Exploiting Position and Contextual Word Embeddings for Keyphrase Extraction from Scientific Papers

Krutarth Patel
Computer Science
Kansas State University
Manhattan, Kansas, United States
kipatel@ksu.edu

Cornelia Caragea
Computer Science
University of Illinois at Chicago
Chicago, Illinois, United States
cornelia@uic.edu

Abstract

Keyphrases associated with research papers provide an effective way to find useful information in the large and growing scholarly digital collections. In this paper, we present KPRank, an unsupervised graph-based algorithm for keyphrase extraction that exploits both positional information and contextual word embeddings into a biased PageRank. Our experimental results on five benchmark datasets show that KPRank that uses contextual word embeddings with additional position signal outperforms previous approaches and strong baselines for this task.

1 Introduction

Keyphrase extraction is the task of automatically extracting a small set of descriptive words or phrases that can accurately summarize the topics discussed in a document (Hasan and Ng, 2010, 2014). Keyphrases are useful in many applications such as document indexing and summarization (Abu-Jbara and Radev, 2011; Qazvinian et al., 2010; Turney, 2003), topic tracking (Augenstein et al., 2017), contextual advertising (Yih et al., 2006), and opinion mining (Berend, 2011).

Most of the previous approaches to keyphrase extraction are either supervised or unsupervised. While supervised approaches perform generally better (Kim et al., 2013), the unsupervised ones have the advantage that they do not require large human-annotated corpora for training reliable models. Unsupervised keyphrase extraction methods usually use graph-based ranking algorithms such as PageRank that work on the word graph constructed from the target document (Mihalcea and Tarau, 2004). Various PageRank extensions have been proposed that incorporate different types of information (Wan and Xiao, 2008; Gollapalli and Caragea, 2014). For example, Wan and Xiao (2008) proposed to incorporate a local neighborhood of the target document into the graph construction, with the neighborhood being determined based on the textual similarity between documents. Liu et al. (2010) exploited topical information to select keyphrases from all major topics. More recently, Mahata et al. (2018) proposed a theme-weighted biased PageRank, called Key2Vec, for keyphrase extraction. In Key2Vec, a theme-vector is computed by averaging the embeddings of words and phrases from the title of a scientific document to capture its theme and the PageRank is biased based on the similarity of candidate words or phrases to the computed theme vector. However, this model is oblivious to the position of words in a scientific document, in which more important words appear not only frequently, but also close to the beginning of the document (Florescu and Caragea, 2017).

Inspired by the Transformer models (Vaswani et al., 2017) that infuse positional information into the word embeddings to produce embeddings with time signal, we propose an extension of Key2Vec that incorporates words' positions into a biased PageRank. Moreover, different from Mahata et al. (2018), who used non-contextual FastText embeddings (Mikolov et al., 2018), we propose to integrate SciBERT contextual embeddings (Beltagy et al., 2019) into our biased PageRank extension. Our contributions are as follows: (1) We propose KPRank, an unsupervised graph-based algorithm that exploits both the position of words in a document and the contextual word embeddings for computing a biased PageRank score for ranking candidate phrases; (2) We show empirically that infusing position information into our biased KPRank model yields better performance compared with its counterpart that does not use the position information. In addition, KPRank with contextual SciBERT embeddings performs better than FastText-based KPRank; (3) Finally, we show that KPRank outperforms many previous unsupervised models.

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 1585–1591
April 19 - 23, 2021.

2 Proposed Approach

In this section, we describe our unsupervised graph-based algorithm called KPRank, that exploits both position information of the words in a document along with contextual word embeddings for computing a biased PageRank score for each candidate word. Our approach consists of three steps: (1) candidate word selection and word graph construction; (2) word scoring by biased PageRank; and (3) candidate phrase formation.

