

where the first item is reconstruction loss, second is Kullback-Leibler divergence loss. We employ KL cost annealing [2] to tackle the latent variable vanishing problem.

3 Experiments

3.1 Dataset and Baselines

We conduct experiments on a public dataset collected from scientific domain: KP20k [10]. Each of which comprised of a title, an abstract as source text, and a set of keyphrases. For our experiments, KP20k dataset is randomly split into training, validation and testing with 530k, 20k and 20k source-keyphrases pairs respectively. We compare our model with recent generation models including catSeqD [14], catSeqCorr [4], catSeqTG [6], catSeqTG-2RF1 [3] and DivKGen [1]. We utilize both quantitative and diversity metrics to evaluate the performance of our model.

3.2 Preprocessing and Implementation Details

The preprocessing includes: tokenize the sentences in source text with the NLTK tokenizer; accomplish dependency parsing using Stanford CoreNLP Tools; keep the most 50k frequent words as the vocabulary. We train the model with the following hyperparameters. We set the size of word embedding to 300 and is shared between encoder and decoder, and initialize it by GloVe pretrained word embedding. The hidden state size of GRU-based encoder, GCN-based encoder and decoder are 150, 100, 300 respectively. The number of layers for GRU are set to 2 and 1 respectively in encoder and decoder, and the layers for GCN is 3. The prior network and the posterior network both have one hidden layer with *SeLU* activation function. We set the size of the latent variable z and the number of the subtopic to 50 and 10. To update the model parameters, we adopt the Adam optimizer with initial learning rate of 0.001. The batch size is 64 during training. Our model is implemented with PyTorch 1.10.0. During inference time, we use greedy search and sample latent variable 2 times.

4 Results

4.1 Quantitative Analysis

We employ macro-average F1, precision and recall as quantity evaluation metrics. We report F1@M, P@M and R@M scores for two datasets, where M is the numbers of model generated keyphrases. We apply Porter Stemmer to predicted keyphrases and ground-truth before considering whether the predictions are correct. Table 1 shows that our model significantly outperforms all baselines by larger margin in R@M and F1@M on KP20k dataset, which demonstrates the effectiveness of our model for predicting. Meanwhile, compared with baselines, our model achieves slightly lower scores on P@M. We suspect that when the model attempts to generate more different keyphrases, it may predict new keyphrases that are not match in the ground truth.

4.2 Qualitative Analysis

In order to measure the diversity of the generated keyphrases, we introduce other metrics. Duplicate KPs, which measures the average duplication ratio of the predicted keyphrases. Self-BLEU [15] score, which calculate the word-overlap score between two generated keyphrases. Distinct score, which computes the Levenshtein Distance between two generated keyphrases. EmbSim score, which measures the cosine distance of the keyphrases embedding vectors. We use Sent2Vec [11] pretrained model to map the generated keyphrase to embedding. The experimental results on KP20k are displayed in right side of Table 1. The table shows that the diversity on Dup KPs, Self-BLEU and EmbSim of generated keyphrases with our method scores comparable to the DivKGen. The results reveal that our model outperforms baseline methods on Distinct metric. On the other hand, we can find that our model can predict more keyphrases and increase in the quality of predictions compared to DivKGen. These experiments show that our model can generate more diversity keyphrases, and meanwhile outperforms baselines on quantity metrics.

Table 1. Performance comparison of different models on KP20k datasets.

