

PerKey: A Persian News Corpus for Keyphrase Extraction and Generation

Ehsan Doostmohammadi

Graduate Student

*Computational Linguistics Group
Sharif University of Technology
Tehran, Iran*

e.doostm72@student.sharif.edu

Mohammad Hadi Bokaei

Assistant Professor

*Information Technology Department
ICT Research Institute
Tehran, Iran*

mh.bokaei@itrc.ac.ir

Hossein Sameti

Associate Professor

*Computer Engineering Department
Sharif University of Technology
Tehran, Iran*

sameti@sharif.edu

Abstract—Keyphrases provide an extremely dense summary of a text. Such information can be used in many Natural Language Processing tasks, such as information retrieval and text summarization. Since previous studies on Persian keyword or keyphrase extraction have not published their data, the field suffers from the lack of a human extracted keyphrase dataset. In this paper, we introduce PerKey¹, a corpus of 553k news articles from six Persian news websites and agencies with relatively high quality author extracted keyphrases, which is then filtered and cleaned to achieve higher quality keyphrases. The resulted data was put into human assessment to ensure the quality of the keyphrases. We also measured the performance of different supervised and unsupervised techniques, e.g. TFIDF, MultipartiteRank, KEA, etc. on the dataset using precision, recall, and F₁-score.

Keywords—Keyphrase Extraction, Keyword Extraction, Persian news corpus, supervised Keyphrase Extraction, unsupervised Keyphrase Extraction

I. INTRODUCTION

Keyphrases are single words or sequences of words that express the main topics discussed in a piece of text. High quality keyphrases can help the categorization and retrieval of data, which can be used in many Natural Language Processing (NLP) tasks, including information retrieval, text summarization, text categorization, question answering, and opinion mining.

Keyphrase extraction is quite a well-known task in NLP, the purpose of which is to extract keyphrases from a given text. Although a considerable progress has been made, the effectiveness of the task of keyphrases extraction is still far from many other problems in NLP. Methods used to extract keyphrases vary from unsupervised approaches, e.g. statistical and graph-based scoring methods, to supervised approaches, e.g. classification/sequence labelling methods. Since supervised approaches require labelled data, most of the approaches to the problem of keyphrases extraction are unsupervised. Unsupervised techniques tend to perform better in general and supervised approaches in

specific fields. Additionally, state-of-the-art methods, i.e. artificial neural networks, require training on labelled data. The problem with both unsupervised and supervised approaches is that they are unable to extract implicit keyphrases. Not all the keyphrases are always explicitly mentioned in the text, but some of them are implicitly spoken of. To extract such keyphrases we need generative models, e.g. sequence to sequence methods as in neural machine translation.

For the abovementioned reasons, a need for a high quality keyphrase corpus is strongly felt in Persian natural language processing. As hand-annotating data is not easily feasible for large corpora, we decided to gather author-keyphrase-annotated data from news websites and agencies with relatively high quality keyphrases. This method is preceded in the task of keyphrase extraction from tweets [1] and keyphrase generation from scientific papers [2], the latter collecting 567,830 papers in the domain of computer science from multiple websites. As of our work, we assessed more than 30 news websites and agencies and chose 6 of them having the highest quality keyphrases. After collecting the data, we did some cleaning and pre-processing to improve the quality of data which resulted in 553,111 news articles, their keyphrases, summaries, and some other information. The resulted data was put into human assessment, in a way which will be described, to ensure the quality of the keyphrases. The dataset is available at this project's GitHub repository ¹.

In the next section we discuss the different approaches to keyphrase extraction and generation in English and Persian. In section III, the dataset of news articles and their keyphrases will be introduced in details and the method of collecting and cleaning them will be discussed. In section IV different methods tested on the dataset are described in details. In section V and VI, the results are represented and discussed. We conclude this work and discuss the future work in section VII.