2.1 Candidate Word Selection and Graph Construction

For a target doucment D, we first apply a part-of-speech filter[1] and select only nouns and adjectives as candidate words, consistent with previous works (Gollapalli and Caragea, 2014; Mihalcea and Tarau, 2004; Wan and Xiao, 2008). We build a word graph $G = (N, E)$ for D using the candidate words as nodes in G. N and E are the sets of nodes and edges, respectively. We consider an edge $(n_i, n_j) \in E$ between two nodes n_i and n_j in N if the words corresponding to these nodes appear within a window of k consecutive words in the content of D. We experimented with values of k from 1 to 10 and obtained best results with $k = 10$, which is consistent with (Wan and Xiao, 2008). The weight of an edge (n_i, n_j), denoted as w_{ij}, is computed based on the co-occurrence count of the two words within k consecutive words in D ($k = 10$). Here, we build undirected graphs because prior work (Mihalcea and Tarau, 2004; Liu et al., 2010) observed that the type of graph (directed or undirected) used to represent the text does not significantly influence the performance of the keyphrase extraction task.

2.2 Biased PageRank

Preliminaries. PageRank (Page et al., 1998) is a graph-based ranking algorithm that iteratively calculates the importance of each node in a graph through endorsements from its neighbors. For document D, we construct an undirected graph G as explained above. Initially, the score of each node in G is set to $\frac{1}{|N|}$. This score is then iteratively updated using PageRank. That is, the score s for node n_i is obtained by applying the equation:

$$s(n_i) = (1-\alpha)\widetilde{p}_i + \alpha \sum_{n_j \in Adj(n_i)} \frac{w_{ji}}{O(n_j)} s(n_j) \quad (1)$$

where $O(n_j) = \sum_{n_k \in Adj(n_j)} w_{jk}$ and $Adj(n_j)$ is the set of all adjacent nodes of node $n_j \in N$. $\widetilde{p}_i$ is defined below.

In order to prevent the PageRank from getting stuck in cycles or dead ends, a dumping factor α was added to Eq. (1) to allow the PageRank to randomly jump to any node in the graph ($\alpha = 0.85$). Let $\widetilde{p} = [\widetilde{p}_1, \cdots, \widetilde{p}_i, \cdots, \widetilde{p}_{|N|}]$ be the probability distribution of randomly jumping to any node in the graph. For an unbiased PageRank, this is a uniform distribution, with $\widetilde{p}_i = \frac{1}{|N|}$, for all i from 1 to $|N|$. For a biased PageRank, this probability distribution is not uniform, but rather the nodes in the graph are visited preferentially, with some nodes being visited more often than others, depending on the $\widetilde{p}_i$ value for node n_i (Haveliwala, 2003). Key2Vec is an example of (topic) biased PageRank for keyphrase extraction that computes $\widetilde{p}_i$ for node n_i using the cosine similarity between the embedding of word/phrase corresponding to node n_i and a theme vector for the entire document, which corresponds to the aggregated word/phrase embeddings from the document's title (Mahata et al., 2018). That is, $\widetilde{p}_i$ is higher for words/phrases that are topically (semantically) more similar to the overall theme vector for the document. Next, we describe our extension KPRank of Key2Vec.

KPRank. In our proposed approach, we calculate $\widetilde{p}_i$ for node n_i using two types of scores: *theme (or topic) score* and *positional score*. We multiply both scores to assign a final weight to node n_i before running the biased PageRank algorithm. Both scores and their calculation are explained below.

To calculate the *theme score* (ts_i) for node $n_i \in N$, we first calculate a theme vector (T_D) for document D. A *theme vector* is obtained by averaging SciBERT (Beltagy et al., 2019) word embeddings of adjectives and nouns from D's title. The *theme score* for node n_i is calculated using the cosine similarity of the SciBERT word embedding corresponding to node n_i and T_D. The idea is to assign a higher score to a word if that word is closer to the theme (topic) of a given document. For obtaining word embeddings, for all the words with similar stemmed version (obtained with Porter stemmer), we averaged the contextualized word embeddings of a word obtained by using SciBERT. We used the title and abstract of a document as input to the SciBERT model. We also experimented with pretrained BERT (Devlin et al., 2018), and found that the performance of BERT-based KPRank and SciBERT-based KPRank are very similar.

