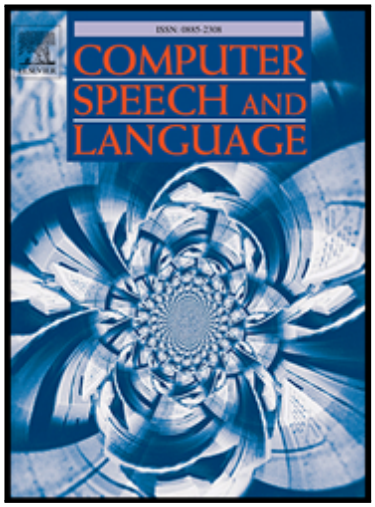
Journal Pre-proof



TOP-Rank: A TopicalPostionRank for Extraction and Classification of Keyphrases in Text

Mubashar Nazar Awan, Mirza O. Beg

PII: S0885-2308(20)30049-8
DOI: https://doi.org/10.1016/j.csl.2020.101116
Reference: YCSLA 101116

To appear in: *Computer Speech & Language*

Received date: 22 February 2019
Revised date: 16 February 2020
Accepted date: 20 May 2020

Please cite this article as: Mubashar Nazar Awan, Mirza O. Beg, TOP-Rank: A TopicalPostionRank for Extraction and Classification of Keyphrases in Text, *Computer Speech & Language* (2020), doi: https://doi.org/10.1016/j.csl.2020.101116

Paper Highlights

February 16, 2020

To Whom it May Concern

Our paper titled *"TOP-Rank: A TopicalPostionRank for Extraction and Classification of Keyphrases in Text"* attempts to provide a methodology for improving keyphrase extraction and classification.

The highlights of our work in this paper are enumerated as follows:

- We propose *TOP-Rank* which incorporates both topical and positional information to rank phrases.
- Classification of phrases in keyphrases and non-keyphrases by identifying topical significance using similarity measures.
- Ranking of keyphrases using intra-cluster and inter-cluster ranking to identify the top most keyphrase from top most cluster.
- Comparison of the similarity measures for keyphrase extraction.
- Improving classification of keyphrases as *Processes*, *Materials* and *Tasks*.

Sincerely,

Mubashar Nazar Awan
Graduate Research Assistant, Department of Computer Science
National University of Computer and Emerging Sciences
A.K. Brohi Road, H-11/4,
Islamabad, Pakistan
E-mail: mubasharnazarawan@gmail.com

TOP-Rank: A TopicalPostionRank for Extraction and Classification of Keyphrases in Text

Mubashar Nazar Awan[a], Mirza O. Beg[b]

National University of Computer and Emerging Sciences, Islamabad

[a]*mubasharnazarawan@gmail.com*
[b]*omer.beg@nu.edu.pk*

Abstract

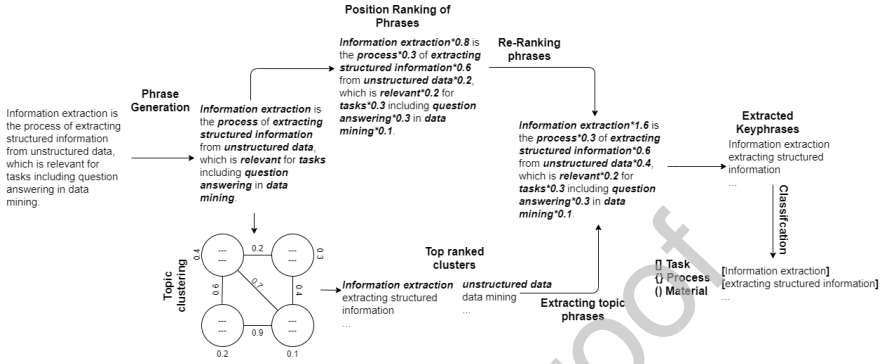
Keyphrase extraction is the task of extracting the most important phrases from a document. Automatic keyphrase extraction attempts to itemize a document content as metainformation and facilitate efficient information retrieval. In this paper we propose TOP-Rank, an approach for keyphrase extraction and keyphrase classification. For keyphrase extraction, we build an approach based on the position of keyphrases in the document and expand it with topical ranking of keyphrases. In particular, keyphrase extraction technique analyzes the documents and extracts keyphrases from the document by giving a higher rank to topical phrases. After keyphrase extraction, we classify keyphrases as *process, material and task*. Our evaluation on diverse datasets shows that TOP-Rank achieves F1-score of 0.73 for keyphrase classification improving upon state-of-the-art methods by a huge margin.

Keywords: keyphrase extraction, topical ranking, position ranking, keyphrase classification

1. Introduction

Extraction of keyphrases from the text is an important problem in the field of Information Retrieval (IR). Keyphrases are a set of important terms that give the high-level description of a piece of text [1]. Keyphrase extraction algorithms are aimed at extracting phrases and words from document that best represent the main topics. It helps the people to quickly identify whether document discusses the topics of interest. Various

Figure 1: This example scenario shows the complete work flow of our study. It starts with collecting text and generating phrases. After phrase generation both Positional and Topical rank algorithms are applied on phrases. The top ranked phrases are classified into *process, materials and tasks*



summarization algorithms are formulated using keyphrases [2]. Deciding what is actually discussed in a document allow us to make clusters of documents having the same topics [3]. Keyphrase extraction is also used in content based recommender systems which help users in discovering information relevant to their previously expressed interests [4].

There are two categories of keyphrase extraction, supervised and unsupervised [5]. In supervised learning, keyphrase extraction is taken as a binary classification problem. The classifiers are trained on textual data, which are annotated with keyphrases and the goal is to determine whether the phrase is a keyphrase. Neural Machine Translation has been used to extract the relevant phrases from the text where title-guided encoding schemes and LSTM have been used [6] [7] [8]. In unsupervised algorithms, citation-based models and topic-based models are used. Citation-based models are used to extract keyphrases from scientific papers, keyphrases are extracted from the citation network of the paper, the information flows from one paper to another through the citation relation [9]. This information flow as well as the influence of one paper on another are specifically captured by the means of citation contexts (i.e., short text segments surrounding a papers mention). In topic-based models, words or keyphrases are ranked based on their relevance to the topic [10]. In our work, we propose an unsupervised

keyphrase extraction which is useful in the areas where finding training data is hard. This work builts upon previously conducted study on position ranking algorithm [11] and extends with TopicalRank. We experimentally evaluate TOP-Rank on diverse datasets and achieve F1-score of up to 0.29 for keyphrase extraction against all the baselines and 0.73 for keyphrase classification against all the state-of-the-art methods. In Figure 1, we show the example scenario of our approach where plain text is used to show the performance of our work.

