Enhancing Pre-Trained Language Representations Based on Contrastive Learning for Unsupervised Keyphrase Extraction

Zhaohui Wang[12], Xinghua Zhang[12], Yanzeng Li[3], Yubin Wang[12], Jiawei Sheng[12], Tingwen Liu[12*], Hongbo Xu[12]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, {wangzhaohui, liutingwen}@iie.ac.cn
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Wangxuan Institute of Computer Technology, Peking University, Beijing, China

***Abstract*—Keyphrase extraction (KPE) aims to obtain a set of phrases from a document that can summarize the main content of the document. Recently, pre-trained language models (LMs), especially BERT and ELMo, have achieved remarkable success, presenting new state-of-the-art results in unsupervised KPE. However, current pre-trained LMs focus on building language modeling objectives to learn a general representation, ignoring the keyphrase-related knowledge. Intuitively, the joint embedding of the keyphrase set should tend to be close to that of the extracted document, and far from those of other documents. In this work, we propose a contrastive learning-based semantic representation task to further improve BERT for unsupervised KPE. Particularly, we design a doc-phrase attention module to generate joint semantic embedding of the keyphrase set as a positive sample and select other semantically similar documents as hard negative samples. In the prediction layer, we further add an accumulated self-attention module to calculate the final scores of candidate phrases. We compare with eight strong baselines, and evaluate our model on three publicly available datasets. Experimental results show that our model is effective and robust on both long and short documents.**

***Index Terms*—keyphrase extraction, contrastive learning, pre-trained language models, unsupervised, attention**

I. Introduction

With a vast amount of scientific or non-scientific articles published online every year, indexing and information retrieval have become challenging. Keyphrase extraction (KPE) is concerned with automatically extracting a set of representative phrases from a document that concisely summarizes its content [1]. It can significantly accelerate the speed of information retrieval and help people get first-hand information from a long text quickly and accurately. As a result, automatic keyphrase extraction is crucial in natural language processing (NLP).

Recently, pre-trained language models (LMs), such as ELMo [2] and BERT [3], have caused a stir in the KPE community. These LMs are pre-trained on unlabeled text and then applied to KPE, in either an embedding-based unsupervised [4], [5] or a supervised [6] manner, both offering substantial performance boosts. Despite refreshing the state-of-the-art performance of KPE, the current pre-trained techniques are not directly optimized for KPE. Typically, these models build unsupervised training objectives to capture dependency between words and learn a general language representation [7], while rarely considering incorporating keyphrase information which can provide rich knowledge for KPE. Due to little knowledge connection between KPE and general language modeling, how to adapt public pre-trained models to be KPE-specific remains an open problem.

The embedding-based unsupervised KPE, which has been widely studied, ensures high retrieval speed and outstanding performance on certain datasets without a large amount of annotated data. However, these methods commonly include the following steps: extracting noun phrases from the document as candidate phrases, utilizing a pre-trained language model to generate document and phrase embeddings, calculating each candidate's final score independently. During the scoring stage, they generally use cosine similarity to assess the relevance between a single candidate phrase and document. However, the semantic information contained in the keyphrase set, which expresses the main content of a document, is ignored.

In this paper, we aim to fully utilize the joint semantic information of the keyphrase set. Inspired by the success of contrastive learning in computer vision (CV), the most recent methods are interested in determining whether it could also assist language models in promoting representation ability [8]. With the simple intuition that the joint semantics of the keyphrase set tends be close to its document and be dissimilar to other document in semantic space, we propose a contrastive learning-based semantic representation task, which leverages triplet loss [9] to effectively optimize the representations of the pre-trained language model.

Specifically, we use BERT [3] language model to encode documents and candidate phrases. In triplet loss, we use the document embeddings as anchors. The doc-phrase attention module is designed to distinguish keyphrases and non-keyphrases and generate joint semantic embeddings of keyphrase sets as positive samples. We also select semantically similar documents as hard negative samples. In the prediction stage, we utilize the linear integration of doc-phrase attention and accumulated self-attention modules to calculate the final scores of candidate keyphrases.

