ISSN 2007-9737

Stop-Word Lists in Keyphrase Extraction: Their Influence and Comparison

Svetlana Popova[1], Mikhail Alexandrov[2,3,*], John Cardiff[1]

[1] Technological University Dublin,
Ireland

[2] Russian Academy of National Economy and Public Administration, Moscow,
Russia

[3] FRUCT Association,
Finland

svp@list.ru, malexandrov@mail.ru, john.cardiff@it-tallaght.ie

Abstract. Keyphrases provide a compact representation of a document's content and are useful in Web search systems, text data mining, and natural language processing applications. The keyphrase extraction domain has been developing for a long time, and achieving further improvements is becoming increasingly challenging. Algorithms compete for minimal gains, highlighting the significance of demonstrating ways to enhance the quality of both existing algorithms and those yet to be developed. This article aims to demonstrate and approve a simple way to enhance keyphrase extraction algorithms by using extended stop words. This enables the improvement of keyphrase extraction algorithms on average by 4% and more.Nevertheless, no studies have been conducted that compare different stop-word lists and their impact on the domain. Our goal is to overcome this gap. We compared the impact of both existing extended and standard stop-word lists on the performance of 10 unsupervised keyphrase extraction algorithms across 5 datasets (a total of 10 sub-datasets were used). We aimed to highlight that researching methods for constructing and using extended stop-word lists deserves attention and could become one of the sub-directions in the keyphrase extraction domain. Extended stop words, when a suitable list is selected, consistently enhance the performance of algorithms in a stable and statistically significant manner. Based on the obtained results, we can assume that knowing the type of text from which keyphrases need to be extracted allows us to select the most appropriate stop-word list.

Keywords. Keyphrase extraction, stop words, NLP.

1 Introduction and Related Work

Keyphrases succinctly summarize a document's content and the process of automatic keyphrase extraction involves the automated identification of significant and topic-relevant phrases from a text. Keyphrases play an important role in enhancing the capabilities of information retrieval systems [15, 3, 37, 6, 12] and contribute to natural language processing applications, e.g. document clustering [16], text classification [20], opinion mining [2], text summarization [21, 9], web tagging [27], and more, making the extraction of keyphrases an important area of data mining. Typically, keyphrases range from one to five words in length. The example of text (scientific abstract) with its keyphrases is presented in Table 1.

This study is dedicated to exploring unsupervised methods based on keyphrase candidate extraction. These methods include two steps: initially, potential keyphrases are identified and extracted from the text; subsequently, these candidates are scored and ranked to determine the final keyphrases. Unsupervised keyphrase extraction techniques can be categorized into

Table 1. Example: scientific abstract and keyphrases from the INSPEC dataset

E-government.The author provides an introduction to the main issues surrounding E-government **modernisation** and **electronic delivery** of all **public services** by 2005. The author makes it clear that E-government is about transformation, not computers and hints at the special **legal issues** which may arise.

graph-based [29, 35, 14, 7, 5], statistical-based [31, 13, 8] and embedding/transformer-based [1, 23, 25, 10, 24, 34, 11, 33] approaches.

The results in the field are still far from high, and algorithms compete with each other for minimal improvements. Therefore, even small improvements play a role when comparing algorithms that extract keyphrases. In [36], the work of [26] is cited as the work where authors remove some words that are too common to be keywords. In our paper we call this kind of words: extended stop words. The list of these words from [26] is not publicly available [36].

[36] reports the 5% drop in performance of [26] approach without this list. It indicates that the influence of incorporating extended stop-words can be tangible. These 5% allow the algorithm from [26] to outperform the results of the algorithm from [36]. Without removing the specified words, the approach from [36] performs better than [26]. However, only a few studies specifically RAKE [31] and our study [30] focus on extended stop words in the keyphrase extraction domain and propose approaches to automate the process of extracting these words from the texts.

The authors of YAKE [8] also use a feature that helps reduce the weight of common words that do not reflect the context. The Word Relatedness to Context feature in YAKE looks like an attempt to find words similar to stop words in a hidden form. There are no studies in the field that compare different extended stop-word lists or examine their impact on the performance of existing algorithms.

To address this issue, we compared different stop-word lists (common and extended) on 10 unsupervised KE algorithms and 10 subsets of five datasets. Obtained results demonstrate that exploiting different stop-word lists affects the quality with which algorithms process test collections, and extended stop-word lists can noticeably improve the evaluation quality of algorithms.

An additional aim of this work is to highlight that researching methods for constructing and using extended stop-word lists deserves attention and could be one of the sub-directions in the KE domain.

Extended stop words enhance the performance of algorithms. At the end of the paper, we will show that improvement is achieved in most cases when extended stop-word lists are used with an average improvement of 4.5%-6% (in some cases, such improvement reaches up to 16%).