This paper makes the following contributions:

- Classification of phrases in keyphrases and non-keyphrases by identifying topical significance using similarity measures.
- Ranking of keyphrases using intra-cluster and inter-cluster ranking to identify the top most keyphrase from top most cluster.
- Improving the classification of keyphrases as *processes, materials and tasks*.

This remaining paper is organized as follows: Section II discusses the Related Work, our methodology is discussed in Section III, Section IV gives the Experiments and Section V concludes the paper.

2. Related Work

We divide the keyphrase extraction approaches into three classes. Citation-based models, topic-based models and graph-based models.

2.1. Citation-based methods

Keyphrases from scientific papers has been extracted by using citation context of the paper which is proposed in various papers [12] [13]. [12] propose an approach which uses citation network of the research papers to extract keyphrases. It extracts all the text from the related work of a paper and from the citations of that paper. This technique is further extended by proposing a supervised citation-enhanced approach for keyphrase extraction [13]. Traditional features like TF-IDF, relativePOS and POS are used with citation network based features.

2.2. Topical-based approaches

The topic based techniques are being used in keyphrase extraction. [10] propose a topic-based model with clustering of topics, which extracts the keyphrases and combines them to form clusters. After making clusters of the topics, TextRank [14] which is based on google's page rank algorithm and is used for keyword extraction and text summarization is used to rank the topics and finally topics having the highest rank are used. It helps to select the keyphrases from the top-ranked topics but individual keyphrases are not ranked, keyphrase from a single cluster is selected randomly which can select an unimportant keyphrase. In most approaches, words are ranked and formed into phrases and those phrases are ranked by adding the rank of the individual words. [15] propose a method which does not consider the length of the keyphrase, it identifies the topics and makes their cluster. Each cluster is assigned words from the document. These words are formed into keyphrase and are ranked based on purity, completeness, phraseness and coverage. [16] propose a method where a separate graph of sentences and topics are generated. If a word in a sentence is present in a topic then an edge is created. Each edge is normalized by its length so that long sentences won't get any benefit. HITS (Hyperlink-Induced Topic Search) algorithm is applied to rank the sentences [17]. Recently, [18] and [19] use multipartite graphs and sentence clusters to incorporate topical knowledge.

2.3. Graph-based methods

[20] propose an unsupervised graph based approach for both summarization and keyword extraction. It generates three sentence-to-sentence, sentence-to-word and word-to-word graphs. Using these three graphs, the sentences and words are ranked and the top most sentences are used in summary, and the top most words are used as keywords. [21] propose an approach similar to position ranking, which combines the statistical and graphical methods to extract keyphrases. It uses TF-IDF, term length, Position of first (PFO) occurrence, subsumlength. The main contribution is that it uses PFO which ranks those keyphrases which passes a certain threshold and subsumcount which decreases the term frequency score of the term if it appears in other terms. In all the above

approaches, linguistics are not taken into account for keyphrase generation. [22] propose a technique which first assigns each word to a syntactic category (noun, adj) and different syntactic patterns are identified, phrases are extracted which belongs to the defined syntactic patterns. These keyphrases are ranked using TF-IDF and TextRank. [11] proposed a positionRank graph based approach for the first time in unsupervised keyphrase extraction. Previously, in supervised approaches only the first position of a word was counted but here all the appearances of a word are used to rank. The individual words are formed into phrases and the phrases are scored by the sum of scores of the individual words. Its limitation is that it does not ensure that phrases are selected from all the main topics. To address this problem, we introduce a topic based approach which rank phrases based on their relevance to the topics. It ensures that keywords are selected from all the main topics. [23] analysed word embeddings with other methods of keyphrase extraction and noticed that it does not make any effect on the results. [24] use prior knowledge for keyphrase extraction, they collected keyphrases vocabulary and use it for candidate phrase selection.

2.4. Classification of keyphrases

In the year 2017, SemEval introduced a task for keyphrase extraction and classification. Keyphrase classification as *process, material and task* was the subtask of the *Task 10: Extracting Keyphrases and Relations from Scientific Publications* [25]. The best results are achieved by MayoNLP with the F1-score of 67% by using SVM [26]. Later on, the best results of semeval are outperformed by using nu-SVM [27].

3. TOP-Rank

To extract the most relevant keyphrases from an article, we propose a technique named TOP-Rank which incorporates both topical and positional information. We divide TOP-Rank into subproblems. First, we extract important keyphrases from a document and then perform classification. In the first phase, our methodology is divided into three steps. Phrase generation, Ranking phrases using modified Position Ranking [11] and our Topical Rank to re-rank the phrases. Previously, individual words were ranked

using position rank instead of ranking phrases. After phrase generation, we lemmatize phrases to reduce noise. In the second phase, we classify keyphrases as *process, material and task*. The following text is used as a running example for describing the steps in our approach.

Information extraction is the process of extracting structured information from unstructured data, which is relevant for tasks including question answering in data mining.

3.1. Phrase Generation

We pre-process our dataset by removing all the irrelevant symbols from the text. Nouns and adjectives are taken as candidate phrases because they succinctly contain the key information [28], We extract the longest sequence of nouns and adjectives as shown in example text 'Information extraction' is selected as a phrase. The steps of phrase generation are given in the Algorithm 1, where the line 3 assigns parts of speech tag to each word, line 8 makes a phrase of nouns and adjectives by combining the words and in line 10, we store the phrases in a list.

Information extraction *is the* ***process*** *of* ***extracting structured information*** *from* ***unstructured data****, which is relevant for tasks including* ***question answering*** *in* ***data mining****.*

3.2. Phrase Position Ranking

Phrases generated by our phrase generator are ranked by using a modified version of position ranking algorithm [11]. Unlike previous work, we rank phrases using position ranking. The complete steps of the position ranking algorithm are shown in Algorithm 2. The position ranking algorithm involves two steps: graph construction and position-biased page rank.

