UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Abdelrhman Elsayed Hassan Eldallal

BibRank: Automatic Keyphrase Extraction Platform Using Metadata

Master's Thesis (30 ECTS)

Supervisor: Eduard Barbu, PhD

Tartu 2021

BibRank: Automatic Keyphrase Extraction Platform Using Metadata

Abstract:
Automatic Keyphrase extraction is the process of automatically identifying the essential phrases from a document. Keyphrases are used in crucial tasks such as document classification, clustering, recommendation, indexing, searching, and summarization. This thesis introduces BibRank, a new semi-supervised automatic keyphrase extraction method that exploits an information-rich dataset collected by parsing bibliographic data in BibTeX format. BibRank combines a novel weighting technique of the bibliographic data with positional, statistical, and word co-occurrence information. We have benchmarked BibRank and state-of-the-art techniques against the dataset. The evaluation indicates that BibRank is more stable and has a better performance than state-of-the-art methods.

Keywords: keyphrase Extraction, Metadata, Natural Language Processing

CERCS: P170 Computer science, numerical analysis, systems, control

BibRank: Automaatne võtmesõnade otsimise platvorm mis kasutab metaandmeid

Lühikokkuvõte:

Automaatne võtmesõnade otsimine on dokumendist automaatselt oluliste fraaside leidmine. Võtmesõnasid kasutatakse tähtsate ülesannete jaoks nagu dokumentide klassifitseerimine, klasterdamine, soovitussüsteemid, indekseerimine, otsimine ja lühikokkuvõtte tegemine. See lõputöö tutvustab BibRank'i, uut poolautomatiseeritud võtmesõnade otsingumeetodit, mis kasutab informatsioonirikast andmestikku, mis on genereeritud bibliograafilistest andmetest BibTeX formaadis. BibRank kombineerib uudse kaalutehnika bibliograafilistele andmetele ja informatsiooni asukohalise, statistilise ja sõnade koosesinemise kohta. Me võrdlesime BibRank'i uusimate lähenemistega tutvustatud andmestikul. Võrdlus ja hindamine näitab, et BibRank toimib paremini ja on stabiilsem kui teised lähenemised.

Võtmesõnad: Võtmesõnade otsimine, metaandmed, Loomuliku keele töötlus

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine

Acknowledgements

I would like to thank my supervisor, Dr Eduard Barbu, for his valuable guidance, support, and kind and understanding attitude throughout the thesis work. I would also like to thank the Tartu NLP research group for the valuable opportunities, courses, and resources they offer.

I extend my thanks to my mentor Dr Gabor Melli for his endless support and guidance. I would also like to thank my dear friend Mostafa Sokar for his support and advice.

My thanks and gratitude to the late Dr Sherif Sakr, who supported and guided me throughout my studies. My thanks are also extended to Dr Riccardo Tommasini and the Data Systems group for their support and guidance.

Finally, I extend my heartiest thanks and gratitude to my parents, family and friends for their endless support and encouragement.

Contents

1 Introduction

1.1 Keyphrase extraction

Automatic keyphrase extraction is the process of automatically identifying a set of important phrases or segments from a document that best describes it [36, 21, 5, 33]. These keyphrases should provide a high-level description and summary of the main topics of the document [25, 17]. Other terms are used to describe the same extracted information, such as keywords or key segments [5]. There are supervised, unsupervised, and semi-supervised approaches for keyphrase extraction (KE) [36, 5]. Supervised approaches represent a classification problem where each word or phrase can have a label positive or negative, indicating whether it is a keyphrase or not. In the case of the Unsupervised approach, the model is domain-independent and is designed to automatically extract these phrases automatically [36]. Semi-supervised approaches use a small amount of labelled data combined with a large amount of unlabeled data or other alternatives [5].

1.2 Applications

Keyphrases are rich sources of information that can be used for various natural language processing (NLP) and information retrieval (IR) tasks [17]. They can be used to cluster or classify documents into different categories by measuring the overlap between the set of keyphrases associated with each document [33]. Extractive text summarization is another application, where the goal is to summarize the text by extracting phrases or sentences from the text best representing it. Thus, the keyphrases represent a minimal summary of the document, and each phrase can help with extracting the related sentences for more extended automatic summarization [36]. Keyphases are used with several other tasks such as document searching, topic detection, named entity recognition, automatic indexing, document and web searching, web mining, recommendation systems, and question answering [31, 2, 9, 21, 46]. Keyphrase extraction task is applied in different fields or domains as social media and web, medicine, law, and agriculture [30, 37].

There are tens of millions of scientific documents that can be found online. Microsoft Academic, which is a search engine for academic publications and literature, has more than 250 million publications indexed [42]. While the rapidly increasing number of publications helps the process of knowledge discovery, it poses a serious challenge for information retrieval or extraction [17]. Since keyphrases are considered to be a minimal summary of the document, they can help with efficient information retrieval. In addition, keyphrases are important for other similar tasks such as scientific papers recommendation, search, and classification [17]. Thus, many keyphrase extraction approaches are designed for scientific documents. Such approaches were paired with many datasets collected from scientific articles. [36, 21].

1.3 Challenges

Although many solutions have been introduced to solve automatic keyphrase extraction, many of these solutions' performance is not satisfactory [30]. Many KE solutions face the problems of sparsity and scalability challenges [5]. These two challenges arise because data grows very rapidly, and manual annotation is very slow and expensive. Another problem is that it is very complex to compare KE models. This complexity is because many benchmark datasets differ in many aspects such as quality of the manually annotated data, domain, size, and language [19]. Therefore, trained KE models fail to generalize across different domains, and the data inconsistencies affect the models' performance badly [19]. Thus, there is a great need for more consistent datasets that take into consideration the large amount of data available and the different domains and features of such data.

1.4 Results and Contributions

The main contributions of the thesis are as follows:

- A new approach for keyphrase extraction using metadata is designed and tested. This approach uses domain data and other features to improve the performance of existing keyphrase extraction models.
- A new dataset for keyphrase extraction is provided. It contains more than 18,000 labeled abstracts of scientific documents with their metadata.
- A platform that implements a full keyphrase extraction pipeline is developed. The platform includes datasets creation, different models processing, and running different evaluations.
- An analysis for several different types of KE models and the new approach. The experiments were performed using the platform and the new dataset.

1.5 Thesis structure

- In Chapter 2, an overview of the task and the related work are outlined. The related work summarizes the different types of keyphrase extraction methods and examples of each type. It includes work about domain-specific models and bibliographic data, which are the new approach's main concepts.
- The keyphrase extraction pipeline is described in chapter 3. The pipeline consists of different steps, which start with data preprocessing, and ends with the keyphrase being identified. The chapter also describes the new approach.

- Chapter 4 explains the process of evaluating keyphrase extraction models, the metrics, and the datasets used. It also introduces a new dataset.
- Chapter 5 presents the results of the experiments and the discussion of the evaluation.
- The conclusion and the future work are presented in chapter 6.

2 Related work and background

2.1 Methods

There are different types of approaches for keyphrase extraction. The two main categories are supervised approaches, and unsupervised approaches [46, 36].

Supervised approaches consider keyphrase extraction a binary classification problem [31]. The goal is to train a model using manually annotated data to classify candidate phrases as a keyphrase or not [21]. This problem definition means that the training data defines the domain of the model. There are traditional supervised approaches that use machine learning models such as decision trees and features such as frequency of the phrase and the position of it in the document to build the classifier [5]. Neural networks based models are also used as supervised approaches, but they require a large amount of labeled data for training [19]. The supervised methods discussed in this chapter are summarized in table 1.

On the other hand, unsupervised approaches do not require labeled training data, and they are domain-independent [36]. Such approaches do not require manually annotated data which is hard to obtain and can limit the models because of inconsistencies [19]. Unsupervised approaches consider the task of keyphrase extraction as a ranking problem [30]. Thus, for unsupervised approaches, the main steps would be to select the candidates, rank them, and filter out the unwanted ones [36]. Since annotated data is not easy to obtain and can limit the trained models' performance, there have been more motivation to develop unsupervised approaches [19]. There are multiple types of unsupervised approaches such as a statistical, linguistic, graph, language models, and embedding based approaches [5, 36]. The unsupervised methods discussed in this chapter are listed in table 2. Some approaches aimed at exploiting both the supervised and unsupervised techniques for keyphrase extraction, and they are called semi-supervised approaches.

2.1.1 Supervised Approaches

Table 1. Supervised keyphrase extraction methods discussed in this chapter

Method	**Year**	**Algorithm/ Model used**
KEA	1999	Naïve Bayes
WINGNUS	2010	Naive Bayes
CeKE	2014	Naive Bayes
Deep Keyphrase Generation	2017	RNN Encoder-Decoder model
Al-Zaidy	2019	Bi-LSTM-CRF

One of the first and most well-known keyphrase extraction models is KEA [45]. The model uses lexical methods and the calculated feature values for each candidate, along

with a machine learning model. This model is used to predict the keyphrases out of the candidate list. KEA has two main stages training and extraction. Both stages choose a set of candidates and calculate feature values for each candidate. The algorithm chooses candidates in three steps text processing, candidate identification, and text stemming. For each candidate in both training and extraction stages, two features are calculated. They are the phrase frequency in the document, which is measured by TFIDF, and the position or the distance of the first occurrence of the phrase in the document. The machine learning model used is Naïve Bayes, and it is trained using manually labeled data. For each candidate, the trained model determines the overall probability of it being a keyphrase. When the model is used on any candidate phrase with feature values t and distance d, two probabilities are computed. First the following expression for $P[yes]$:

$$P[yes] = \frac{Y}{Y+N} P_{TF\times IDF}[t|yes] P_{distance}[d|yes] \tag{1}$$

A similar expression for $P[no]$ is computed. Where Y is the number of positive instances, and N is the number of negative instances. The overall probability that any candidate is a keyphrase is a calculated:

$$p = \frac{P[yes]}{P[yes] + P[no]} \tag{2}$$

All candidates are ranked using this probability value. This ranked candidates are filtered using two post steps, then the first r candidates are selected as keyphrases, where r is the number of keyphrases needed.