We compare our model with eight unsupervised keyphrase

*Corresponding author
DOI reference number: 10.18293/SEKE2022-131

extraction methods on three benchmark datasets. Two datasets contain short documents, and one contains long documents. Experimental results show that our model performs better than or as competitive as the baselines. The main contributions of this paper are summarized as follows:

- We propose a contrastive learning-based semantic representation task to enhance pre-trained LMs for unsupervised KPE;
- We design a doc-phrase attention module to generate joint semantic embedding contained in keyphrase set and combine accumulated self-attention module to calculate the scores for candidate phrases;
- Experimental results show that our model outperforms eight strong baselines and is robust to identify keyphrases from both short and long documents of different domains.

II. Related Work

This section briefly describes prior works about unsupervised keyphrase extraction and contrastive self-supervised learning, which have strong connections with this paper.

A. Unsupervised Keyphrase Extraction

Keyphrase extraction is an important problem in NLP area. Researchers have developed a wide range of solutions for this task in the last few years, including supervised and unsupervised methods. In the supervised setting, keyphrase extraction is usually treated as a classification problem [10], [11] or text generation [12], [13] task, which needs large amounts of annotated data for training and are generally domain-specific. In this paper, we discuss unsupervised keyprhase extraction only.

Traditional unsupervised methods such as TF-IDF and YAKE! [14], are statistic-based methods. YAKE! incorporates five different features for each term to calculate a ranking score after preprocessing the text by splitting it into individual terms.

In addition, graph-based methods via converting a document into graphs are popular. Motivated by Brin and Page [15], TextRank [16] was proposed to rank nodes of graphs constructed by word co-occurrence windows and implements PageRank iteratively. After this, various works attempted to expand TextRank. SingleRank [17] is one of the modifications in which the weight of each edge is equal to the number of co-occurrences of two corresponding words. TopicRank [18] assigned a salience score to each topic by candidate keyphrase clustering.

Embedding-based methods rely on notable new developments in text representation learning by encoding text sequences into low-dimension vectors. Hence, embedding-based unsupervised keyphrase extraction has gained a lot of attention in recent years. EmbedRank [19] proposed to measure the text similarity between phrase and document embeddings to make predictions. Sun et al. [4] proposed SIFRank, which improves the static embedding of EmbedRank with a pre-trained language model and a sentence embedding model SIF [20]. AttentionRank [5] proposed a hybrid attention model to identify keyphrases from a document. These embedding-based methods ignore the information carried in the keyphrase set, while we effectively capture and utilize the information by a doc-phrase attention module and achieve competitive results.

B. Contrastive Self-Supervised Learning

Self-supervised learning has gained popularity due to its ability to avoid the cost of annotated large-scale datasets. It can adopt self-defined pseudo labels as supervision and use the learned representations for several downstream tasks. Specifically, contrastive learning has recently become a dominant component in self-supervised learning methods in CV, NLP, and other domains [21]. Contrastive learning aims to learn effective representation by pulling semantically close neighbors together and pushing non-neighbors apart. CERT [8] applied the back-translation to create augmentations for original sentences. Declutr [22] regarded that different spans inside one document are similar to each others. SCL [23] proposed a supervised contrastive learning objective to increase the distance between categories for the fine-tuning stage. SimCSE [24] described an unsupervised approach, which takes an input sentence and predicts itself in a contrastive objective, with only standard dropout used as noise. Intuitively, the joint semantics of the keyphrase set tends to be close to the entire document, and far from other documents in semantic space. Therefore, we desingn a contrastive learning-based semantic representation task to enhance pre-trained language model for unsupervised KPE.

