Autoregressive Language Model for Zero-shot Constrained Keyphrase Generation

Anonymous ACL submission

Abstract

Recently, most of the state-of-the-art keyphrase prediction models are based on a supervised generative model. It shows significantly better than before. Nevertheless, it still faces domain robustness and building datasets on high-resource. To overcome these limitations, unsupervised methods have also been critical and studied. We analyzed it also have a defect in a necessary process, which extracts candidates beforehand selecting keyphrase. As not including various forms of phrases, we note that the unsupervised method can't ensure oracle keyphrase. In this paper, we present zero-shot constrained keyphrase generation by leveraging a large-scale language model. To generate diverse keyphrases, we explore controlling a phrase during the generation. Finally, we evaluate benchmark datasets of the scholar domain. It results in better performances than unsupervised methods on several datasets without going through the candidate extraction stage. For domain robustness, we evaluate out-of-domain DUC compare with NUS. Since our method doesn't fine-tune to a corpus of a specific domain, it's better than supervised methods based on Sequence-to-Sequence.

1 Introduction

Natural Language Processing (NLP) research has been intensive and remarkable progress recently with a large-scale language model. Autoregressive language models (ALM) (Radford et al., 2019; Raffel et al., 2019; Brown et al., 2020; Lewis et al., 2020), such as GPT-2, T5 and GPT-3, show improved performance in zero-shot and few-shot in various NLP's tasks with the prompt-based multi-task learning (McCann et al., 2018). These language models are capable of another task by giving the context from some examples connected with the source text. But since the input length limited by each model is not scalable (Schick and Schütze, 2020). Especially, it's difficult in the keyphrase prediction task to feed the target source text into the model with not-short documents as examples.

Due to this limitation, we focus on zero-shot rather than few-shot to solve problems of the existing keyphrase prediction system with a challenging approach.

In this work, we present a zero-shot constrained keyphrase generation method. Our proposed method aims to generate a keyphrase from a large-scale language model by finding out the prompt that describes the task from real-world structured documents. With several benchmark datasets, we compare them with existing supervised and unsupervised methods. We also evaluate out-of-domain whether the proposed method is robust.

2 Related Work

Unsupervised Methods Heuristic (Witten et al., 1999; Liu et al., 2011) and statistical (Ramos, 2003; El-Beltagy and Rafea, 2010) methods of extracting candidates are traditionally used for present keyphrases. Candidates are selected by using Part-Of-Speech (POS) tags or spans consisting of multiple words. Recently, embedding-based methods (Bennani-Smires et al., 2018; Sun et al., 2020) has also been studied along with graph-based methods (Mihalcea and Tarau, 2004; Wan and Xiao, 2008; Bougouin et al., 2013; Florescu and Caragea, 2017).

(Bennani-Smires et al., 2018) mentions low performances with boundless candidates and topics in the long document. We define that problem as difficult to ensure oracle keyphrases during processing. It's described in Section 4.

Supervised Methods On the other hand, supervised methods are free from forms of the phrase, and can predict the absent keyphrase of out-of-document (Meng et al., 2017; Zhao and Zhang, 2019; Chen et al., 2019b,a; Liu et al., 2020). (Meng et al., 2017) successfully settles a DNN-based gen-

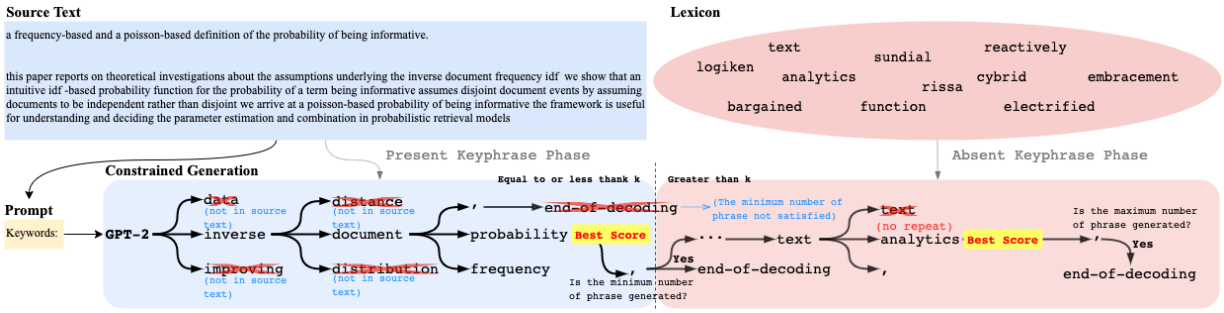

Figure 1: The proposed method's flow for the constrained keyphrase generation during 2-phases.

erative model for the keyphrase generation with Copy Mechanism (Gu et al., 2016).