WINGNUS [34] is a supervised keyphrase extraction method, which chooses the candidate keyphrases using regular expressions that were used by [24]. They also provided a keyphrase distribution study using the training data from sections of documents. The study's conclusion was a proposal for a candidate identification approach that uses logical structures to filter out unneeded candidates while ensuring good coverage. They considered the best choice to be either the full-text or text segments. The segments should include titles, headers, abstracts, and introduction. Additionally, they include related work and conclusion. Using a Naïve Bayes model, they experimented with several combinations of features such as term frequency, length of phrases, and whether a phrase is part of the document's title or has a particular format. They concluded that the best features were the TFIDF, the frequency, the first occurrence, and the length of the phrase.

CeKE is another approach that uses Naïve Bayes model, but with a different set of features [12]. The model is a binary classification model, which means that candidate phrases are classified into two classes representing being a keyphrase or not. Several features are extracted and used, such as TFIDF, the position of the candidate's first occurrence, POS tags, and citation-related features. Citations network-based are novel features that exploit the citation context. They are boolean values that are true if the candidate term appears in a citation context. The TFIDF is also computed using the

citation context. The citation-based features showed improvement compared with the state-of-art approaches.

Recently, deep learning methods are widely used, both [29] and [1] are examples of supervised deep learning based keyphrase extraction models. [29] used a generative model with an encoder-decoder framework for keyphrase prediction. They do not only extract keyphrases from the text, but they use the generative model to predict keyphrases that are not in the text. Their goal is to use deep learning to understand the semantics of the text and generate the keyphrases accordingly. Figure 1 shows an example provided by [29]. The example shows the output of RNN generative keyphrase extraction model. The input is the title and abstract of a research paper. The model can generate two lists, one with keyphrases that present in the text, and the other is based on the meaning of the text. For example, "video retrieval" does not occur in the text, but it is understood by the context.

1. **Input text:** Title: *Towards content-based relevance ranking for video search* Abstract: *Most existing web video search engines index videos by file names, URLs, and surrounding texts. These types of video metadata roughly describe the whole video in an abstract level without taking the rich content, such as semantic content descriptions and speech within the video, into consideration. Therefore the relevance ranking of the video search results is not satisfactory as the details of video contents are ignored. In this paper we propose a novel relevance ranking approach for Web-based video search using both video metadata and the rich content contained in the videos. To leverage real content into ranking, the videos are segmented into shots, which are smaller and more semantic-meaningful retrievable units, and then more detailed information of video content such as semantic descriptions and speech of each shots are used to improve the retrieval and ranking performance. With video metadata and content information of shots, we developed an integrated ranking approach, which achieves improved ranking performance. We also introduce machine learning into the ranking system, and compare them with IR-model (information retrieval model) based method. The evaluation results demonstrate the effectiveness of the proposed ranking methods.*

2. **Present Keyphrase RNN**: 1. information retrieval; 2. video search; 3. search engine; 4. video content; 5. machine learning; 6. web video; 7. content based; 8. semantic content; 9. web based video; 10. web based

3. **Absent Keyphrase RNN**: 1. video retrieval; 2. relevance feedback; 3. video summarization; 4.query expansion; 5.video indexing; 6.semantic web; 7. multimedia retrieval; 8. image retrieval; 9. web search; 10. query processing

Figure 1. Generative Model-based Keyphrase Extraction Example

Deep learning methods definition of the keyphrase extraction problem was different from the traditional classification problem. Given a dataset that contains N keyphrase data instances, the i-th data instance $(x^{(i)}, p^{(i)}$ has one source text $x^{(i)}$, and M_i target keyphrases $p^{(i)} = (p^{(i,1)}, p^{(i,2)}, ..., p^{(i,M_i)})$. Where, both source text $x^{(i)}$ and keyphrase $p^{(i,j)}$ are sequences of words:

$$X^{(i)} = x_1^{(i)}, x_2^{(i)}, ..., x_{L_x i}^{(i)} \tag{3}$$

$$P^{(i,j)} = y_1^{(i,j)}, y_2^{(i,j)}, ..., y_{L_p(i,j)}^{(i,j)} \tag{4}$$

Where, $L_x(i)$ and $L_p(i,j)$ are the length of the word sequence and of $X^{(i)}$ and $p^{(i,j)}$ respectively. Each data instance has one source text sequence and more than one target phrase sequences. The data is converted to pairs of text and keyphrases, by creating multiple paris of the same text. The pairs are used to trained the seq2seq RNN Encoder-Decoder model.This is different from the traditional keyphrase extraction as classification problem where each candidate phrase can be either a keyphrase or not.

[1] also tried to capture the hidden semantics in text by exploiting Bidirectional Long Short Term Memory networks. They also extracted label dependencies through a transition parameter matrix which consists of the transition probabilities from one label to the neighboring using Conditional Random Fields. They formulated the problem as a sequence labeling task. Given an input sequence $x = x_1, \ldots, x_n$ where x_i it the input vector of the ith word, and the goal is to predict target sequence is of labels $y = y_1, \ldots, y_n$, where a label for each word in the input. The label y_i either is KP (keyphrase word) or Non-KP (not keyphrase word). The formulation of the problem using sequence labeling helps with the correlations between neighboring labels and rather than decode labels for the input sequence independently, it allows jointly decoding them.

2.1.2 Unsupervised Approaches

Supervised approaches training data with manually annotated keyphrases, which is time-consuming, and costly [5]. Therefore, there has been much research with a focus on unsupervised methods. These methods can be grouped into three main categories Statistical-based, graph-based, and neural networks-based approaches, where each category can be split into multiple other subcategories.

Statistical-based approaches

TFIDF [18] is one of the most common baseline methods for the task [36]. The model is widely used since it does not require any resources or labeled data, as it uses statistics based on unlabeled data to weight keyphrase candidates [19]. The methods assign scores

Table 2. Unsupervised keyphrase extraction methods discussed in this chapter

Method	**Year**	**Approach Type**
TFIDF	1999	Statistical
KPMiner	2010	Statistical
YAKE	2020	Statistical
TextRank	2004	Graph based
CollabRank	2008	Graph based
TopicRank	2013	Graph based
PositionRank	2017	Graph based
SGRank	2015	Hybrid Statistical-graphical
EmbedRank	2018	Sentence Embedding
KeyBERT	2021	Sentence Embeddings

as weights for candidate keyphrases, which can be used for ranking or as a feature for a different model [11]. The score is calculated according to the formula:

$$TFIDF = TF \times IDF \tag{5}$$

Where TF is the term or phrase frequency, and it is calculated by adding up all the occurrences of a phrase in each document in the corpus.IDF is the inverse document frequency and it is calculated using the following formula:

$$IDF = \log \frac{N}{d_{phrase}} \tag{6}$$

Where N is the total number of documents in the corpus, and d_{phrase} is the number of documents that have an occurrence of the phrase [11]. There are a number of variations of TFIDF. For example, raw phrase frequency is used instead of phrase frequency or using some normalization methods for the scores. [36].

KPMiner [15] is another statistical keyphrase extraction method, which does not require training and uses a modified version of TFIDF with more statistical information [30]. The model can be configured by users based on their understanding of the nature of both documents and keyphrases [15]. The method has three main steps candidate keyphrase selection, candidate keyphrases weight calculation, and final candidate phrase list filtering. The candidate selection is based on a number of linguistic and statistical heuristics, defined by two main conditions. First condition, a candidate phrase will not be separated by punctuation marks and will rarely have stop words. They identified a total of 187 stop words such as (the, then, in, above, etc), which are words that are commonly used in most of the English sentences. The second condition is related to the position first occurrence of the candidate phrase in the document. They have noticed that phrases the first appear after a certain threshold position are very rarely to be keyphrases. They

defined a constant cutoff value to be used as part of the candidate selection process. The second step is to calculate weights for each candidate phrase. The weights are calculated using the following formula:

$$w_{ij} = tf_{ij} * idf * B_i * P_f \tag{7}$$

Where w_{ij} is the weigh for the candidate phrase j in Document i, tf_{ij} is the frequency of the phrase, idf is the inverse document frequency defined in 6, B_i is the boosting factor for document i, and P_f is the phrase position factor. The boosting factor is a score calculated for each candidate phrase based based on the total number of such phrases in a document, the number of phrases whose length exceeds one in the same document, and two weight adjustment constants. The final step, is to first rank the candidate phrases using the calculated weights, and then choose the first n phrases after applying some filters to be the keyphrases.

YAKE [10] is a statistical keyphrase extraction method that exploits both statistics and context information. YAKE algorithm includes three steps preprocessing, feature extraction, keyphrases selection. Preprocessing is done by cleaning the text, splitting it into terms, and identifying stop words. The algorithm identifies five different features for each term. The features are term casing (T_{case}), term position ($T_{Position}$), term frequency (T_{freq})), term relatedness to context (T_{Rel}), and term different sentence ($T_{Sentence}$). Term casting is a feature related to the casing aspect of the term. The term context relatedness is calculated by counting the number of different terms that appear to the left and the right of the candidate term. The term different sentence is based on how often the term is part of different sentences. The five scores is used to calculate one score for each term. The score is calculated as follows:

$$\mathrm{S(t)} = \frac{\mathrm{T_{Rel}} * \mathrm{T}_{Position}}{\mathrm{T}_{Case} + \frac{\mathrm{TF}_{Nam}}{\mathrm{T}_{Rel}} + \frac{\mathrm{T}_{sentence}}{\mathrm{T}_{Rel}}} \tag{8}$$

Where the smaller the value of the score is, the more important this term would be. The final step is to calculate the score for sequences of terms using a sliding window and rank the terms using the scores.

Graph-based approaches

Graph-based keyphrase extraction methods are probably the most popular methods in the literature [19]. In graph based methods, a document is represented as a graph where terms or words are represented as vertices (nodes), and the edges represent the relations between these terms [36]. These edges can represent a number of different relations exploiting different information and scopes of the text to construct the graph [5]. Co-occurrence edges connect terms the co-occurring in the text within certain context or a text window of a specific size in a sentence, paragraph, or a document. Syntax edges are

connections based on relations of terms syntax dependency. Semantic relations are edges that connect terms based on their meaning, and form. [5] identified those three types of edges which define the graph structure, and they suggested that there are more different possibilities for different graph structures and terms relations.

