DAKE: Document-Level Attention for Keyphrase Extraction

Tokala Yaswanth Sri Sai Santosh[1], Debarshi Kumar Sanyal[2(✉)], Plaban Kumar Bhowmick[3], and Partha Pratim Das[1]

[1] Department of Computer Science and Engineering, IIT Kharagpur, Kharagpur 721302, India
santoshtyss@gmail.com, ppd@cse.iitkgp.ac.in
[2] National Digital Library of India, IIT Kharagpur, Kharagpur 721302, India
debarshisanyal@gmail.com
[3] Centre for Educational Technology, IIT Kharagpur, Kharagpur 721302, India
plaban@cet.iitkgp.ac.in

Abstract. Keyphrases provide a concise representation of the topical content of a document and they are helpful in various downstream tasks. Previous approaches for keyphrase extraction model it as a sequence labelling task and use local contextual information to understand the semantics of the input text but they fail when the local context is ambiguous or unclear. We present a new framework to improve keyphrase extraction by utilizing additional supporting contextual information. We retrieve this additional information from other sentences within the same document. To this end, we propose Document-level Attention for Keyphrase Extraction (DAKE), which comprises Bidirectional Long Short-Term Memory networks that capture hidden semantics in text, a document-level attention mechanism to incorporate document level contextual information, gating mechanisms which help to determine the influence of additional contextual information on the fusion with local contextual information, and Conditional Random Fields which capture output label dependencies. Our experimental results on a dataset of research papers show that the proposed model outperforms previous state-of-the-art approaches for keyphrase extraction.

Keywords: Keyphrase extraction · Sequence labelling · LSTM · Document-level attention

1 Introduction

Keyphrase extraction is the task of automatically extracting words or phrases from a text, which concisely represent the essence of the text. Because of the succinct expression, keyphrases are widely used in many tasks like document retrieval [13,25], document categorization [9,12], opinion mining [3] and summarization [24,31]. Figure 1 shows an example of a title and the abstract of a research paper along with the author-specified keyphrases highlighted in bold.

© Springer Nature Switzerland AG 2020
J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12036, pp. 392–401, 2020.
https://doi.org/10.1007/978-3-030-45442-5_49

Present methods for keyphrase extraction follow a two-step procedure where they select important phrases from the document as potential keyphrase candidates by heuristic rules [18,28,29] and then the extracted candidate phrases are ranked either by unsupervised approaches [17,21,27] or supervised approaches [18,22,29]. Unsupervised approaches score those candidate phrases based on individual words comprising the candidate phrases. They utilize various scoring measures based on the informativeness of the word with respect to the whole document [10]. Other paradigms utilize graph-based ranking algorithms wherein each word in the document is mapped to a node in the graph and the connecting edges in the graph represent the association patterns among the words in the document. Then, the scores of the individual words are estimated using various graph centrality measures [6,21,27]. On the other hand, supervised approaches [4,14] use binary classification to label the extracted candidate phrases as keyphrases or non-keyphrases, based on various features such as, tf-idf, part-of-speech (POS) tags, and the position of phrases in the document. The major limitation of these supervised approaches is that they classify the labels of each candidate phrase independently without taking into account the dependencies that could potentially exist between neighbouring labels and they also ignore the semantic meaning of the text. To overcome the above stated limitation, [8] formulated keyphrase extraction as a sequence labeling task and used linear-chain Conditional Random Fields for this task. However, this approach does not explicitly take into account the long-term dependencies and semantics of the text. More recently, to capture both the semantics of the text as well as the dependencies among the labels of neighboring words [1] used a deep learning-based approach called BiLSTM-CRF which combines a bi-directional Long Short-Term Memory (BiLSTM) layer that models the sequential input text with a Conditional Random Field (CRF) layer that captures the dependencies in the output.

Title: **DCE-MRI** data analysis for cancer area classification.
Abstract: The paper aims at improving the support of medical researchers in the context of in-vivo cancer imaging. [..] The proposed approach is based on a three-step procedure: i) robust feature extraction from raw time-intensity curves, ii) voxel segmentation, and iii) voxel **classification** based on a learning-by-example approach. Finally, in the third step, a support vector machine (**SVM**) is trained to classify voxels according to the labels obtained by the clustering phase. [..]