Algorithm 1 PHRASE GENERATION

1: **procedure** PHRASEGENERATION(x)
2: $x \leftarrow preprocessing(x)$
3: $tagged \leftarrow$ POS_TAGGING(X)
4: $phrase \leftarrow \{\}$
5: $phrases \leftarrow \emptyset$
6: **for each** $p \in tagged$ **do**
7: **if** $TAG(p) == Noun || TAG(p) == Adj$ **then**
8: $phrase = phrase + p$
9: **else if** $len(phrase)! = 0$ **then**
10: $phrases[] \leftarrow phrase$
11: $phrase \leftarrow \{\}$
12: **return** $phrases$ ▷ phrases from a single document

3.2.1. Graph Construction

We built a graph $G = (V, E)$ for each document (x) as given in line 2 of Algorithm 2, where phrase is a vertex (V) and edge (E) between vertices $v_i, v_j \epsilon V$ is created if the phrases co-occur within a window size k and edge is weighted based on the co-occurance of phrases within a window size k.

3.2.2. Position-Biased Page Rank

Let G be an undirected graph contructed above, line 3 normalizes the edges between vertices. An edge $E_{i_j} \epsilon E$, where i,j are the vertices is normalized by taking all the edge weights of vertices connecting with j as given in equation 1:

$$\widetilde{E_{i_j}} = \begin{cases} E_{i_j} / \sum_{k=1}^{|V|} E_{j_k} & \text{if } \sum_{j=1}^{|V|} E_{i_j} \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Let Vec be the vector containing scores of phrases and its size is $|V|$ as given in line 4. Initially, Vec is set to 1/ $|V|$ for all $v_i \epsilon V$. The scores in Vec at step t+1 are updated in line 5 by using equation 2, where G is used as an adjacency matrix of the

contructed graph and matrix multiplication is computed with the Vector (Vec) which updates the scores of phrases in Vector (Vec):

$$Vec(t+1) = G.Vec(t) \tag{2}$$

Algorithm 2 POSITION RANK

1: **procedure** POSITIONRANKING(x)
2: $G \leftarrow GraphConstruction(x)$
3: $G \leftarrow Normalization(G)$
4: $Vec \leftarrow VectorCreation(G)$
5: $Vec \leftarrow VectorUpdate(Vec, G)$
6: $P \leftarrow PositionRank(x)$
7: $Vec \leftarrow FinalPositionRank(Vec, P)$
8: **return** Vec ▷ vector of PositonRank scores

A position rank vector P is created of size $|V|$ in line 6, where all the phrases are ranked based on their position in document. If a phrase appears at the 3rd and 50th positions then its rank is calculated as $1/3 + 1/50 = 0.353$. The equation 3 shows how the Position of a single phrase in P is calculated, where $count(P_i)$ is total count of a phrase P_i in document and $POS(P_{i_j})$ is the position of phrase P_i. The final PositionRank scores (Vec) given in line 7 are calculated by equation 4 and 5, where α is hyperparameter which enforces the weights on the position rank vector (P) and on vector of scores (Vec), w_{j_i} is the weight of edge between two vertices:

$$P_i = \sum_{j=1}^{|count(P_i)|} 1/POS(P_{i_j}) \tag{3}$$

$$Vec(v_i) = (1-\alpha).P_i + \alpha \sum_{v_j \epsilon Adj(v_i)} \frac{w_{j_i}}{O(v_j)} Vec(v_j) \tag{4}$$

$$O(v_j) = \sum_{v_k \epsilon Adj(v_j)} w_{j_k} \tag{5}$$

The phrases are ranked using the final scores of position rank vector (Vec) as given in example below:

Information extraction*0.8 *is the* ***process*0.3*** *of* ***extracting structured information*0.6*** *from* ***unstructured data*0.2****, which is relevant for tasks including* ***question answering*0.3*** *in* ***data mining*0.1****.*

3.3. TopicalRank

Phrases have been ranked using Position Ranking. In this step, we want to re-rank them based on topical significance. It is important that keyphrases cover all the important topics. To determine this, we apply clustering of phrases which are similar to each other. We computed synctactual similarity between phrases by using TF-IDF, unlike word embeddings because it was most suitable for our goal. Lets take two words *computer* and *science* we dont want them to be in one cluster because both are different words but word embeddings of both these two words would be very much similar. We compute the cosine and jaccard similarity separately between all the generated phrases as they are considered to be good text similarity measures [29]. If the value is greater than the threshold (t), it means they are tagged into the same topic. We made clusters of all the phrases which are similar to each other and discard the outliers. The complete steps of Topical Rank are shown in Algorithm 3.

Algorithm 3 TOPICAL RANK

procedure TOPICALRANK(x)
  $cluster \leftarrow Clusters(x)$
  $G \leftarrow Graph(cluster)$
  $K \leftarrow VectorCreation(G)$
  $prediction \leftarrow KeypExtr(K, G, 10)$
  return $prediction$ ▷ TopicalRank phrases

3.3.1. Graph Construction

After computing the similarity and making cluster of phrases at line 2, We built a graph $G = (V, E)$ at line 3, where each vertex (V) is a cluster and edge between two clusters is weighted based on the reciprocal distance of phrases in each cluster. The weight is assigned using the equation 6 and 7. The clusters are represented by (c_i, c_j) and relevance(p_i, p_j) is the closeness between two phrases. Each occurrance of the phrase in a document is counted to calculate relevance. The relevance between two phrases is computed based on the reciprocal distance between them as given in equation 7. The m,n is the total number of phrases in both clusters, which is used to normalize distance computed between clusters. The $pos(p_i), pos(p_j)$ are positions of the phrase in a document. These equations are inspired by [10]. Let K be the vector of size $|V|$ given in line 4, where V is the number of clusters and it contains the scores of each cluster. Initially the vector (K) is set to $1/|V|$ for all elements. The scores are computed by the equation 8.

$$w_{i,j} = \frac{1}{(m+n)} \sum_{p_i \epsilon c_i} \sum_{p_j \epsilon c_j} relevance(p_i, p_j) \tag{6}$$

$$relevance(p_i, p_j) = \sum_{pt_i \epsilon pos(p_i)} \sum_{pt_j \epsilon pos(p_j)} \frac{1}{|pt_i - pt_j|} \tag{7}$$

$$K(v_i) = \sum_{v_j \epsilon Adj(v_i)} \frac{w_{j_i}}{\sum_{v_k \epsilon Adj(v_j)} w_{j_k}} K(v_j) \tag{8}$$