All studies in this work were conducted using datasets consisting of abstracts of scientific papers. Keyphrase extraction from these types of texts has attracted attention due to the active development of electronic libraries and e-learning platforms. Experiments were performed based on the Python-based Keyphrase Extraction framework (PKE) [4], which guarantees their correctness and reproducibility.

2 Keyphrase Extraction Problem, PKE Settings, Evaluation, and Datasets

2.1 Keyphrase Extraction

Keyphrase Extraction is defined as follows. Let $D = \{d_1,\ d_2,\ \ldots,\ d_n\}$ be a set of n documents. Each document $d_i \in D$ has reference phrases (also called gold standard) – a set of keyphrases predefined by the experts $C_i = \{c_{i_1},\ c_{i_2},\ \ldots,\ c_{i_m}\}$. The goal of an unsupervised keyphrase extraction approach is for each text $d_i \in D$ automatically extract a list of keyphrase candidates, score them, create a ranked list, and select k (@k) top-ranked phrases as keyphrases $G_i = \{g_{i_1},\ g_{i_2},\ \ldots,\ g_{i_k}\}$ that should match the set of reference phrases as precisely as possible in terms of exact match Precision, Recall and $F1_{score}$ - F1 (described below in subsection 2.3 Evaluation).

2.2 PKE and Algorithms Settings

To guarantee the correctness of the implementation of the algorithms exploited in the research and the reproducibility we used PKE framework. PKE [4], a comprehensive Python-based Keyphrase Extraction framework, incorporates implementations of keyphrase extraction techniques that were state-of-the-art at the time of its creation.

To operate identically with all the unsupervised keyphrase extraction algorithms involved in the study we exploited all of them in the same way: as candidate-based approaches similar to how it was done in PKE paper [4]. We exploited the following unsupervised methods implemented in PKE:

Graph-based (TextRank, SingleRank, TopicRank, PositionRank, TopicalPageRank, MultipartiteRank) and statistics-based (FirstPhrase, TfIdf, KP-Miner, YAKE) [4].

In all experiments, candidate phrases are extracted from the texts as continuous sequences of nouns and adjectives that are not stop words and satisfy the following pattern: $< \mathrm{ADJ} >^* < \mathrm{NOUN} | \mathrm{PROPN} >+$. Consequently, all exploited methods differ only in the ranking step.

2.3 Evaluation

2.3.1 Keyphrase Extraction Evaluation

PKE evaluates an algorithm using the exact match macro-average $\mathrm{F1}_{\mathrm{score}}$@k, "@k" means that only k keyphrases can be extracted for each text. $\mathrm{F1}_{\mathrm{score}}$ comparing a set of automatically extracted keyphrases for the text with a set of keyphrases marked for the same text by experts (reference keyphrases).

The score for each text d_i is calculated by comparing the set of keyphrases extracted for the text (G_i) with the set of reference phrases from that text (C_i). $\mathrm{Precision}_{d_i}$ and Recall_{d_i} are calculated, based on which the $\mathrm{F1}_{\mathrm{score}_{d_i}}$ is determined as follows:

$$\mathrm{Precision}_{d_i} = \frac{|(C_i \cap G_i)|}{|G_i|}, \quad (1)$$

$$\mathrm{Recall}_{d_i} = \frac{|(C_i \cap G_i)|}{|C_i|}, \quad (2)$$

$$\mathrm{F1}_{\mathrm{score}_{d_i}} = \frac{2 \times \mathrm{Precision}_{d_i} \times \mathrm{Recall}_{d_i}}{\mathrm{Precision}_{d_i} + \mathrm{Recall}_{d_i}}, \quad (3)$$

where $|C_i \cap G_i|$ - is the number of correctly extracted phrases for text d_i, $|G_i|$ - is the number of phrases automatically extracted by the algorithm from the text, $|C_i|$ - is the number of reference phrases for text d_i.

We use k=10 as the most frequent way of evaluation in the domain. We do not remove phrases that do not occur in the corresponding text from reference keyphrases. This practice exists in the domain, e.g. in [19, 35] and it makes Recall and $\mathrm{F}_{\mathrm{score}}$ higher. Following standard practice in the domain, we performed stemming on all phrases before evaluating their quality.

PKE uses an exact match $\mathrm{F1}_{\mathrm{score}}$: An automatically extracted keyphrase is considered a true positive if the reference phrases contain the same phrase. If there is a semantically equivalent but visually distinct phrase, it is considered a false positive. This is an evaluation error, one of the errors described in [17] that causes low-performance quality in the domain.

Despite this, the $\mathrm{F1}_{\mathrm{score}}$ is exploited in most papers to compare keyphrase extraction algorithms and is the main and standard evaluation approach in the domain. Throughout the text, the quality of the performance of keyphrase extraction algorithms will be understood as their $\mathrm{F1}_{\mathrm{score}}$ evaluation on test datasets.