III. Problem Definition

Keyphrase extraction is the task of automatically selecting a small set of phrases that summarize the document's main content. Formally, given a document $d = \{w_1, w_2, \ldots, w_n\}$ in dataset D consisting of n words, candidate phrases can be selected as set $C = \{c_1, c_2, \ldots, c_m\}$, where m is the number of candidates. Each candidate c_i consists of several words $c_i = \{c_i^1, c_i^2, \ldots, c_i^l\}$. Keyphrase extraction is to select Top-K candidates from C forming a keyphrase set $K = \{k_1, k_2, \ldots, k_t\}$ according to their scores, usually $t < m$.

IV. Methodology

In this section, we introduce our method in detail. The overall architecture is illustrated in Figure 1, which can be divided into three parts: (a) extracting a candidate set C from a document for the contrastive framework and prediction stage; (b) our contrastive architecture to enhance BERT model and (c) the final score calculation strategy for each candidate.

A. Candidate Generation

We use the candidate generation module implemented in EmbedRank [19]. Firstly, the document is tagged to a sequence of words with part-of-speech tags. Then, we extract the noun phrases (NPs) from the sequence according to the part-of-speech tags using NP-chunker (pattern written by regular

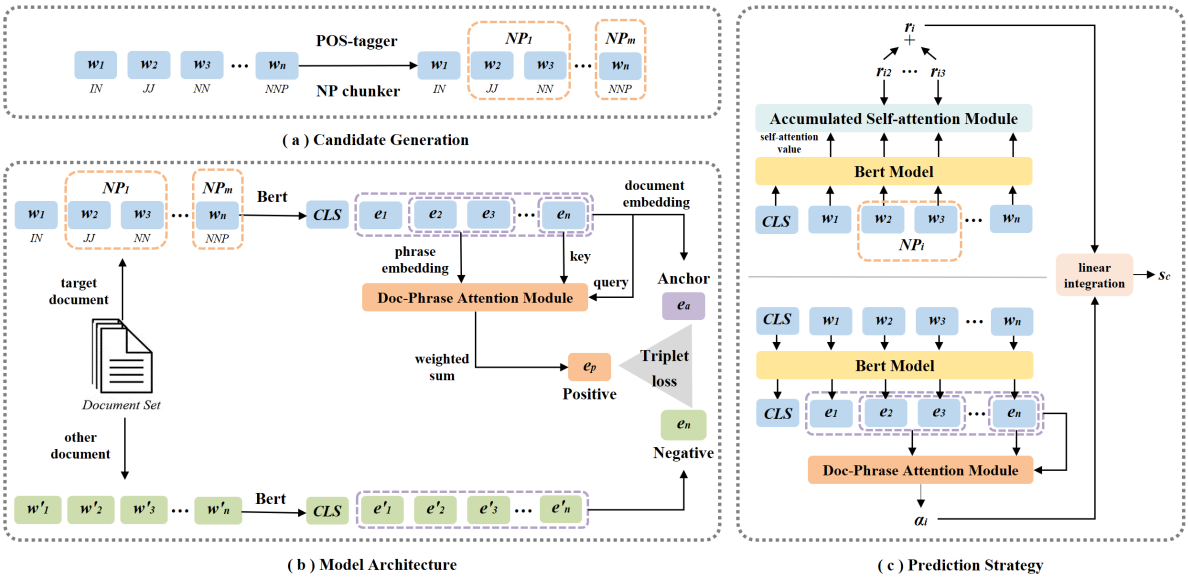

Fig. 1. Overview of Our Method.

expression). The NPs extracted from the document are the candidate keyphrases. Specifically, we use the Stanford CoreNLP to tokenize part-of-speech tags. And regular expression <NN.*|JJ>*<NN.*> is used to extract noun phrases as candidate keyphrases.

B. Model Architecture

Existing embedding-based works measure the semantic similarity between each candidate phrase and the document independently, which ignores the semantic information contained in the keyphrase set. Intuitively, the joint semantics of the keyphrase set, which can better describe the main content of a document, tends to be close to the entire document, but far from other documents in semantic space. In this part, we introduce how to generate a joint semantic embedding from a phrase set and how to model the relationship between keyphrase set and document embeddings using ontrastive learning.