3.3.2. Keyphrase Selection

In this step, we rank phrases in top rank clusters and select the top ranked phrase from each cluster. For each cluster, a graph $G = (V, E)$ is built, where each vertex is a phrase. The edge weight between phrases is computed by using the same equation 7 where p_i, p_j are two phrases. Let T be the vector of size $|V|$ and initially set to $1/|V|$ for all elements. Finally, the vertices are ranked using equation 9 and top ranked phrase is selected from a cluster. The difference between eq. 8 and 9 is that eq. 8 is used to rank clusters and eq. 9 is used to rank phrases inside each cluster. The result of our TopicalRank are given below, the top ranked phrases in a cluster are highlighted. The

phrases are selected in line 5, where the argument 10 means that we want to select 10 phrases from the top 10 clusters.

$$T(v_i) = \sum_{v_j \epsilon Adj(v_i)} \frac{w_{j_i}}{\sum_{v_k \epsilon Adj(v_j)} w_{j_k}} T(v_j) \tag{9}$$

information extraction *extracting structured information* ...	***unstructured data*** *data mining* ...

3.4. Re-Ranking of keyphrases

The Position Rank vector PV which contains the ranking of phrases is re-ranked by using Topical Rank phrases. We doubled the phrase score, if the same phrase is extracted with topical rank. The top n best phrases are selected as keyphrases as shown below. The complete working of TOP-Rank has been explained in Algorithm 4:

Information extraction*1.6 *is the* ***process*0.3*** *of* ***extracting structured information*0.6*** *from* ***unstructured data*0.4****, which is relevant for tasks including* ***question answering*0.3*** *in* ***data mining*0.1****.*

3.5. Keyphrase classification

After keyphrase extraction, our goal is to classify the keyphrases and for this purpose we focus on the preprocessing of dataset. First, we lowercase all the phrases to avoid multiple copies of words. e.g 'Information' and 'information' are taken as same words. Next step is to remove punctuation as it does not convey important information while dealing with the text data. Lemmatization is the last step to reduce noise. We discard the meaningless phrases consisting of a single or double letter e.g (a, bg). After preprocessing, we extract features from the text and then apply word embedding which we discuss in the next sections.

Algorithm 4 ALGORITHM OF TOP RANK

procedure TOPRANK($docs$)
  $keyphrases \leftarrow \emptyset$
  for each $d \in docs$ **do**
    $keyphrases[] \leftarrow$ PHRASEGENERATION(d)
  $PV \leftarrow \emptyset$
  $Topics \leftarrow \emptyset$
  for each $k \in keyphrases$ **do**
    $PV[] \leftarrow$ POSITIONRANKING(k)
    $Topics[] \leftarrow$ TOPICALRANK(k)
  for each $i \in range(len(docs))$ **do**
    for each $t \in Topics[i]$ **do**
      if $t \in PV[i]$ **then**
        $PV[i][t] * = 2$
  return PV ▷ phrases extracted by TOP Rank

3.5.1. N-grams

N-grams are a combination of multiple words together. The words combined can give more useful information than by using them standalone. Ngrams (N=2) are bigrams and (N=3) are trigrams. We use bigrams and trigrams as most of the keyphrases in the dataset are having length of up to 3.

3.5.2. POS tagging

Syntactic patterns are used to identify keyphrases [22], we apply the same idea in keyphrase classification by using the parts of speech tags as a separate feature. The intuition behind this is that there is a possibility that the same type of patterns are followed while classifying keyphrases e.g a keyphrase of tags (noun adj noun) has a higher probability of being classified as *task*.

Table 1: Summary of datasets with the number of documents and the original number of keyphrases specified

Dataset	No. of documents	No. of Keyphrases
Nguyen	150	652
KDD	755	3093
News Articles	450	21732
SemEval dataset(classificaton)	450	8564

3.5.3. Word embeddings

In this study, we use GloVe [30] word embeddings of 300 dimensions trained on wikipedia. We found that almost 25-30% data is missing in the embeddings so we create our own word embeddings with gensim by using training data of plain text sentences for keyphrase extraction provided by semeval [25]. As word embeddings require lot of documents and we use only 350 documents of the training set which leads to a huge training loss but we use those embeddings with GloVe to reduce missing entries. We added only those embeddings of words which are missing in Glove.

Finally, we model our study into a multiclass classification problem, all the steps described above are performed and passed to the classifier as an input.

4. Evaluation

4.1. Datasets and Evaluation metrics

Evaluation measures are the vital part in order to assess the performance of the classifier and building model. Almost all evaluation measures depends on the nature of the data. To calculate the performance of our work, we perform experiments on the variety of datasets as shown in Table 1. The first dataset consist of research papers and is provided by [31]. The second dataset consists of research papers from ACM conference on Knowledge Discovery and Data Mining(KDD) [12]. Our research work is not limited to these datasets only as we evaluate our approach on the other datasets to increase the diversity of work. We use the News articles dataset [32], to check the relevancy of our

work in other areas and semeval dataset for keyphrase classification. The details of the datasets are given in Table 1 along with the total number of documents and the number of keyphrases originally specified. As in all the previous works the precision, Recall and F1-score are used to compute the results, we also use it as a evaluation metrics. We also use cosine and jaccard similarity [29] to evaluate the correctness of our predictions with the original assigned keyphrases.

4.2. Benchmarks

In order to examine the performance of our algorithm, we use the PositionRank [11] for the comparison purpose. We also compare our approach TOP-Rank with other baselines techniques like TextRank and TF-IDF. For classification, we use three classifiers Naive Bayes, SVM, and Random Forest. Naive Bayes is used because it is considered to be a good classifier to handle textual data [33], SVM performs exceptionally well as shown in the previous work [27], and Random forest is the last classifier used in our work [34].