2.3.2 Statistical Tests

We use statistical tests to demonstrate statistically significant differences in the performance of 10 unsupervised keyphrase extraction algorithms when they exploit different stop-word lists. We compare the results of the algorithms that exploited two different stop-word lists pairwise. The Wilcoxon signed-rank test is used.

By the 'better quality of a stop-word list' or 'more suitable list', we mean the following. Consider two stop-word lists - list A and list B. If, when exploiting list A, the keyphrase extraction algorithm performs better in terms of evaluation in $\mathrm{F1}_{\mathrm{score}}$ compared to the same algorithm using stop words from list B, then we assert that stop-words list A has better quality and is more suitable.

Table 2. Datasets description: ”k.p. per text” = average number of keyphrases per text in the subset, ”pr.” = present (indicate what % of keyphrases from references appear in the document), ch. = characters

Datasets	num. of doc.	doc. descriptions	assigned by	k.p. per text	pr. %
INSPEC	domains: Computers and Control and Information Technology				
train	1,000	title+abstract	reader	9.79	78.00
validation	500	title+abstract	reader	9.15	77.96
test	500	title+abstract	reader	9.83	78.70
SemEval (TA)	domains: Distributed Systems, Information Searchand Retrieval, Distributed Artificial Intelligence - Multiagent Systems, Socialand Behavioral Sciences - Economics				
train	144	title+abstract	reader+author	15.44	42.16
test	100	title+abstract	reader+author	14.66	40.11
kp20k	domains: Computer Science				
test	20,000	title+abstract	author	5.28	58.40
validation	20,000	title+abstract	author	5.27	58.20
PubMed test dataset	domain: Biomedical				
first 500 doc.	500	title+first 1,200 ch.	author	5.40	84.54
last 500 doc	500	title+first 1,200 ch.	author	5.40	84.54
KPBiomed test dataset	domains: Biomedical				
first 2,000 doc.	2,000	title+abstract	author	5.22	66.59
second 2,000 doc.	2,000	title+abstract	author	5.22	66.59

2.4 Datasets

All test collections contain short texts and are from a scientific domain. We rely on PKE built-in well-known datasets. All collections are taken from a single repository that PKE works with [1]. These include:

- **INSPEC** [2] [19]: as test collections we use ”test” and ”validation” subsets. There are 500 scientific publication titles with abstracts in each subset with uncontr (reader) manually assigned reference keyphrases.
- **SemEval2010** [3] [22] dataset with 100 full texts in the ”test” subset and 144 full texts in the ”train” subset. Both subsets for each text have combined manual author- and reader-assigned keyphrases as reference keyphrases. We exploited the SemEval2010 dataset in the following format. SemEval2010(TA) includes only titles and abstracts of articles, making it similar to the INSPEC collection.
- **kp20k** [4] [28]: ”test” and ”validation” subsets were used for evaluation. Each subset includes 20,000 abstracts with titles from scientific articles with author-assigned keyphrases.
- **Pubmed** [5] [32]: dataset contains 1,320 articles with full text and author-assigned keyphrases. Titles are separated from full texts but abstracts are not. For each text, we took the title and the first 1,200 characters of the full text, assuming that in this way we would be able to use most of the abstracts. We created two subsets: the first

[1] huggingface.co/taln-ls2n
[2] huggingface.co/datasets/taln-ls2n/inspec
[3] huggingface.co/datasets/taln-ls2n/semeval-2010-pre
[4] huggingface.co/datasets/taln-ls2n/kp20k
[5] huggingface.co/datasets/taln-ls2n/pubmed

subset includes the first 500 documents from the database and the the second subset consists of the last 500 texts from the database. These subsets have no intersection.

– **KPBiomed**[6] [18]: ”test” subset of this dataset includes 20,000 abstracts and titles with author-assigned keyphrases. We created two subsets from ”test”: the first subset includes the first 2,000 texts from the database and the the second subset consists of the second 2,000 documents from the database. These subsets have no intersection. We chose this subset size because processing a collection of 20,000 documents across all experiments takes quite a long time.

The first three collections contain texts primarily from Computer Science, while the latter two contain texts from the domain of Biomedicine. From the same repository where these collections are available, we took statistics, which are combined into a Table 2.

3 Experiment Description and Results

In this section, we will compare how different stop-word lists affect the quality of the keyphrase extraction algorithms. Standard and extended stop-word lists will be used. Before we move on to the description of the experiment, consider the methods for building extended stop-word lists.

3.1 Extended Stop Word Lists Extraction

There are only two algorithms in RAKE [31] and in our research [30] for automatic extraction words that are too common to be a part of a keyphrase. In both articles, these words act as delimiters between phrases. In experiments, we exploited these words in the same way as stop words and we call them: extended stop words. Both algorithms [31] and [30] in original papers extract additional stop words (phrase delimiters) using the same source: INSPEC ”train” documents set [19]. Therefore, the extended stop word lists obtained by each approach differ only due to the differences between the methods used to extract these lists.