1) Encoder: BERT model encodes document d into a sequence of vectors $E_d = \{e_d^1, e_d^2, e_d^3, \ldots, e_d^n\}$, where e_i^d is the i-th word's contextualized embedding in document d from the last transformer layer.

$$E = \text{Bert}\left(w_1, w_2 \ldots, w_n\right) \tag{1}$$

MeanPooling method is adopted to obtain document-level and phrase-level embeddings.

$$e_d = \text{MeanPooling}\left(e_d^1, e_d^2, e_d^3, \ldots, e_d^n\right) \tag{2}$$

$$c_i = \text{MeanPooling}\left(c_i^1, c_i^2, \ldots, c_i^l\right) \tag{3}$$

2) Doc-Phrase Attention Module: The doc-phrase attention is designed to measure the importance between candidate phrases. Considering that only part of the candidate phrases are keyphrases, we design a doc-phrase attention module to distinguish keyphrases from non-keyphrases. The input of doc-phrase attention module is the embedding of document d and the embeddings of candidate phrases in set C, where the former is used as query and the latter is considered as key and value. The joint semantic embedding of phrase set for target document d can be calculated as the weighted sum of c_i.

$$\text{Att}\left(e_d, c_i\right) = e_d^T W c_i \tag{4}$$

$$\alpha_i = \frac{e^{\text{Att}(e_d, c_i)}}{\sum_{i=1}^{m} e^{\text{Att}(e_d, c_i)}} \tag{5}$$

$$e_t = \sum_{i=1}^{m} \alpha_i c_i \tag{6}$$

3) Contrastive supervision: Triplet loss [9] was used as the overall training objective. In order to ensure fast convergence, we design an effective strategy to select hard negative samples.

Given the triplets (e_a, e_p, e_n) where a, p, n represent anchor, positive and negative examples respectively, triplet loss aims to narrow the gap between anchor and positive examples and distinguish between anchor and negative examples. The constraint is shown as Equation 7. Typically, a document's main content tends to be close to itself and far from other documents in semantic space. We take the target document embedding as the anchor, the joint semantic embedding of the target document as the positive sample and the embedding of other documents in dataset as the negative samples.

$$\left\|e_a - e_p\right\|_2^2 + m < \left\|e_a - e_n\right\|_2^2, \forall \left(e_a, e_p, e_n\right) \in \mathbb{R} \tag{7}$$

The loss function can be defined as the following:

$$L = \sum_{d \in D} \left[\left\|e_d^a - e_d^p\right\|_2^2 - \left\|e_d^a - e_d^n\right\|_2^2 + m\right]_+ \tag{8}$$

https://stanfordnlp.github.io/CoreNLP/

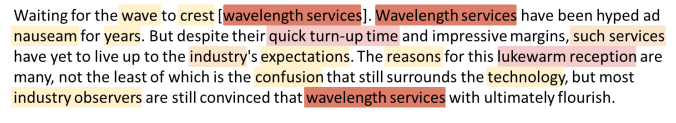

Fig. 2. Visualization Example of The Accumulated Self-attention Module.

where e_d^a and e_d^n represent the embedding of target document and the embedding of other documents calculated by Equation 2. Furthermore, e_d^p represents the joint semantic embedding of target document's phrase set calculated by Equation 6 and m denotes margin.

In addition, it is crucial to select or mining triplets that violate the triplet constraint in Equation 7. This means that, given e_a we need select an e_p (*hard positive*) such that $\text{argmax}_{e_p} \|e_a - e_p\|_2^2$ and similarly e_n (*hard negative*) such that $\text{argmax}_{e_n} \|e_a - e_n\|_2^2$. We select documents that are semantically closer to the target document as negative examples. As for the positive example, according to experimental results, the joint semantic embedding generated by doc-phrase attention module can meet the triplet loss requirements.