4.3. Analysis of results

We show the results for top k = {2,4,6,8,10} predicted keyphrases in Table 2 and 3. We set the α in equation 4 to 0.85 as in the default implementation of PositionRank [11]. The threshold (t) for computing the phrase similarity is set to 0.01, so that we combine the phrases even with less similarity. We use the default value of co-occurrance (k) which is 2. The results for both Cosine and Jaccard similarity are given in Table 2 and 3. We can see that on both Nguyen and KDD datasets, our TOP-Rank is outperforming all the other models. In News Articles dataset, Precision is quite high and recall is low, because this dataset contains large number of original keyphrases given for each document as shown in Table 1. Each document contains an average of 40 original assigned keyphrases and we predict only Top2 to Top10 keyphrases. To calculate recall, we divide the correct predictions by total number of original assigned keyphrases that results

Table 2: Precision, Recall and F1-score comparions of TOP Rank against the PositionRank,TextRank and TF-IDF with different window sizes using Cosine similarity

Datasest	Unsupervised	Top2			Top4			Top6			Top8			Top10		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
KDD	TOP-Rank	**0.311**	**0.152**	**0.204**	**0.252**	**0.241**	**0.246**	**0.221**	**0.309**	**0.258**	**0.198**	**0.365**	**0.257**	**0.183**	**0.420**	**0.255**
	PositionRank	0.309	0.151	0.203	0.229	0.220	0.224	0.193	0.271	0.225	0.170	0.312	0.220	0.158	0.361	0.220
	TextRank	0.144	0.071	0.095	0.142	0.136	0.136	0.143	0.200	0.167	0.143	0.263	0.185	0.144	0.331	0.201
	TF-IDF	0.212	0.104	0.140	0.200	0.191	0.195	0.190	0.266	0.222	0.182	0.335	0.236	0.173	0.397	0.241
Nguyen	TOP-Rank	**0.361**	**0.164**	**0.226**	**0.301**	**0.273**	**0.286**	**0.270**	**0.368**	**0.311**	**0.244**	**0.440**	**0.314**	**0.216**	**0.488**	**0.295**
	PositionRank	0.338	0.153	0.211	0.264	0.239	0.251	0.230	0.313	0.265	0.226	0.371	0.265	0.190	0.429	0.263
	TextRank	0.139	0.063	0.087	0.137	0.124	0.130	0.139	0.189	0.160	0.139	0.250	0.179	0.139	0.314	0.193
	TF-IDF	0.230	0.104	0.143	0.225	0.204	0.214	0.220	0.299	0.253	0.205	0.370	0.264	0.199	0.448	0.276
News Articles	TOP-Rank	0.603	**0.025**	**0.048**	**0.577**	**0.048**	**0.089**	**0.559**	**0.069**	**0.123**	**0.539**	**0.089**	**0.153**	**0.521**	**0.108**	**0.179**
	PositionRank	**0.613**	0.025	0.048	0.540	0.045	0.083	0.511	0.064	0.114	0.501	0.083	0.142	0.493	0.102	0.169
	TextRank	0.423	0.018	0.034	0.423	0.035	0.065	0.423	0.053	0.094	0.424	0.070	0.120	0.423	0.088	0.146
	TF-IDF	0.483	0.020	0.038	0.494	0.041	0.076	0.489	0.061	0.108	0.484	0.080	0.130	0.488	0.101	0.167

Table 3: Precision, Recall and F1-score comparions of TOP Rank against the PositionRank,TextRank and TF-IDF with different window sizes using Jaccard similarity

Datasest	Unsupervised	Top2			Top4			Top6			Top8			Top10		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
KDD	TOP-Rank	**0.255**	**0.124**	**0.167**	**0.207**	**0.198**	**0.202**	**0.180**	**0.254**	**0.211**	**0.160**	**0.300**	**0.209**	**0.148**	**0.344**	**0.207**
	PositionRank	0.252	0.123	0.165	0.189	0.180	0.184	0.158	0.223	0.185	0.138	0.258	0.180	0.127	0.296	0.178
	TextRank	0.111	0.057	0.075	0.121	0.115	0.118	0.114	0.161	0.133	0.108	0.203	0.151	0.104	0.243	0.146
	TF-IDF	0.154	0.075	0.101	0.152	0.145	0.148	0.147	0.207	0.172	0.141	0.263	0.184	0.135	0.313	0.189
Nguyen	TOP-Rank	**0.297**	**0.135**	**0.186**	**0.248**	**0.225**	**0.236**	**0.223**	**0.304**	**0.257**	**0.197**	**0.356**	**0.254**	**0.174**	**0.393**	**0.241**
	PositionRank	0.280	0.127	0.175	0.215	0.195	0.205	0.188	0.256	0.217	0.166	0.299	0.213	0.154	0.347	0.213
	TextRank	0.111	0.051	0.070	0.137	0.124	0.130	0.132	0.179	0.157	0.137	0.248	0.176	0.128	0.288	0.177
	TF-IDF	0.166	0.075	0.103	0.164	0.149	0.156	0.163	0.222	0.188	0.156	0.281	0.201	0.128	0.346	0.187
News Articles	TOP-Rank	0.516	0.021	0.040	**0.493**	**0.041**	**0.076**	**0.479**	**0.059**	**0.105**	**0.465**	**0.077**	**0.132**	**0.448**	**0.093**	**0.154**
	PositionRank	**0.523**	**0.022**	**0.042**	0.459	0.038	0.070	0.436	0.054	0.096	0.429	0.071	0.122	0.424	0.088	0.146
	TextRank	0.362	0.015	0.014	0.391	0.032	0.059	0.411	0.051	0.091	0.414	0.069	0.118	0.412	0.085	0.141
	TF-IDF	0.334	0.014	0.027	0.353	0.029	0.054	0.358	0.044	0.078	0.358	0.059	0.101	0.365	0.076	0.126

into low recall because of the large number of original keypharases. The precision of news articles dataset is high as compared to the results of other datasets because of the high number of original given keyphrases, the chance of predicting a correct keyphrase also increases. At Top2, our TOP-Rank is underperforming against PositionRank in some cases using both the similarity measures because we extract 10 keyphrases from our Topical Rank and then re-rank the original Position Rank vector by predicting only 2 keyphrases do not show significant effect. Hence, when predicting Top10 keyphrases our TOP-Rank leads to improve the results.

The more detail representation of the results are shown in Figure 2 and 3. The x-axis shows the predicted keyphrases from 2 to 10, the y-axis shows the score of Precision, Recall and F1. We notice that recall is continuously increasing with the number of predicted keyphrases on all the datasets, because extraction of large number of keyphrases increases the chance of correct predictions. The Precision decreases when the number of predicted keyphrases increases, which shows that by increasing the size of our predicted keyphrases give us more inaccurate predictions. If original assigned keyphrases are 20 and we predict Top2 then it has a more chance of predicting them right. When predicting Top10 keyphrases some of the predictions results into predicting wrong because probability of predicting correct keyphrases get low as shown in figure 2 and 3. Our Top-Rank is outperforming all the other baselines in all the three datasets, however in news articles the results are quite neck to neck. The TOP-Rank is showing significant improvement when predicted keyphrases are increased, which is because prediction of only Top 2 keyphrases doesn't add much in prediction of TOP-Rank as compared to Position Ranking. The jaccard similarity is giving low score as compared to cosine similarity in all the datasets which shows that cosine similarity is a good measure in our case. Our method to embed topical information with the positional information has outperformed all the other baselines.