¹<https://github.com/edoost/perkey>

II. PREVIOUS WORK

A. Work on English

Although a substantial number of approaches to keyphrase extraction are unsupervised [3]–[6], recent supervised algorithms outperform the conventional unsupervised ones [1], [2]. Achieving such performance requires a large set of training data which is not achievable without a great deal of human resources, time, and financial cost. Instead, one can make use of already tagged data in the internet, even though they are not as accurate as such gold dataset. Such approach is taken in keyphrase extraction from tweets [1] and keyphrase generation in the field of scientific papers [2]. Some other publicly available datasets in English are [7], [8] for scientific papers, [9] for news articles, and [10] for web pages, none of which containing more than 1,000 samples, which makes them unsuitable for training neural networks.

B. Work on Persian

Literature of keyphrase/keyword extraction from Persian texts is quite minimal. Most of the papers on Persian take an unsupervised approach and non of them has released their data.

Mohammadi and Analoui [11], after removing stop words and stemming, extracts keywords using TF, TTF (Total Term Frequency), and DF matrices and fuzzification.

Khozani and Bayat [12], after stemming and removing stop words, uses $W(t_k) = \frac{tf(d_i, t_k) \times N}{df(t_k)}$ to obtain TFIDF weights of the words in documents. W being the weight, t_k the k 'th token, tf the term frequency, d_i the i 'th document, N number of tokens in the current document, and df the document frequency. After calculating the weights of the tokens, the weight of the bi-token keyphrases is calculated using a co-occurrence matrix.

Kian and Zahedi [13] uses attention attractive strings to improve keyword extraction from 800 documents from Hamshahri news collection [14]. Three to seven keyphrases were assigned to each document, the length of each keyphrase ranging from two to four words. The best result, an F_1 of 40.23, is achieved using attention attractive strings and training on 400 documents.

III. DATASET DESCRIPTION

We assessed more than 30 news websites and agencies and chose 6 of them having the highest quality keyphrases. The dataset comprises 553,111 news articles crawled from 6 news websites and agencies which provided high-quality keyphrases. Number of news articles crawled from each website/agency can be found in table I.

The dataset is stored in JSON format, each news article containing the following information:

```
{title, body, summary, keyphrases, category, url}
```

After character and word normalization, the articles with bodies containing less than 40 and more than 500

TABLE I
NUMBER OF ARTICLES BASED ON WEBSITE/AGENCY

website/agency	# of articles	% of total
Khabaronline	165,839	29.98%
Fararu	161,588	29.21%
Ilina	145,420	26.29%
Mashreq	33,621	6.07%
Alef	33,621	6.07%
Aftab	1,517	0.27%
total	553,111	100%

tokens were removed. This was to make sure that the articles are mostly news, as articles longer than 500 words tend to be interview or news analysis. More information on number of tokens in the body of each article can be found in table II.

TABLE II
NUMBER OF ARTICLES BASED ON NUMBER OF TOKENS IN THEM

# of tokens	# of articles	% of total
40-100	123,913	22.40%
100-150	96,919	17.52%
150-200	81,857	14.79%
200-250	66,738	12.06%
250-300	54,119	9.78%
300-350	42,826	7.74%
350-400	34,457	6.23%
400-450	28,194	5.09%
450-500	24,088	4.35%
total	553,111	100%

We also removed news articles with less than 2 keyphrases. It appears that most of the articles (65.11%) contain 2 or 3 keyphrases. More information on this can be found in table III.

TABLE III
NUMBER OF ARTICLES BASED ON NUMBER OF KEYPHRASES

# of keyphrases	# of articles	% of total
2	157,466	28.46%
3	202,748	36.65%
4	81,500	14.73%
5	56,278	10.17%
6	7,060	1.27%
7	2,883	0.52%
8	1,382	0.25%
9	1,032	0.18%
9+	42,762	7.73%
total	553,111	100%

Table IV shows number of keyphrases based on the number of characters in them. It appears that most of the keyphrases contain something between 5 to 15 characters. News articles containing keyphrases with less than 3 characters length were removed.

Table V shows that most of the keyphrases contain either 1 or 2 tokens. News articles containing keyphrases

TABLE IV
NUMBER OF KEYPHRASES BASED ON THE NUMBER OF CHARACTERS IN THEM

# of characters	# of keyphrases	% of total
3-5	276,141	13.52%
5-10	830,036	40.65%
10-15	574,945	28.16%
15-20	233,032	11.41%
20-25	82,354	4.03%
25-30	25,771	1.26%
30-35	14,955	0.73%
35-40	4,392	0.21%
total	2,041,626	100%

with more than 7 tokens in them were removed.