C. Prediction Strategy

1) Accumulated Self-attention Module: Motivated by [5], [25], we extract self-attention weights of the words from the BERT. As shown in Figure 2, we sum the attention weights that a phrase received in the document, and all the noun phrases are highlighted. The higher self-attention it receives, the darker the noun chunk is. Intuitively, noun phrases with darker colors should be selected as keywords with higher probabilities. The calculating method is introduced as follows.

To obtain the attention value r_w of the word w within a sentence, we sum the attentions $r_{w'w}$ that a word w received from other words w' within the same sentence s, shown as Equation 9. This attention value r_w represents the importance of the word within the context of a sentence.

$$r_w = \sum_{w' \in s} r_{w'w} \tag{9}$$

To calculate the self-attention of a candidate c in sentence j, we add up the attentions of the words in c, shown as Equation 10.

$$r_j^c = \sum_{w \in c} r_w \tag{10}$$

The document level self-attention value of candidate c is computed as the sum of all self-attention values of c in each sentence of document d, shown as Equation 11.

$$r_c = \sum_{j \in d} r_j^c \tag{11}$$

2) Final Score Calculation: For document d, the doc-phrase attention value α_c and the accumulated self-attention value r_c are calculated and normalized separately for each candidates. The final score of a candidate is generated by linear integration of these two values using Equation 12, where $\beta \in [0, 1]$.

$$S_c = \beta * r_c + (1 - \beta) * \alpha_c \tag{12}$$

TABLE I
STATISTICS OF THE THREE DATASETS.

Dataset	Documents				Keyphrases		
	Total	Type	AveWords	AveSentences	Total	AveNumber	AveLength
Inspec	500	Abstracts	134	6	4912	9.8	2.3
SemEval2017	493	Paragraph	168	7	8529	17.3	3
SemEval2010	243	Full papers	8154	369	3662	15.1	2.1

V. EXPERIMENTS

In this section, we first set up the experiments by preparing the datasets and introducing the comparison methods, and then report the results of conducted experiments to demonstrate the effectiveness of the proposed method.

A. Datasets and Evaluation Metrics

To fully evaluate the performance of our model, we testify it on three benchmark datasets. The statistics of the three datasets are shown in Table I. Datasets Inspec [26] and SemEval2017 [27] contain short documents, whereas SemEval2010 [28] contains long documents.

The **Inspec** dataset consists of 2000 short documents selected from scientific journal abstracts. There are 1000 documents for training, 500 for validation and 500 for test. We use the test part to validate our model in this paper.

The **SemEval2017** dataset is the Task 10 in SemEval2017 competition. It contains 493 paragraphs selected from ScienceDirect journal, covering computer science, materials science and physics. Each document is annotated with keyphrases by an undergraduate and an expert.

The **SemEval2010** dataset consists of 243 full papers from the ACM Digital Library. The articles are purposefully selected from four different areas.

B. Baselines

We compared our model with eight keyphrase extraction methods which are all unsupervised models in three types: statistic-based model, graph-based model and embedding-based model. The statistic-based models are TF-IDF and YAKE! [14]. The graph-based models are TopicRank [18], PositionRank [29] and SingleRank [17]. The embedding-based models are EmbedRank [19], SIFRank [4] and AttentionRank [5].These baselines all generate candidates using noun phrases without any additional steps. We used PKE to run SingleRank, RAKE, and TopicRank. The published GitHub code of YAKE!, PositionRank, EmbedRank, SIFRank and AttentionRank were used to produce the results on the selected

https://github.com/boudinfl/pke
https://github.com/LIAAD/yake
https://github.com/ymym3412/position-rank
https://github.com/swisscom/ai-research-keyphraseextraction
https://github.com/sunyilgdx/SIFRank
https://github.com/hd10-iupui/AttentionRank

TABLE II
MODEL COMPARISION WITH PRECISION(P), RECALL(R), AND F-SCORE(F1) @5, @10, @15 ON THREE BENCHMARK DATASETS. N IS THE NUMBER EXTRACTED FROM A SINGLE DOCUMENT BY THE MODELS. THE BEST PERFORMANCES ARE BOLD.

