Reimplement of *SGG* and Further Work

Written by Team[1]*
Lingfeng Hu-31520211154050, Fenghao Qu-32920192201094, Zhou Li-32920192201076, Qiuyu Dai-23020211153889, Zongyang Liu-34720201151374

[1]Association for the Advancement of Artificial Intelligence
1092055613@qq.com

Abstract

Keyphrase Prediction task aims to generating keyphrases that can summarize the main idea of the source text. Most existing keyphrase generation approaches can be categorized into two ways: extracting words from the source text, and generating new words absent in the source text. Some methods utilize both of the ways, however, ignore the diversity among these two ways. Inspired by the latest article *SGG*: *Learning to Select*, *Guide*, *and Generate for Keyphrase Generation*, which treats different ways with different mechanism to get performance boosted, we decided to reimplement the work to get a solution for this problem. Besides reimplementation, we'll evaluate the method on some other datasets that the origin paper didn't try, to test the generalization ability of the model. Experimental results on four keyphrase generation benchmarks and a large scale test dataset KP20k we added demonstrate the effectiveness of the model, which outperforms the strong baselines for both present and absent keyphrases generation.

Introduction

Keyphrases are short phrases that indicate the core information of a document. The keyphrase generation (KG) problem focuses on automatically producing a keyphrase set (a set of keyphrases) for the given document.The keyphrase extraction and generation can play an important role in academic field and information processing field. Because of the condensed expression, keyphrases can benefit various downstream applications including opinion mining, document clustering, and text summarization.

Keyphrases of a document can be categorized into two groups: present keyphrase that appears in the document and absent keyphrase that does not appear in the document. Recent generative methods for KG apply the attentional encoder-decoder framework but they include the following two problems:

- They complicate the identification of present keyphrases. Specifically, they search for words over the entire predefined vocabulary containing a vast amount of words (e.g., 50,000 words) to generate a present keyphrase verbatim, which is over parameterized since a present keyphrase can be simply selected from a continuous subsequence of the source text containing limited words (e.g.,less than 400 words).
- They weaken the generation of absent keyphrases. Existing models for absent keyphrase generation are usually trained on datasets mixed with a large proportion of present keyphrases. Table 1 shows that nearly half of the training data are present keyphrases, which leads to the extremely low proportions of absent keyphrases generated by such a model, i.e., CopyRNN. The above observation demonstrates that these methods are biased towards replicating words from source text for present keyphrase generation, which will inevitably affect the performance on generating absent keyphrases.

To address the aforementioned problems, we choose to reimplement a Select-Guide-Generate (SGG) approach, which is the state of the art method. The architecture of the Select-Guide Generate (SGG) approach is illustrated in Figure. The model is the extension of Seq2Seq framework which consists of a text encoder, a selector, a guider, and a generator. The text encoder converts the source text x into a set of hidden representation vectors $\{h_i\}_{i=1}^{L}$ with a bidirectional Long Short-term Memory Network (bi-LSTM) (Hochreiter and Schmidhuber 1997), where L is the length of source text sequence. The selector is a uni-directional LSTM, which predicts the present keyphrase sequence y^p based on the attention distribution over source words. After selecting present keyphrases, a guider is produced by a guider to memorize the prediction information of the selector, and then fed to the attention module of a generator to adjust the information it pays attention to. The selection-guided generator is also implemented as a uni-directional LSTM, which produces the absent keyphrase sequence y^a based on two distributions over predefined-vocabulary and source words, respectively. At the same time, a soft switch gate p_{gen} is employed as a trade-off between the above two distributions.

Related Work

The extraction and generation methods are two different research directions in the field of keyphrase prediction. The existing extraction methods can be broadly classified into supervised and unsupervised approaches. The supervised ap-

*With help from the AAAI Publications Committee.

proaches using treat keyphrase extraction as a binary classification task, which train the models with the features of labeled keyphrases to determine whether a candidate phrase is a keyphrase. TF-IDF and TextRank are typical. (Medelyan, Frank, and Witten 2009; Gollapalli, Li, and Yang 2017). In contrast, the unsupervised approaches treat keyphrase extraction as a ranking task, scoring each candidate using some different ranking metrics, such as clustering (Liu et al. 2009), or graph-based ranking (Mihalcea and Tarau 2004; Wang et al. 2014; Gollapalli and Caragea 2014; Zhang et al. 2017).