TextRank [32] is one of the first graph-based keyphrase extraction methods, which was an inspiration for many researchers to build methods based on it [36]. TextRank is a ranking method similar to PagRank algorithm designed for keyphrase extraction task [5]. Preprocessing is the first step in the algorithm, where the text is tokenized and processed with Part-of-speech tagging model and during this step only singles words are considered as candidate keyphrases [32]. The next step is to apply syntactic filters which process the terms (text units) and only keep nouns and adjectives. The output of the filters is lexical units, which are the candidate terms used to represent the nodes of the graph. The edges between the terms (nodes) represent the co-occurrence of the terms within a window of a fixed size of N words. The output constructed graph of this process is unweighted and undirected one, and each vertex has a score of 1. Then, the PageRank ranking algorithm is run for several iterations until it converges which assigns each vertex a new score. The score is calculated for a nodes V_i using the following formula:

$$S\left(V_i\right) = (1-d) + d * \sum_{j \in \mathrm{In}(V_i)} \frac{1}{\left|Out\left(V_j\right)\right|} S\left(V_j\right) \tag{9}$$

Where d is a damping factor that controls the algorithm movement between different vertices, and $|Out\left(V_j\right))$ is the set of vertices which the node V_i points to. Once the ranking algorithm converges the nodes are sorted using the scores.

CollabRank is a graph-based method that exploited the mutual information between several documents to improve the process of keyphrase extraction [41]. They assume that similar documents contain similar keyphrases. The approach has two main steps document clustering and collaborative keyphrase extraction. The document clustering step is the process of clustering D documents into a set of groups. Then, for each C group or cluster, two processes are conducted to extract keyphrases for single documents in the cluster C using batches. The first process is to evaluate candidate terms in each cluster using a graph-based ranking algorithm similar to the one used in TextRank method. Then scores are calculated for each candidate term within a document by summing the cluster level scores. POS tagging is used as a post-processing step, where any term that is not a noun or adjective is ignored. The final list is ranked using the scores calculated.

TopicRank is a graph-based keyphrase extraction approach that exploit the information related to the topics of the document [8]. TopicRank is an unsupervised approach that extracts the keyphrases from the topics of the document. They define topics as a group of similar keyphrases. The algorithm is divided into five steps preprocessing, candidate phrases extraction, candidate phrases clustering, graph-based ranking, and keyphrases selection. Preprocessing includes text tokenization, POS tagging, and candidate phrases

extraction. Then, similar candiadte phrases are grouped into different sets of topics using hierarchical agglomerative clustering. A weighted graph is then constructed where the topics are the vertices and the edges are weighted using on a measure that considers phrases' offset positions in the text. The weights of the edges are calculated using the following formula:

$$w_{i,j} = \sum_{c_i \in t_i} \sum_{c_j \in t_j} \operatorname{dist}(c_i, c_j) \tag{10}$$

where distance is calculated as following:

$$\operatorname{dist}(c_i, c_j) = \sum_{p_i \in \operatorname{pos}(c_i)} \sum_{p_j \in \operatorname{pos}(c_j)} \frac{1}{|p_i - p_j|} \tag{11}$$

In the document, $\operatorname{dist}(c_i, c_j)$ refers to the inverse distances between the offset positions of the candidate phrases c_i and c_j and $\operatorname{pos}(c_j)$ represents the offset positions of the candidate phrase c_i. TextRank method is used to rank topics by assigning a significance score to each topic. Finally, from the N most important topics a term is selected as a keyphrase.

PositionRank [17] aims at improving overall performance by exploiting positional, statistical, and word co-occurrence information [36]. The algorithm has three main steps constructing the graph at the word level, designing of position-biased PageRank, and extracting candidate phrases. First, a POS filter is used only to choose noun and adjective terms to construct the graph. Then the terms are used as vertices, and the edges represent the co-occurrence count between any two terms (vertices) in the document. The weight of every node is initially set to zero. Then, the PageRank scores are recursively calculated by summing all nodes' scores connected to each vertex. The algorithm assigns high scores (probabilities) to the frequent terms and appears early in the document. The score is computed using the following formula:

$$S(v_i) = (1 - \alpha) \cdot \widetilde{p}_i + \alpha \cdot \sum_{v_j \in \operatorname{Adj}(v_i)} \frac{w_{ji}}{O(v_j)} S(v_j) \tag{12}$$

$\widetilde{p}_i$ for for vertex v_i is found in vector $\widetilde{p}$ which has the information about each candidate word inverse position in the document. w_{ji} is the PageRank score between node v_j and v_i and similarly w_{jk} is computer in formula $O(v_j) = \sum_{v_k \in Adj(v_j)} w_{jk}$. Candidate words in a document are concatenated to form candidate phrases, by considering only noun phrases using are filtered using regular expressions. The final scores are the sum of the scores of the candidate words in a phrase. The highest-scoring N phrases are considered the output of the algorithm.

MultipartiteRank [7] is a very similar approach to TopicRank, which is more recent and has more advanced features [36]. MultipartiteRank capture position information by adjust edge weights using an in-between step. The algorithm is based on TopicRank

where keyphrase candidates and topic are identified. Additionally, it includes constructing a directed multipartite graph where keyphrase candidates are the nodes which are connected through an edge, only if they belong to different topics.

SGRank is an unsupervised method that combines both statistical and graph-based techniques [13]. The algorithm has four stages to process an input document. First, it extracts all the n-grams and eliminate those that have low probability to be keyphrases. The elimination is done by checking if the phrase contains stop words, punctuation marks, or POS tags that are not nouns or adjectives. Additionally, a threshold for the frequency of each phrase is used to filter out infrequent terms. Second stage is ranking candidate phrases using a modified version of TFIDF similar to the one used in KPMiner. In the next stage, the top T ranking candidate phrases are reranked using additional heuristics. The additional statistical heuristics are the position of the first occurrence of the candidate, candidate length, and subsumption count. The final stage is to use the ranking from stage three to construct a graph, where candidates with positive scores are nodes and the edge between two nodes means they co-occur within a window of width d. Then, PageRank algorithm is used to obtain the final ranking, and choose the top N candidates as keyphrases.

Sentence Embeddings based approaches

Sentence embedding is a popular method that is used for words representation, which is utilized by keyphrase extraction approaches [36]. EmbedRank is one of the approaches that uses sentence embeddings to rank keyphrase candidates [6]. The algorithm has three main steps. First, the selection of candidate phrases based on POS sequences. Second step is to rank the selected phrases using sentence embeddings. The step includes computing a document embedding, which is a noise reduction process, as only the adjectives and nouns in the document are kept. Then, Using the same algorithm an embedding is computed for each candidate phrase separately. The candidates ranking is computed using the cosine distance between the document embedding and the phrase embedding. Maximal Marginal Relevance (MMR) is used to ensure diversity between the selected keyphrases. The cosine similarity based ranking ensures informativeness and the MMR score ensures dissimilarity among the selected keyphrases. MMR is calculated using the following formula:

$$\text{MMR} := \underset{C_i \in C \setminus K}{\arg\max} \left[\lambda \cdot \widetilde{cos_{\text{sim}}} \left(C_i, \text{doc} \right) \quad -(1-\lambda) \max_{C_j \in K} \left(\widetilde{cos_{sim}} \left(C_i, C_j \right) \right] \right. \tag{13}$$

Where C is the set of all candidate keyphrases, K is the set of the top selected ones, doc is the document embedding, C_i and C_j are the candidate phrases i and j embeddings, and $\widetilde{cos_{sim}}$ is the normalized cosine distance. λ is a diversity factor that controls the dissimilarity within the selected keyphrases.

Another similar approach using sentence embeddings combined with BERT was presented in [20]. BERT is a bi-directional transformer model that convert words or segments into embeddings based on their meaning [14]. BERT is a pre-trained model that captures the contextual relations between text segments. [20] used such embeddings with an algorithm similar to EmbedRank.

2.1.3 Semi-Supervised Approaches

Semi-supervised methods combine both supervised and unsupervised techniques to extract keyphrases [5]. A semi-supervised approach was presented in [26]. The model depends on the assumption that the document's title reflects the content. Thus both the document and the keyword should be semantically similar to the title. The algorithm requires constructing a semantic network, which is a hyper-graph. In the graph, phrases are the vertices, and the connections are weighted edges that measure the semantic relatedness among phrases. The semantic relations between phrases are computed using the universal knowledge base, Wikipedia. The approach exploits the document's intrinsic semantic features, which are represented by the relation between the title of the document and the topic. Additionally, it uses an external knowledge source to relate the different phrases semantically by building the graph. The semantic network is constructed by first mapping each candidate phrase to a Wikipedia article. The weights between two phrases (nodes) are calculated using the mutual links between every two articles representing the phrases. The weights for the edges are calculated as:

$$w(e) = \frac{\alpha}{|e|} \sum_{e_{ij} \subseteq e} w\left(e_{ij}\right) \tag{14}$$

Where $|e|$ is the total number of the vertices (phrases) in vector e, e_{ij} is an edge, and α is a balancing factor.

2.1.4 Domain specific models

Several keyphrase extraction approaches are designed for a certain domain that can be represented by corpora, which aims at capturing the knowledge, and the patterns in this corpora to extract keyphrases [39]. An example of an approach designed for the medical domain was presented in [37]. The approach aims to extract keyphrases from medical documents using document features and weights calculated based on external knowledge related to medical keyphrases.

Many keyphrase extraction approaches were evaluated and designed based on academic domain data (scientific articles) such as, Key2vec [27] a phrase embedding approach, TextRank [32], KP-Miner [15], CeKE [12], SGRank [13], PositionRank[17], and TextRank [32]. Keyphrase extraction for scientific articles is a popular task because there are millions of scientific articles on the web and in many cases authors provide

manually annotated keyphrases [17]. The task itself is a popular baseline for keyphrase extraction since it was part of the workshop on semantic evaluation 2010, where systems for keyphrase extraction for scientific articles [25].

