Keyphrase Generation: A Text Summarization Struggle

Erion Çano
Institute of Formal and Applied Linguistics, Charles University, Prague, Czech Republic
cano@ufal.mff.cuni.cz

Ondřej Bojar
Institute of Formal and Applied Linguistics, Charles University, Prague, Czech Republic
bojar@ufal.mff.cuni.cz

Abstract

Authors' keyphrases assigned to scientific articles are essential for recognizing content and topic aspects. Most of the proposed supervised and unsupervised methods for keyphrase generation are unable to produce terms that are valuable but do not appear in the text. In this paper, we explore the possibility of considering the keyphrase string as an abstractive summary of the title and the abstract. First, we collect, process and release a large dataset of scientific paper metadata that contains 2.2 million records. Then we experiment with popular text summarization neural architectures. Despite using advanced deep learning models, large quantities of data and many days of computation, our systematic evaluation on four test datasets reveals that the explored text summarization methods could not produce better keyphrases than the simpler unsupervised methods, or the existing supervised ones.

1 Introduction

A valuable concept for searching and categorizing scientific papers in digital libraries is the *keyphrase* (we use *keyphrase* and *keyword* interchangeably), a short set of one or few words that represent concepts. Scientific articles are commonly annotated with keyphrases based on taxonomies of concepts and the authors' judgment. Finding keyphrases that best describe the contents of a document is thus essential and rewarding.

Most of the proposed keyphrase extraction solutions tend to be unsupervised (Florescu and Caragea, 2017; Nguyen and Luong, 2010; Rose et al., 2010; Bougouin et al., 2013; Campos et al., 2018) and generate terms by selecting the most appropriate candidates, ranking the candidates based on several features and finally returning the top N. Another way is to utilize datasets of texts and keywords for training supervised models with linguistic or other features to predict if candidates are keywords or not (Witten et al., 1999; Turney, 2000; Medelyan, 2009; Hulth, 2003).

All above methods propose N keyphrases for each article which are joined together with "," (or other separator like ";") to form the *keyphrase string* of that article. They suffer from various problems or discrepancies. First, they are unable to find an optimal value for N and require it as a preset parameter. Furthermore, semantic and syntactic properties of article phrases are analyzed separately. The meaning of paragraphs, sections or entire document is thus missed. Lastly, only phrases that do appear in the article are returned. Meng et al. (2017) recently proposed a deep supervised keyphrase generation solution trained on a big dataset. It successfully solves the last two problems above, but not the first one.

Motivated by recent advances in neural machine translation and abstractive text summarization (Vaswani et al., 2017; Foster et al., 2018; Rush et al., 2015; See et al., 2017), in this paper, we explore the possibility of considering keyphrase generation as an abstractive text summarization task. Instead of generating keywords one by one and linking them to form the keyphrase string, we consider the later as an abstractive summary of the concatenated paper title and abstract. Different recently-proposed text summarization architectures are tried on four test datasets of article keyphrases (Tanti et al., 2017; Rush et al., 2015; See et al., 2017). We trained them with a newly created dataset of 2.2 million article titles, abstracts and keyphrase strings that we processed and released.[1]

The selected text summarization models are compared with popular unsupervised and supervised methods using ROUGE (Lin, 2004) and full-match F_1 metrics. The results show that though

[1]http://hdl.handle.net/11234/1-2943

trained with large data quantities for many days, the tried text summarization methods could not produce better keywords than the existing supervised or deep supervised predictive models. In our opinion, a possible explanation for this is the fact that the title and the abstract may not carry sufficient topical information about the article, even when joined together. In contrast, when assigning keywords annotations of their paper, authors are highly influenced by the topic aspects of it.

This paper carries several contributions, despite the fact that no progressive result scores were reached. It is the first work that considers keyphrase generation as an abstractive text summarization task. We produced a large dataset of article titles, abstracts, and keywords that can be used for keyword generation, text summarization or similar purposes. Finally, we evaluated the performance of different neural network architectures on summarization of article keyword strings, comparing them with popular unsupervised methods.