[1] We used Python's NLTK toolkit for POS tagging.

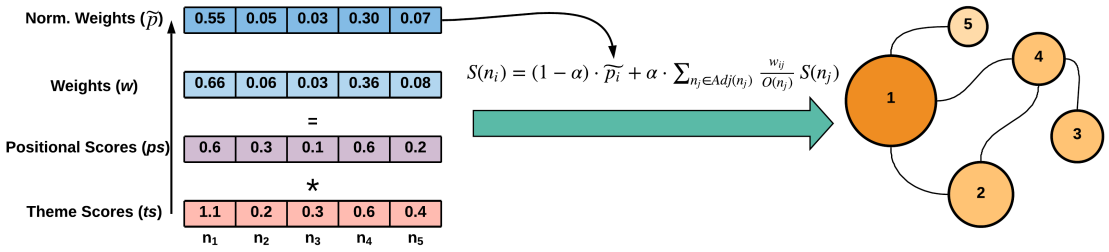

Figure 1: Illustration of our approach.

To calculate the *positional score* (ps_i) for node n_i, we consider the set P_i that contains all the positions of occurrence in the text of the word corresponding to node n_i. Then, ps_i is calculated as $ps_i = \sum_{j \in P_i} \frac{1}{j}$. For example, for a word occurring on positions 1 and 10 in the text, its ps_i score is $\frac{1}{1} + \frac{1}{10}$, whereas for a word occurring on position 100, its ps_i score is $\frac{1}{100}$. The intuition behind this weighting scheme is to give higher weight to words appearing in the beginning of a document since in scientific writing, authors tend to use keyphrases very early in the document (even from the title) (Florescu and Caragea, 2017). Based on these considerations, the first position of a phrase/word and its relative position are also used in many supervised approaches as powerful features (Patel and Caragea, 2019; Hulth, 2003; Wu et al., 2005)

To calculate the weight w_i for n_i, we perform multiplication of both the *theme score* (ts_i) and the *positional score* (ps_i). The intuition is that we give preference to words that appear near the beginning of the document and are more frequent as compared with less frequent words appearing later in document even though both words may be equally close to the theme of the document or may have similar theme score. The vector $\widetilde{p}$ is last set to the normalized weights for each node as follows:

$$\widetilde{p} = \left[\frac{w_1}{\sum_{i=1}^{|N|} w_i}, \frac{w_2}{\sum_{i=1}^{|N|} w_i}, ..., \frac{w_{|N|}}{\sum_{i=1}^{|N|} w_i} \right] \quad (2)$$

The biased PageRank scores for each node n_i are finally calculated by iteratively applying Eq. (1) with $\widetilde{p}$ as in Eq. (2). Figure 1 shows the illustration of our approach. As can be seen, even though both n_1 and n_4 have similar *theme score*, final weights are different based on different *positional scores*.

In our experiments, the PageRank scores are updated until the difference between two consecutive iterations is ≤ 0.001 or for 100 iterations.

2.3 Candidate Phrases Formation

Candidate words appearing continuously in the document are concatenated to generate candidate phrases. We consider phrases with the regular expression (adjective)*(noun)+, of length up to four words, to generate candidate phrases. We used stemmed version of each word using Porter stemmer. We use POS tagger from Python's NLTK toolkit. The score for each candidate phrase is calculated by summing up the scores of its individual words (Wan and Xiao, 2008). The top-scoring phrases are output as predicted keyphrases for a given document.

Dataset	# Test Papers	Avg. # KP per doc.
SemEval	100	6.87
Inspec	500	8.82
Krapivin	400	3.48
NUS	211	5.49
ACM	20,000	2.67

Table 1: The summary of the datasets along with their gold-standard keyphrases (KP).