TABLE V
NUMBER OF KEYPHRASES BASED ON THE NUMBER OF TOKENS IN THEM

# of tokens	# of keyphrases	% of total
1	943,485	46.21%
2	746,148	36.54%
3	222,339	10.90%
4	89,746	4.40%
5	25,933	1.27%
6	10,800	0.52%
7	3,175	0.15%
total	2,041,626	100%

Finally, it appears that around 717k of the keyphrases, which constitute 35.13% of all keyphrases, are not present in the body of the news articles. This number shows that the dataset is suitable for the task of keyphrase generation, as well as extraction. More on this in table VI.

TABLE VI
NUMBER OF ABSENT AND PRESENT KEYPHRASES

	# of keyphrases	% of total
present	1,324,305	64.86%
absent	717,321	35.13%
total	2,041,626	100%

The dataset is divided to three parts, 25,000 for test set, 25,000 for validation set, and 503,111 for training set. The dataset can be downloaded from this project's GitHub repository referenced in the first page. To ensure the quality of the keyphrases, we conducted a survey, asking 5 participants to rate the precision and recall of the reference keyphrases. Participants were asked to answer two questions from 1 to 5 for 50 randomly chosen news articles, the two questions being:

- How much are the keyphrases related to the subjects discussed in the news article?
- To what extent are the subjects discussed in the article reflected in the keyphrases?

The average of the answers to the first question was 4.592 and to the second 3.980, the harmonic mean of which being 4.264, which ensures the quality of the keyphrases.

IV. METHODOLOGY OF THE TESTED ALGORITHMS

In this section the approaches to extract keyphrases from news articles will be discussed. We can divide the approaches into two main models, unsupervised and supervised. All the models are implemented using pke (an open source python-based keyphrase extraction toolkit) [15].

A. Unsupervised Models

The most important merit of unsupervised models is that they can work in a new area or language with little or no adjustments. We can divide the unsupervised techniques into two subsets: statistical models and graph-based models.

1) *Statistical Models*: Statistical models are models that select candidates based on their statistical features.

- TFIDF [16] calculates the importance of n -gram phrases in a corpus of documents. Here, IDF is calculated using below formula:

$$idf = \log(1 + \frac{N}{n_t})$$

N being the total number of documents in the corpus, and n_t the number of documents containing the term t .

- KPMine [17] works in three phases: (1) candidate selection, (2) candidate weight calculation, and (3) keyphrase refinement. The first phase, candidate selection, follows three heuristics: (a) the phrases should not be separated by punctuation or stop word, (b) the resulted phrases should have appeared at least n time(s) in the document (LASF: least allowable seen frequency), and (c) as phrases happening after a certain threshold in a document are very rarely keyphrases, they will be cut off. The candidate weight calculation phase is done using $w_{ij} = tf_{ij} \times idf \times B_i \times P_f$ where w_{ij} is weight of term t_j in document D_i , tf_{ij} is the frequency of term t_j in document D_i , idf is $\log_2(N/n)$ where N is the number of documents in the collection and n is number of documents where term t_j occurs at least once, P_f is the position factor, and B_i is the boosting factor associated with document D_i calculated using below formula:

$$B_d = |Nd|(|P_d| \times \alpha)$$

where if $B_d > \sigma$ then $B_d = \sigma$ and $|P_d|$ is the number of candidate terms whose lengths exceed one in the document and α and σ are hyperparameters. Finally, the weights of the top n keyphrases are refined by seeing if any of them is a sub-phrase of another. If so, its count is decremented by the frequency of the term of which it is a part and weights are recalculated. Here the cutoff threshold and LASF were tuned on

the development set and were set to 250 words and 3, respectively. We left α and σ to their default values, i.e. 2.3 and 3.0 respectively.