In the following studies, the keyphrase generation is studied from various perspectives. Most recent (Meng et al., 2020) studied related factors for filling the gap of performances from differences in model design. With the comprehensive comparison, we note that supervised methods fit specific domain isn't robust despite the using huge datasets. This problem is also mentioned in (Gallina et al., 2019).

3 Methods

Manual Prompt Design A keyphrase sometimes appears sequentially with "Keywords" and "Index Terms" in real-world structured documents like papers and news. As the prompt connected with the special token ':', a language model generates from conditional distribution $P(y|x, T)$ using the prompt as tokens T to describe the task. We finally use "Keywords" as the final prompt, which has higher performance for generating keyphrases.

Constrained Keyphrase Generation Our work is inspired by (Pascual et al., 2020). To satisfy lexical constraints during the decoding process, a language model forces to generate a certain word. To generate a suitable keyphrase, a strategy is devised by generating keyphrases in a lexicon. It's shown as a graphical summary in Figure 1.

With Byte Pair Encoding (BPE) (Sennrich et al., 2015) tokenizer, we build a set V_s composed of tokens of phrases. During 2-phases according to the specified number k, 1. To focus on a present keyphrase, only a set V_p composed of tokens that appeared in the source text is allowed when the number of generated phrases are equal to or less than k. 2. Only V_s is allowed when the number of generated phrases is greater than k. To encourage a diverse keyphrase, the set V_{prefix} composed of the first prefix's tokens of each phrase generated at the previous steps is concurrently constrained.

According to number t of delimiter token S_{idx}, which separates each phrase, allowed tokens $V_{allowed}$ are given by:

$$V_{allowed} = \begin{cases} V_p & t \leq k \\ V_s - V_{prefix} & t > k \end{cases} \quad (1)$$

The number of minimum and maximum phrases, which are P_{min} and P_{max} respectively, are controlled by the end-of-decoding token E_{idx} during the generation. $V_{allowed}$ according to the number of the currently generated t is given by:

$$V_{allowed} = \begin{cases} V_{allowed} - E_{idx} & t \leq P_{min} \\ E_{idx} & t > P_{max} \end{cases} \quad (2)$$

If the token generated at the previous step $t-1$ is a plural noun, S_{idx} is generated by force to complete the concatenation of the phrase on the next step.

To generate not verbose keyphrase, location prepositions[1], such as "in", "with", are additionally constrained at all steps.

Phrase Control This section describes several phrase control methods that alleviate the problem that ALM repeats the same word (Holtzman et al., 2019) and encourage diverse keyphrases.

For handling the phrase, generated tokens are split as phrase-level by S_{idx}. Then the last phrase P_{Cur} currently being generated is controlled by tokens computed from previous phrases P_{Gen}. For alleviating a language model stuck when repeating copiously, tokens of the last phrase in P_{Gen} and tokens of P_{Cur} are constrained. For dealing with diversity, we use exhausting vocabulary and

[1] https://www.cambridge.org/kr/academic/subjects/languages-linguistics/grammar-and-syntax/cambridge-grammar-english-language

Dataset	Domain	#Doc	#AvgTok	#Present	#Absent	#Candidate	#Gain	%Gain
KP20k	Scholar	19982	156.4	65936	39004	640891/753835/646925	31253/33495/28945	47.4/50.8/43.9
Inspec	Scholar	500	121.9	3836	1064	12948/15238/12983	2424/2470/2351	63.2/64.4/61.3
NUS	Scholar	211	164.4	1106	1173	6975/8347/7084	585/602/527	52.9/54.5/47.7
Krapivin	Scholar	460	159.3	1473	1163	14515/17181/14765	799/811/776	54.3/55.1/52.7
DUC	News	308	686.0	2314	165	38724/45996/39091	1779/1832/1733	76.9/79.2/74.9

Table 1: Statistics of 4 scholars and 1 news test dataset. #AvgTok is the average number of words in documents, and #Present and #Absent are each number of present and absent keyphrases. #Candidate denotes the number according to three forms of the phrase: the first allows only noun phrase, the second allows also the gerund or present participle, and the third allows up to the past participle. #Gain and %Gain are the number and ratio of candidates that correspond to oracle keyphrases respectively.

expanding the phrase. As the last word connected with modifier is significant determining semantics of the phrase, used tokens of the last word of each phrase in P_{Gen} are constrained. And if P_{Cur} was used, S_{idx} and E_{idx} are constrained for expanding phrase to not duplicated.

