Proceedings of the Third International Conference on Computing Methodologies and Communication (ICCMC 2019)
IEEE Xplore Part Number: CFP19K25-ART; ISBN: 978-1-5386-7808-4

A Comparative Study on different Keyword Extraction Algorithms

M G Thushara
Department of Computer Science and Applications
Amrita School of Engineering
Amrita Vishwa Vidyapeetham
Amritapuri, Kerala, India
thusharamg@am.amrita.edu

Tadi Mownika
Department of Computer Science and Engineering
Amrita School of Engineering
Amrita Vishwa Vidyapeetham
Amritapuri, Kerala, India
tadimownika@gmail.com

Ritika Mangamuru
Department of Computer Science and Engineering
Amrita School of Engineering
Amrita Vishwa Vidyapeetham
Amritapuri, Kerala, India
m.ritikareddy@gmail.com

***Abstract*—Growth in the number of research documents getting published is increasing. Finding a research document under interested domain by referring the whole paper has become a tedious task. Keywords, Keyphrases gives the summary of the text. Keywords and keyphrases help in understanding the information described in the research document. The domain of a research document can be determined based on the keywords and keyphrases extracted. Extracting keywords and keyphrases manually is a tedious task. Automatic keyphrase extraction techniques help in overcoming this challenging task. This paper is a comparative study of unsupervised keyphrase extraction algorithms without using corpus. It compares the performance of PositionRank which considers the position of the all words occurrences in the document with TextRank and RAKE (Rapid Automatic Keyword Extraction).**

***Keywords*—Automatic keyphrase extraction, Unsupervised technique, relevant keywords.**

I. Introduction

Now a days large amounts of research documents are getting published and this document space is also increasing. Searching documents related to our area of interests is a challenging task. Summarization conveys the content of research paper without loosing it's originality and thereby helps in finding documents related to our area of interests [1]. Keywords, Keyphrases gives the summary of the text. A keyphrase denotes a multi-word lexeme (e.g. computer science engineering, machine learning), whereas a keyword represents a single word (e.g. computer, keyboard) [2]. Keyword extraction is an application of document classification, summarizing etc [3]. Extraction of relevant keywords helps in choosing the document in the area of interest to read upon [3]. Extracting keywords and keyphrases manually is a tedious task when document length is long and when multiple documents are there. Finding the relevant keyphrases which describe the information of the document is a challenging task. Automatic keyphrase extraction reduces these efforts and saves time.
Machine Learning approaches for keyphrase extraction are [2]:

- Supervised
- Unsupervised
- Semi-Supervised

In the supervised technique, the model is trained on data to identify if a phrase in the text is a keyphrase or not [4]. Supervised technique requires a lot of training data to extract appropriate key phrases which is one of the biggest challenge [4].

Recently the growth in the number of research documents getting published is increasing. If one has to find out a concept described in the paper, keywords will be helpful [3]. In these scenarios, keyword extraction without having the corpus of documents belongs to a similar domain is useful which is the unsupervised technique [3]. Unsupervised technique for keyphrase extraction does not use training data. It determines the keywords and keyphrases using the properties of text in the research document [5]. Even though supervised algorithms gives better results than unsupervised techniques, the collection of corpus belongs to the same domain is a challenging task [6]. This paper is a comparative study of various keyphrase extraction algorithms like PositionRank, TextRank and RAKE (Rapid Automatic Keyword Extraction).

Lately, TextRank algorithm scores keywords based on the co-occurrence connections between words [7]. In Rapid Automatic Keyword Extraction (RAKE), keywords have multiple content words which are informative rather than punctuation and stop words [8]. An unsupervised model, PositionRank is a biased PageRank which extracts keywords and keyphrases from documents that contain information from all positions of a words occurrences [6].

II. BACKGROUND STUDY

In Supervised technique, keyphrase extraction can be considered as classification problem which predicts whether a phrase in the research paper is a keyphrase or not [9].

TF*IDF-based baseline, in which keyphrases are scored and selected based on TF*IDF, which is used by both supervised and unsupervised techniques [10], [11], [12], [13]. In supervised learning, TF*IDF scores the keyphrases based on its rate of occurrence in the research paper and number of

times the word or a phrase occurs in the corpus of the same kind of research papers [14]. The words which are having less TF*IDF value are significant [15]. This performance is might not be efficient because the keyphrases are not necessarily to be the frequent terms in the document [9]. In unsupervised approach, tf (term frequency) is only taken in TF*IDF because when a single document is considered, idf value will be the same for every candidate word. We can find a better method to extract keyphrases when compared to this.

For unsupervised keyphrase extraction, graph-based ranking methods and centrality measures incorporated the newest ideas and perform the best [16]. These techniques construct a graph with words as nodes/ vertices according to the word co-occurrences in the text document and after that rank the nodes according to their scores [16]. Subsequently, key words with the highest scores are considered [16].