3 Data

For evaluation, we use five datasets, which we describe below. We use the combination of controlled (author assigned) and uncontrolled (reader assigned) keyphrases as gold-standard phrases. We used uncontrolled keyphrases when available. Table 1 shows the summary of the datasets.

SemEval (Kim et al., 2010) contains 288 research papers with a train and test split consisting of 188 and 100 papers, respectively.

Inspec (Hulth, 2003) contains 2,000 research papers. It has a train-validation-test split of 1,000, 500 and 500 papers, respectively.

Krapivin (Krapivin et al., 2009) contains 2,304 ACM research papers with full text and author-assigned keyphrases. Similar to (Meng et al., 2017), since the dataset does not have a train-test split, we sampled 400 papers as the test set.

NUS (Nguyen and Kan, 2007) contains 211 research papers. This dataset does not have a train and test split and it is relatively small. Hence, consistent with (Meng et al., 2017) we used entire dataset as the test set.

ACM (Patel and Caragea, 2019) This dataset

	SemEval		Inspec		Krapivin		NUS		ACM	
	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10	F1@5	F1@10
KPRank(SB)	**22.51**	**25.76**	**27.72**	32.30	17.74	18.57	**21.09**	**22.36**	**14.79**	**15.17**
KPRank(SB−POS)	17.33	23.97	26.88	32.72	15.81	16.97	16.79	19.68	12.13	13.14
KPRank(FastText)	22.04	25.39	27.28	32.12	18.20	**18.91**	20.69	22.12	14.77	15.08
PositionRank	**22.51**	24.99	26.73	31.84	**18.49**	18.30	18.65	20.99	13.04	14.09
Key2Vec	17.54	23.63	26.97	**32.82**	15.50	16.68	16.80	20.10	12.08	13.07
Tf-Idf	18.13	21.82	24.73	31.41	15.67	17.23	14.22	18.05	11.07	12.90
TextRank	12.12	17.90	22.81	30.47	09.15	12.91	09.04	12.64	05.02	07.54
SingleRank	11.22	18.24	24.11	31.96	10.25	13.53	08.69	13.78	05.66	08.32
ExpandRank	14.65	20.46	24.81	31.45	12.32	15.32	10.90	15.29	10.02	11.28
TopicRank	10.77	11.04	14.65	17.46	08.11	07.82	11.79	11.45	09.39	07.62

Table 2: The comparison of SciBERT (SB) based KPRank, and previous works.

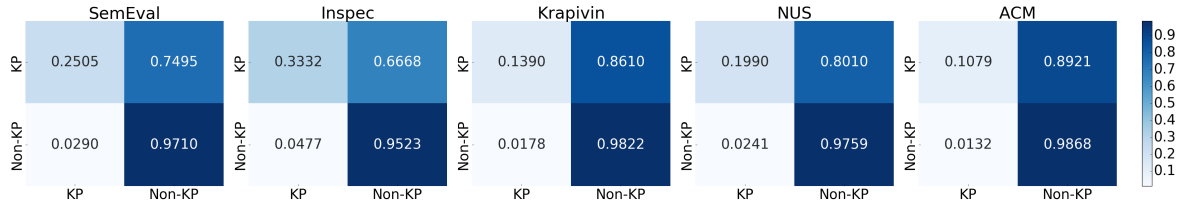

Figure 2: Keyphrase extraction confusion matrices of KPRank(SB) using @5 predictions on all the datasets. The darker the blue on the main diagonal, the more accurate the model is.

contains 30,000 papers published in ACM conferences with a train and test split consisting of 10,000 and 20,000 papers, respectively.

For each dataset we use its test set for evaluation.