Fig. 1. An example of keyphrase extraction with author-specified keyphrases highlighted in bold.

The above mentioned approaches treat keyphrase extraction as a sentence-level task where sentences in the same document are viewed as independent. When labeling a word, local contextual information from the surrounding words is crucial because the context gives insight to the semantic meaning of the word. However, there are many instances in which the local context is ambiguous or lacks sufficient information. If the model has access to supporting information that provides additional context, the model may use this additional supporting

information to predict the label correctly. Such additional supporting information may be found from other sentences in the same document from which the query sentence is taken. To utilize this additional supporting information, we propose a document-level attention mechanism inspired from [20,30]; it dynamically weights the additional supporting information emphasizing the most relevant information from each supporting sentence with respect to the local context. But leveraging this additional supporting information has a downside of introducing noise into the representations. To alleviate this problem, we use a gating mechanism [20,30] that balances the influence of the local contextual representations and the additional supporting information from the document-level contextual representations.

To this end, in this paper, we propose Document-level Attention for Keyphrase Extraction (DAKE). It initially produces representations for each word that encode the local context from the query sentence using BiLSTM, then uses a document-level attention mechanism to incorporate the most relevant information from each supporting information with respect to the local context, and employs a gating mechanism to filter out the irrelevant information. Finally, it uses a CRF layer which captures output label dependencies to decode the gated local and the document-level contextual representations to predict the label. The main contributions of this paper are as follows:

- We propose DAKE, a BiLSTM-CRF model augmented with document-level attention and a gating mechanism for improved keyword extraction from research papers.
- Experimental results on a dataset of research papers show that DAKE outperforms previous state-of-the-art approaches.

2 Problem Formulation

We formally describe the keyphrase extraction task as follows: Given a sentence, $s = \{w_1, w_2, \ldots, w_n\}$ where n is the length of the sentence, predict the labels sequence $y = \{y_1, y_2, \ldots, y_n\}$ where y_i is the label corresponding to word w_i and it can KP (keyphrase word) or Not-KP (not a keyphrase word). Every longest sequence of KP words in a sentence is a keyphrase.

3 Proposed Method

The main components in our proposed architecture, DAKE, are: Word Embedding Layer, Sentence Encoding Layer, Document-level Attention mechanism, Gating mechanism, Context Augmenting Layer and Label Sequence Prediction Layer. The first layer produces word embeddings of the sentence from which the second layer generates word representations that encode the local context from the query sentence. Then the document-level attention mechanism extracts supporting information from other sentences in the document to enrich the current word representation. Subsequently, we utilize a gating mechanism to filter out

the irrelevant information from each word representation. The next layer fuses the local and the global contexts into each word representation. Finally, we feed these word representations into the CRF layer which acts as a decoder to predict the label, KP or Not-KP, associated with each word. The model is trained in an end-to-end fashion.

3.1 Word Embedding Layer

Given a document $D = \{s_1, s_2, \ldots, s_m\}$ of m sentences, where a sentence $s_i = \{w_{i1}, w_{i2}, \ldots, w_{in}\}$ is a sequence of n words, we transform each word w_{ij} in the sentence s_i into a vector $\mathbf{x}_{ij}$ using pre-trained word embeddings.