[6] huggingface.co/datasets/taln-ls2n/kpbiomed

3.1.1 Phrase Delimiters Extraction in RAKE

RAKE is one of the most rapid algorithms. The authors proposed a method for extracting words that act as phrase delimiters. RAKE uses them together with other phrase delimiters, e.g. punctuation or common stop words, to split the longest sequences of continuous words that are extracted as candidates.

A phrase delimiters list is created based on the INSPEC train dataset [19]: the set of documents with labeled keyphrases. The method picks words with a document frequency higher than a threshold $df>10$ that occur more frequently as adjacent to keyphrases than within them and appends these words to the delimiters list. The obtained delimiters list merged with standard stop words we call **”RAKE-PD”**.

The obtained list of delimiters improves RAKE’s performance on the INSPEC test dataset compared to using the Fox stop words [31]. Examples of words from this list: ”the, and, of, a, in, is, for, to, we, this, are, with, as, on, it, an, that, which, by, using, can, paper, from, be, based, has, was, have, or, at, such, also, but, results, proposed, show, new, these, used, however, our, were, when, one, not, two, study, present, ...”(this list is the first part from example in the original paper [31]).

3.1.2 Extended Stop-word List Extraction: Alternative Approach

We suggested another way to extract extended stop words for keyphrase extraction [30]. Keyphrases are extracted as longest sequences of contiguous nouns and adjectives split at phrase delimiters: punctuation, extended, and common stop-word positions. There is no ranking step in [30]. Here the extended stop-word list is built based on a set of documents annotated with keyphrases (based on the INSPEC train dataset similar to RAKE).

The approach iterates over a set of nouns and adjectives in the training dataset vocabulary. It measures the $F1_{score}$ increase in performance produced by the keyphrase extraction algorithm on INSPEC train if this current word is considered as a stop word.

Table 3. Evaluation of keyphrase extraction approaches exploited different stop-words lists in terms of $F1_{score}@10$. "SW list" = Stop-Word list name. The best results are highlighted in bold. The second-highest results are marked in italics with *. ESW = ExtentedSW list, RAKE = RAKE-PD list

SW list− >	NLTK	ESW	RAKE	Fox	Smart	NLTK	ESW	RAKE	Fox	Smart
INSPEC			**test**					**validation**		
FirstPhr.	28.48	**30.09**	28.66	28.40	* *28.75*	28.75	29.37	26.70	28.25	* *28.78*
TextR.	34.78	**37.26**	35.64	35.39	* *36.15*	33.60	**35.12**	32.62	33.49	* *34.68*
SingleR.	34.77	**36.51**	35.23	35.03	* *35.66*	33.90	**34.92**	32.44	33.65	* *34.56*
TopicR.	28.43	**29.56**	27.65	28.45	* *28.52*	27.78	**28.51**	25.90	27.47	* *27.91*
Multipar.R.	29.34	**30.47**	28.71	* *29.38*	29.37	28.82	**29.64**	27.35	28.76	* *29.20*
PositionR.	33.48	**34.83**	33.38	33.47	* *34.05*	33.09	**33.82**	31.30	32.87	* *33.66*
TopicalP.R.	34.44	**36.28**	35.14	34.88	* *35.31*	33.54	* *34.15*	31.88	33.43	**34.32**
Tf-Idf	35.46	**36.29**	34.36	35.24	* *35.72*	33.71	* *34.22*	31.99	33.80	**34.41**
KP-Miner	33.81	**35.07**	33.99	34.52	* *34.94*	32.62	* *33.49*	31.10	32.90	**33.61**
YAKE	35.08	**36.00**	34.06	34.78	* *35.49*	33.60	**33.96**	31.32	32.98	* *33.82*
SemEval2010(TA)			**test**					**train**		
FirstPhr.	15.37	**16.61**	* *15.46*	15.94	15.40	* *17.11*	**17.74**	16.47	16.63	16.56
TextR.	13.95	* *15.84*	**16.23**	14.99	15.13	16.01	**17.58**	* *17.20*	16.25	15.95
SingleR.	17.38	**18.40**	* *17.93*	17.80	18.30	18.00	**19.31**	* *18.68*	18.35	17.99
TopicR.	14.79	**15.06**	14.35	* *14.91*	14.81	16.40	**17.04**	15.97	* *16.45*	15.95
Multipar.R.	15.38	**16.06**	14.82	* *15.95*	15.35	* *17.27*	**18.24**	16.51	17.03	16.88
PositionR.	17.58	**18.22**	17.04	17.80	* *18.00*	18.94	**20.29**	* *19.81*	19.11	18.40
TopicalP.R.	16.82	**18.10**	* *17.28*	17.84	17.83	18.26	**19.56**	* *18.61*	18.54	18.17
Tf-Idf	16.35	**16.83**	15.82	* *16.64*	16.27	* *18.61*	**19.12**	18.12	18.12	17.79
KP-Miner	17.22	**17.88**	* *17.54*	17.59	17.53	18.56	**19.63**	18.53	18.40	* *18.66*
YAKE	18.64	* *18.55*	17.10	18.43	18.38	19.63	* *19.49*	18.76	19.46	19.33
kp20k			**test**					**validation**		
FirstPhr.	13.50	**13.99**	13.42	* *13.66*	13.53	13.58	**14.13**	13.55	* *13.74*	13.63
TextR.	10.01	**10.95**	**10.95**	10.60	10.49	10.18	**11.11**	* *11.05*	10.85	10.75
SingleR.	12.52	**13.16**	12.91	* *13.00*	12.92	12.64	**13.31**	13.01	* *13.17*	13.06
TopicR.	11.97	**12.35**	11.91	* *12.17*	12.06	12.00	**12.41**	11.92	* *12.19*	12.09
Multipar.R.	13.55	**13.95**	13.40	* *13.77*	13.66	13.60	**14.02**	13.42	* *13.78*	13.66
PositionR.	14.08	**14.57**	14.17	* *14.38*	14.33	14.10	**14.65**	14.19	* *14.45*	14.37
TopicalP.R.	12.80	**13.40**	13.18	* *13.28*	13.19	12.95	**13.58**	13.27	* *13.44*	13.33
Tf-Idf	12.14	**12.64**	* *12.52*	12.49	12.46	12.27	**12.80**	* *12.59*	12.60	12.59
KP-Miner	14.05	**14.46**	14.03	* *14.30*	14.29	14.26	**14.66**	14.11	* *14.51*	14.46
YAKE	14.68	**15.08**	14.60	* *14.88*	14.81	14.75	**15.22**	14.64	* *14.95*	14.89