The statistical significance test is computed to understand the improvement of results. We have taken F1-score of position rank as null hypothesis and F1-score of TOP-

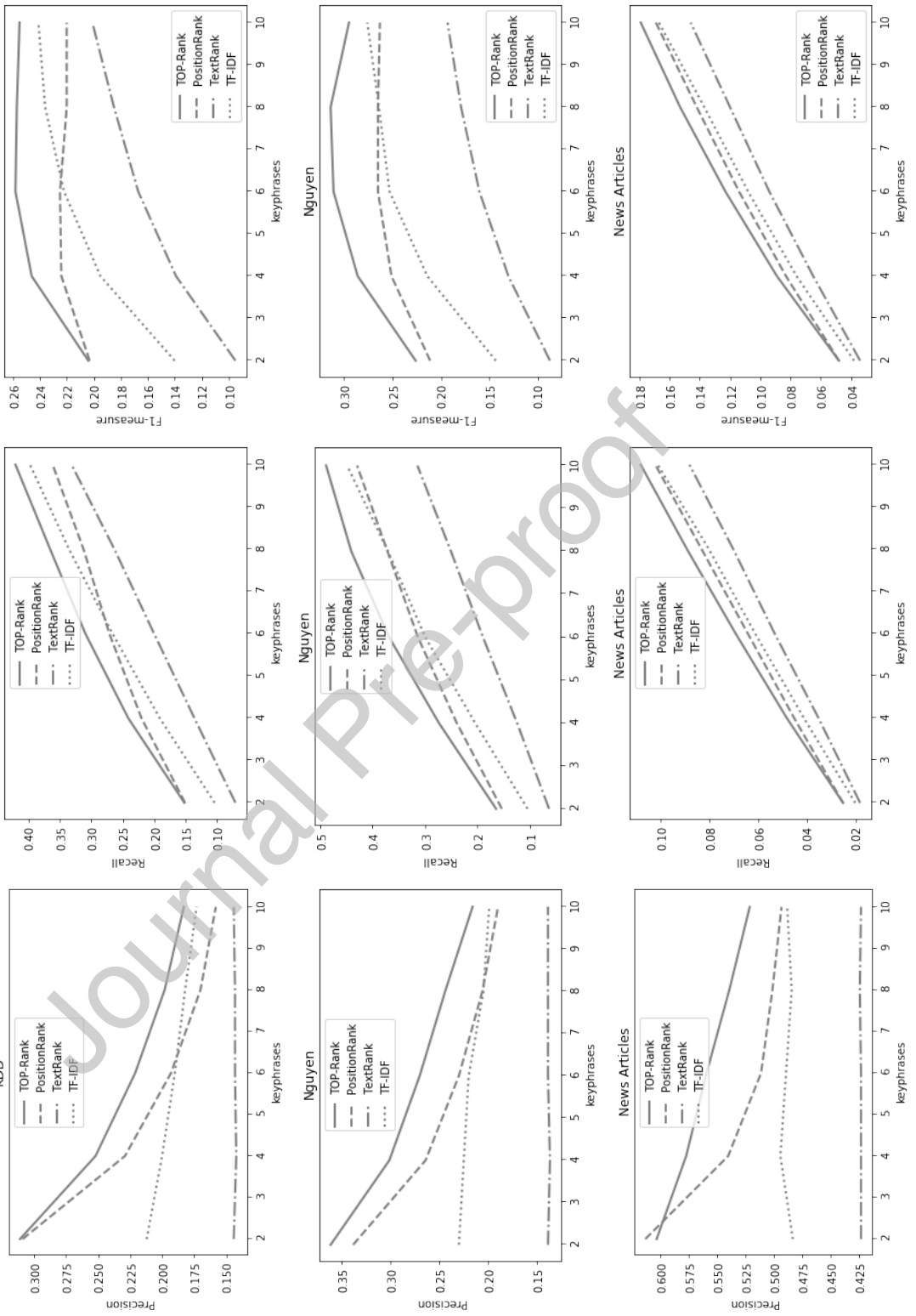


Figure 2: The graphical representation of TOP-Rank comparisons against the PositionRank, TextRank and TF-IDF with different window sizes using Cosine similarity