N	Method	Inspec			SemEval2017			SemEval2010		
		P	R	F1	P	R	F1	P	R	F1
5	**TF-IDF**	16.71	8.51	11.28	28.31	8.18	12.69	14.93	4.72	7.17
	YAKE!	25.04	11	15.29	24.79	7.96	12.05	16.87	5.65	8.46
	TopicRank	27.4	11.93	16.62	38.13	11.02	17.1	10.37	3.52	5.26
	PositionRank	29.8	12.15	17.26	40.65	11.75	18.23	5.16	1.62	2.47
	SingleRank	30.26	12.24	17.43	40.57	12.6	19.23	2.33	1.41	1.76
	EmbedRank	33.77	12.43	18.17	44.72	12.93	20.06	3.29	1.1	1.65
	AttentionRank	35.44	12.72	18.72	45.27	13.15	20.38	19.51	6.3	9.52
	SIFRank	**39.64**	13.83	20.51	45.16	13.23	20.46	11.44	3.83	5.74
	Ours	39.04	**14.01**	**20.61**	**47.30**	**13.74**	**21.30**	**22.22**	**7.18**	**10.85**
10	**TF-IDF**	13.76	14.01	13.88	22.19	12.83	16.26	13.18	9.59	11.1
	YAKE!	19.48	16.67	17.97	23.33	14.86	18.16	14.94	10	11.98
	TopicRank	27.11	22.27	24.45	30.87	17.84	22.61	9.26	6.2	7.43
	PositionRank	28.04	23.25	25.42	33.1	20.2	25.09	4.61	3.05	3.67
	SingleRank	28.32	23.43	25.64	35.25	20.38	25.83	2.23	2.53	2.37
	EmbedRank	29.97	22.3	25.57	37.48	23.21	28.67	3.58	2.34	2.83
	AttentionRank	31.47	23.15	26.68	39.66	23.03	29.14	16.83	10.87	13.21
	SIFRank	**35.89**	**24.77**	**29.31**	40.31	23.32	29.55	7.82	5.18	6.23
	Ours	34.06	24.25	28.33	**41.52**	**24.12**	**30.52**	**19.18**	**12.39**	**15.05**
15	**TF-IDF**	11.44	17.47	13.83	18.01	15.62	16.73	12.16	11.39	11.76
	YAKE!	17.12	21.7	19.14	21.41	20.07	20.72	12.87	12.86	12.86
	TopicRank	24.09	29.04	26.33	26.85	23.17	24.87	7.98	8.06	8.02
	PositionRank	24.59	28.53	26.41	29	26.5	27.69	4.15	4.02	4.08
	SingleRank	27.34	27.26	27.3	32.95	28.48	30.55	2.62	4.37	3.28
	EmbedRank	26.41	30.2	28.18	34.68	29.61	31.95	3.65	3.63	3.64
	AttentionRank	28.43	29.09	28.76	35.26	30.64	32.79	14.24	13.79	14.01
	SIFRank	**30.84**	**30.98**	**30.91**	35.90	31.10	33.33	6.20	6.11	6.15
	Ours	29.53	30.57	30.04	**37.05**	**32.25**	**34.48**	**16.27**	**15.76**	**16.01**

datasets. It is worth noting that the produced results of the baselines are slightly higher or lower than the results presented in the original papers.

C. Experiment Settings

In the experiment, we use "bert-base-uncased" as our pre-trained model. Our model is optimized using Adam with 1e-5 learning rate and 8 batch sizes. We use maximum sequence length 512. The number of negative samples is 2. For all datasets, we set the linear combination ratio β to be 0.5 for Inspec and 0.8 for SemEval2017 and SemEval2010. For the baseline methods, the parameters published on the corresponding GitHub were used. All of the models are implemented under PyTorch running on 2 NVIDIA Tesla T4 GPUs.

D. Results

Table II shows the results of Precision, Recall and F1 @5, 10, and 15 using our model and baseline models on three datasets.