If a given improvement exceeds the threshold h (h=0.0001), the word is labeled as a stop word. After every word is iterated, all words labeled as stop words are added to the final extended stop-word list as well as standard stop words from NLTK.

This final stop-words list we call **"ExtendedSW"**. Examples of words from this list: "entire, results, various, extensions, input, main, many, number, different, way, available, large, certain, ..." (this list is the first part from the original paper [30]).

Table 4. Evaluation of keyphrase extraction approaches exploited different stop-words lists in terms of $F1_{score}@10$. "SW list" = Stop-Word list name. The best results are highlighted in bold. The second-highest results are marked with an underscore. ESW = ExtentedSW list, RAKE = RAKE-PD list

SW list$->$	NLTK	ESW	RAKE	Fox	Smart	NLTK	ESW	RAKE	Fox	Smart
Pubmed			**first 500 doc.**					**last 500 doc.**		
FirstPhr.	14.71	* *15.45*	**16.39**	14.83	14.89	15.74	* *16.19*	**16.55**	15.66	15.83
TextR.	7.55	* *8.20*	**8.68**	8.03	7.98	8.20	8.47	**8.92**	8.38	* *8.55*
SingleR.	11.73	* *12.40*	**12.71**	11.96	11.87	12.37	* *13.25*	**13.45**	12.74	12.78
TopicR.	14.04	14.02	**14.74**	* *14.31*	14.21	14.37	**14.52**	14.43	14.29	* *14.44*
Multipar.R.	15.81	* *16.15*	**17.07**	16.09	15.98	16.32	**16.84**	* *16.75*	16.28	16.24
PositionR.	14.57	* *15.18*	**15.88**	14.84	14.81	15.12	* *15.70*	**16.09**	15.31	15.45
TopicalP.R.	12.15	* *12.74*	**13.21**	12.61	12.59	12.77	* *13.45*	**13.98**	13.07	13.17
Tf-Idf	15.92	* *16.38*	**16.83**	16.11	16.16	16.34	* *16.59*	**16.93**	16.30	16.58
KP-Miner	16.49	* *16.59*	**16.82**	16.38	16.48	16.97	17.05	**17.31**	17.04	* *17.15*
YAKE	16.05	* *16.61*	**17.42**	16.09	16.35	16.46	* *16.96*	**17.22**	16.60	16.75
KPBiomed			**first 2000 doc.**					**second 2000 doc.**		
FirstPhr.	15.72	* *16.26*	**16.42**	15.74	15.92	15.60	* *16.22*	**16.26**	15.63	15.69
TextR.	6.91	* *7.57*	**7.92**	7.54	7.26	6.92	* *7.63*	**7.84**	7.32	7.30
SingleR.	10.95	* *11.44*	**11.77**	11.43	11.29	11.15	* *11.91*	**12.21**	11.63	11.56
TopicR.	13.49	* *13.94*	**14.06**	13.80	13.76	13.41	* *13.72*	**13.75**	13.62	13.61
Multipar.R.	15.71	* *16.10*	**16.28**	15.93	15.86	15.77	* *16.19*	**16.22**	16.00	15.91
PositionR.	13.84	* *14.38*	**14.66**	14.22	14.10	14.29	* *14.80*	**15.11**	14.61	14.59
TopicalP.R.	11.10	* *11.73*	**12.09**	11.53	11.43	11.23	* *12.15*	**12.16**	11.79	11.71
Tf-Idf	15.83	* *16.06*	**16.24**	16.05	16.00	16.08	16.31	**16.44**	* *16.37*	16.35
KP-Miner	16.68	* *16.95*	**17.09**	16.79	16.75	16.70	16.86	**17.00**	* *16.91*	16.87
YAKE	15.88	* *16.34*	**16.54**	16.18	16.10	16.19	* *16.69*	**16.73**	16.51	16.42