2 Scientific Paper Datasets

Because of the open source and open data initiatives, many public datasets from various domains can be found online (Çano and Morisio, 2015). Among the several collections of scientific articles, some of them have gained considerable popularity in research literature. In Meng et al. (2017), we found a recent and big collection of 20K paper abstracts and keyphrases. These metadata belong to articles of computer science from ACM Digital Library, ScienceDirect, and Web of Science. In Hulth (2003), we found a collection of 2000 (1500 for train/val and 500 for testing) abstracts in English, together with titles and authors' keywords. The corresponding articles were published from 1998 to 2002 and belong to the discipline of *Information Technology*. Furthermore, Krapivin et al. (2010) released a dataset of 2000 (1600 for train/val and 400 for testing) full articles published by ACM from 2003 to 2005 in Computer Science domain. More information about similar keyphrase data collections or other available resources can be found in Hasan and Ng (2014) and in online repositories.[2]

Regarding text summarization, some of the most popular datasets are: DUC-2004 [3] mainly used for testing, English Gigaword (Napoles et al., 2012), CNN/Daily Mail described in Section 4.3 of (Nallapati et al., 2016) and Newsroom, a heterogeneous bundle of news articles described in Grusky et al. (2018). These datasets are frequently used for the task of predicting titles from abstracts or short stories. However, no keyphrases are provided; they do not serve to our purpose. ArnetMiner is a recent attempt to crawl scientific paper data from academic networks (Tang et al., 2008). The system extracts profiles of researchers from digital resources and integrates their data in a common network. A spin-off is the Open Academic Graph (OAG) data collection (Sinha et al., 2015).

Attribute	Train	Val	Test	Fullset
Records	2M	100K	100K	2.2M
Keyphrases	12M	575K	870K	13.4M
Title tokens	24M	1.3M	1.6M	27M
Abstract tokens	441M	21M	37M	499M
Av. Keyphrase	6	5.8	8.7	6.1
Av. Title	12.1	12.8	15.9	12.3
Av. Abstract	220	211	372	227

Table 1: Statistics of OAGK dataset

To produce a usable collection for our purpose, we started from OAG. We extracted *title*, *abstract* and *keywords*. The list of keywords was transformed into a comma-separated string and a language identifier was used to remove records that were not in English. Abstracts and titles were lowercased, and Stanford CoreNLP tokenizer was used for tokenizing. Short records of fewer than 20 tokens in the abstract, 2 tokens in the title and 2 tokens in the keywords were removed. For the test portion, we selected documents of at least 27, 3 and 2 tokens in each field. Data preprocessing stopped here for the release version (no symbol filtering), given that many researchers want to filter text in their own way. This new dataset named OAGK can be used for both text summarization (predicting title from abstract) and keyphrase extraction (unsupervised, supervised or deep supervised) tasks. Some rounded measures about each set of released data are presented in Table 1.

3 Keyphrase Extraction Strategies

3.1 Unsupervised and Supervised Methods

TopicRank is an extractive method that creates topic clusters using the graph of terms and phrases (Bougouin et al., 2013). Obtained topics are then ranked according to their importance in the document. Finally, keyphrases are extracted by pick-

[2]https://github.com/LIAAD/KeywordExtractor-Datasets
[3]https://duc.nist.gov/duc2004/

ing one candidate from each of the most important topics. A more recent, unsupervised and feature-based method for keyphrase extraction is YAKE! (Campos et al., 2018). It heuristically combines features like *casing*, *word position* or *word frequency* to generate an aggregate score for each phrase and uses it to select the best candidates.

One of the first supervised methods is KEA described by Witten et al. (1999). It extracts those candidate phrases from the document that have good chances to be keywords. Several features like *TF-IDF* are computed for each candidate phrase during training. In the end, Naïve Bayes algorithm is used to decide if a candidate is a keyword or not (binary classification). An improvement and generalization of KEA is MAUI (Medelyan, 2009). Additional features are computed, and bagged decision trees are used instead of Naïve Bayes. The author reports significant performance improvements in precision, recall and F_1 scores.

The above keyphrase extraction methods and others like Florescu and Caragea (2017) or Nguyen and Luong (2010) reveal various problems. First, they are not able to find an optimal value for N (number of keywords to generate for an article) based on article contents and require it as a preset parameter. Second, the semantic and syntactic properties of article phrases (considered as candidate keywords) are analyzed separately. The meaning of longer text units like paragraphs or entire abstract/paper is missed. Third, only phrases that do appear in the paper are returned. In practice, authors do often assign words that are not part of their article.