2.2 Bibliographic Data

A bibliography is the list of the sources that were referred to in the research work. BibTex is a tool that can be used for automating and managing such list [16]. A set of key-value pairs, a unique citation key, and a document type identify each entry in the list [3]. An example of a BibTex entry is illustrated in figure 2. The entry describes an article as the entry starts with `@` followed by the type of the entry, in the figure the text `@Article` defines the type and is the start of the entry description [3]. A unique identifier follows the type description for the entry, then a set of key-value pairs describing the entry. These values are manually described by the authors, or the publisher [16]. In the example, in figure 2 the set of values describes information about the article, including keywords, which is a set of phrases provided by the user to describe the article. Other important information is provided, such as year of publication, journal, abstract, and more. Such a list of values differs from one in terms of the information described. However, the format is similar and stable, which resulted in BibTex data being available in large amounts [16].

In [4] , the authors introduced two new programs *bibtosql* and *bibsql*, which are used to load BibTex entries or files into relational databases with an SQL interface. The interface makes that searchable data [4]. This work was the start of the TUG bibliography archive, which currently has more than 1.63 million BibTeX entries [35]. The archive is organized by topic and by the journal. All the data is formatted using BibTex. Therefore programs can be used to parse, load, and process the data for different applications. Another BibTex dataset was introduced in [43], the dataset has more than 600 thousand papers. The dataset is available online and has been used for large-scale coreference resolution.

BibTex dataset has been used for machine learning tasks such as automatically generate labeled data for citation field extraction [40]. A BibTex dataset was used to generate a large-scale data set with 41 million labeled string (CFE dataset). They tested LSTM model and a variation of BERT model trained using the dataset for CFE task. The BERT model trained with the dataset showed a 24.48% relative error reduction compared with the previous state-of-art performance.

```
@Article{Bailey:1998:FNM,
  author =        "David H. Bailey",
  title =         "Finding New Mathematical Identities via
                  Numerical
                  Computations",
  journal =       j-SIGNUM,
  volume =        "33",
  number =        "1",
  pages =         "17--22",
  month =         jan,
  year =          "1998",
  CODEN =         "SNEWD6",
  DOI =           "https://doi.org/10.1145/381866.381887",
  ISSN =          "0163-5778 (print), 1558-0237 (electronic)",
  ISSN-L =        "0163-5778",
  bibdate =       "Tue Apr 12 07:50:30 MDT 2005",
  bibsource =     "http://portal.acm.org/;
                  http://www.math.utah.edu/pub/tex/bib/signum.bib"
                  ,
  abstract =      "A recent development in computational
                  mathematics is
                  the use of high-precision numerical computations
                  ,
                  together with advanced integer relation
                  algorithms, to
                  discover heretofore unknown mathematical
                  identities.
                  One of these new identities, a remarkable new
                  formula
                  for $\pi$, permits one to directly compute the
                  $n$-th
                  hexadecimal digit of $\pi$, without computing
                  the first
                  $n - 1$ digits, and without the need of
                  multiple-precision arithmetic software.",
  acknowledgement = ack-nhfb,
  fjournal =      "ACM SIGNUM Newsletter",
  journal-URL =   "http://portal.acm.org/browse\_dl.cfm?idx=J690",
  keywords =      "BBP (Bailey, Borwein, Plouffe) formula; PSQL",
}
```

Figure 2. BibTex Example

3 Methodology: Keyphrase Extraction Pipeline

Keyphrase extraction pipelines are designed to provide an end-to-end implementation of the process [36]. In this chapter, the different components of the pipeline are discussed based on the analysis of different keyphrase approaches. In addition, the chapter describes a new semi-supervised keyphrase extraction technique, which represents a new proposed pipeline.

3.1 The Pipeline

Many surveys have analyzed and discussed the different approaches for automatic keyphrase extraction [31, 36, 21, 30, 5, 39, 46, 9]. Based on these studies, the typical keyphrase extraction pipeline is illustrated in figure 3. The pipeline has four main steps preprocessing, candidate selection, feature selection, and keyphrase ranking. Keyphrase selection can be considered to be part of the ranking step.

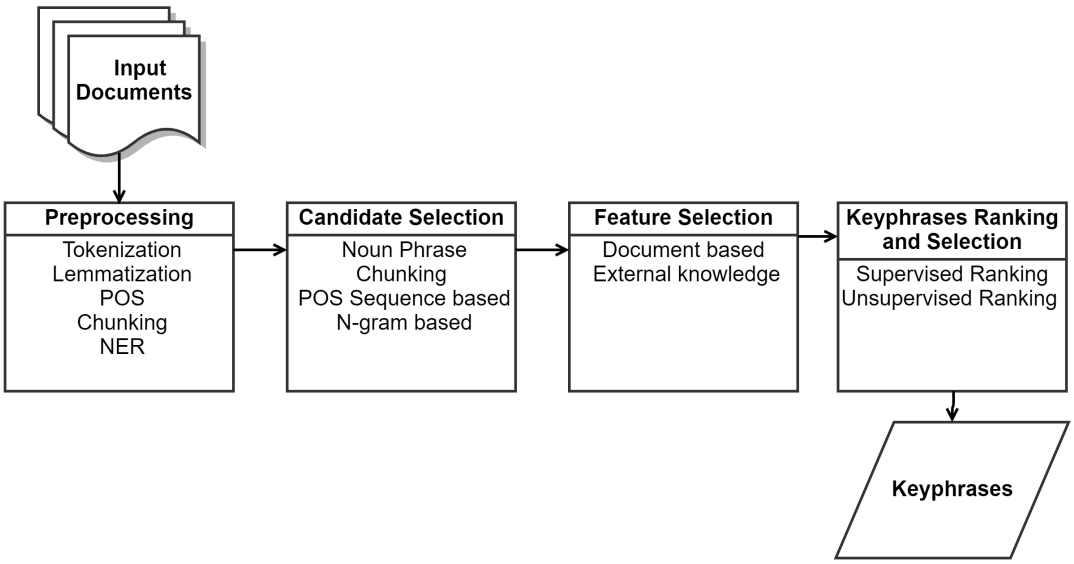

Figure 3. Keyphrase Extraction Pipeline

3.1.1 Preprocessing

The preprocessing step aims to make the text ready for further processing by converting it into a more machine-readable format, which reduces its complexity [31]. Preprocessing includes several operations such as tokenization, lemmatization, stemming, POS tagging, stop words removal, and NER. Typically, the first step of preprocessing is segmenting the

text into sentences and then tokenizing the sentences into terms. Most of the approaches discussed in chapter 2 implemented the tokenization step by splitting sentences into space-separated terms (words). Stop word removal is eliminating the words that occur in most English sentences since they do not add much information to the text. A few examples of such words are "the", "a", and "an". POS tagging is applied by assigning a tag describing the term based on its definition and context, such as "noun", "verb", and "adjective". Similarly named entity extraction process where the tags are assigned to one or more words (phrases), representing entities such as organization, locations, and names. The output of the preprocessing step is a list of tagged or untagged terms of the processed text. Figure 4 is an example of two processes from preprocessing step of an input sentence, where the POS tag are "PRON" for pronouns, "AUX" for auxiliary verbs, "ADJ" for adjectives, "NOUN" for nouns, "ADP" for adpositions (prepositions and postpositions), and "PROPN" for proper nouns.

1. Input Text: There are many different types of approaches for keyphrase extraction.
2. Tokenziation: "There", "are", "many", "different", "types", "of", "approaches", "for", "keyphrase", "extraction", "."
3. POS tagging: PRON, AUX, ADJ, ADJ, NOUN, ADP, NOUN, ADP, PROPN, NOUN

Figure 4. Preprocessing example

3.1.2 Candidate Selection

Candidate phrases are selected from the processed text using multiple techniques. The two most used techniques are n-gram sequencing and noun-phrase chunking (NP-chunking). N-gram sequencing is based on choosing "n" number of terms, for example, two or three (bi-grams, tri-grams). Then, several rules are used to limit the number of n-grams used as candidate phrases. [15] used n-gram sequencing for candidate selection. Some rules were used to filter the phrases and output the candidate terms. For example, the phrase would not be separated by a punctuation mark. Another rule is that the phrase would not contain a stop word. Additional conditions were used based on the frequency of terms in the documents and positions of the term. Other approaches used similar techniques such as [13]. NP-chunking is a process of extracting noun phrases from a text, and mainly it is done by exploiting POS tags [31]. For example, rules would be defined based on the POS tags, that a noun preceded by a set of adjectives is a noun phrase. The output of the candidate selection step is a list of terms or phrases which are the candidates to be keyphrases.

3.1.3 Feature Extraction

Keyphrase extraction methods exploit different types of features to evaluate the importance of candidate phrases [36]. Features are part of both supervised and unsupervised approaches, but they have been a critical element of the supervised ones as they affect the performance of such models [31]. There are different types of features. Typically, a keyphrase extraction method uses a set of features of different types. Features can be grouped into two main categories external-knowledge based features and in-document based features. External features are based on external knowledge sources such as Wikipedia. [26] exploited external features by using Wikipedia to create relatedness weights between phrases. External features are not widely used as they can be more computationally expensive than in-document features.

In-document features are extracted using the text of the document. They can be divided into several subcategories statistical features, positional features, linguistic features, and context features. Statistical features depend on the frequency and statistical scores. $TFIDF$ score is one of the most popular statistical score used in keyphrase extraction methods [36]. The score is calculated as explained in equation 5, where TF is the term frequency, and IDF is defined in equation 6. Other statistical features are used, such as term frequency, term length, and term entropy [39]. Examples of methods that have used statistical features are [45], [34], [12], and more. Positional features are designed based on the document's structure, where the position of a sentence or phrase is used as a feature [31]. The position features can include information related to titles, sections, abstracts, and citation contexts [36]. The first occurrence of a term in a document is one of the most popular position features [39]. PositionRank [17] exploits both position features, and statistical features. They assign weights to terms by evaluating the first occurrences of the term and its frequency. Linguistic features are designed based on the format of the text, the context, and the definition of the term [36]. POS tagging is an example of linguistic features. Context features are main features that consider a sequence of terms that defines the context [31]. For example, deep learning methods with sentence embeddings defines this context as the sentence. Most of the approaches discussed in chapter 2 use different features or combine such features. The goal of the feature extraction process is to extract features and use them to assign weights or values of features associated with candidate phrases.