Short document. The results show that the embedding-based methods, including our model, perform better than the statistic-based (TF-IDF and YAKE!) and graph-based algorithms (SingleRank, TopicRank, and PositionRank) on short document sets (Inspec and SemEval 2017). Statistic-based and graph-based unsupervised methods, despite their simplicity, do not perform as well as other methods on short documents, for which semantic information is assumed to be very important.

SIFRank performs slightly better than our model on Inspec. It works better than our model when K is set to 10 or 15.

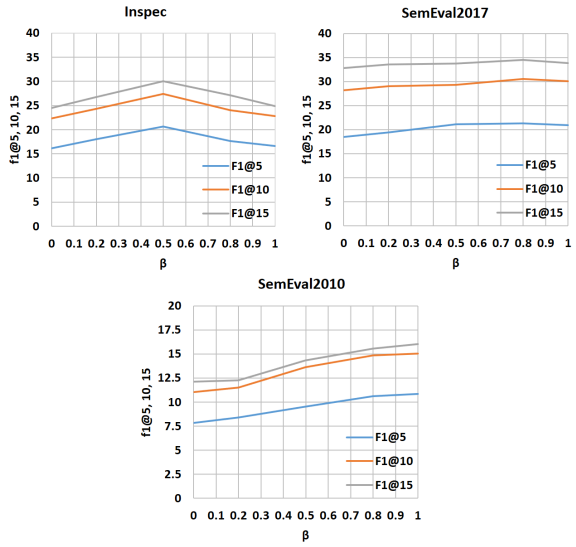

Fig. 3. Evaluation of the Linear Proportions' Impact on Performance.

Nevertheless, our model has a slightly better F1 than SIFRank and other baselines when the top 5 candidates are used for evaluation. Moreover, our model outperforms SIFRank on SemEval2017 dataset. It shows that our method performs competitively with SIFRank.

Long document. Our method shows advantage on long document set SemEval2010. The F1 value is at least 1.3% better than the highest baseline. Statistic-based methods achieve prominent results than other baselines on long documents. This may indicate that for long documents, statistical features such as word frequency and inverse document frequency are more important for the selection of keyphrases. Existing graph-based methods and embedding-based methods have difficulty in capturing these features, which can be well solved in our method.

E. Impact of hyperparameters

Our model linearly integrates the doc-phrase attention value and the accumulated self-attention value to measure the importance of a candidate phrase. We study the influence of the two modules by adjusting β (in Equation 12) from 0 to 1. Figure 3 shows that the best ratio is different for different datasets.

For short document datasets such as Inspec and SemEval2017, the addition of both parts of the attention values can improve the model performance. Specifically, for dataset Inspec, F1 value is highest when β is round 0.5. For SemEval2017, the best performance can be achieved when β is set to 0.5, 0.8 and 0.8. However, the contribution of accumulated self-attention value is higher than doc-phrase attention value for long ducument dataset-SemEval2010. When β is set to 1, the model achieves the best performance, which means only accumulated self-attention value is needed to find the keyphrases.

We consider that the accumulated self-attention module captures the repetition of the keyphrases implicitly through

the self-attention weights accumulation over the document. However, for short document dataset like Inspec, the doc-phrase attention value has more impact. Since there are only a few sentences in a document, the repetition of the phrases is low. Nonetheless, the contextual relevance among keyphrases and sentences and documents still needs to be emphasized.