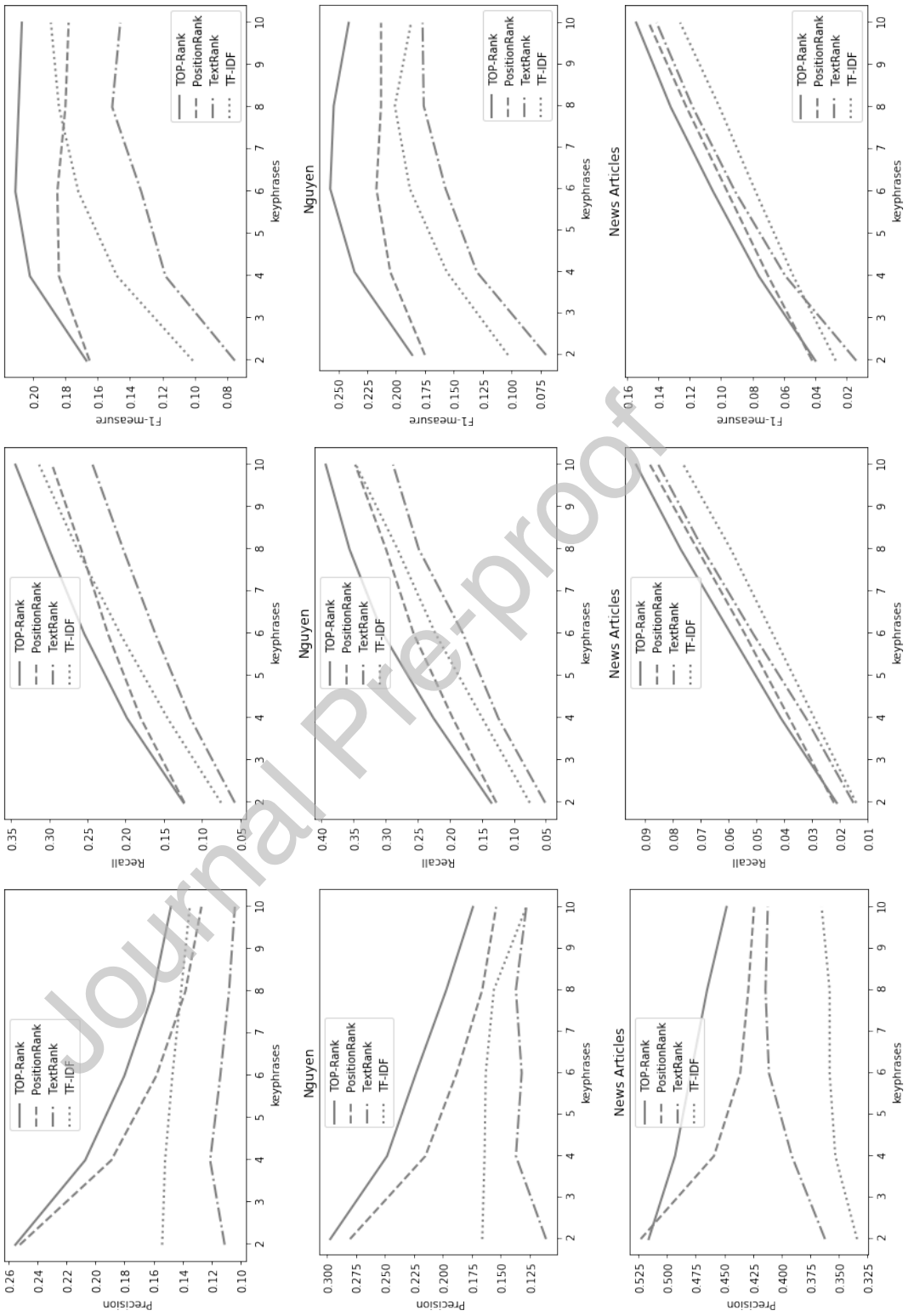


Figure 3: The graphical representation of TOP-Rank comparisons against the PositionRank, TextRank and TF-IDF with different window sizes using Jaccard similarity

Table 4: t-Test of F1-Score results using cosine similarity

Datasest	KDD		Nguyen		News Articles	
Unsupervised	TOP-Rank	Positon Rank	TOP-Rank	Position Rank	TOP-Rank	Postion Rank
1	0.204	0.203	0.226	0.211	0.048	0.048
2	0.246	0.224	0.286	0.251	0.089	0.083
3	0.258	0.225	0.311	0.265	0.123	0.114
4	0.257	0.220	0.314	0.265	0.153	0.142
5	0.255	0.220	0.295	0.263	0.179	0.169
P-Value	0.009		0.002		0.011	

Table 5: t-Test of F1-Score results using jaccard similarity

Datasest	KDD		Nguyen		News Articles	
Unsupervised	TOP-Rank	Positon Rank	TOP-Rank	Position Rank	TOP-Rank	Postion Rank
1	0.167	0.165	0.186	0.175	0.040	0.042
2	0.202	0.184	0.236	0.205	0.076	0.070
3	0.211	0.185	0.257	0.217	0.105	0.054
4	0.209	0.180	0.254	0.213	0.132	0.122
5	0.207	0.178	0.241	0.213	0.154	0.146
P-Value	0.007		0.002		0.09	

Rank as our alternate hypothesis. The α value is set to 0.05, if the computed P-value is less than 0.05 then we can reject the null hypothesis and if the P-value is more than 0.05 than null hypothesis is accepted. The P-value on all datasets is less than 0.05 as shown in table 4, which clearly shows that we can reject null hypothesis on all datasets results shown in table 2. The t-Test results shown in table 5 are computed on table 3 F1-score, and indicates that P-value for KDD and Nguyen dataset results is less than 0.05 which clearly rejects the null hypothesis but for news articles P-value is greater than 0.05 which accepts the null hypothesis. The t-Test conducted in Table 4, 5 overall shows that we can reject the null hypothesis with 95% confidence.

The results of keyphrase classification are shown in Table 6. SVM achieves the best F1 score as compared to Naive Bayes and Random Forest. The Confusion matrix is shown in Figure 4, which gives a clear view that most of the Tasks are confused with the *process* because in the dataset some keyphrases are same, sometimes they are labeled as *process* and sometimes as a *task*. As the data is imbalanced and the *task* has limited amount of keyphrases as compared to *process* and *material*, so the classification also gets affect. Naive Bayes classify most of the keyphrases as *material* and less amount of keyphrases are classified as *task*, its because of the prior probability use in naive bayes algorithm. Naive Bayes use the prior probability when classifying a keyphrase which means that if training set contains most of the keyphrases *material* then prior probability of classifying a keyphrase as *material* is high. As the number of Tasks are low, Naive bayes predicts very small amount of keyphrases as task.

SVM outperforms all other classifiers and performs with the F1-score of 0.730 and performs better against all the previous state of the art implementations of keyphrase classification, semeval [25] results and nu-SVM [27].

5. Conclusion and Future Directions

We propose an approach for keyphrase extraction and classification. Our work is divided into two parts: keyphrase extraction and classification. For the first part, the

Figure 4: Confusion matrix of keyphrase classification using Naive Bayes, SVM and Random Forest

Table 6: Results of keyphrase classification using Naive Bayes, SVM, Random Forest and comparison with the state-of-the-art methods

	Material	Process	Task	F1-score
SVM	**0.778**	0.742	**0.420**	**0.730**
Naive Bayes	0.736	0.582	0.069	0.606
Random Forest	0.644	0.606	0.238	0.591
nu-SVM	0.710	**0.778**	0.381	0.716
best@SemEval2017	0.660	0.760	0.280	0.670

keyphrases are ranked by Position and Topical ranking algorithms. We perform experiments on various datasets given in Table 1 to examine diverse space of related work. We got promising results on all the datasets, we outperform the position rank by achieving F1-score up to 0.29.