3.1.4 Keyphrases Ranking and Selection

Ranking is the process of sorting the candidate phrases according to their importance. Mainly the ranking is based on the weights calculated from the feature extraction step. The problem of keyprhase extraction can be considered a ranking problem, where supervised or unsupervised approaches are used to rank candidate keyphrases [31]. Graph methods are designed so that terms are represented as nodes, and features are used

to build the connections. Then an algorithm such as PageRank is used to rank the terms in the graph. This ranking method was used in TextRank [32], PositionRank [17], and SGRank [13]. The output of the ranking process is a sorted list of candidate phrases, where the first N phrases are selected to be keyphrases.

3.2 BibRank

BibRank is a new semi-supervised automatic keyphrase extraction approach that exploits bibliographic data. The new algorithm introduces a new weighting technique using bibliographic data and combines features from PositionRank approach [17] and citations-context-based approaches. CeKE [12] keyphrase extraction method that uses citations in a scientific article to define a context and build a citation-based network as a feature for candidate phrases ranking. The method was proven to improve the performance compared with the other state-of-art models. This approach was an inspiration to exploit the bibliographic data, which can be used to create a similar context, but with more information captured. The proposed algorithm has three main steps (1) preprocessing; (2) candidate selection; (4) feature selection; (3) ranking and selection.

3.2.1 Preprocessing and Candidate Selection

Preprocessing consists of two steps tokenization and stemming. Tokenization is done by segmenting the text into sentences first, then split each sentence into words. Words are defined as space-separated tokens. Similar to the PositionRank approach, the second step in preprocessing is word stemming. Word stemming is the process of reducing words with the same root to the same form by removing suffixes and prefixes to the root [44]. Most stemming algorithms focus on removing suffixes or additions to the right-hand of the word. Porter stemmer is a simple stemming approach that was proven to work well in practice [44]. The stemmer is used in the preprocessing step. Lemmatization is the process of converting words into a meaningful base form, which can be based on context [28]. It was considered for the second preprocessing step instead of stemming. Both stemming and lemmatization resulted in similar performance when the model was evaluated.

For candidate selection, the NP-chunking-based technique is used. A POS tagger is used to first assigns tags to the words, and then it extracts the noun phrases from the text. The tagger used is part of Stanford CoreNLP Natural Language Processing Toolkit [28]. The toolkit has a tokenizer that can tag the text and extract the NP chunks, which are the candidate phrases.

The feature selection process has two main parts. One is extracting position score, and the second is extracting Bib score. PositionRank approach is used to get the position scores. The approach constructs a graph where the candidate phrases are the nodes, and the edges represent the occurrence information [17]. PageRank algorithm is then applied,

and the output score is the position score. The scores are calculated using the equation 12. The second score is based on bibliography data.

3.2.2 Bib Weights

Bib weights are first calculated across the related document (bib context) to get the bib weights for each candidate term. This context is defined by the publication year, journal, or the topic of the article. Then a set of bib records of documents representing this context is retrieved. The keywords values are used to calculate the bib weights. Since the bib records are sets of key-value pairs, the values of keywords can be extracted. For each keyword that appears in a record, a weight value is calculated as follows:

$$\lambda_p = \frac{\alpha}{|P|} \sum_{d \subseteq D} c_{pd} \tag{15}$$

Where λp is bib weight, α is a normalization factor, P is the set of all terms, d is a document, and c is the occurrence of a phrase in a document. The list of the weights from the bib context data is applied on the input document candidate phrases and update the position scores, and the equation 12 is updated with the bib weights as follows:

$$S\left(v_i\right) = (1-\alpha) \cdot \widetilde{p}_i + \alpha \cdot \sum_{v_j \in \mathrm{Adj}(v_i)} \frac{w_{ji}}{O\left(v_j\right)} S\left(v_j\right) + \lambda_i \tag{16}$$

Where $S\left(v_i\right)$ is the final weight value used by BibRank, which combines position scores and bib scores.

3.2.3 Ranking and Selection

Finally, the candidate keyphrases are ranked using the combined scores. The keyphrases are identified by selecting the first N phrases. Since the weights are normalized, they are used as a confidence measure for the keyphrases. The model output is a set of pairs of phrases and scores.

4 Evaluation and Metrics

Evaluation is an additional step that can be added to the keyphrase extraction pipeline, where the goal is to evaluate the extracted keyphrases [31]. There are two main approaches for the evaluation stage, manual evaluation, and automatic evaluation. In case of manual evaluation, the list of extracted keyphrases is presented to a number of reviewers then they are asked to rank each phrase in terms of its relevance [33]. The problem with the Human-based evaluation approach is that it is time-consuming, very expensive, and subjective [31, 33]. On the other hand, automatic evaluation assigns a score to measure how well the list of extracted keyphrases matches a predefined gold-standard list of keyphrases [36]. This approach requires a dataset of texts with their associated gold-standard keyphrases and evaluation metrics to assign scores for the extracted keyphrases.

In this research, a number of automatic evaluation experiments were run to evaluate the proposed method, and compare it to a number of keyphrase extraction methods. This chapter discusses several datasets and evaluation metrics that are used in the evaluation process. In addition, a new dataset that exploits bibliographic data is introduced.

4.1 Datasets

Several datasets have been used to evaluate keyphrase extraction systems, which are from different sources, such as scientific publications, news articles, or web posts [36] . In this research, the focus is on scientific publications based datasets, since the proposed keyphrase extraction approach is designed for scientific articles. The datasets used in the evaluation process are given in Table 3. For each dataset, the name of the dataset, the number of documents, the type of documents, and the annotation technique are stated. Type of document is whether the full paper is stored or only the abstract of the paper is stored. Annotation information describes the source of the annotation which can be the author of the paper, professional indexers, or readers of the paper [36].

ACM dataset was proposed in [38]. The dataset was created using 2,304 articles from 254 different journals published, ranging from World Journal of Urology to Abdominal Imaging. The dataset consists of pairs of the full text of the paper and the list of gold standard keyphrases that represents the text. The dataset does not contain other related information such as publication year or journal name. Similarly, NUS dataset is another full text-based dataset that was introduced in[33]. The dataset consists of 120 documents. Each document has two lists of keyphrases, one provided by the original author and the other by readers. The creators of the dataset mentioned that they used Google SOAP API to download the 211 academic papers. However, they did not mention any criteria of the selection process or more information about the sources of the articles. Inspec is a dataset that consists of scientific documents abstracts [22]. The dataset

contains abstracts ranging from the year 1998 to 2002. The documents are journal papers selected from different disciplines such as Computers and Control and Information Technology. The exact journal and each paper publication year is not included in the dataset. Professional indexers annotated the abstracts with a set of gold standard of keyphrases. WWW and KDD are two similar datasets that were introduced in [12]. They are created using scientific articles from machine learning conferences: World Wide Web (WWW), Knowledge Discovery and Data Mining (KDD). The included the keyphrases are described by the authors.

Table 3. Evaluation Datasets

Dataset	Documents	Type	Annotation
ACM [38]	2,304	Full papers	Authors
NUS [33]	211	Full papers	Authors/Readers
Inspec [22]	2,000	Abstracts	Indexers
WWW [12]	1,330	Abstracts	Authors
KDD [12]	755	Abstracts	Authors
BIB Dataset	**18,193**	**Abstracts and Metadata**	**Authors**

These datasets have been included in many evaluations, and surveys of automatic keyphrase extraction task [36]. This led to one of the drawbacks of the current keyphrase extraction systems, that is, they are not tested on many datasets [31]. The mentioned datasets are simply lists of pairs of text and a set of keyphrases, which means that important information about the data is missing, such as publication year, journals, and topics. For example, if a dataset is created using papers from one topic only, such as KDD or WWW, it would be hard to be confident about the model performance in different field or topic. This missing data is a drawback of the current methods of evaluation for keyphrase extraction approaches.

4.1.1 BibRank "Bibliography" Dataset

The proposed dataset aims at solving the two mentioned drawbacks by providing information-rich and extendable data. The dataset utilizes the bibliographic data that is available on the web. This data is available in BibTeX format, where the files contain hundreds and sometimes thousands of Bib records of research papers. The records contain metadata about the papers. Millions of records were parsed, processed, filtered, and formatted to build BibRank dataset.

The TUG bibliography archive was built based on the work in [4]. The Archive has more than 1.63 million BibTeX entries that are structured into several files. All the available bib files were processed to produce the dataset. The dataset has 18,193 processed document. Each entry has 22 attributes. The list of attributes represents values

that appear in a bib record. Figure 5 lists the most frequent fields. Bib records do not have identical fields, making some of the fields more frequent than the others. The data was filtered so that they must include three attributes which are ***title***, ***abstract***, and ***keywords*** (gold standard keyphrases). This means each entry in the dataset must have a title, abstract, and list of manually annotated keyphrases. Each bib record is parsed as a set of key-value pair. Where the key is the label, and the value is the information stored. All the information was extracted and stored as metadata for the record.

The data can be easily clustered using metadata stored for each record. For example, the sources or the journals of each abstract can be obtained by using an attribute called ***bibfile***. Then the attribute is used to map the file into one of the following topics: science history journals, Computer science journals and topics, ACM Transactions, Cryptography, Fonts and typography, IEEE journals, Computational/quantum chemistry/physics, Numerical analysis, Probability and statistics, SIAM journals, Mathematics, and Mathematical and computational biology. This means that each data entry in the BibRank dataset has a list of attributes that describes the source and the text itself, unlike other datasets that consist of random texts.

1. ***title*** the title of the paper
2. ***abstract***: the abstract of the paper
3. ***keywords***: gold standard keyphrases
4. ***year***: publication year of the paper
5. ***journal***: journal that paper was published in
6. ***bibfile***: Bib filename that contains the records
7. ***bibsource***: Databse or archive source of data

Figure 5. BibRank Dataset Frequent Attributes

Statistics about the dataset is described in Table 4. The dataset has more than 18,000 abstracts from several journals, Bib sources, and topics. The average word count per document and the average number of keyphrases is described in the table.