3.2 Sentence Encoding Layer

We use a BiLSTM [11] to obtain the hidden representation H_i of the sentence s_i. A BiLSTM comprises a forward-LSTM which reads the input sequence in the original direction and a backward-LSTM which reads it in the opposite direction. We apply forward-LSTM on the sentence $s_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{in})$ to obtain $\overrightarrow{H_i} = (\overrightarrow{\mathbf{h}_{i1}}, \overrightarrow{\mathbf{h}_{i2}}, \ldots, \overrightarrow{\mathbf{h}_{in}})$. The backward-LSTM on s_i produces $\overleftarrow{H_i} = (\overleftarrow{\mathbf{h}_{i1}}, \overleftarrow{\mathbf{h}_{i2}}, \ldots, \overleftarrow{\mathbf{h}_{in}})$. We concatenate the outputs of the forward and the backward LSTMs to obtain the local contextual representation $H_i = \{\mathbf{h}_{i1}, \mathbf{h}_{i2}, \ldots, \mathbf{h}_{in}\}$ where $\mathbf{h}_{ij} = [\overrightarrow{\mathbf{h}_{ij}} : \overleftarrow{\mathbf{h}_{ij}}]$; here, : denotes concatenation operation. Succinctly, $\mathbf{h}_{ij} = \text{BiLSTM}(\mathbf{x}_{ij})$

3.3 Document-Level Attention

Many keyphrase mentions are tagged incorrectly in current approaches including the BiLSTM-CRF model [1] due to ambiguous contexts present in the input sentence. In cases where a sentence is short or highly ambiguous, the model may either fail to identify keyphrases due to insufficient information or make wrong predictions by using noisy context. We hypothesize that this limitation can be alleviated using additional supporting information from other sentences within the same document. To extract this global context, we need vector representations of other sentences in the same document D. We utilize BERT [5] as a sentence encoder to obtain representations for the sentences in D. Given an input sentence s_l in D, we extract the final hidden state of the [CLS] token as the representation $\mathbf{h}'_l$ of the sentence, where [CLS] is the special classification embedding in BERT. Then, for each word, w_{ij} in the input sentence s_i, we apply an attention mechanism to weight the supporting sentences in D as follows

$$e^l_{ij} = \mathbf{v}^\top \tanh(W_1 \mathbf{h}_{ij} + W_2 \mathbf{h}'_l + \mathbf{b}_1) \tag{1}$$

$$\alpha^l_{ij} = \frac{\exp(e^l_{ij})}{\sum_{p=1}^{m} \exp(e^p_{ij})} \tag{2}$$

where W_1, W_2 are trainable weight matrices and $\mathbf{b}_1$ is a trainable bias vector. We compute the final representation of supporting information as $\tilde{\mathbf{h}}_{ij} = \sum_{l=1}^{m} \alpha_{ij}^{l} \mathbf{h}'_l$. For each word w_{ij}, $\tilde{\mathbf{h}}_{ij}$ captures the document-level supporting evidence with regard to w_{ij}.

3.4 Gating Mechanism

Though the above supporting information from the entire document is valuable to the prediction, we must mitigate the influence of the distant supporting information as the prediction should be made primarily based on the local context. Therefore, we apply a gating mechanism to constrain this influence and enable the model to decide the amount of the supporting information that should be incorporated in the model, which is given as follows:

$$\mathbf{r}_{ij} = \sigma(W_3\tilde{\mathbf{h}}_{ij} + W_4\mathbf{h}_{ij} + \mathbf{b}_2) \tag{3}$$

$$\mathbf{z}_{ij} = \sigma(W_5\tilde{\mathbf{h}}_{ij} + W_6\mathbf{h}_{ij} + \mathbf{b}_3) \tag{4}$$

$$\mathbf{g}_{ij} = \tanh(W_7\mathbf{h}_{ij} + \mathbf{z}_{ij} \odot (W_8\tilde{\mathbf{h}}_{ij} + \mathbf{b}_4)) \tag{5}$$

$$\mathbf{d}_{ij} = \mathbf{r}_{ij} \odot \mathbf{h}_{ij} + (1 - \mathbf{r}_{ij}) \odot \mathbf{g}_{ij} \tag{6}$$

where $\odot$ denotes Hadamard product and W_3,W_4,W_5,W_6,W_7,W_8 are trainable weight matrices and $\mathbf{b}_2$,$\mathbf{b}_3$,$\mathbf{b}_4$ are trainable bias vectors. $\mathbf{d}_{ij}$ is the representation for the gated supporting evidence for w_{ij}.