Meng et al. (2017) overcome the second and third problem using an encoder-decoder model (COPYRNN) with a bidirectional Gated Recurrent Unit (GRU) and a forward GRU with attention. They train it on a datasets of hundred thousands of samples, consisting of abstract-keyword (one keyword only) pairs. The model is entirely data-driven and can produce terms that may not appear in the document. It still produces one keyword at a time, requiring N (first problem) as parameter to create the full keyphrase string.

3.2 Text Summarization Methods

To overcome the three problems mentioned in Section 3.1, we explore abstractive text summarization models proposed in the literature, trained with article abstracts and titles as sources and keyword strings as targets. They are expected to learn and paraphrase over entire source text and produce a summary in the form of a keyphrase string with no need for extra parameters. They should also introduce new words that do not appear in the abstract. Two simple encoder-decoder variants based on LSTMs are described in Figure 3 of Tanti et al. (2017). MERGE (Figure 3.a) encodes input and the current summary independently and merges them in a joint representation which is later decoded to predict the next summary token. INJECT model (Figure 3.b) on the other hand injects the source document context representation to the encoding part of the current summary before the decoding operation is performed.

ABS is presented in Figure 3.a of Rush et al. (2015). The encoder (Figure 3.b) takes in the input text and a learned soft alignment between the input and the summary, producing the context vector. This soft alignment is the attention mechanism (Bahdanau et al., 2014). To generate the summary words, Rush et al. apply a beam-search decoder with a window of K candidate words in each position of the summary.

Pointer-Generator network (POINTCOV) depicted in Figure 3 of See et al. (2017) is similar to ABS. It is composed of an attention-based encoder that produces the context vector. The decoder is extended with a pointer-generator model that computes a probability p_{gen} from the context vector, the decoder states, and the decoder output. That probability is used as a switch to decide if the next word is to be generated or copied from the input. This model is thus a compromise between abstractive and extractive (copying words from input) models. Another extension is the coverage mechanism for avoiding word repetitions in the summary, a common problem of encoder-decoder summarizers (Tu et al., 2016).

4 Results

We performed experiments with the unsupervised and supervised methods of Section 3 on the first three datasets of Section 2 and on OAGK. All supervised methods were trained with the 2M records of OAGK train part. An exception was MAUI which could be trained on 25K records at most (memory limitation). In addition to the processing steps of Section 2, we further replaced digit symbols with # and limited source and tar-

	Hulth (500)		Krapivin (400)		Meng (20K)		OAGK (100K)	
Method	**F_1@5**	**F_1@7**	**F_1@5**	**F_1@7**	**F_1@5**	**F_1@7**	**F_1@5**	**F_1@7**
YAKE!	19.35	21.47	17.98	17.4	17.11	15.19	15.24	14.57
TOPICRANK	16.5	20.44	6.93	6.92	11.93	11.72	11.9	12.08
MAUI	20.11	20.56	23.17	23.04	22.3	19.63	19.58	18.42
COPYRNN	**29.2**	**33.6**	**30.2**	**25.2**	**32.8**	**25.5**	**33.06**	**31.92**
MERGE	6.85	6.86	4.92	4.93	8.75	8.76	11.12	13.39
INJECT	6.09	6.08	4.1	4.11	8.09	8.09	9.61	11.22
ABS	14.75	14.82	10.24	10.29	12.17	12.09	14.54	14.57
POINTCOV	22.19	21.55	19.87	20.03	20.45	20.89	22.72	21.49

Table 2: Full-match scores of predicted keyphrases by various methods

	Hulth (500)		Krapivin (400)		Meng (20K)		OAGK (100K)	
Method	R_1F_1	R_LF_1	R_1F_1	R_LF_1	R_1F_1	R_LF_1	R_1F_1	R_LF_1
YAKE!	37.48	24.83	26.19	18.57	26.47	17.36	20.38	14.54
TOPICRANK	32.0	20.36	14.08	11.47	21.68	15.94	17.46	13.28
MAUI	36.88	27.16	28.29	23.74	34.33	28.12	32.16	25.09
COPYRNN	**44.58**	**35.24**	**39.73**	**30.29**	**42.93**	34.62	**43.54**	**36.09**
MERGE	15.19	9.45	9.66	7.14	16.53	12.31	17.3	14.43
INJECT	14.15	8.81	9.58	6.79	15.6	11.21	14.3	11.08
ABS	27.54	19.48	25.59	18.2	28.31	22.16	29.05	25.77
POINTCOV	37.16	33.69	35.81	29.52	38.47	**35.06**	38.66	34.04

Table 3: Rouge scores of predicted keyphrases by various methods

get text lengths to 270 and 21 tokens, respectively. Vocabulary size was also limited to the 90K most frequent words.