BibRank dataset is easy to extend by processing any bibliography data files in BibtTex format. Such files are easy to obtain, and they can be chosen with different constraints allowing better evaluation for the keyphrase extraction methods. Millions of BibTex files can be processed, parsed, filtered easily using the same technique. The records are compared to prevent any data duplicates from appearing in the data. The final output

Table 4. BibRank Dataset

Data	Count
Entries (abstract)	18,193
Journals	693
Bib Files	285
Topics	12
Avg Words	121
Avg Keyphrases	9

is information-rich data that can be used to run different experiments for better model evaluation.

4.2 Evaluation Process and Metrics

The evaluation process is done after choosing a dataset by comparing the extracted keyphrases to the list of gold standard phrases. Metrics assign a score for the comparison, which shows how similar the extracted phrases from the pre-annotated ones.

4.2.1 Evaluation Description

BibRank model was automatically evaluated using the annotated BibTex-based dataset and using datasets described in Table 3. Also, several supervised and unsupervised were evaluated too. The supervised and unsupervised models included in the experiments are a subset of the models listed in Table 1, and Table 2.

The proposed BibRank dataset allowed running different experiments by setting specific parameters for the testing data. Since the dataset includes more information about the data sources, the information can be used to choose a specific part of the data. For example, performance can be evaluated for each topic separately. It also allows BibRank model to use part of the data for tuning by generating the Bib scores for the model. Nevertheless, data leakage is prevented since BibRank dataset does not include duplicates, and each record has a unique set of values. In other words, one record cannot belong to two topics or have two publication years at the same time.

The goal of the evaluation process is to compare models' performance in different settings. The comparison is made by utilizing the information-rich BibRank dataset. The dataset allowed evaluation using data that belong to specific categories. Such evaluations are possible using BibRank dataset since it includes the information needed to cluster the data records into different groups. Commonly, the whole dataset is used for the evaluation process.

4.2.2 Evaluation Metrics

Evaluation metrics are used to assign a score to the keyphrase extracted by a method in comparison to gold standard phrases [21]. The scores are obtained by first, do an exact matching step to check whether the list of gold standard keyphrases and the list of extracted keyphrases are similar according to a matching strategy [31, 25]. After the matching step, evaluation metrics are used to assign scores according to the matching step output [21]. Recall (R), precision (P), and F1 score (F1) are well-known evaluation that have been adopted by many keyphrase extraction approaches [31]. Some approaches adopted metrics from other tasks, such as R-Precision, which is adopted from information retrieval task [23]. These metrics are described as follows [36].

Recall is defined as the fraction of the successfully extracted keyphrases, relevant to total number of gold standard keyphrases. It as calculated as:

$$recall = \frac{number\ of\ correct\ matches}{number\ of\ goldstandrd\ keyphrases} = \frac{TP}{TP + FN} \tag{17}$$

Where TP is the number of true positives, and FN is the number of False negatives.

Precision is the as fraction of the correctly extracted keyphrases, relevant to the total number of extracted keyphrases.

$$precision = \frac{number\ of\ correct\ matches}{number\ of\ gold\ standrd\ keyphrases} = \frac{TP}{TP + FP} \tag{18}$$

Where TP is the number of true positives, and FP is the number of False positives. F1 score is a weighed average precision and recall, and it is calculated as:

$$F1score = 2 \times \frac{precision \times recall}{precision + recall} \tag{19}$$

R-Precision is mainly used in information retrieval, and it is defined as the precision score when the count of retrieved documents is the same as the number of relevant or correct documents [31]. The score was adopted for keyphrase extraction evaluation in [23] and it is computed as follows:

$$RPrecision = \frac{number\ of\ overlapping\ word(s)}{length\ of\ keyphrase/candidate} \tag{20}$$

Where the length is for the longest keyphrase in list the extracted phrases. They introduced a modified version of the score where the position of the word in the phrase affects the score, and the total number of words is part of the calculation too. [23].

In the experiments presented in this research, *recall*, *precision*, and *F1 score* were adopted as the primary evaluation metrics. *R-Precision* was adopted only in cases where the three scores are very similar, and an extra score is needed. R-Precision is mainly an information retrieval metric and is not widely used by keyphrase extraction methods [21].

For each experiment, two sets of before-mentioned scores are calculated using the four evaluation metrics. One using the full list of gold standard keyphrases, and the other one is the same set of evaluation metrics with a different version of the gold standard list. The gold standard keyphrases list is adjusted to include only phrases that occur in the input text. Since the task is keyphrase extraction, the phrases should occur in the text to be extracted using the presented methods.

5 Experiments Results and Discussion

The experiments presented in this chapter are designed to evaluate the proposed approach (BibRank) in different settings and compare the performance to other approaches. Since BibRank is a semi-supervised approach, the data used to generate the Bib weights is described for each experiment. The data used to generate the weights was not part of the testing data for the same evaluation. The data was also chosen in a way that practically easy to obtain. For example, if the testing data is chosen between a specific set of years, then the data used for training the weights is chosen from a range that precedes the testing range. Experiments were conducted similar to the evaluation process described in chapter 4.

5.1 Keyphrase Extraction Platform

A platform was developed to support the full pipeline of keyphrase extraction. The platform is available online [1]. The platform includes the implementation of the newly proposed method BibRank. In addition, it includes the 18 different implementations of keyphrase methods, which are mainly listed in table 1 and table 2. The platform can be used to evaluate any of the 19 methods, using any of the datasets described in table 3. The data used for evaluation can be filtered using parameters that utilize the metadata stored in BibRank dataset. For example, the evaluation can be run for a certain topic(s), journal(s), or a range of years. The platform evaluates the performance using four evaluation metrics described in section 4.2. The output is formatted as a JSON file. All the information about the data used and the performance scores are stored.

5.2 Results

Table 5 describes three sets of experiments, using three different parts of the Bib dataset. The dataset has 12 categories or topics which are described in section 4.1.1. The table describes experiments using three categories *Computer science (compsci)*, *ACM*, and *history, philosophy, and science*. *Computer science (compsci)* category includes papers that were published in different journals for computer science. For Computer Science

[1] https://github.com/dallal9/keyphrase

category, 5,671 were used for testing filtered by publication year where the range used was between 1988 and 2020. This range was set to prevent data leakage, as papers published between 1980 and 1987 were used to generate the Bib weights used by BibRank model. Similarly, *ACM* category was split into testing part, which is 2,516 filtered by years 1987 to 2020. Second part is for generating Bib weights, which is 270 documents filtered by years 1986 to 1987. Finally, *history, philosophy, and science* category had similar split, but it had 254 test documents. Each document in the testing data is represented by an abstract and a list of gold-standard keyphrases. Each model is used to extract keyphrases using the default parameters. The output is evaluated according to the process described in 4.2. The list of the extracted keyphrases is compared to the gold-standard list and assigned a score. The scores are represented by three evaluation metrics which are precision (P), recall (R), F1 score (F1). Each model was evaluated using the default parameters, and

Table 5. Methods Evaluation

	Computer science			**ACM**			**History and Science**		
	P	R	F1	P	R	F1	P	R	F1
YAKE	0.064	0.114	0.068	0.031	0.123	0.0476	0.056	0.195	0.082
SGRank	0.117	0.186	0.121	0.038	0.159	0.0594	0.070	0.270	0.105
TextRank	0.113	0.177	0.117	0.033	0.141	0.0519	0.068	0.253	0.100
EmbedRank	0.106	0.190	0.113	0.044	0.190	0.0688	0.068	0.226	0.098
keyBert	0.089	0.158	0.094	0.041	0.162	0.0632	0.069	0.245	0.101
BibRank	**0.161**	**0.284**	**0.170**	**0.066**	**0.286**	**0.1030**	**0.093**	**0.346**	**0.137**

Similarly, Table 6 describes a set of experiments with more focus on BibRank method. The testing data is the data papers from computer science journals, which are labeled by "Compsci" in the dataset. The data was filtered by year and only the year 2019 was considered for testing. This part consists of 30 labeled documents. Each model listed was evaluated using the data. BibRank model was evaluated using a set of different parameters. The parameters were changes for the criteria of choosing data for generating Bib weights. The different settings were (1) data from math journals filtered by years from the range 2012 to 2014, (2) data from history journals filtered by years 2017 and 2018, (3) data from the same testing topic (computer science), but filtered by years 1999 and 2000, and (4) finally data from the same topic and filtered by years 2017 and 2018 which are the two years preceding the one used to filter testing data.

5.3 Discussion and Analysis

The goal of the first set of experiments was to evaluate and compare the models in different settings. The results are described in table 5. YAKE [10] which is a statistical

Table 6. Bib Weights Criteria Evaluation

	Compsci - year 2019			
	P	R	F1	Weights data
YAKE	0.031	0.030	0.030	
SGRank	0.125	0.132	0.123	
TextRank	0.1375	0.1522	0.138	
EmbedRank	0.184	0.217	0.187	
KeyBert	0.075	0.121	0.0794	
BibRank	0.199	0.216	0.199	No Bib weights
	0.20	0.216	0.207	Weights from 2012 and 2014 - math data
	0.203	0.218	0.210	Weights from 2017 and 2018 - history data
	0.202	0.220	0.211	Weights from 1999 and 2000 - compsci data
	0.237	**0.266**	**0.240**	**Weights from 2017 and 2018 - compsci data**

method that exploits several features such as frequency and position, had the lowest performance across the three different experiments. It scored an average F1 score of 0.065. The results suggest that statistical approaches have limitations because they do not exploit any relations between the different parts of the text. on the other hand, SGRank had the second-best F1 score in the case of Computer Science journals and History and Science journals. Although SGRank uses statistical techniques similar to YAKE, it had much better results. SGRank scored average F1 score of 0.095. SGRank exploits both statistical and graph techniques, which led to the improvement in the performance. TextRank method, which uses only graph features, had comparable results to SGRank. It had an average score of 0.089. This shows that the graph features are effective, and they can improve performance. Besides, it suggests that combining different types of features can improve performance further.