PageRank can be applied on the graph built with the adjacent words in a document to find the scores of keyphrases [17]. TextRank does not depend on a text unit's (vertex) native context, but depends upon the entire text (graph) from which the information is drawn recursively [17]. With the graph it builds on texts, TextRank connects different entities to text and develops the concept of suggestion. [17]. Words that are more informative for the given text is assigned high score as those will be highly recommendable [17]. TextRank needn't bother with profound semantic learning, nor area or language explicit clarified corpora, which can be effectively utilized in different domains, genres, or languages [17]. This methodology might not give the relevant keywords which best describes the concepts in the research document [9].

In Rapid Automatic Keyword Extraction (RAKE), keywords have multiple content words which are informative rather than punctuation and stop words [8].

In research documents, keyphrases frequently occur in the beginning of a document (e.g: Title, Abstract) [6]. The author predicted keyphrases are highlighted in the Fig.1. Notice that the keyphrase Markov chain frequently occurs at the beginning of the document (even from its title) [6].

Position Rank, an unsupervised graph-based algorithm, considers both the position of the word and its frequency in a document for keyphrase extraction [6]. All positional occurrences of words are considered in position rank which performs better than the methods which consider only first position of a word [6].

Topic modeling, an unsupervised algorithm that learns regarding concealed topics from an expansive set of research documents [18].

Factorizing Personalized Markov Chains for Next-Basket Recommendation

Steffen Rendle*
Department of Reasoning for Intelligence
The Institute of Scientific and Industrial Research
Osaka University, Japan
rendle@ar.sanken.osaka-u.ac.jp

Christoph Freudenthaler
Information Systems and Machine Learning Lab
Institute for Computer Science
University of Hildesheim, Germany
freudenthaler@ismll.uni-hildesheim.de

Lars Schmidt-Thieme
Information Systems and Machine Learning Lab
Institute for Computer Science
University of Hildesheim, Germany
schmidt-thieme@ismll.uni-hildesheim.de

ABSTRACT

Recommender systems are an important component of many websites. Two of the most popular approaches are based on matrix factorization (MF) and Markov chains (MC). MF methods learn the general taste of a user by factorizing the matrix over observed user-item preferences. On the other hand, MC methods model sequential behavior by learning a transition graph over items that is used to predict the next action based on the recent actions of a user. In this paper, we present a method bringing both approaches together. Our method is based on personalized transition graphs over underlying Markov chains. That means for each user an own transition matrix is learned – thus in total the method uses a transition cube. As the observations for estimating the transitions are usually very limited, our method factorizes the transition cube with a pairwise interaction model which is a special case of the Tucker Decomposition. We show that our factorized personalized MC (FPMC) model subsumes both a common Markov chain and the normal matrix factorization model. For learning the model parameters, we introduce an adaption of the Bayesian Personalized Ranking (BPR) framework for sequential basket data. Empirically, we show that our FPMC model outperforms both the common matrix factorization and the unpersonalized MC model both learned with and without factorization.

Categories and Subject Descriptors

I.2.6 [Artificial Intelligence]: Learning—*Parameter Learning*

General Terms

Algorithms, Experimentation, Measurement, Performance

Keywords

Basket Recommendation, Markov Chain, Matrix Factorization

*Steffen Rendle is currently on leave from the Machine Learning Lab, University of Hildesheim, Germany.

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

1. INTRODUCTION

A core technology of many recent websites are recommender systems. They are used for example to increase sales in e-commerce, clicking rates on websites or visitor satisfaction in general. In this paper, we deal with the problem setting where sequential basket data is given per user. An obvious example is an online shop where a user buys items (e.g. books or CDs). In these applications, usually several items are bought at the same time, i.e. we have a set/basket of items at one point of time. The target is now to recommend items to the user that he might want to buy in his next visit.

Recommender systems based on a Markov chain (MC) model utilize such sequential data by predicting the users next action based on the last actions. Therefore a transition matrix is estimated that gives the probability of buying an item based on the last purchases of the user. The transition matrix of the MC models is assumed to be the same over all users. The personalization is made by applying the (general) transition matrix on the user's last actions. On the other hand, one of the most successful model classes are factorization methods (MF) based on matrix or tensor decomposition. The best approaches [3, 4] for the 1M$ Netflix challenge[1] are based on this model class. Also on the ECML/PKDD discovery challenge[2] for tag recommendation, a factorization model based on tensor decomposition has outperformed the other approaches [8]. These models learn the general taste of the user disregarding sequential information. Both MF and MC have their advantages: MF uses all data to learn the general taste of the user whereas MC can capture sequence effects in time by using a non-personalized transition matrix, i.e. the transition matrix is learned over all data of all users.