3.5 Context Augmenting Layer

For each word w_{ij} of sentence s_i, we concatenate its local contextual representation $\mathbf{h}_{ij}$ and gated document-level supporting contextual representation $\mathbf{d}_{ij}$ to obtain its final representation $\mathbf{a}_{ij} = [\mathbf{h}_{ij} : \mathbf{d}_{ij}]$, where : denotes concatenation operation. These final representations $A_i = \{\mathbf{a}_{i1}, \mathbf{a}_{i2}, \ldots, \mathbf{a}_{in}\}$ of sentence s_i are fed to another BiLSTM to further encode the local contextual features along with supporting contextual information into unified representations $C_i = \{\mathbf{c}_{i1}, \mathbf{c}_{i2}, \ldots, \mathbf{c}_{in}\}$ where $\mathbf{c}_{ij} = \text{BiLSTM}(\mathbf{a}_{ij})$. The output of this encoding captures the interaction among the context words conditioned on the supporting information. This is different from the initial encoding layer, which captures the interaction among words of the sentence independent of the supporting information.

3.6 Label Sequence Prediction Layer

The obtained contextual representations C_i of query sentence s_i are given as input sequence to a CRF layer [16] that produces a probability distribution over the output label sequence using the dependencies among the labels of the entire input sequence. In order to efficiently find the best sequence of labels for an input sentence, the Viterbi algorithm [7] is used.

4 Experiments

4.1 Dataset

We use the dataset from [19] which comprises metadata of papers from several online digital libraries. The dataset contains metadata for 567,830 papers with a clear split as train, validation, and test sets provided by the authors, as follows: 527,830 were used for model training, 20,000 were used for validation and the rest 20,000 were used for testing. We refer to these sets as kp527k, kp20k-v and kp20k respectively. The metadata of each paper consists of title, abstract, and author-assigned keyphrases. The title and abstract of each paper are used to extract keyphrases, whereas the author-input keyphrases are used as gold-standard for evaluation.

4.2 Baselines and Evaluation Metrics

We compare our approach, DAKE with the following baselines: Bi-LSTM-CRF [1], CRF [8], Bi-LSTM [1], copy-RNN [19], KEA [29], Tf-Idf, TextRank [21] and SingleRank [27]. We also carry out an ablation test to understand the effectiveness of document-level attention and gating mechanism components by removing them. Similar to previous works, we evaluate the predictions of each method against the author-specified keyphrases that can be located in the corresponding paper abstracts in the dataset ("gold standard"). We present results for all our experiments using the precision, recall, and F1-score measures. For comparison of the methods, we choose the F1-score, which is the harmonic mean of precision and recall.

4.3 Implementation Details

We use pre-trained word embedding vectors obtained using GloVe [23]. We use SciBERT [2], a BERT model trained on scientific text for the sentence encoder. For word representations, we use 300-dimensional pre-trained word embeddings and for sentence encoder, we use 768 dimensional representation obtained using SciBERT. The hidden state of the LSTM is set to 300 dimensions. The model is trained end-to-end using the Adam optimization method [15]. The learning rate is initially set as 0.001 and decayed by 0.5 after each epoch. For regularization to avoid over-fitting, dropout [26] is applied to each layer. We select the model with the best F1-score on the validation set, kp20k-v.

5 Results and Discussion

Table 1a shows the results of our approach in comparison to various baselines. Our approach, DAKE outperforms all baselines in terms of the F1-score. Tf-Idf, TextRank and SingleRank are unsupervised extractive approaches while KEA, Bi-LSTM-CRF, CRF, Bi-LSTM follow supervised extractive approach.

copyRNN is a recently proposed generative model based on sequence-to-sequence learning along with a copying mechanism. For the unsupervised models and the sequence-to-sequence learning model, we report the performance at top-5 predicted keyphrases since top-5 showed highest performance in the previous works for these models. From Table 1a, we observe that the deep learning-based approaches perform better than the traditional feature-based approaches. This indicates the importance of understanding the semantics of the text for keyphrase extraction. BiLSTM-CRF yields better results in terms of the F1-score over CRF (improvement of F1-score by 18.17% from 17.46% to 35.63%) and BiLSTM (improvement of F1-score by 18.88% from 16.75% to 35.63%) models alone. This result indicates that the combination of BiLSTM, which is powerful in capturing the semantics of the textual content, with CRF, which captures the dependencies among the output labels, helped boost the performance in identifying keyphrases. Our proposed method, DAKE outperforms the BiLSTM-CRF (improvement of F1-score by 6.67% from 35.63% to 42.30%) approach, which indicates that the incorporation of additional contextual information from other sentences in the document into the BiLSTM-CRF model helps to further boost the performance.