4 Experimental Setup and Results

Evaluation metrics. To evaluate the performance of different methods, we use micro avg. F1-score. We report the performance for the top 5 and 10 candidate phrases returned by different methods as in (Meng et al., 2017). To create a word graph for a given document, we use its title and abstract. To match the predicted keyphrases with gold-standard keyphrases, we do exact match between the stemmed version of each.

The effect of position, contextual embeddings, and the comparison with previous works. To see the effect of positional information, we compare the performance of KPRank that uses contextual SciBERT (SB) embeddings along with positional information (denoted as KPRank(SB)) with that of its counterpart that does not use positional information (denoted as KPRank(SB−POS)). Moreover, to see the effect of contextual embeddings, we compare the performance of SciBERT-based KPRank (KPRank(SB)) with that of KPRank that uses FastText non-contextual word embeddings (Mikolov et al., 2018) (denoted as KPRank(FastText)). For FastText, we used pretrained 300 dimensional embeddings trained on subword information on Common Crawl. Note that KPRank(SciBERT) and KPRank(FastText) use positional information along with the theme score. Last, we compare the performance of KPRank with Tf-Idf and six PageRank based unsupervised methods as baselines: PositionRank (Florescu and Caragea, 2017), Key2Vec (Mahata et al., 2018), TextRank (Mihalcea and Tarau, 2004), SingleRank (Wan and Xiao, 2008), ExpandRank (Wan and Xiao, 2008), TopicRank (Bougouin et al., 2013).

Tables 2 shows these comparisons on SemEval, Inspec, Krapivin, NUS, and ACM. It can be seen from the table that adding position information shows much higher improvement in the performance of KPRank, i.e. KPRank(SB) substantially outperforms KPRank(SB−POS). Moreover, KPRank(SB) outperforms KPRank(FastText) on all the datasets except for Krapivin. Importantly, KPRank(SB) outperforms most baseline methods, including Key2Vec (by a large margin) e.g., on SemEval, KPRank(SB) achieves an F1@5 of 22.51% as compared with 17.54% achieved by Key2Vec. We can also notice from Table 2 that KPRank(SB) achieves comparable performance whenever any baseline method achieves the best performance.

Figure 2 shows the confusion matrices of KPRank(SB) using @5 predictions on all five datasets. Each matrix is represented as a heat map, i.e., the darker the blue color the higher the value at that position and the darker the blue on the main diagonal, the more accurate the model is.

Organization design: The continuing influence of *information technology*

Drawing from an *information processing perspective*, this paper examines how *information technology* (IT) has been a catalyst in the development of new forms of *organizational structures*. [...] to the present *environmental instability* that now characterizes many industries. Specifically, the authors suggest that advances in IT have enabled managers to adapt existing forms and create new models for *organizational design* that better fit requirements of an unstable environment. [...]. IT has gone from a support mechanism to a substitute for *organizational structures* in the form of the shadow structure. [...]

Gold-standard keyphrases: *Organization design, Information processing perspective, Organizational structures, Environmental instability, Information technology*

Predicted keyphrases: ***Organization design***, ***Information technology***, ***Information processing perspective***, ***Organizational structures***, *Organizational design, Organization, Information processing, Shadow structure, New forms, Bureaucratic structure*

Figure 3: The title, abstract, gold-standard keyphrases and predicted keyphrases of a paper. The phrases marked with cyan in the title and abstract shown on the top of the figure are gold-standard keyphrases.

Comparison with a supervised approach. Usually, the performance of the supervised keyphrase extraction models is better than the unsupervised models (Kim et al., 2013). We compare the performance of KPRank(SB) with the CRF based sequence classification model for the keyphrase extraction (Patel and Caragea, 2019) that uses word embeddings as features along with document specific features. The CRF model outperforms KPRank(SB) on all five datasets, e.g., CRF model achieves an F1 of 45.73% as compared with 25.76% achieved by KPRank(SB) on SemEval.