In this paper, we present a model based on an underlying MC where the transitions are user-specific. We model a transition cube where each slice is a user-specific transition matrix of an underlying MC on the users basket history. With this personalization, we bring together both advantages of MC and MF: (1) the sequential data is captured by the transition matrix and (2) as all transition matrices are user-specific, the user-taste over all data is captured. Besides introducing personalized MCs, the central contribution of

[1] http://www.netflixprize.com/
[2] http://www.kde.cs.uni-kassel.de/ws/dc09

Fig. 1. Title and Abstract of a paper and the author generated keyphrases. The highlighted phrases denotes the author generated keyphrases [6].

III. RELATED WORK

Syntactic filters are applied in TextRank to a document text to distinguish content words and amass how often a word co-occur within a window size of 2 [8]. RAKE collects word co-occurrences within candidate keywords [8]. The algorithms calculate keyword scores according to their respective strategies after co-occurrences are counted [8]. The performance of RAKE is better than TextRank [9]. This is because of RAKE capacity to rank keywords in a solitary pass while TextRank needs continual iterations to obtain convergence on word scores [9]. The advantages of RAKE are its computational efficiency, speed, its precision despite its simplicity and ability to work on individual documents [8].

PositionRank, an unsupervised algorithm for keyphrase extraction from research papers considers all the positional occurrences of a word into a biased PageRank [6]. PositionRank's performance is better when compared to PageRank models that do not consider word position [6]. It accomplishes improvements as high as 29.09% [6].

IV. PROPOSED METHODOLOGY

Automatic keyphrase extraction deals with the automatic selection of salient phrases from the research paper [9]. In other words, its objective is to obtain a group of phrases that

are associated with the primary subjects mentioned in the given research paper [9]. Document keyphrases have empowered quick and exact looking for a given research paper from a substantial collected text, and have displayed their ability in enhancing numerous natural language processing (NLP) and information retrieval (IR) tasks, such as text summarization [19], text categorization [20], opinion mining [21], and document indexing [22].

Keyphrase extraction is a two-step process:

- Finding the relevant keywords and keyphrases which best describes the information present in the document [9].
- Scoring these keywords and keyphrases using supervised or unsupervised techniques [9].

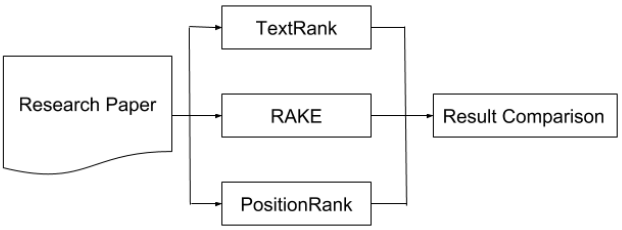

Fig. 2. System Architecture

A. Brute Force

Brute Force methodology considers all the words and phrases in the research paper as candidate words or phrases [9]. But all the words and phrases in the document might not describe the information in the document [9].

Key words and phrases are extracted by following a set of rules as follows:

- Removing stop words and punctuation's from the document [9].
- Extracting only noun, adjective, verb phrases and lower casing all the words [9].

When the document size is large, brute force method considers large number of phrases as candidate phrases. But most of those phrases does not convey the information described in the document. Therefore this method is not applicable for a lengthy document.

Unsupervised machine learning methods does not need training dataset. Graph-based ranking approach to extract keywords and keyphrases from research documents is a unsupervised technique in which the frequency of co-occurrence between words is considered to determine the phrase as a candidate phrase [9]. This approach ensures that the selected phrases describes the information present in the document [9].

The whole research paper is represented as a network whose nodes/ vertices are candidate key phrases (mainly only key words) and whose edges represents connection between candidates [9]. Then, a graph-based ranking algorithm, such as Googles PageRank, can be run over the network, and the highest-scoring terms are taken to be the documents keyphrases. We can instantiate Textrank algorithm in place of Pagerank [9].

B. TextRank

1) Graph Building.: TextRank is a graph based model which uses the theory behind the PageRank algorithm. It is used to extract the summary of a text. After preprocessing the text in the research paper, unique words will form vocabulary. Each word in the vocabulary acts as a vertex for the graph. The information of connection between the vertices is present in the weighted edge matrix. Weighted undirected edges graph is built. Information about the connections(edges) between all vertices will be present in the weighted edge matrix. Weighted edge[i][j] represents weight of the edge between the word vertex represented by vocabulary index i and the word vertex represented by vocabulary j. There will not be any edge between words represented by index i and j, if the weighted edge[i][j] value is zero. If the words are present within a window of particular window size, that means there is an edge between those words. If a connection is found between the same set of two words in different position in the text, the value of corresponding weighted edge[i][j] is increased by

$$\frac{1}{Distance\ between\ positions\ of\ words}$$

The score of all vertices are initialized to one and self-loops are not countable, their value is assigned to zero.