The few parameters of the unsupervised methods (length and windows of candidate keyphrases for YAKE!, ranking strategy for TOPICRANK) were tuned using the validation part of each dataset. For the evaluation, we used F_1 score of full matches between predicted and authors' keywords. Given that the average number of keywords in the data is about 6, we computed F_1 scores on top 5 and top 7 returned keywords (**F_1@5, F_1@7**).

Before each comparison, both sets of terms were stemmed with Porter Stemmer and duplicates were removed. In the case of summarization models, keyphrases were extracted from their comma-separated summaries. We also computed ROUGE-1 and ROUGE-L F_1 scores (R_1F_1, R_LF_1) that are suitable for evaluating short summaries (Lin, 2004). The keywords obtained from the unsupervised methods were linked together to form the keyphrase string (assumed summary). This was later compared with the original keyphrase string of the authors.

Full-match results on each dataset are reported in Table 2. From the unsupervised models, we see that YAKE! is consistently better than TOPICRANK. The next two supervised models perform even better, with COPYRNN being discretely superior than MAUI.

Results of the four summarization models seem disappointing. MERGE and INJECT are the worst on every dataset, with highest score 13.39 %. Various predictions of these models are empty or very short, and some others contain long word repetitions which are discarded during evaluation. As a result, there are usually fewer than five predicted keyphrases. This explains why **F_1@5** and **F_1@7** scores are very close to each other.

ABS works slightly better reaching scores from 10.24 to 14.75 %. POINTCOV is the best of the text summarizers producing keyphrase predictions that are usually clean and concise with few repetitions. This is probably the merit of the coverage mechanism. There is still a considerable gap between POINTCOV and COPYRNN. Rouge-1 and Rouge-L F_1 scores are reported in Table 3. COPYRNN is still the best but POINTCOV is close. ABS scores are also comparable to those of MAUI and YAKE!. TOPICRANK, MERGE and INJECT are again the worst.

Regarding the test datasets, the highest result scores are achieved on Hulth and the lowest on Krapivin. We checked some samples of the later and observed that each of them contains separation tags (e.g., –T, –A, –B, Figure etc.) for indicating different parts of text in the original paper. A more intelligent text cleaning step may be required on those data.

5 Discussion

The results show that the tried text summarization models perform poorly on full-match keyword predictions. Their higher ROUGE scores further indicate that the problem is not entirely in the summarization process. Observing a few samples, we found differences between the two evaluation strategies. For example, suppose we have the predicted keyword *"intelligent system"* compared against authors' keyword *"system design"*. Full-match evaluation adds nothing to $\mathbf{F_1@5}$ and $\mathbf{F_1@7}$ scores. However, in the case of ROUGE evaluation, the prediction is partially right and a certain value is added to $\boldsymbol{R_1F_1}$ score. In follow up works, one solution to this discrepancy could be to try partial-match comparison scores like overlap coefficients.

Another detail that has some negative effect in full-match scores is keyword separation. The predicted string:

"health care,,,,immune system; human -; metabolism, immunity,,,,"

produces ["health care", "immune system", "human", "metabolism", "immunity"] as the list of keywords after removing the extra separators. Instead, we expected ["health care", "immune system", "human metabolism", "immunity"]. This again penalizes full-match scores but not $\boldsymbol{R_1F_1}$ score. A more intelligent keyword separation mechanism could thus help for higher full-match result scores.

A third reason could be the fact that we used the title and abstract of papers only. This is actually what most researchers do, as it is hard to find high quantities of article full texts for free. Article body is usually restricted. Abstractive summarization methods could still benefit from longer source texts. Using default hyperparameters for the models may have also influenced the results. Some parameter tuning could be beneficial, though.