Preferred N-gram We manipulate the generation of a concise or lengthy keyphrase. The initial preferred size of n-gram β and weight of penalty α, the negative log probability $\log \widetilde{P}_s$ of S_{idx} adjusted by the penalty ρ computed by the size of the n-gram of P_{Cur}, which is n, is given by:

$$\log \widetilde{P}_s\left(y|x,T\right) = \log P_s\left(y|x,T\right) * \frac{1}{\rho} \quad (3)$$

$$\rho = \frac{(1+n)^\alpha}{(1+\beta)^\alpha} \quad (4)$$

If n is less than β, the generation of a longer phrase is encouraged, and if it is greater than β, S_{idx} is encouraged.

4 Experiments and Results

Datasets Our proposed method is evaluated on 4 benchmark datasets, namely KP20k (Meng et al., 2017), Inspec (Hulth, 2003), Krapivin (Krapivin et al., 2009) and NUS (Nguyen and Kan, 2007), consisting of scientific publications commonly used in the keyphrase prediction. And we use additionally one news dataset is used as the out-of-domain, namely DUC (Wan and Xiao, 2008). Scholar datasets include title and abstract only (Meng et al., 2017). Table 1 shows statistics of documents in each dataset and of a present and absent keyphrase.

As mentioned in Section 2, statistics of candidates that can be ensured according to each noun and participial phrase using CoreNLP[2] as POS tagger are also included. Existing unsupervised extractive methods can obtain 47.4 (%Gain) less than half of oracle keyphrases in KP20k, and about half of NUS and Krapivin can be obtained, compared with DUC and Inspec. As unsupervised methods encounter the upper limit, these are difficult to achieve a perfect score.

Details and Performance Comparison V_s described in Section 3 is composed of tokens of phrases appearing in KP20k's training dataset, which is V_{KP20k}. We use a beam search with width 6 during the inference, and macro average F1 score as evaluation metrics proposed in (Bennani-Smires et al., 2018). We build a benchmark on several datasets comparing with traditional and recent methods in supervised (Meng et al., 2017; Chen et al., 2019b,a; Yuan et al., 2020; Meng et al., 2020) and unsupervised (Mihalcea and Tarau, 2004; Boudin, 2018; Bennani-Smires et al., 2018) methods. Table 2 shows performances on scholar datasets. It shows that larger language model (GPT-2 xl) improves F_1@10 from 19.0 to 25.3 compared to smaller model (GPT-2 base).

To verify that our method has domain robustness, we compare our method to supervised methods. We evaluate additionally DUC of out-of-domain, consisting of news articles. Also, since V_{KP20k} is composed of phrases in the scholar domain, we build additionally V_{DUC} composed of phrases appearing in DUC. Table 3 shows that our approach is more robust to out-of-domain, compared with S2S-based Recurrent Neural Network (Gu et al., 2016) and Transformer (Vaswani et al., 2017; Meng et al., 2020) incorporated by Copy Mechanism that is trained on KP20k.

Existing unsupervised methods achieve lower performance on Krapivin and NUS compared to Inspec. They tend to show better performance on the dataset with the high ratio of oracle keyphrases

[2] https://github.com/stanfordnlp/CoreNLP

	KP20k			Inspec			Krapivin			NUS		
Supervised Methods												
	F_1@5	F_1@10	F_1@O	F_1@5	F_1@10	F_1@O	F_1@5	F_1@10	F_1@O	F_1@5	F_1@10	F_1@O
CopyRNN (Meng et al., 2017)	31.7	27.3	33.5	24.4	28.9	29.0	30.5	26.6	32.5	37.6	35.2	40.6
TG-Net (Chen et al., 2019b)	37.2	**31.5**	-	31.5	38.1	-	34.9	**29.5**	-	40.6	37.0	-
KG-KE-KR-M (Chen et al., 2019a)	31.7	28.2	38.8	25.7	28.4	31.4	27.2	25.0	31.7	28.9	28.6	38.4
CatSeqD(RNN) (Yuan et al., 2020)	-	26.1	31.2	-	**38.7**	38.8	-	26.9	33.5	-	36.6	39.2
CatSeqD(TRANS) (Meng et al., 2020)	-	29.0	36.2	-	36.6	36.9	-	28.1	36.4	-	**37.3**	42.3
Unsupervised Methods												
TextRank (Mihalcea and Tarau, 2004)	6.6†	9.5†	-	14.7	15.2	-	7.4†	11.0†	-	9.5†	15.1†	-
Multipartite (Boudin, 2018)	16.0†	13.7†	-	25.7	30.0	-	15.4†	13.6†	-	22.7†	20.0†	-
EmbedRank d2v (Bennani-Smires et al., 2018)	-	-	-	31.5	**37.9**	-	-	-	-	2.3	3.5	-
EmbedRank s2v (Bennani-Smires et al., 2018)	10.2†	10.4†	-	29.8	37.0	-	11.7†	11.7†	-	14.8†	16.1†	-
GPT-2 base (Ours)	14.6	15.5	12.4	20.8	22.3	21.1	14.7	15.4	12.1	21.7	23.0	22.7
GPT-2 xl (Ours)	17.7	**19.4**	15.0	25.9	27.6	26.3	20.9	**22.4**	18.3	29.8	**32.0**	29.2