VI. Conclusion

This paper proposes a contrastive learning-based semantic representation task to enhance pre-trained language model for unsupervised keyphrase extraction, which takes advantage of triple loss to combine the target document, joint information of keyphrase set, and other documents in semantic space. We utilize a doc-phrase attention module and an accumulated self-attention module to rank candidate phrases. The doc-phrase attention is designed to measure the importance between candidate phrases. The accumulated self-attention module aims to determine the importance of a candidate phrase in the context of the document. We compared the proposed model with eight strong baselines on three benchmark datasets, including two short document datasets and one long document dataset. Our model gains a better or competitive F1@5, 10, and 15 on all datasets. The ablation study shows that accumulated self-attention has a higher contribution to the long document set. The linear integration of the two attention modules shows the best results for short documents. In conclusion, our model is an efficient and robust unsupervised method for keyphrase extraction task, which regards the keyphrase set as a whole and fully leverages the semantic information from keyphrase set and document.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (grant No.2021YFB3100600), the Strategic Priority Research Program of Chinese Academy of Sciences (grant No.XDC02040400) and the Youth Innovation Promotion Association of CAS (Grant No. 2021153).

References

[1] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2014, pp. 1262–1273.

[2] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, Jun. 2018, pp. 2227–2237.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[4] Y. Sun, H. Qiu, Y. Zheng, Z. Wang, and C. Zhang, "Sifrank: a new baseline for unsupervised keyphrase extraction based on pre-trained language model," *IEEE Access*, vol. 8, pp. 10 896–10 906, 2020.

[5] H. Ding and X. Luo, "Attentionrank: Unsupervised keyphrase extraction using self and cross attentions," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 1919–1928.

[6] S. Sun, Z. Liu, C. Xiong, Z. Liu, and J. Bao, "Capturing global informativeness in open domain keyphrase extraction," in *CCF International Conference on Natural Language Processing and Chinese Computing*. Springer, 2021, pp. 275–287.

[7] H. Tian, C. Gao, X. Xiao, H. Liu, B. He, H. Wu, H. Wang, and F. Wu, "Skep: Sentiment knowledge enhanced pre-training for sentiment analysis," *arXiv preprint arXiv:2005.05635*, 2020.

[8] H. Fang, S. Wang, M. Zhou, J. Ding, and P. Xie, "Cert: Contrastive self-supervised learning for language understanding," *arXiv preprint arXiv:2005.12766*, 2020.

[9] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[10] R. Alzaidy, C. Caragea, and C. L. Giles, "Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents," in *The world wide web conference*, 2019, pp. 2551–2557.

[11] D. Sahrawat, D. Mahata, M. Kulkarni, H. Zhang, R. Gosangi, A. Stent, A. Sharma, Y. Kumar, R. R. Shah, and R. Zimmermann, "Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings," *arXiv preprint arXiv:1910.08840*, 2019.

[12] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi, "Deep keyphrase generation," *arXiv preprint arXiv:1704.06879*, 2017.

[13] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," *arXiv preprint arXiv:1603.06393*, 2016.

[14] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, "Yake! collection-independent automatic keyword extractor," in *European Conference on Information Retrieval*. Springer, 2018, pp. 806–810.

[15] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

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

[17] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge." in *AAAI*, vol. 8, 2008, pp. 855–860.

[18] A. Bougouin, F. Boudin, and B. Daille, "Topicrank: Graph-based topic ranking for keyphrase extraction," in *International joint conference on natural language processing (IJCNLP)*, 2013, pp. 543–551.

[19] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi, "Simple unsupervised keyphrase extraction using sentence embeddings," *arXiv preprint arXiv:1801.04470*, 2018.

[20] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," in *International conference on learning representations*, 2017.

[21] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2021.

[22] J. M. Giorgi, O. Nitski, G. D. Bader, and B. Wang, "Declutr: Deep contrastive learning for unsupervised textual representations," *arXiv preprint arXiv:2006.03659*, 2020.

[23] B. Gunel, J. Du, A. Conneau, and V. Stoyanov, "Supervised contrastive learning for pre-trained language model fine-tuning," *arXiv preprint arXiv:2011.01403*, 2020.

[24] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," *arXiv preprint arXiv:2104.08821*, 2021.

[25] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," *arXiv preprint arXiv:1906.04341*, 2019.

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

[27] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, "Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications," *arXiv preprint arXiv:1704.02853*, 2017.

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

[29] C. Florescu and C. Caragea, "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)*, 2017, pp. 1105–1115.