SGG work is mainly related to keyphrase generation approaches. CopyRNN (Rui et al. 2017) can not only generate words based on the original text, but also generate words that the original text does not have. The disadvantage is that a large amount of annotation data is required. To address this problem, Ye and Wang (2018) proposed a semi-supervised keyphrase generation methods by leveraging both labeled data and large-scale unlabeled samples for learning. In addition, CopyRNN uses the concatenation of article title and abstract as input, ignoring the leading role of the title. To address this deficiency, Chen et al. (2019) proposed a title-guided Seq2Seq network to sufficiently utilize the already summarized information in title. In addition, some research attempts to introduce external knowledge into keyphrase generation, such as syntactic constraints (Zhao and Zhang 2019) and latent topics (Wang et al. 2019).

These approaches do not consider the one-to-many(with the concatenation of all keyphrases) relationship between the input text and target keyphrases, and thus fail to model the correlation among the multiple target keyphrases. To overcome this drawback, Chen† et al. (2018) propose a new sequence-to-sequence architecture for keyphrase generation named CorrRNN, which captures correlation among multiple keyphrases. To avoid generating duplicate keyphrases, Chen et al. (2020) proposed an exclusive hierarchical decoding framework that includes a hierarchical decoding process and either a soft or a hard exclusion mechanism. Similar to the above, SGG separately models one-to-many relationship between the input text and present keyphrases and absent keyphrases, and our method deploys a guider to avoid the generator generating duplicate present keyphrases.

Proposed Solution

To address the above problems, we reproduced the Select-Guide-Generate approach, as shown in Figure 1, which can deal with present and absent keyphrase generation separately with different stages based on different mechanisms (Zhao and Zhang 2019). Specifically, our model is implemented with a hierarchical neural network which performs Seq2Seq learning by applying a multi-task learning strategy. This network consists of a selector at low layer, a generator at high layer, and a guider at middle layer for information transfer. The selector generates present keyphrases through a pointing mechanism, which adopts attention distributions to select a sequence of words from the source text as output. The generator further generates the absent keyphrases through a pointing-generating mechanism. Because present keyphrases have already been generated by the

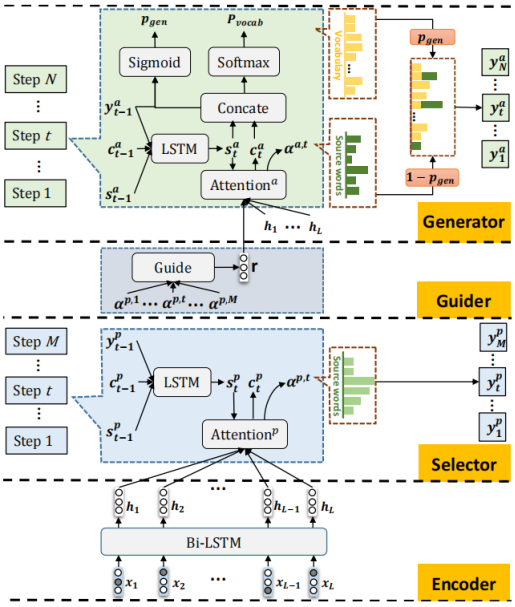

Figure 1: The architecture of the proposed SGG which is implemented with a hierarchical neural network.

selector, they should not be generated again by the generator. Therefore, a guider is designed to memorize the generated present keyphrases from the selector, and then fed into the attention module of the generator to constrain it to focus on generating absent keyphrases. We summarize our main contributions as follows:

- We reproduce the SGG approach which models present and absent keyphrase generation separately in different stages, i.e., select, guide, and generate, without sacrificing the end-to-end training through back-propagation. The origin paper didn't provide source code, however, we reproduced the work through modifying the code provided in another paper(Chen et al. 2020).
- Extensive experiments are conducted to verify the effectiveness of our model, which not only improves present keyphrase generation but also dramatically boosts the performance of absent keyphrase generation. In addition, we apply this model to other large scale datasets that the origin paper didn't referred to verify the generalization of the model.