3.2 Different Stop-word Lists Comparison

3.2.1 Experiment Description

We examined the impact of various stop-word lists on the performance of keyphrase extraction methods. We took the standard stop-word list from the NLTK, as well as the FOX[7] and SMART[8] stoplists previously tested in the domain of keyphrase extraction, and extended stop-word lists referenced in [31] RAKE (RAKE-PD) and in [30] (ExtendedSW). Each stop-word list was exploited in the work of each of the 10 unsupervised keyphrase extraction algorithms: TextRank, SingleRank, TopicRank, PositionRank, TopicalPageRank, MultipartiteRank, FirstPhrase, TfIdf, KP-Miner, and YAKE. Table 3 and Table 4 presents the results for each dataset. We conducted the Wilcoxon signed-rank tests to check whether some specific stop-word lists statistically significantly improved the quality of the algorithms compared with exploiting other stop-word lists.

[7] github.com/aneesha/RAKE/blob/master/FoxStoplist.txt

[8] github.com/aneesha/RAKE/blob/master/SmartStoplist.txt

3.2.2 Results and Discussion

The results presented in Table 3 and Table 4 allow us to draw the following conclusions.

– On all five datasets, the extended stop word lists help the algorithms achieve the best results (there are only several exceptions). On the first three datasets, the best algorithm performance is achieved with the ExtendedSW list (ESW).

On the remaining two datasets, the RAKE-PD list outperforms ExtendedSW; however, ExtendedSW consistently remains the second most effective for these collections.

– The ExtendedSW stop word list allows algorithms to achieve the highest results for datasets related to the field of Computer Science, with only a few exceptions across all experiments. In the case of datasets from the Biomedical Sciences, the ExtendedSW list almost always yields the second-best results compared to the RAKE-PD stop word list.

  In other words, the ExtendedSW list consistently shows the best or second-best quality in nearly all experiments (except for 5 cases out of 100). The RAKE-PD stop word list enables algorithms to achieve the highest results on the two datasets from the field of Biomedicine, but in most cases, on three Computer Science datasets, this stop word list performs worse than the SMART or FOX lists.

  Therefore, we assume that the ExtendedSW list generally performs better than the RAKE-PD. Additionally, note that RAKE is a patented algorithm.

– The results obtained for stop-word lists on different subsets of the same datasets are closely similar. We can assume that an optimal stop word list for a given type of text can be selected using a subset of such texts for which reference keyphrases are available.

– On average, across all combinations of datasets and algorithms, ExtendedSW improves the performance of keyphrase extraction algorithms by 4% compared to the commonly used NLTK stop-word list. When considering only the datasets related to Computer Science, this improvement is 4.5%.

  ExtendedSW improves the performance of the algorithms in 98 out of 100 experiments. Compared to NLTK, the RAKE-PD list improved the performance of keyphrase extraction algorithms by 6% on Biomedical datasets. However, in more than half of the experiments on the datasets from Computer Science, exploiting RAKE-PD did not improve keyphrase extraction. Here, RAKE-PD falls behind ExtendedSW.

We conducted statistical tests to demonstrate that the results obtained on each Computer Science dataset using the list ExtendedSW statistically significantly improved algorithms' results achieved with the other stop-words lists: NLTK, FOX, SMART, and RAKE-PD. The Wilcoxon signed-rank test was used. The statistical test revealed statistically significant differences at the p-values 0.01 or 0.05 in all cases except one.

It is the comparison case with SMART list on the INSPEC "validation" subset where p-value=0.08. All other cases indicate that using the list ExtendedSW statistically significantly improved algorithms' results obtained with the other stop-words lists. The same for the RAKE-DP stop words on Biomedical datasets.

4 Conclusions

This work aimed to compare different stop-word lists and their impact on the keyphrase extraction domain. We compared standard and extended stop-word lists. We want to highlight that researching methods for constructing and using extended stop-word lists deserves attention. Experiments with 10 different unsupervised keyphrase extraction algorithms on 10 subsets from 5 different datasets show that extended stop-word lists allow the algorithms to achieve the best performance.