EmbedRank method had an average F1 score of 0.093, which is very close to SGRank. EmbedRank had the second-best performance score in the case of ACM papers and the second-worst performance score in the case of History papers. The results suggest that methods that use pre-trained models such as BERT have certain biases that affect these methods' results. Similarly, KeyBert did not perform the same across the three types of data. Keybert had an average F1 score of 0.086, which is only better than YAKE approach. This suggests that using the number of terms to be included as candidate phrases as an input has lowered the model performance. Although both EmbedRank and KeyBert use sentence embeddings and BERT language model, EmbedRank had a better average F1 score. The reason is that EmbedRank utilizes more features that KeyBert. The latter model converts all the phrases in the document and the document itself into embeddings. Then it compares the cosine distance between each phrase embedding and the document embedding and rank the phrases according to the distance. EmbedRank

has a similar process, but it has two extra steps. One during candidate selection, where it uses noun-phrase chunking (NP-chunking) technique for the process. The second is by improving the cosine distance measurements by adding a trade-off parameter that ensures diversity between the selected keyphrases. This is another evidence that combining features improves the performance of keyprhase extraction emthods.

BibRank scored an average F1 score of 0.136, which is the highest average score across all the tested methods. BibRank exploits statistical, positional, and word co-occurrence information. Also, it utilizes the context of information of the related documents by using Bib weights. The results suggest that the combination of such features improved the model performance.

Table 6 describes a different experiment. The results of the tested method are similar to the ones in table 5, Where YAKE has the lowest F1 score of 0.03. SGRank and TextRank had comparable F1 scores of 0.123 and 0.138, which is higher than YAKE. EmbedRank a higher score of the three which is 0.187, but KeyBert performance dropped to 0.0794. In this experiment, the focus is more on BibRank. First, the model is evaluated without Bib weight. The F1 score 0.199, which is a bit higher than EmbedRank. When data from a different type of journals is used to generate Bib weights, only a slight improvement occurs. Where in the case of math journals, the score was 0.210, and when History and Science journals were used the score was 0.210. Similarly, when the data used is out-of-date, data filters by years 1999 and 2000, the performance had a slight improvement, and the score was 0.211. The F1 score reached 0.240 when the same type and close data in terms of publication years was used. This suggests that the results are sensitive to context. Also, the utilization of the related data to that context and improve the results.

Figure 6 summarizes the information in table 5, and table 6. The figure shows that BibRank model with the Bib weights had a stable and high performance compared to other methods. Where it had the highest F1 score in all the experiments. Other tools have a fluctuating performance. This suggests that context information utilized by BibRank, affects the results and is included in the different model biases. For example, the difference in performance between YAKE and EmbedRank is large in case of Computer Science data, but it is much less in case of History data. BibRank did not have such cases, where the model surprisingly drops. The Bib weights helped the model to adapt to different types of data.

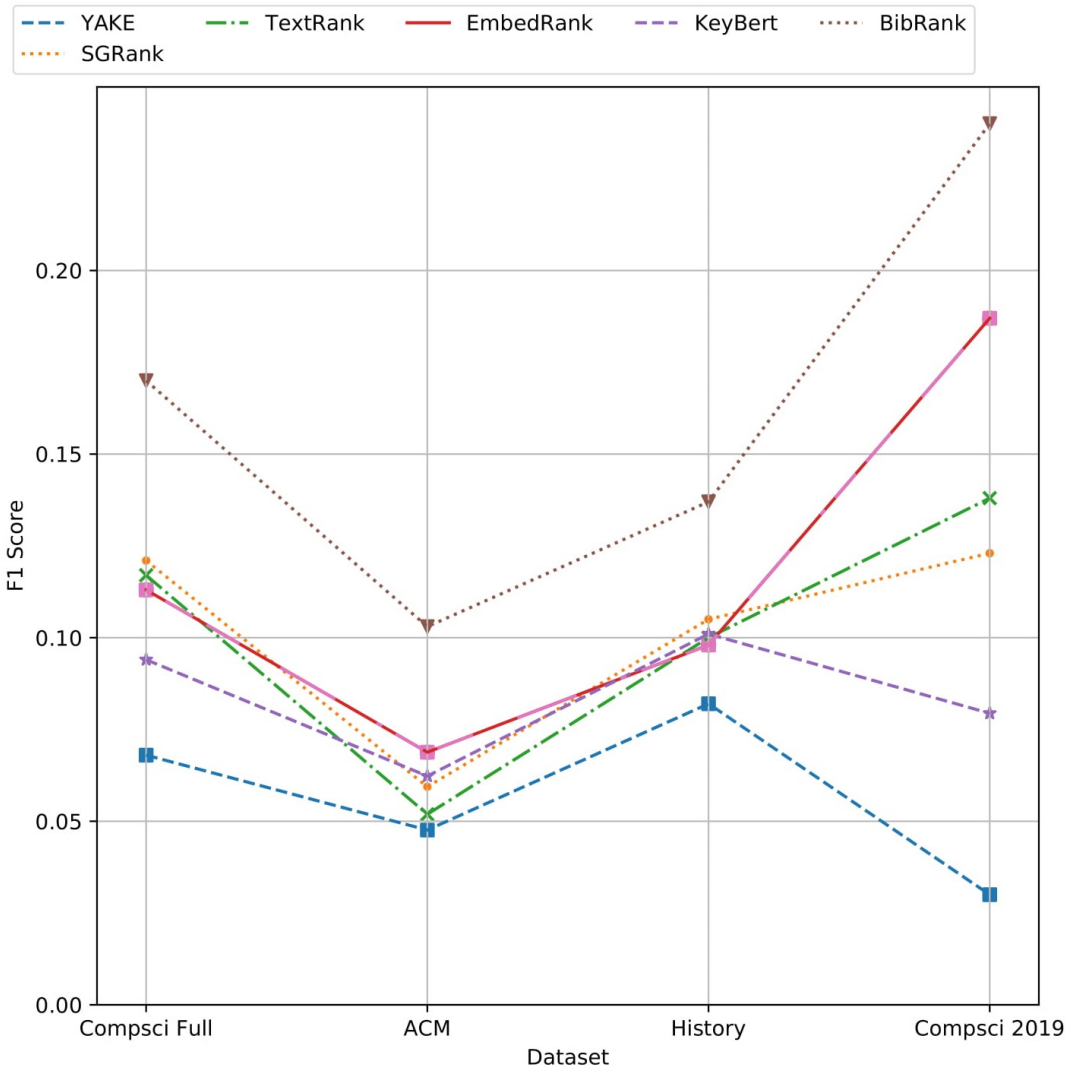

Figure 6. Experiments Summary

5.4 Detailed Example

Figure 7 describes a real example of keyphrase extraction. The title and the abstract of a research paper is described. They are the input to the keyphrase extraction method. The figure lists a set of gold standard keyphrases that are used to evaluate the extracted keyphrases. The figures also describes the actual output of each model.

5.4.1 Methods Output

YAKE model extracted the frequent words, and they were sorted according to position too. N-gram construction method was used to combine words. However, the heuristic measure used to determine the relevance of phrases seems more biased towards single terms or words. Another explanation is that these words are the most frequent ones. The model had one exact match, which is the word "vlsi". The fact that the model depends mainly on statistical features made the model extract phrases that should not be counted as keyphrase in general, but they are frequent. Such words are, for example, "knowledge" and "paper". The two words need context to be more meaningful.

SGRank and TextRank had similar output. Both models extracted several keyphrases that vary in length, position, and frequency. The phrases seem to be more meaningful than the ones extracted by YAKE method. Although they do not include the gold standard phrases, they are still a good representation of the text. For example, "design automation assistan" and "vlsi design" are good phrases that both models extracted. However, they also had the same error of extracting the word "paper" because of the text's frequency and position.

EmbedRank had some more complicated and longer keyphrases. It suggests that the sentence embedding model used helps the model processing sentences and long phrases. Some of the extracted phrases represent the text, but they did not include the gold standard ones.

KeyBert has a primary parameter that caused the model to perform poorly. The model has the range of n-grams used as input. When the parameter is set to 1, it means it looks for words to be keyphrases. The model is similar to the YAKE model but with better features by using deep learning. When the range is set to a larger value such as 2 or 3, the model becomes biased towards keyphrases with more words only. KeyBert seems to have a problem balancing the number of terms included in a keyphrase. The two cases are illustrated in the figure.

BibRank with no weights has similar results to TextRank and SGRank. When Bib weights were used, the model extracted two exact matches: "vlsi" and "expert systems". The results show how the Bib weights can help the model to outperform the graph-based methods by utilizing a larger context.

5.4.2 Process Limitations

On average, the performance of keyphrase extraction methods seems low as the F1 scores are lower than the ones that can be achieved in other classification tasks such as POS tagging [36, 31]. One of the reasons is evident in the example illustrated in figure 7. The evaluation of the extracted keyphrases is done by matching and scoring them compared to the gold standard keyphrases. Exact or partial matching is used. However, any other phrases that can be a good representation of the topic are not considered as good classification. For example, in figure 7, the phrase "vlsi design" was extracted by the methods TextRank, BibRank, KeyBert, and BibRank, and it seems like a keyphrase, but the author did not manually annotate it. Another example is "design automation assistant", which was extracted by TextRank, SGRank, and BibRank. The phrase is a good representation of the paper and not also included in the gold standard list. The way gold standard terms are used limits the evaluation process since it only scores the matching between the extracted and annotated lists. Even if other phrases are important, they are not considered as such if they are not part of the gold standard list.