Anecdotal example. To see the quality of predicted phrases by the KPRank(SB), we randomly selected a paper from the Inspec dataset and evaluated the KPRank(SB) on it. We manually inspected the top-10 predictions by the KPRank(SB) and contrasted them with the gold-standard keyphrases. The title, abstract, gold-standard keyphrases and top-10 predicted keyphrases for this paper are shown in Figure 3. Precisely, in the figure, the cyan italic phrases shown in the text on the top of the figure represent gold-standard keyphrases, whereas the bottom of the figure shows gold-standard keyphrases and the top-10 predicted keyphrases by KPRank(SB) (shown in the order of their prediction). It can be seen from the figure that four out of five gold-standard keyphrases are present in the top-5 predicted keyphrases.

We can also see that KPRank(SB) did not predict gold-standard phrase "environmental instabily." A closer inspection of the document and both types of scores (*theme score* and *positional score*) assigned by KPRank(SB) to both constituent words of the gold-standard phrase that was not ranked in top-10 predictions revealed that these constituent words have lower values of *theme score* and they both appear only once in the document. Hence, the Pagerank algorithm will not boost these words. Inspecting other errors, we found that KPRank can fail to predict phrases that contain words that are less frequent in the document and their word embeddings are far from the *theme vector*.

5 Conclusion and Future Work

In this paper, we proposed a novel unsupervised graph-based algorithm, named KPRank, which incorporates both positional appearances of the words along with contextual word embeddings for computing a biased PageRank score for each candidate word. Our experimental results on five datasets show that incorporating position information into our biased KPRank model yields better performance compared with a KPRank that does not use the position information, and SciBERT-based KPRank usually outperforms FastText-based KPRank on this task. Moreover, KPRank outperforms strong baseline methods. In the future, it would be interesting to explore KPRank on other domains, such as Biology, and Social Science.

Acknowledgments

We thank our anonymous reviewers for their constructive comments and feedback, which helped improve our paper. This research is supported in part by NSF. Any opinions, findings, and conclusions expressed here are those of the authors and do not necessarily reflect the views of NSF.

References

Amjad Abu-Jbara and Dragomir Radev. 2011. Coherent citation-based summarization of scientific papers. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 500–509.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. 2017. Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 546–555.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Gábor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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, Vancouver, Canada.

Sujatha Das Gollapalli and Cornelia Caragea. 2014. Extracting keyphrases from research papers using citation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, pages 1629–1635.

Kazi Saidul Hasan and Vincent Ng. 2010. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 365–373.

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 (Volume 1: Long Papers)*, pages 1262–1273.

Taher H Haveliwala. 2003. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE transactions on knowledge and data engineering*, pages 784–796.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 216–223.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2013. Automatic keyphrase extraction from scientific articles. *Language resources and evaluation*, 47(3):723–742.

Mikalai Krapivin, Aliaksandr Autaeu, and Maurizio Marchese. 2009. Large dataset for keyphrases extraction. Technical report, University of Trento.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 366–376.

Debanjan Mahata, John Kuriakose, Rajiv Ratn Shah, and Roger Zimmermann. 2018. Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. 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 634–639.

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.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 404–411.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Asian Digital Libraries*, pages 317–326.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1998. The pagerank citation ranking: bringing order to the web. *Technical report, Standford Digital Library Technologies Project*.

Krutarth Patel and Cornelia Caragea. 2019. Exploring word embeddings in crf-based keyphrase extraction from research papers. In *Proceedings of the 10th International Conference on Knowledge Capture*, pages 37–44.

Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. 2010. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd international conference on computational linguistics (COLING 2010)*, pages 895–903.

Peter D Turney. 2003. Coherent keyphrase extraction via web mining. *arXiv preprint cs/0308033*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, pages 855–860.

Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen. 2005. Domain-specific keyphrase extraction. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 283–284.

Wen-tau Yih, Joshua Goodman, and Vitor R. Carvalho. 2006. Finding advertising keywords on web pages. In *Proceedings of the 15th international conference on World Wide Web*, pages 213–222.