- YAKE [18], [19] is a multilingual online single-document keyphrase extraction tool which relies only on statistical features, and a list of stopwords, to extract keyphrases from unlabelled data. The process comprises six steps: (1) preprocessing: splitting the tokens based on word boundaries; (2) feature extraction, consisting of:
 - 1) Casing: regards the casing aspect of a word;
 - 2) Word Positional: values words occurring at the beginning;
 - 3) Word Frequency: scores more the words occurring more often;
 - 4) Word Relatedness to Context: computes the number of different terms that occur in the context of the candidate word on the basis that the number of different terms that co-occur with the candidate word has a direct relationship with the meaningfulness of it. In other words, the more diverse the context of the candidate, the more meaningless it is;
- 5) Word DifSentence: counts the number of times a candidate word appears within different sentences. (3) Individual terms are scored using the above-mentioned features; (4) Candidate keywords list are generated to the size of the window (a hyperparameter) and the scores are calculated as below such that the smaller the score the better the keyword will be:

$$S(kw) = \frac{\prod_{w \in kw} S(w)}{TF(kw) \times (1 + \sum_{w \in kw} S(w))}$$

where $S(kw)$ is the score of the candidate phrase calculated as the product of the scores of the constituent words normalized by the length and TF of the keyphrase, such that longer n-grams will not be scored higher and less frequent ones will be penalised. (5) Data is deduplicated using Levenshtein distance, and finally (6) the keyphrases are ranked such that the lower the $S(kw)$ score, the more important the keyword will be.

2) *Graph-based Models*: These approaches typically consist of two steps: (1) building a graph representation of the document with words as nodes and semantic relation between them as edges; (2) ranking nodes using a graph-theoretic measure and using the top-ranked ones as keyphrases.

- SingleRank [20] extracts candidates from sequences of nouns and adjectives and builds a word-based graph of the document and scores them using TextRank [21].
- TopicRank [22] improves SingleRank by grouping lexically similar candidates into topics and ranking

topics. After extracting sequences of nouns and adjectives as noun phrases, TopicRank clusters them into topics using a hierarchical agglomerative clustering algorithm with average linkage. Next, similar phrases are removed and the graph is built and weighted and final keyphrases are selected from first occurring candidates of the n highest ranked topics.

- Multipartite Rank [23] encodes topical information within a multipartite graph structure. This method works in two steps like other graph-based approaches. However, it also has a middle step in which weights are adjusted in order to capture position information. In the first step, keyphrase candidates are selected from what we define as noun phrase. As we are dealing with Persian language in this paper, the pattern for noun phrase would be the sequence of nouns with zero or more adjectives (/Noun+Adj*/). Then, based on the stemmed form of the words, the topics are found using hierarchical agglomerative clustering with average linkage. The candidate similarity is calculated as below:

$$sim(c_i, c_j) = \frac{|stems(c_i) \cap stems(c_j)|}{|stems(c_i) \cup stems(c_j)|}$$

A directed multipartite graph is built, the nodes of which are keyphrase candidates that are connected only if they belong to different topics and edges are weighted according to the distance between two candidates:

$$w_{ij} = \sum_{c_i \in t_i} \sum_{c_j \in t_j} dist(c_i, c_j)$$

c being the candidate and t the topic. Next, the most representative keyphrase candidates for each topic are selected according to their position in the document. In other words, candidates occurring at the beginning are promoted to the other candidates of the same topic. Finally, keyphrase candidates are ordered by TextRank ranking algorithm [21]:

$$S(c_i) = (1 - \lambda) + \lambda \cdot \sum_{c_j \in \mathcal{I}(c_i)} \frac{w_{ij} \cdot S(c_j)}{\sum_{c_k \in \mathcal{O}(c_j)} w_{jk}}$$

where $\mathcal{I}(c_i)$ is the set of predecessors of c_i , $\mathcal{O}(c_j)$ is the set of successors of c_j , and λ is a damping factor set to 0.85.

B. Supervised Model

Supervised models need annotated data and it takes time to train them. However, they often show superior performance and effectiveness comparing to unsupervised techniques.