Our results can be further improved by adding a weight factor while re-ranking phrases which are extracted using Topical Rank because we double the score of phrases extracted by Topical Rank, the results can be more accurate by increasing the weight to the topical phrases. We extracted top 10 keyphrases from Topical Rank method and re-rank them in the Position Rank vector, by increasing the values of the extracted keyphrases results can be improved. We can also extract n keyphrases where n is the number of clusters which means that all the cluster/topics are incorporated in extracting keyphrases and keyphrases from all the topics are extracted. By using this method there will not be any need to rank clusters. We can make graphs for separate clusters and can extract keyphrases from them. In the second part of our work, we improve the classification score of keyphrases, as we have only one dataset for the classification of keyphrases, there is a need to create more datasets in this area.

In the end, We perform comparison of the cosine and jaccard similarity measures and found that cosine similarity performs better.

References

References

[1] M. Grineva, M. Grinev, D. Lizorkin, Extracting key terms from noisy and multi-theme documents (2009) 661–670.

[2] E. DAvanzo, B. Magnini, A. Vallin, Keyphrase extraction for summarization purposes: The lake system at duc-2004, in: Proceedings of the 2004 document understanding conference, 2004.

[3] K. M. Hammouda, D. N. Matute, M. S. Kamel, Corephrase: Keyphrase extraction for document clustering, in: International Workshop on Machine Learning and Data Mining in Pattern Recognition, Springer, 2005, pp. 265–274.

[4] P. Lops, M. De Gemmis, G. Semeraro, Content-based recommender systems: State of the art and trends, in: Recommender systems handbook, Springer, 2011, pp. 73–105.

[5] K. S. Hasan, 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.

[6] M. Passon, M. Comuzzo, G. Serra, C. Tasso, Keyphrase extraction via an attentive model, in: Italian Research Conference on Digital Libraries, Springer, 2019, pp. 304–314.

[7] W. Chen, Y. Gao, J. Zhang, I. King, M. R. Lyu, -guided encoding for keyphrase generation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 6268–6275.

[8] R. Alzaidy, C. Caragea, C. L. Giles, Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents, in: The world wide web conference, 2019, pp. 2551–2557.

[9] X. Shi, J. Leskovec, D. A. McFarland, Citing for high impact, in: Proceedings of the 10th annual joint conference on Digital libraries, ACM, 2010, pp. 49–58.

[10] A. Bougouin, F. Boudin, B. Daille, Topicrank: Graph-based topic ranking for keyphrase extraction, in: International Joint Conference on Natural Language Processing (IJCNLP), 2013, pp. 543–551.

[11] C. Florescu, C. Caragea, Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vol. 1, 2017, pp. 1105–1115.

[12] S. D. Gollapalli, C. Caragea, Extracting keyphrases from research papers using citation networks., in: AAAI, 2014, pp. 1629–1635.

[13] C. Caragea, F. A. Bulgarov, A. Godea, S. D. Gollapalli, Citation-enhanced keyphrase extraction from research papers: A supervised approach., in: EMNLP, Vol. 14, 2014, pp. 1435–1446.

[14] R. Mihalcea, P. Tarau, Textrank: Bringing order into text, in: Proceedings of the 2004 conference on empirical methods in natural language processing, 2004, pp. 404–411.

[15] M. Danilevsky, C. Wang, N. Desai, X. Ren, J. Guo, J. Han, Automatic construction and ranking of topical keyphrases on collections of short documents, in: SIAM International Conference on Data Mining, SIAM, 2014, pp. 398–406.

[16] D. Parveen, H.-M. Ramsl, M. Strube, Topical coherence for graph-based extractive summarization, in: Conference on Empirical Methods in Natural Language Processing, 2015, pp. 1949–1954.

[17] J. M. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM (JACM) 46 (5) (1999) 604–632.

[18] F. Boudin, Unsupervised keyphrase extraction with multipartite graphs, in: 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.

[19] M. R. Alfarra, A. Alfarra, Graph-based technique for extracting keyphrases in a single-document (gtek), in: International Conference on Promising Electronic Technologies (ICPET), IEEE, 2018, pp. 92–97.

[20] X. Wan, J. Yang, J. Xiao, Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction, in: ACL, Vol. 7, 2007, pp. 552–559.

[21] S. Danesh, T. Sumner, J. H. Martin, Sgrank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction., in: * SEM@ NAACL-HLT, 2015, pp. 117–126.

[22] C. B. Ali, R. Wang, H. Haddad, A two-level keyphrase extraction approach., in: CICLing (2), 2015, pp. 390–401.

[23] J. Mothe, F. Ramiandrisoa, M. Rasolomanana, Automatic keyphrase extraction using graph-based methods, in: ACM Symposium on Applied Computing (SAC). ACM, 2018.

[24] G. He, J. Fang, H. Cui, C. Wu, W. Lu, Keyphrase extraction based on prior knowledge, in: Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries, ACM, 2018, pp. 341–342.

[25] I. Augenstein, M. Das, S. Riedel, L. Vikraman, A. McCallum, Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications, In Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pages 546555, Vancouver, Canada. Association for Computational Linguistics.

[26] S. Liu, F. Shen, V. Chaudhary, H. Liu, Mayonlp at semeval 2017 task 10: Word embedding distance pattern for keyphrase classification in scientific publications, in: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp. 956–960.

[27] D. Buscaldi, S. D. Hernandez, T. Charnois, Classification of keyphrases from scientific publications using wordnet and word embeddings, in: VADOR (Valorisation et Analyse des Données de la Recherche) 2017, 2017.

[28] S. S. Kathait, S. Tiwari, A. Varshney, A. Sharma, Unsupervised key-phrase extraction using noun phrases, International Journal of Computer Applications 162 (1).

[29] S. Hariharan, R. Srinivasan, A comparison of similarity measures for text documents, Journal of Information & Knowledge Management 7 (01) (2008) 1–8.

[30] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[31] T. D. Nguyen, M. Kan, Keyphrase extraction in scientific publications, in: International conference on Asian digital libraries, Springer, 2007, pp. 317–326.

[32] L. Marujo, A. Gershman, J. Carbonell, R. Frederking, J. ao P. Neto, Supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization, in: Proceedings of the LREC, 2012.

[33] P. Liu, H. Yu, T. Xu, C. Lan, Research on archives text classification based on naive bayes, in: 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), IEEE, 2017, pp. 187–190.

[34] D. Xue, F. Li, Research of text categorization model based on random forests, in: Computational Intelligence & Communication Technology (CICT), 2015 IEEE International Conference on, IEEE, 2015, pp. 173–176.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.