Table 1. Performance analysis of DAKE

(a) Performance of different keyphrase extraction algorithms.

Method	Precision	Recall	F1-score
Tf-Idf	8.97	13.49	10.77
TextRank	15.29	23.01	18.37
SingleRank	8.42	12.70	10.14
KEA	15.14	22.78	18.19
copyRNN	27.71	41.79	33.29
CRF	66.67	10.04	17.46
BiLSTM	9.41	**76.24**	16.75
BiLSTM-CRF	64.19	24.66	35.63
DAKE	**68.21**	30.66	**42.30**

(b) Ablation Study: BiLSTM-CRF used as baseline.

Method	Precision	Recall	F1-score
DAKE without document-level attention	64.19	24.66	35.63
DAKE without gating mechanism	65.26	25.31	36.47
DAKE without context augment-ing layer	66.74	26.45	38.09
DAKE without CRF layer	61.38	28.81	39.21
DAKE	**68.21**	**30.66**	**42.30**

Table 1b shows the ablation study. We observe that document-level attention increases the F1-score of the baseline BiLSTM-CRF by 0.84% (from 35.63% to 36.47%). This validates our hypothesis that additional supporting information boosts the performance for keyphrase extraction. But leveraging this additional supporting information has a downside of introducing noise into the representations, and to alleviate this, we used a gating mechanism which boosted the F1-score by 1.62% (from 36.47% to 38.09%). Document-level attention did not show great improvement when it has only one layer of BiSLTM because the

final tagging predictions mainly depend on the local context of each word while additional context only supplements extra information. Therefore, our model needs another layer of BiLSTM to encode the sequential intermediate vectors containing additional context and local context, as evidenced from our F1-score improvement by 4.21% (from 38.09% to 42.30%). When CRF is removed from DAKE, the F1-score falls by 3.09%, showing that CRF successfully captures the output label dependencies.

6 Conclusion and Future Work

We proposed an architecture, DAKE, for keyword extraction from documents. It uses a BiLSTM-CRF network enhanced with a document-level attention mechanism to incorporate contextual information from the entire document, and gating mechanisms to balance between the global and the local contexts. It outperforms existing keyphrase extraction methods on a dataset of research papers. In future, we would like to integrate the relationships between documents such as those available from a citation network by enhancing our approach with contexts in which the document is referenced within a citation network.

Acknowledgements. This work is supported by *National Digital Library of India* Project sponsored by Ministry of Human Resource Development, Government of India at IIT Kharagpur.

References

1. Alzaidy, R., Caragea, C., Giles, C.L.: Bi-LSTM-CRF sequence labeling for keyphrase extraction from scholarly documents. In: Proceedings of The World Wide Web Conference, pp. 2551–2557. ACM (2019)
2. Beltagy, I., Cohan, A., Lo, K.: Scibert: pretrained contextualized embeddings for scientific text. arXiv preprint arXiv:1903.10676 (2019)
3. Berend, G.: Opinion expression mining by exploiting keyphrase extraction. In: Proceedings of the 5th International Joint Conference on Natural Language Processing. Asian Federation of Natural Language Processing (2011)
4. Caragea, C., Bulgarov, F.A., Godea, A., Gollapalli, S.D.: Citation-enhanced keyphrase extraction from research papers: a supervised approach. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1435–1446 (2014)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Florescu, C., Caragea, C.: PositionRank: an unsupervised approach to keyphrase extraction from scholarly documents. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, vol. 1: Long Papers, pp. 1105–1115 (2017)
7. Forney, G.D.: The Viterbi algorithm. Proc. IEEE **61**(3), 268–278 (1973)
8. Gollapalli, S.D., Li, X.L., Yang, P.: Incorporating expert knowledge into keyphrase extraction. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence (2017)