- **Title** : *The VLSI Design Automation Assistant - A synthesis expert*
- **Abstract** : *This paper describes an approach to VLSI design synthesis that uses knowledge-based expert systems to proceed from an algorithmic description of a VLSI system to a list of technology-dependent registers, operators, data paths, and control signals. It outlines how the Design Automation Assistant uses large amounts of expert knowledge to design an architecture with little back-tracking. This paper takes a retrospective look at that codified knowledge base, examining what has been learned about VLSI design. Finally, the paper gives an overview of current work in using bottom-up design information in the synthesis of integrated circuits.*
- **Gold Standard keyphrases** : 'artificial intelligence', 'circuit cad', 'expert systems', 'integrated circuit manufacture', 'automation', 'vlsi'
- ***YAKE***: 'design', 'knowledge', 'paper',**'vlsi'**, 'large', 'expert', 'base', 'system'
- ***SGRank***: 'codified knowledge base', 'design automation assistant', 'paper', 'control signal', 'expert knowledge', 'data path', 'large amount', 'dependent register'
- ***TextRank***: 'vlsi design', 'design automation assistant', 'design synthesis', 'codified knowledge base', 'design information', 'expert knowledge', 'vlsi system', 'expert system'
- ***EmbedRank***: 'this paper', 'an approach', 'very-large-scale-integration (vlsi) design synthesis', 'knowledge-based expert systems', 'an algorithmic description', 'a vlsi system', 'a list', 'technology-dependent registers', 'operators', 'integrated circuits'
- ***KeyBert (n-gram range 1:1)***: 'tracking', 'systems', 'control', **'vlsi'**, 'architecture', 'operators', 'automation', 'outlines'
- ***KeyBert (n-gram range 1:3)***:'vlsi design synthesis', 'vlsi design', 'design synthesis uses', 'based expert systems', 'current work using', 'knowledge design architecture', 'expert knowledge design', 'synthesis integrated circuits'
- **BibRank (No weights)**: 'vlsi design', 'design automation assistant', 'design synthesis', 'up design information', 'vlsi system', 'paper', 'large amounts', 'expert knowledge'
- **BibRank**: 'vlsi design', 'up design information', 'design synthesis', 'design automation assistant', 'vlsi system', **'vlsi'**, 'circuits', **'expert systems'**

Figure 7. Example for Keyphrase Extraction process

6 Conclusion and Future Work

This paper introduced BibRank, a new semi-supervised keyphrase extraction approach that exploits context, statistical, positional, and word co-occurrence information. The context is defined by several related documents discovered by utilizing metadata in Bib dataset. The proposed model utilizes a newly introduced dataset, which was constructed using BibTex formatted data. Unlike other keyphrase extraction datasets, the Bib dataset contains metadata about the documents. The model and the dataset are part of a platform that was developed to implement the full pipeline of the keyphrase extraction task.

Bib dataset was constructed using archives of Bibtex data which is a rich source of data. For each entry metadata was stored including publication year, topic, journal, and other information. BibRank was built to utilize the information stored in the data in addition to other features related to graph, statistical, and positional techniques. The model assumes that context represented by metadata can help identifying better keyphrase for input document. A number of experiments were conducted to test this assumption and evaluate BibRank and other methods . The results showed that other keyphrase extraction methods had a varying performance when the context changes, however BibRank had a more stable performance. Also, models with combined features and complex pipelines outperformed simple models. The newly proposed technique improved the performance when compared to other methods. The experiments showed that the combination of features used by BibRank is effective. The results proved that the context utilized by BibRank affects the performance and can improve it.

BibRank outperformed other methods evaluated in the experiments. However, overall the average F1 scores relatively low compared to other NLP classification tasks. This is due to limitations in the evaluation process, where the gold-standrd data do not include all possible cases of keyphrases. Also, this suggests that keyphrase extraction task is complex, and both keyphrase extraction methods and evaluation processes can be improved.

6.1 Future Work

Bib dataset can be extended to include more documents and topics. The process can be automated, where the Bib archives can be periodically downloaded or crawled, parsed, and appended to the dataset file. Thus, the data would be up-to-date. Similar automation can be implemented for BibRank model, especially the process of defining the context of the data used for generating Bib weights. This means a classification problem should be solved first. The problem would be to classify the input document to a certain category mapped to a context. A context is represented by the metadata for a set of documents related to the input. The goal of automation is to make BibRank less dependent on manual processes.

References

[1] Rabah Alzaidy, Cornelia Caragea, and C Lee Giles. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In *The world wide web conference*, pages 2551–2557, 2019.

[2] Marco Basaldella, Elisa Antolli, Giuseppe Serra, and Carlo Tasso. Bidirectional lstm recurrent neural network for keyphrase extraction. In *Italian Research Conference on Digital Libraries*, pages 180–187. Springer, 2018.

[3] Nelson HF Beebe. Bibliography prettyprinting and syntax checking. *TUGBoat*, 14(4):395–419, 1993.

[4] Nelson HF Beebe. Bibtex meets relational databases. *j TUGboat*, 30:252–271, 2009.

[5] Slobodan Beliga, Ana Meštrović, and Sanda Martinčić-Ipšić. An overview of graph-based keyword extraction methods and approaches. *Journal of information and organizational sciences*, 39(1):1–20, 2015.

[6] Kamil Bennani-Smires, Claudiu Musat, Andreea Hossmann, Michael Baeriswyl, and Martin Jaggi. Simple unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470*, 2018.

[7] Florian Boudin. Unsupervised keyphrase extraction with multipartite graphs. *arXiv preprint arXiv:1803.08721*, 2018.

[8] Adrien Bougouin, Florian Boudin, and Béatrice Daille. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International joint conference on natural language processing (IJCNLP)*, pages 543–551, 2013.

[9] Florin Bulgarov and Cornelia Caragea. A comparison of supervised keyphrase extraction models. In *Proceedings of the 24th international conference on World Wide Web*, pages 13–14, 2015.

[10] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alipio Jorge, Célia Nunes, and Adam Jatowt. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289, 2020.

[11] Erion Çano and Ondřej Bojar. Keyphrase generation: A multi-aspect survey. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pages 85–94. IEEE, 2019.

[12] Cornelia Caragea, Florin Bulgarov, Andreea Godea, and Sujatha Das Gollapalli. Citation-enhanced keyphrase extraction from research papers: A supervised approach. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1435–1446, 2014.

[13] Soheil Danesh, Tamara Sumner, and James H Martin. Sgrank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction. In *Proceedings of the fourth joint conference on lexical and computational semantics*, pages 117–126, 2015.

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

[15] Samhaa R El-Beltagy and Ahmed Rafea. Kp-miner: Participation in semeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 190–193, 2010.

[16] Jurgen Fenn. Managing citations and your bibliography with bibtex. *The PracTEX Journal,(4)*, 2006.

[17] Corina Florescu and Cornelia 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)*, pages 1105–1115, 2017.

[18] E FRANK. Domain-specific keyphrase extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence, 1999*, pages 668–673, 1999.

[19] Ygor Gallina, Florian Boudin, and Béatrice Daille. Large-scale evaluation of keyphrase extraction models. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, pages 271–278, 2020.

[20] Maarten Grootendorst. Maartengr/keybert. `https://zenodo.org/record/4461265`, 2021.

[21] Kazi Saidul Hasan and Vincent 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)*, pages 1262–1273, 2014.

[22] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223, 2003.

[23] Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. Evaluating n-gram based evaluation metrics for automatic keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 572–580, 2010.

[24] Su Nam Kim and Min-Yen Kan. Re-examining automatic keyphrase extraction approaches in scientific articles. In *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications (MWE 2009)*, pages 9–16, 2009.

[25] Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, 2010.

[26] Decong Li, Sujian Li, Wenjie Li, Wei Wang, and Weiguang Qu. A semi-supervised key phrase extraction approach: learning from title phrases through a document semantic network. In *Proceedings of the ACL 2010 conference short papers*, pages 296–300, 2010.

[27] Debanjan Mahata, John Kuriakose, Rajiv Shah, and Roger Zimmermann. Key2vec: Automatic ranked keyphrase extraction from scientific articles using phrase embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 634–639, 2018.

[28] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[29] Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879*, 2017.

[30] Zakariae Alami Merrouni, Bouchra Frikh, and Brahim Ouhbi. Automatic keyphrase extraction: An overview of the state of the art. In *2016 4th IEEE international colloquium on information science and technology (CiSt)*, pages 306–313. IEEE, 2016.

[31] Zakariae Alami Merrouni, Bouchra Frikh, and Brahim Ouhbi. Automatic keyphrase extraction: a survey and trends. *Journal of Intelligent Information Systems*, pages 1–34, 2019.

[32] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

[33] Thuy Dung Nguyen and Min-Yen Kan. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, pages 317–326. Springer, 2007.

[34] Thuy Dung Nguyen and Minh-Thang Luong. Wingnus: Keyphrase extraction utilizing document logical structure. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 166–169, 2010.

[35] University of Uta. The tug bibliography archive. `http://ftp.math.utah.edu/pub/tex/bib/`.

[36] Eirini Papagiannopoulou and Grigorios Tsoumakas. A review of keyphrase extraction. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(2):e1339, 2020.

[37] Kamal Sarkar. A hybrid approach to extract keyphrases from medical documents. *arXiv preprint arXiv:1303.1441*, 2013.

[38] Alexander Thorsten Schutz et al. Keyphrase extraction from single documents in the open domain exploiting linguistic and statistical methods. *M. App. Sc Thesis*, 2008.

[39] Sifatullah Siddiqi and Aditi Sharan. Keyword and keyphrase extraction techniques: a literature review. *International Journal of Computer Applications*, 109(2), 2015.

[40] Dung Thai, Zhiyang Xu, Nicholas Monath, Boris Veytsman, and Andrew McCallum. Using bibtex to automatically generate labeled data for citation field extraction. *arXiv preprint arXiv:2006.05563*, 2020.

[41] Xiaojun Wan and Jianguo Xiao. Collabrank: towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 969–976, 2008.

[42] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 1(1):396–413, 2020.

[43] Michael Wick, Sameer Singh, and Andrew McCallum. A discriminative hierarchical model for fast coreference at large scale. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 379–388, 2012.

[44] Peter Willett. The porter stemming algorithm: then and now. *Program*, 2006.

[45] Ian H Witten, Gordon W Paynter, Eibe Frank, Carl Gutwin, and Craig G Nevill-Manning. Kea: practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, pages 254–255, 1999.

[46] Torsten Zesch and Iryna Gurevych. Approximate matching for evaluating keyphrase extraction. In *Proceedings of the International Conference RANLP-2009*, pages 484–489, 2009.

Appendix

I. Source Code

The source code of the platform that includes Bibrank, and other methods implementations, datasets files and scripts, and evaluation scripts available in the following GitHub repository:

`https://github.com/dallal9/keyphrase`

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Abdelrhman Elsayed Hassan Eldallal**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   BibRank: Automatic Keyphrase Extraction Platform Using Metadata,

   supervised by Eduard Barbu.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Abdelrhman Elsayed Hassan Eldallal
30/04/2021