In order to evaluate our reproduced model comprehensively, we test models on four widely used public datasets from the scientific domain, namely Inspec(Hulth and Megyesi 2006), Krapivin, SemEval 2010(Su et al. 2010) and NUS(Nguyen and Kan 2007). We set maximal length of source sequence as 400, 25 for target sequence of selector and generator, and 50 for the decoders of all generation baselines. We choose the top 50,000 frequently occurred words as our vocabulary. The dimension of the word embedding is 128. The dimension of hidden state in encoder, selector and generator is 512. The word embedding is randomly initialized and learned during training.

We initialize the parameters of models with uniform dis-

tribution in [-0.2,0.2]. The model is optimized using Adagrad with learning rate = 0.15, initial accumulator = 0.1 and maximal gradient normalization = 2. In the inference process, we use beam search to generate diverse keyphrases and the beam size is 200 same as baselines.

Experiments

Some details and Results

We set all the parameters as stated in the paper, all the models are trained on a single RTX 2080Ti. In this section, we present the results of present and absent keyphrase generation separately. The results of predicting present keyphrases are shown in Table 1, in which the F1 at top-5 are given. We first compare our reimplemented model with the conventional keyphrase extraction methods. The results show that our model performs better than extraction methods with a large margin, demonstrating the potential of the Seq2Seq-based generation models in automatic keyphrase extraction task. We then compare our model with the generation baselines, and the results are shown in Table 2, which indicates that our model still outperforms these baselines. The better performance of SGG illustrates the necessity of distinguishing the extraction approach and generation approach while generating keyphrases. In the mean time, it also shows our success in reproducing the behavior of the model.

Table 1: F1@5 results of predicting present keyphrases of different models on four datasets. The performance data of the comparison model in the chart are all taken from the original SGG paper, and only the last line of SGG data is our experimental result.

method	Inspec	Krapivin	NUS	SemEval
	F1@5	F1@5	F1@5	F1@5
TF-IDF	22.1	12.9	13.6	12.8
TextRank	22.3	18.9	19.5	17.6
KEA	9.8	11.0	6.9	2.5
CopyRNN	27.8	31.1	33.4	29.3
CopyTrans+	21.1	26.4	35.1	29.5
CorrRNN	-	**31.8**	35.8	32.0
CatSeq	29.0	30.7	35.9	30.2
SGG	**29.9**	26.8	**36.1**	**32.7**

Results Analysis

From the results, it can be seen that there is a certain gap between our reproducible results and the index results given in the original paper.There are two reasons for speculation.

- First. When recording data in the original paper, the author used multiple tests to obtain the optimal results, while we only ran it once. It is normal for the results to fluctuate.
- Second. The original paper may have introduced some tricks in code writing, which was not explained in the paper.We lack trick to improve performance, so the final data is not as good as the results shown in the paper.

Table 2: Recall@50 results of predicting absent keyphrases of different models on four datasets. The performance data of the comparison model in the chart are all taken from the original SGG paper, and only the last line of SGG data is our experimental result.

Method	Inspec	Krapivin	NUS	SemEval
CopyRNN	10.0	20.2	11.6	**6.7**
CopyTrans+	5.6	16.9	8.9	4.1
CorrRNN+	8.5	15.2	8.0	3.5
CatSeq	2.9	7.4	3.1	2.5
SGG	**10.7**	**22.1**	**11.9**	4.8

In addition to reproducing the original paper, we also applied the model to other datasets to test the generalization ability of the model.