The main reason could be even more fundamental. We trained abstractive summarization models on abstracts and titles with authors' keyphrases considered as golden ones. There might be two issues here. First, when setting their keywords, authors mostly consider the topical aspects of their work rather than paraphrasing over the contents. Abstracts and titles we used may not carry enough topical information about the article, even when joined together. Second, considering authors' keywords as golden ones may not be reasonable. One solution is to employ human experts and ask them to annotate each article based on what they read. This is however prohibitive when hundred thousands of samples are required. Extensive experiments on this issue may provide different facts and change the picture. For the moment, a safe way to go seems developing deep supervised generative models like the one of Meng et al. (2017) that predict one keyphrase at each step independently.

6 Conclusions

In this paper, we experimented with various unsupervised, supervised, deep supervised and abstractive text summarization models for predicting keyphrases of scientific articles. To the best of our knowledge, this is the first attempt that explores the possibility of conceiving article string of keywords as an abstractive summary of title and abstract. We collected and produced a large dataset of 2.2 million abstracts, titles and keyphrase strings from scientific papers available online. It can be used for future text summarization and keyphrase generation experiments. Systematic evaluation on four test datasets shows that the used summarization models could not produce better keywords than the supervised predictive models. Extensive experiments with more advanced summarizaiton methods and better parameter optimization may still reveal a different view of the situation.

Acknowledgments

This research work was supported by the project No. CZ.02.2.69/0.0/0.0/16_027/0008495 (International Mobility of Researchers at Charles University) of the Operational Programme Research, Development and Education, grant 19-26934X (NEUREM3) of the Czech Science Foundation and H2020-ICT-2018-2-825460 (ELITR) of the EU.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language*

Processing, pages 543–551. Asian Federation of Natural Language Processing.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. 2018. Yake! collection-independent automatic keyword extractor. In *Advances in Information Retrieval*, pages 806–810. Springer International Publishing.

Erion Çano and Maurizio Morisio. 2015. Characterization of public datasets for recommender systems. In *2015 IEEE 1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*, pages 249–257.

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*, pages 1105–1115. Association for Computational Linguistics.

George Foster, Ashish Vaswani, Jakob Uszkoreit, Wolfgang Macherey, Lukasz Kaiser, Orhan Firat, Llion Jones, Noam Shazeer, Yonghui Wu, Ankur Bapna, Melvin Johnson, Mike Schuster, Zhifeng Chen, Macduff Hughes, Niki Parmar, and Mia Xu Chen. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *ACL (1)*, pages 76–86. Association for Computational Linguistics.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 708–719. Association for Computational Linguistics.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1262–1273.

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. Association for Computational Linguistics.

Mikalai Krapivin, Aliaksandr Autayeu, Maurizio Marchese, Enrico Blanzieri, and Nicola Segata. 2010. Keyphrases extraction from scientific documents. In *The Role of Digital Libraries in a Time of Global Change*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, page 10.

Olena Medelyan. 2009. *Human-competitive automatic topic indexing*. The University of Waikato, Phd Thesis.

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*, pages 582–592. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290. Association for Computational Linguistics.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12, pages 95–100, Stroudsburg, PA, USA. Association for Computational Linguistics.

Thuy Dung Nguyen and Minh-Thang Luong. 2010. Wingnus: Keyphrase extraction utilizing document logical structure. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 166–169, Stroudsburg, PA, USA. Association for Computational Linguistics.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. In Michael W. Berry and Jacob Kogan, editors, *Text Mining. Applications and Theory*, pages 1–20. John Wiley and Sons, Ltd.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389. Association for Computational Linguistics.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1073–1083. Association for Computational Linguistics.

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 243–246, New York, NY, USA. ACM.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: Extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 990–998, New York, NY, USA. ACM.

Marc Tanti, Albert Gatt, and Kenneth P. Camilleri. 2017. What is the role of recurrent neural networks (rnns) in an image caption generator? *CoRR*, abs/1708.02043.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 76–85. Association for Computational Linguistics.

Peter D. Turney. 2000. Learning algorithms for keyphrase extraction. *Inf. Retr.*, 2(4):303–336.

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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

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*, DL '99, pages 254–255, NY, USA. ACM.