Obtained results show that the stop-word lists that allow keyphrase extraction algorithms to achieve the highest performance are very similar across different subsets of the same datasets. Additionally, we observed that the choice of a stop word list depends on the domain. For all datasets related to Computer Science, the best algorithm performance was achieved using the same extended stop-word list.

For Biomedical datasets, a different extended stop-word list proved to be the most suitable, but it was the same list across all Biomedical datasets. We assume that if we know the type of texts from which keyphrases need to be extracted, we can select the most appropriate stop-word list. On the domain-specific

datasets used in this study, extended stop-word lists enabled keyphrase extraction algorithms to achieve maximum performance, improving their quality by an average of 4.5% to 6%, with some algorithms showing up to a 16% improvement compared to using the standard NLTK stop-word list. These improvements justify the development of approaches for the automatic extraction of extended stop-word lists for keyphrase extraction tasks. The results also allow us to assume that the extended stop word list ExtendedSW is a good alternative to the extended stop words (phrase delimiters) extracted in the patented RAKE algorithm.

References

1. **Bennani-Smires, K., Musat, C., Hossmann, A., Baeriswyl, M., Jaggi, M. (2018).** Simple unsupervised keyphrase extraction using sentence embeddings. DOI: 10.48550 /arXiv.1801.04470.
2. **Berend, G. (2011).** Opinion expression mining by exploiting keyphrase extraction. Proceedings of the 5th International Joint Conference on Natural Language Processing, pp. 11621170.
3. **Bernardini, A., Carpineto, C., D'Amico, M. (2009).** Full-subtopic retrieval with keyphrase-based search results clustering. Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, Vol. 1, pp. 206–213. DOI: 10.1109/WI-IAT.2009.37.
4. **Boudin, F. (2016).** pke: An open source python-based keyphrase extraction toolkit. Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations, Osaka, Japan, pp. 69–73.
5. **Boudin, F. (2018).** Unsupervised keyphrase extraction with multipartite graphs. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Vol. 2, pp. 667–672. DOI: 10.18653/v1/N18-2105.
6. **Boudin, F., Gallina, Y., Aizawa, A. (2020).** Keyphrase generation for scientific document retrieval. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, pp. 1118–1126. DOI: 10.1 8653/v1/2020.acl-main.105.
7. **Bougouin, A., Boudin, F., Daille, B. (2013).** TopicRank: Graph-based topic ranking for keyphrase extraction. Proceedings of the Sixth International Joint Conference on Natural Language Processing, Asian Federation of Natural Language Processing, Nagoya, Japan, pp. 543–551.
8. **Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., Jatowt, A. (2020).** Yake! keyword extraction from single documents using multiple local features. Information Sciences, Vol. 509, pp. 257–289. DOI: 10.1016/j.ins.2019.09.013.
9. **D'Avanzo, E., Magnini, B. (2005).** A keyphrase-based approach to summarization: The lake system at DUC-2005. Proceedings of DUC.
10. **Ding, H., Luo, X. (2021).** Attentionrank: Unsupervised keyphrase extraction using self and cross attentions. Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1919–1928. DOI: 10.18653/v1/2021.emnlp-main.146.
11. **Ding, H., Luo, X. (2022).** AGRank: Augmented graph-based unsupervised keyphrase extraction. Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, Association for Computational Linguistics, pp. 230–239.
12. **Du, H., Thudumu, S., Giardina, A., Vasa, R., Mouzakis, K., Jiang, L., Chisholm, J., Bista, S. (2023).** Contextual topic discovery using unsupervised keyphrase extraction and

hierarchical semantic graph model. Journal of Big Data, Vol. 10, No. 1. DOI: 10.1186/s40537-023-00833-1.