- KEA [24] calculates two features for each sample: TFIDF and first occurrence. TFIDF is calculated using:

$$tfidf = \frac{freq(P, D)}{size(D)} \times -\log_2 \frac{df(P)}{N}$$

where P stands for phrase and $size()$ is the number of words. The second feature, first occurrence, is calculated as the number of words that precede the phrase's first appearance, divided by the number of words in the document. It results in a number between 0 and 1 indicating the percentage of the document that precedes phrase's first appearance. After discretization of the values of these two features for gold keyphrases, the probability of a phrase being a keyphrase is calculated as:

$$P(yes) = \frac{Y}{Y + N} \times P_{tfidf}(t|yes) \times P_{dist}(d|yes)$$

where t is the feature value for $tfidf$ and d for *distance*. Finally, the probability of keyphraseness is calculated as:

$$P(keyphraseness) = \frac{P(yes)}{P(yes) + P(no)}$$

V. RESULTS

We conducted an empirical study on 7 models, the results of which are shown in table VII, @5 meaning the results on the top five keyphrases and @10, top ten. We also repeated the experiment this time only on news articles with 3 and more keyphrases. The results are available in table VIII. The best results are in bold font, second best are underlined, and the third best results are in italics.

TABLE VII
EMPIRICAL RESULTS ON ALL THE DATA

Method	P@5	R@5	F ₁ @5	P@10	R@10	F ₁ @10
TFIDF	<i>.1459</i>	<u>.1959</u>	<i>.1673</i>	<i>.1036</i>	<u>.2758</u>	<i>.1507</i>
KPM.	.1625	<i>.1843</i>	<u>.1727</u>	.1404	<i>.2371</i>	.1763
YAKE	.0622	.0806	.0702	.0568	.1459	.0818
S.Rank	.0500	.0767	.0605	.0552	.1605	.0822
T.Rank	.0846	.1189	.0989	.0566	.1541	.0828
M.Rank	.0922	.1268	.1068	.0652	.1772	.0953
KEA	<u>.1564</u>	.2125	.1802	<u>.1110</u>	.2990	<u>.1620</u>

The results in table VII and also table VIII show that supervised method, KEA, performs the best, and after that, statistical methods tend to perform better in comparison to graph-based ones.

Comparing the results in table VII and VIII shows that keyphrases in news articles with fewer keyphrases do not cover all the subjects discussed in the news article, hence are of low quality.

TABLE VIII
EMPIRICAL RESULTS ON NEWS ARTICLES WITH AT LEAST THREE KEYPHRASES

Method	P@5	R@5	F ₁ @5	P@10	R@10	F ₁ @10
TFIDF	<i>.1724</i>	<u>.2060</u>	<i>.1877</i>	<i>.1216</i>	<u>.2877</u>	<i>.1710</i>
KPM.	.1900	<i>.1948</i>	<u>.1924</u>	.1632	<i>.2513</i>	.1979
YAKE	.0726	.0820	.0770	.0658	.1481	.0911
S.Rank	.0532	.0671	.0594	.0623	.1533	.0886
T.Rank	.0986	.1208	.1086	.0665	.1583	.0937
M.Rank	.1093	.1319	.1196	.0771	.1835	.1086
KEA	<u>.1837</u>	.2226	.2013	<u>.1300</u>	.3115	<u>.1835</u>

VI. DISCUSSION

Experimental results show that the supervised method, as opposed to what expected, cannot outperform KP-Miner. Apparently, this method is not powerful enough to benefit from such training data which emphasizes the need for more powerful state-of-the-art techniques. The tables also show that statistical unsupervised approaches outperform graph-based ones. That is also the case in English standard datasets [8].

Another experiment we performed, was removing news articles having fewer than three keyphrases from the data and repeating the evaluations. Supervised and unsupervised techniques showed higher precision, recall, and F₁-score when we eliminated those news articles. One reason could be the higher quality of the news articles with more keyphrases, in general, also supported by the quality assessment performed by human on the dataset.

VII. CONCLUSION AND FUTURE WORK

In this paper, we tested 7 keyphrase extraction models on two kinds of data. Limiting the news articles to those that have more keyphrases not only keeps news articles with more representative keyphrases, but also results in better performance in training of the supervised technique, i.e. KEA. Such results are promising for training state of the art supervised approaches, e.g. sequence labelling with artificial neural networks, as they performance drop when learning from very sparse data. We also observed that 35.13% of all the keyphrases are absent from news body. Absent keyphrases are prevalent in real texts as the subjects discussed are not always mentioned explicitly in them. Tackling such problem requires generative models, e.g. sequence to sequence models.