Table 2: Present keyphrase prediction results in 4 scholar datasets. The highest F_1@10 among supervised/unsupervised methods is marked bold in red/blue. And the performance with † represents it is evaluated by us using only title and abstract.

	DUC		NUS	
	F_1@5	F_1@10	F_1@5	F_1@10
RNN-O2S-KP20k (Meng et al., 2020)	11.9	16.0	37.3	36.6
TF-O2S-KP20k (Meng et al., 2020)	10.1	11.4	40.1	**37.3**
V_{KP20k}	15.3	18.0	29.8	32.0
V_{DUC}	17.0	**20.0**	29.1	31.4

Table 3: Performance comparison of NUS of scholar domain and DUC of out-of-domain.

	P@10	R@10	F_1@10
+All Constraint	34.4	**36.5**	32.0
-Minimum Number of Phrases	**39.9**	32.8	**32.2**
-Phrase Control	37.3	28.4	28.1
-Minimum Number of Phrases No Phrase Control	39.1	26.1	27.4
-All Constraints	39.8	21.3	25.1

Table 4: Performance comparison on NUS according to a combination of several constraints.

in candidates shown in Table 2. Since the proposed method doesn't go through the process of extracting candidates in advance, F_1@10 is improved on NUS by 12.0 compared to Multipartite (Boudin, 2018). And KP20k and Krapivin are evaluated by us to directly compare existing unsupervised methods with ours, the proposed method improves performances on them.

Ablation Study Table 4 shows performances on NUS according to several combinations of proposed constraints. R@10 is 15.2 higher with the constraint of phrase controls than without any constraints. If the minimum number of phrases isn't limited, performance is more precise. However, we evaluated final performances using all constraints for the diverse keyphrase generation. Examples for absent keyphrases are provided respectively in appendix A.

Performances per Preferred N-gram We fix $\alpha = 1.7$ and compare β according to integers from $[1, 5]$ range. Since the length of each phrase is different for annotators, it is significant to use a parameter to control it in the keyphrase generation. Figure 2 shows the Average F1 score with different values of beta on NUS dataset.

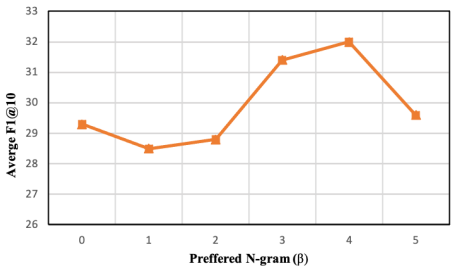

Figure 2: Performance comparison on NUS according to preferred N-gram.

5 Conclusion

In this work, we explore how to generate a keyphrase using ALM. The proposed method builds up the model aware of the keyphrase prediction task with a prompt found out from real-world structured documents and can generate diverse keyphrases using several constraints. The proposed method is novel compared to existing methods, and our results show improving the performance when utilizing more large language models. It shows that the improved effect is feasible by leveraging future language models with our method. Our method shows better domain robustness than supervised methods. Moreover, it overcomes an unsupervised method performance that can't ensure oracle keyphrases during the extraction process.

References

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

Florian Boudin. 2018. Unsupervised Keyphrase Extraction with Multipartite Graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 667–672, New Orleans, Louisiana. Association for Computational Linguistics.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and Others. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Wang Chen, Hou Pong Chan, Piji Li, Lidong Bing, and Irwin King. 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)*, pages 2846–2856, Minneapolis, Minnesota. Association for Computational Linguistics.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R Lyu. 2019b. Title-Guided Encoding for Keyphrase Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6268–6275.

Samhaa R El-Beltagy and Ahmed Rafea. 2010. KP-Miner: Participation in SemEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 190–193, Uppsala, Sweden. Association for Computational Linguistics.

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

Ygor Gallina, Florian Boudin, and Beatrice Daille. 2019. KPTimes: A large-scale dataset for keyphrase generation on news documents. *arXiv preprint arXiv:1911.12559*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O K Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Anette Hulth. 2003. Improved Automatic Keyword Extraction Given More Linguistic Knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223.