13. **El-Beltagy, S. R., Rafea, A. A. (2010).** Kp-miner: Participation in SemEval-2. Proceedings of the 5th international workshop on semantic evaluation, pp. 190–193.
14. **Florescu, C., Caragea, C. (2017).** PositionRank: An unsupervised approach to keyphrase extraction from scholarly documents. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Vancouver, Canada, pp. 1105–1115. DOI: 10.18653/v1/P17-1102.
15. **Gutwin, C., Paynter, G., Witten, I., Nevill-Manning, C., Frank, E. (1999).** Improving browsing in digital libraries with keyphrase indexes. Decision Support Systems, Vol. 27, pp. 81–104. DOI: 10.1016/S0167-9236(99)00038-X.
16. **Hammouda, K. M., Matute, D. N., Kamel, M. S. (2005).** Corephrase: Keyphrase extraction for document clustering. Proceedings of the 4th International Conference on Machine Learning and Data Mining in Pattern Recognition, Springer-Verlag, pp. 265–274. DOI: 10.1007/11510888_26.
17. **Hasan, K. S., Ng, V. (2014).** Automatic keyphrase extraction: A survey of the state of the art. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Baltimore, Maryland, pp. 1262–1273. DOI: 10.3115/v1/P14-1119.
18. **Houbre, M., Boudin, F., Daille, B. (2022).** A large-scale dataset for biomedical keyphrase generation. Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI), Association for Computational Linguistics, pp. 47–53. DOI: 10.18653/v1/2022.louhi-1.6.
19. **Hulth, A. (2003).** Improved automatic keyword extraction given more linguistic knowledge. Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, pp. 216–223. DOI: 10.3115/1119355.1119383.
20. **Hulth, A., Megyesi, B. B. (2006).** A study on automatically extracted keywords in text categorization. Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, USA, pp. 537–544. DOI: 10.3115/1220175.1220243.
21. **Jones, S., Lundy, S., Paynter, G. W. (2002).** Interactive document summarization using automatically extracted keyphrases. Proceedings of the 35th Annual Hawaii International Conference on System Sciences, pp. 1160–1169. DOI: 10.1109/HICSS.2002.994038.
22. **Kim, S. N., Medelyan, O., Kan, M. Y., Baldwin, T. (2010).** SemEval-2010 task 5: Automatic keyphrase extraction from scientific articles. Proceedings of the 5th International Workshop on Semantic Evaluation, Association for Computational Linguistics, pp. 21–26.
23. **Le, Q., Mikolov, T. (2014).** Distributed representations of sentences and documents. Proceedings of the 31st International Conference on Machine Learning, Vol. 32, No. 2, pp. 1188–1196. DOI: 10.48550/arXiv.1405.4053.
24. **Li, T., Hu, L., Sun, C., Li, S., Chi, L. (2021).** Triplerank: An unsupervised keyphrase extraction algorithm. Knowledge-Based Systems, Vol. 219, pp. 106846. DOI: 10.1016/j.knosys.2021.106846.
25. **Liang, X., Wu, S., Li, M., Li, Z. (2021).** Unsupervised keyphrase extraction by jointly modeling local and global context. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana,

Dominican Republic, pp. 155–164. DOI: 10.18653/v1/2021.emnlp-main.14.

26. **Liu, Z., Huang, W., Zheng, Y., Sun, M. (2010).** Automatic keyphrase extraction via topic decomposition. Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp. 366–376.
27. **Medelyan, O., Frank, E., Witten, I. H. (2009).** Human-competitive tagging using automatic keyphrase extraction. Association for Computational Linguistics, pp. 1318–1327.
28. **Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., Chi, Y. (2017).** Deep keyphrase generation. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, pp. 582–592. DOI: 10.18653/v1/P17-1054.
29. **Mihalcea, R., Tarau, P. (2004).** TextRank: Bringing order into texts. Proceedings of the Empirical Methods in Natural Language Processing.
30. **Popova, S., Kovriguina, L., Mouromtsev, D., Khodyrev, I. (2013).** Stop-words in keyphrase extraction problem. Proceedings of the Conference of Open Innovation Association, FRUCT, pp. 113–121. DOI: 10.1109/FRUCT.2013.6737953.
31. **Rose, S., Engel, D., Cramer, N., Cowley, W. (2010).** Automatic keyword extraction from individual documents. Text Mining: Applications and Theory, pp. 1–20. DOI: 10.1002/9780470689646.ch1.
32. **Schutz, A. (2008).** Keyphrase extraction from single documents in the open domain exploiting linguistic and statistical methods. M. App. Sc Thesis.
33. **Song, M., Xu, P., Feng, Y., Liu, H., Jing, L. (2023).** Mitigating over-generation for unsupervised keyphrase extraction with heterogeneous centrality detection. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, pp. 16349–16359. DOI: 10.18653/v1/2023.emnlp-main.1017.
34. **Sun, Y., Qiu, H., Zheng, Y., Wang, Z., Zhang, C. (2020).** SIFRank: A new baseline for unsupervised keyphrase extraction based on pre-trained language model. IEEE Access, Vol. 8, pp. 10896–10906. DOI: 10.1109/ACCESS.2020.2965087.
35. **Wan, X., Xiao, J. (2008).** Single document keyphrase extraction using neighborhood knowledge. Proceedings of the 23rd National Conference on Artificial Intelligence, AAAI Press, Vol. 2, pp. 855–860.
36. **Wang, R., Liu, W., Mcdonald, C. (2015).** Using word embeddings to enhance keyword identification for scientific publications. pp. 257–268. DOI: 10.1007/978-3-319-19548-3_21.
37. **Zeng, H. J., He, Q. C., Chen, Z., Ma, W. Y., Ma, J. (2004).** Learning to cluster web search results. Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 210–217. DOI: 10.1145/1008992.1009030.

Article received on 22/05/2024; accepted on 07/08/2024.
**Corresponding author is Mikhail Alexandrov.*