It is also important to improve evaluation schemes. Current evaluation convention in keyphrase extraction, i.e. precision, recall, and F₁-score, is too much strict, because different words can mean the same thing and refer to the same subject. [3] suggests some improvements to the criteria, for instance, using machine translation evaluation techniques, e.g. BLEU and Rouge, to achieve a more realistic evaluation for the task of keyphrase extraction and generation.

REFERENCES

- [1] Q. Zhang, Y. Wang, Y. Gong, and X. Huang, "Keyphrase extraction using deep recurrent neural networks on twitter," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 836–845.
- [2] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi, "Deep keyphrase generation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2017, pp. 582–592.
- [3] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2014, pp. 1262–1273.
- [4] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998.
- [5] Z. Liu, P. Li, Y. Zheng, and M. Sun, "Clustering to find exemplar terms for keyphrase extraction," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 2009, pp. 257–266.
- [6] Z. Liu, W. Huang, Y. Zheng, and M. Sun, "Automatic keyphrase extraction via topic decomposition," in *Proceedings of the 2010 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2010, pp. 366–376.
- [7] O. Medelyan, E. Frank, and I. H. Witten, "Human-competitive tagging using automatic keyphrase extraction," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*. Association for Computational Linguistics, 2009, pp. 1318–1327.
- [8] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2010, pp. 21–26.
- [9] X. Wan and J. Xiao, "Single document keyphrase extraction using neighborhood knowledge," in *AAAI*, vol. 8, 2008, pp. 855–860.
- [10] W.-t. Yih, J. Goodman, and V. R. Carvalho, "Finding advertising keywords on web pages," in *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006, pp. 213–222.
- [11] M. Mohammadi and M. Analoui, "Extracting keywords from persian documents," in *13th Annual Conference of Computer Society of Iran*. Computer Society, 2008.
- [12] S. M. H. Khozani and H. Bayat, "Specialization of keyword extraction approach to persian texts," in *Soft Computing and Pattern Recognition (SoCPaR), 2011 International Conference of*. IEEE, 2011, pp. 112–116.
- [13] H. Kian and M. Zahedi, "Improving precision in automatic keyword extraction using attention attractive strings," *Arabian Journal for Science and Engineering*, vol. 38, no. 8, pp. 2063–2068, 2013.
- [14] A. AleAhmad, H. Amiri, E. Darrudi, M. Rahgozar, and F. Oroumchian, "Hamshahri: A standard persian text collection," *Knowledge-Based Systems*, vol. 22, no. 5, pp. 382–387, 2009.
- [15] F. Boudin, "pke: an open source python-based keyphrase extraction toolkit," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*. Osaka, Japan: The COLING 2016 Organizing Committee, December 2016, pp. 69–73. [Online]. Available: <http://aclweb.org/anthology/C16-2015>
- [16] S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Automatic keyphrase extraction from scientific articles," *Language resources and evaluation*, vol. 47, no. 3, pp. 723–742, 2013.
- [17] S. R. El-Beltagy and A. Rafea, "Kp-miner: A keyphrase extraction system for english and arabic documents," *Information Systems*, vol. 34, no. 1, pp. 132–144, 2009.
- [18] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, "A text feature based automatic keyword extraction method for single documents," in *European Conference on Information Retrieval*. Springer, 2018, pp. 684–691.
- [19] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and a. Jatowt, "Yake! collection-independent automatic keyword extractor," in *European Conference on Information Retrieval*. Springer, 2018, pp. 806–810.
- [20] X. Wan and J. Xiao, "Collabrank: towards a collaborative approach to single-document keyphrase extraction," in *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, 2008, pp. 969–976.
- [21] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.
- [22] A. Bougouin, F. Boudin, and B. Daille, "Topicrank: Graph-based topic ranking for keyphrase extraction," in *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, 2013, pp. 543–551.
- [23] F. Boudin, "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)*, vol. 2, 2018, pp. 667–672.
- [24] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: Practical automated keyphrase extraction," in *Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*. IGI Global, 2005, pp. 129–152.