Mikalai Krapivin, Aliaksandr Autaeu, and Maurizio Marchese. 2009. Large dataset for keyphrases extraction.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics.

Rui Liu, Zheng Lin, and Weiping Wang. 2020. Keyphrase prediction with pre-trained language model. *arXiv preprint arXiv:2004.10462*.

Zhiyuan Liu, Xinxiong Chen, Yabin Zheng, and Maosong Sun. 2011. Automatic keyphrase extraction by bridging vocabulary gap. *CoNLL 2011 - Fifteenth Conference on Computational Natural Language Learning, Proceedings of the Conference*, (June):135–143.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Rui Meng, Xingdi Yuan, Tong Wang, Sanqiang Zhao, Adam Trischler, and Daqing He. 2020. An empirical study on neural keyphrase generation. *arXiv preprint arXiv:2009.10229*.

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

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

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, pages 317–326. Springer.

Damian Pascual, Beni Egressy, Florian Bolli, and Roger Wattenhofer. 2020. Directed Beam Search: Plug-and-Play Lexically Constrained Language Generation. *arXiv preprint arXiv:2012.15416*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and Others. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1:9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Juan Ramos. 2003. Using TF-IDF to Determine Word Relevance in Document Queries. *Proceedings of the first instructional conference on machine learning*, 242(1):29–48.

Timo Schick and Hinrich Schütze. 2020. It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. *arXiv preprint arXiv:2009.07118*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Yi Sun, Hangping Qiu, Yu Zheng, Zhongwei Wang, and Chaoran Zhang. 2020. SIFRank: A New Baseline for Unsupervised Keyphrase Extraction Based on Pre-Trained Language Model. *IEEE Access*, 8:10896–10906.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xiaojun Wan and Jianguo Xiao. 2008. Single Document Keyphrase Extraction Using Neighborhood Knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 8, pages 855–860.

Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. 1999. KEA: Practical Automatic Keyphrase Extraction. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, pages 254–255, New York, NY, USA. Association for Computing Machinery.

Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. 2020. One Size Does Not Fit All: Generating and Evaluating Variable Number of Keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7961–7975, Online. Association for Computational Linguistics.

Jing Zhao and Yuxiang Zhang. 2019. Incorporating Linguistic Constraints into Keyphrase Generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5224–5233, Florence, Italy. Association for Computational Linguistics.

A Examples

Title: Learning Query Languages of **Web Interfaces**. **Abstract:** This paper studies the problem of **automatic acquisition** of the **query languages** supported by a Web information resource. We describe a system that automatically probes the search interface of a resource with a set of test queries and analyses the returned pages to recognize supported **query operators**. The **automatic acquisition** assumes the availability of the number of matches the resource returns for a submitted query. The match numbers are used to train a **learning** system and to generate classification rules that recognize the **query operators** supported by a provider and their syntactic encodings. These classification rules are employed during the automatic probing of new providers to determine **query operators** they support. We report on results of experiments with a set of real **Web resources**.
Oracle Keyphrases: query operators, automatic acquisition, learning, hidden web, search interface, web resources, machine learning, search engine, query languages, hidden web, web interfaces
Predicted Keyphrases: **query operators**, learning system, web information resources, **query language**, **automatic acquisition**, ..., web information resource, web search interface, **search engines**

Table 5: Example 1. Keyphrases in the document are bold. Present/Absent Keyphrases by the proposed method are marked bold in blue/red.

Title: Web **Taxonomy Integration** through Co-**Bootstrapping**. **Abstract:** We address the problem of integrating objects from a source taxonomy into a master taxonomy. This problem is not only currently pervasive on the web, but also important to the emerging **semantic web**. A straightforward approach to automating this process would be to learn a classifier that can classify objects from the source taxonomy into categories of the master taxonomy. The key insight is that the availability of the source taxonomy data could be helpful to build better classifiers for the master taxonomy if their categorizations have some semantic overlap. In this paper, we propose a new approach, co-**bootstrapping**, to enhance the **classification** by exploiting such implicit knowledge. Our experiments with real-world web data show substantial improvements in the performance of **taxonomy integration**.
Oracle Keyphrases: taxonomy integration, bootstrapping, semantic web, classification, ontology mapping, machine learning, boosting
Predicted Keyphrases: semantic search, **semantic webs**, taxonomy, web classification, ..., **machine learning**

Table 6: Example 2. Keyphrases in the document are bold. Present/Absent Keyphrases by the proposed method are marked bold in blue/red.