9. Hammouda, K.M., Matute, D.N., Kamel, M.S.: CorePhrase: keyphrase extraction for document clustering. In: Perner, P., Imiya, A. (eds.) MLDM 2005. LNCS (LNAI), vol. 3587, pp. 265–274. Springer, Heidelberg (2005). https://doi.org/10.1007/11510888_26
10. Hasan, K.S., Ng, V.: Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, pp. 365–373. Association for Computational Linguistics (2010)
11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
12. Hulth, A., Megyesi, B.B.: A study on automatically extracted keywords in text categorization. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, pp. 537–544. Association for Computational Linguistics (2006)
13. Jones, S., Staveley, M.S.: Phrasier: a system for interactive document retrieval using keyphrases. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 160–167. ACM (1999)
14. Kim, S.N., Medelyan, O., Kan, M.Y., Baldwin, T.: Automatic keyphrase extraction from scientific articles. Lang. Res. Eval. **47**(3), 723–742 (2013)
15. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
16. Lafferty, J., McCallum, A., Pereira, F.C.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning, pp. 282–289 (2001)
17. Le, T.T.N., Nguyen, M.L., Shimazu, A.: Unsupervised keyphrase extraction: introducing new kinds of words to keyphrases. In: Kang, B.H., Bai, Q. (eds.) AI 2016. LNCS (LNAI), vol. 9992, pp. 665–671. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50127-7_58
18. Medelyan, O., Frank, E., Witten, I.H.: Human-competitive tagging using automatic keyphrase extraction. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, vol. 3, pp. 1318–1327. Association for Computational Linguistics (2009)
19. Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y.: Deep keyphrase generation. arXiv preprint arXiv:1704.06879 (2017)
20. Miculicich, L., Ram, D., Pappas, N., Henderson, J.: Document-level neural machine translation with hierarchical attention networks. arXiv preprint arXiv:1809.01576 (2018)
21. Mihalcea, R., Tarau, P.: Textrank: bringing order into text. In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp. 404–411 (2004)
22. Nguyen, T.D., Kan, M.-Y.: Keyphrase extraction in scientific publications. In: Goh, D.H.-L., Cao, T.H., Sølvberg, I.T., Rasmussen, E. (eds.) ICADL 2007. LNCS, vol. 4822, pp. 317–326. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-77094-7_41
23. Pennington, J., Socher, R., Manning, C.: GloVe: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1532–1543 (2014)
24. Qazvinian, V., Radev, D.R., Ozgur, A.: Citation summarization through keyphrase extraction. In: Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010), pp. 895–903 (2010)

25. Sanyal, D.K., Bhowmick, P.K., Das, P.P., Chattopadhyay, S., Santosh, T.Y.S.S.: Enhancing access to scholarly publications with surrogate resources. Scientometrics **121**(2), 1129–1164 (2019). https://doi.org/10.1007/s11192-019-03227-4
26. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
27. Wan, X., Xiao, J.: Single document keyphrase extraction using neighborhood knowledge. In: AAAI, vol. 8, pp. 855–860 (2008)
28. Wang, M., Zhao, B., Huang, Y.: PTR: phrase-based topical ranking for automatic keyphrase extraction in scientific publications. In: Hirose, A., Ozawa, S., Doya, K., Ikeda, K., Lee, M., Liu, D. (eds.) ICONIP 2016. LNCS, vol. 9950, pp. 120–128. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46681-1_15
29. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G.: KEA: practical automated keyphrase extraction. In: Design and Usability of Digital Libraries: Case Studies in the Asia Pacific, pp. 129–152. IGI Global (2005)
30. Zhang, B., Whitehead, S., Huang, L., Ji, H.: Global attention for name tagging. In: Proceedings of the 22nd Conference on Computational Natural Language Learning, pp. 86–96 (2018)
31. Zhang, Y., Zincir-Heywood, N., Milios, E.: World wide web site summarization. Web Intell. Agent Syst. Int. J. **2**(1), 39–53 (2004)