Models	Quantity evaluation			Quality evaluation				
	P@M	R@M	F1@M	#KPs	Dup KPs	Self-BLEU	Distinct	EmbSim
Ground Truth	-	-	-	5.3	0.1	3.8	32.7	0.159
CatSeq	29.1	26.0	27.4	7.3	26.6	26.6	45.6	0.328
CatSeqD	29.4	25.7	27.4	6.7	25.7	27.0	45.3	0.325
CatSeqCorr	28.3	26.4	27.3	7.0	23.2	24.5	44.0	0.309
CatSeqTG	**29.5**	26.2	27.8	6.8	24.7	26.2	45.2	0.323
CatSeqTG-2RF1	27.4	28.6	28.0	7.5	30.9	30.7	46.7	0.341
DivKGen	27.3	24.0	25.6	4.6	**4.9**	**8.8**	35.2	**0.185**
Ours	24.5	**35.6**	**29.3**	**7.6**	6.8	10.7	**34.6**	0.195

5 Conclusion

In this paper, we focus on the diversify of keyphrases from the perspectives of content selection of source text and means of expression. We propose a novel model that breaks the keyphrase generation task into three steps, namely encoding, selection and decoding. In the selection step, the model produces different subtopics about source text to guide decoder to generate a variety of keyphrases. We introduce a conditional variational generation network to improve the diversity of model expression by sampling latent variables from probability distribution. Experimental results of both quantitative and diversity evaluations on public datasets confirms the effectiveness of our approach.

Acknowledgments. This work was supported in part by the Consulting Project of Chinese Academy of Engineering under Grant 2020-XY-5, 2018-XY-07, and in part by the Fundamental Research Funds for the Central Universities and the Academy-Locality Cooperation Project of Chinese Academy of Engineering under Grant JS2021ZT05.

References

1. Bahuleyan H, El Asri L. Diverse Keyphrase Generation with Neural Unlikelihood Training[C]//Proceedings of the 28th International Conference on Computational Linguistics. 2020: 5271-5287.
2. Bowman S, Vilnis L, Vinyals O, et al. Generating Sentences from a Continuous Space[C]//Proceedings of the 20th SIGNLL. 2016: 10-21.
3. Chan H P, Chen W, Wang L, et al. Neural Keyphrase Generation via Reinforcement Learning with Adaptive Rewards[C]//Proceedings of the 57th Annual Meeting of the ACL. 2019: 2163-2174.
4. Chen J, Zhang X, Wu Y, et al. Keyphrase Generation with Correlation Constraints[C]//Proceedings of the 2018 Conference on EMNLP. 2018: 4057-4066.

5. Chen W, Chan H P, Li P, et al. Exclusive Hierarchical Decoding for Deep Keyphrase Generation[C]//Proceedings of the 58th ACL. 2020: 1095-1105.
6. Chen W, Gao Y, Zhang J, et al. -Guided Encoding for Keyphrase Generation[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33(01): 6268-6275.
7. Cho K, van Merriënboer B, Gülçehre Ç, et al. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation[C]//Proceedings of the 2014 Conference on EMNLP. 2014: 1724-1734.
8. Kingma D P, Welling M. Auto-Encoding Variational Bayes[J]. stat, 2014, 1050: 1.
9. Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[C]// International Conference on Learning Representations. 2017: 1-14.
10. Meng R, Zhao S, Han S, et al. Deep Keyphrase Generation[C]//Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1). 2017: 582-592.
11. Pagliardini M, Gupta P, Jaggi M. Unsupervised Learning of Sentence Embeddings Using Compositional n-Gram Features[C]//Proceedings of the 2018 Conference of the NAACL, Volume 1. 2018: 528-540.
12. Qi P, Zhang Y, Zhang Y, et al. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages[C]//Proceedings of the 58th Annual Meeting of the ACL. 2020: 101-108.
13. Ye J, Gui T, Luo Y, et al. One2Set: Generating Diverse Keyphrases as a Set[C]//Proceedings of the 59th Annual Meeting of the ACL (Volume 1: Long Papers). 2021: 4598-4608.
14. Yuan X, Wang T, Meng R, et al. One Size Does Not Fit All: Generating and Evaluating Variable Number of Keyphrases[C]//Proceedings of the 58th Annual Meeting of the ACL. 2020: 7961-7975.
15. Zhu Y, Lu S, Zheng L, et al. Texygen: A benchmarking platform for text generation models[C]//The 41st International ACM SIGIR Conference. 2018: 1097-1100.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.