Here, we select the KP20k dataset which is very popular in the field of KeyPhrase generation. The dataset includes almost 550k pieces of data in total, among which 20,000 pieces are used for verification and 20,000 pieces for testing.This dataset is two orders of magnitude larger than the four SGG datasets used in the original paper (i.e., 2000 Inspec data and 2303 Krapivin data), so the results are less affected by fluctuations and more reliable.The test results are shown in Table 3.

Table 3: Since the original SGG paper did not show the performance of other comparative models on KP20k,We select the data of KP20k data set comparison model performance from another paper *Exclusive Hierarchical Decoding for Deep Keyphrase Generation* (Chen et al. 2020). Then we also make a comparison with SGG. There is no comparison model for recall@50 in that paper, so only SGG results are displayed under absent KP: recall@50.

method	KP20k	
	present kp: F1@5	absent kp: Recal@50
Transformer	22.1	
catSeq	22.3	
catSeqD	9.8	
catSeqCorr	27.8	
SGG	29.5	10.1

As it can be seen, SGG still performs better than Sota even on large datasets.But the degree of optimization is mediocre, just over a percentage point higher.

Conclusion

In this paper, a Select-Guide-Generate (SGG) approach is reimplemented with a hierarchical neural model for keyphrase generation, which separately deals with the generation of present and absent keyphrases and obtained remarkable improvement on the task. Comprehensive empirical studies demonstrate the effectiveness of SGG. Further-

more, an evaluation on a large scale dataset indicates the extensibility of SGG model.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *Computer Science*.

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

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

Chen†, J.; Zhang, X.; Wu†, Y.; Yan‡, Z.; and Li†, Z. 2018. Keyphrase Generation with Correlation Constraints.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

Gollapalli, S. D.; and Caragea, C. 2014. Extracting keyphrases from research papers using citation networks. *AAAI Press*.

Gollapalli, S. D.; Li, X.-L.; and Yang, P. 2017. Incorporating expert knowledge into keyphrase extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Gu, J.; Lu, Z.; Li, H.; and Li, V. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780.

Hulth, A.; and Megyesi, B. B. 2006. A Study on Automatically Extracted Keywords in Text Categorization. In *ACL 2006, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006*.

Krapivin, M.; Autaeu, A.; and Marchese, M. ???? Large Dataset for Keyphrases Extraction.

Liu, Z.; Li, P.; Zheng, Y.; and Sun, M. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, 257–266.

Medelyan, O.; Frank, E.; and Witten, I. H. 2009. Human-competitive tagging using automatic keyphrase extraction - Usage Statistics.

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.

Rui, M.; Zhao, S.; Han, S.; He, D.; and Yu, C. 2017. Deep Keyphrase Generation.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Su, N.; Kim; Medelyan, O.; Kan, M. Y.; and Baldwin, T. 2010. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. *Language Resources and Evaluation*, 47(3): 21–26.

Subramanian, S.; Wang, T.; Yuan, X.; Zhang, S.; Bengio, Y.; and Trischler, A. 2017. Neural models for key phrase detection and question generation. *arXiv preprint arXiv:1706.04560*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; and Li, H. 2016. Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. *arXiv preprint arXiv:1506.03134*.

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

Wang, F.; Wang, Z.; Wang, S.; and Li, Z. 2014. Exploiting description knowledge for keyphrase extraction. In *Pacific Rim International Conference on Artificial Intelligence*, 130–142. Springer.

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. *arXiv preprint arXiv:1906.03889*.

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.

Ye, H.; and Wang, L. 2018. Semi-supervised learning for neural keyphrase generation. *arXiv preprint arXiv:1808.06773*.

Yuan, X.; Wang, T.; Meng, R.; Thaker, K.; Brusilovsky, P.; He, D.; and Trischler, A. 2018. One size does not fit all: Generating and evaluating variable number of keyphrases. *arXiv preprint arXiv:1810.05241*.

Zhang, Y.; Chang, Y.; Liu, X.; Gollapalli, S. D.; Li, X.; and Xiao, C. 2017. Mike: keyphrase extraction by integrating multidimensional information. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1349–1358.

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.