2) Summation of weights of edges connected to a vertex.: inout[i] represents the total number edges connected with that particular vertex represented by i.

3) Vertices score calculation.: The formula used for scoring a vertex represented by i is [17], [23]:

$$score[i] = (1-d) + d\,[Summation(j)((\frac{weighted\ edge[i][j]}{inout[j]})\ score[j])]$$

where j belongs to the list of vertices that has a connection with i. d is the damping factor. The score is iteratively updated until convergence.

4) Scoring Keyphrases.: Adding the score of individual word in a phrase will give the score of key phrases.

5) Sorting Keyphrases based on score.: Sorting the key phrases based on its scores will help to find most relevant key phrases of the document.

This methodology might not give relevant key phrases which best describes the relevant concepts described in the research document [9]. In research documents which is an example of structured document, there are specific locations (Title, Abstract, and Introduction) in which keyphrases appear mostly [9]. The concept of the position rank is to assign larger probabilities to the words which frequently occurred in early positions of the document [6].

C. RAKE

RAKE is an unsupervised technique for obtaining keywords from the research paper [24]. In Rapid Automatic Keyword Extraction (RAKE), keywords have multiple content words which are informative rather than punctuation and stop words [8]. RAKE can be relevant to any type of documents independent of domains especially those that don't pursue explicit language tradition [24]. The inputs parameters for RAKE are a list of stop words, phrase delimiters [24]. The words that don't seem to be in stopword list is taken into account as candidate keywords, however won't be the particular keywords [25].

The RAKE algorithm includes the following steps:

1) Candidate Selection.: It removes all stop words and phrase delimiters and then extracts all possible keyphrases [24].

2) Feature Calculation.: The feature value in RAKE is keyword score-matrix rather than TF-IDF scores used by most of the other frameworks. [24].

The distinctive estimates utilized for the construction of score-weight matrix:

- *Word frequency:* It shows the count of the occurrences of a word in the research paper [24].
- *Word degree:* The degree of a word represents how often it co-occurs with different words within the candidate keywords. [24].
- *Ratio of the degree to frequency:* The score-weight matrix consequently made is utilized in the selection of keywords from the research paper [24].

3) Keyword Selection.: The best-ranked words are considered as the keywords among the extracted candidate phrases [24].

In RAKE, co-occurrences of words within these candidate keywords are significant and enable us to recognize co-occurrence without the use of a self-assertively estimated sliding window [8]. But in TextRank, it considers fixed-size sliding window [8].

D. Position Rank

The phrases which frequently occurs at the beginning of the document are mostly found as keyphrases or candidate phrases [6]. We can notice that keyphrases occur mostly at the beginning of the document, even in the title of the document [6]. The concept of the position rank is to assign larger probabilities to the words which frequently occurred in early positions of the document [6].

The PositionRank algorithm includes the following steps:

1) Building Graph.: Consider document d for extracting keyphrases. An Undirected graph G =(V, E) is built, where every distinctive word that goes through parts-of-speech filtering represents vertex v_i belongs to V [6]. Two vertices v_i and v_j are connected by an edge (v_i,v_j) belongs to E, if these words co-occur within a window of w contiguous tokens in d [6]. The weight of an edge (w_{ij}) is calculated based on how often the two words co-occur in a window size of w consecutive tokens in d [6].
The score (s) for vertex v_i is obtained by recursively computing the equation [26]:

$$S(v_i) = \alpha.\ p(v_i)\ +\ (1-\ \alpha) \sum_{v_j \epsilon Adj(v_i)} \frac{w_{ji}}{\sum_{v_k \epsilon Adj(v_j)} w_{jk}} S(v_j)$$

Where is a damping factor, usually set to 0.85, and $p(v_i)$ is a weight assigned to vertex v_i [6]. The term $p(v_i)$ is added to ensure that the PageRank algorithm does not get stuck into cycles and can jump to another vertex in the graph with probability $p(v_i)$ [6]. In unbiased PageRank, each vertex has equal probability whereas, in biased PageRank (PositionRank), some vertices have higher probability than others [6].
For example, a word x occurring in the following positions: 3rd, 6th and 11th, has a

$$weight\ p(v_i) = \frac{1}{3}\ +\ \frac{1}{6}\ +\ \frac{1}{11} =\ 0.59$$

The word found in 60th position will have a lesser probability to be considered as the keyword when compared to the word found in 2nd position.

2) Forming Keyphrases: The sum of scores of contagious individual words gives the score of keyphrases [6]. PositionRank totals the information from all positions of word occurrences - whereas other methodologies utilize just the first position of a word alluded as PositionRank-f_p [6]. Considering the above example, a word present in the positions 2nd, 5th, and 10th has a weight of

$$\frac{1}{3}\ +\ \frac{1}{6}\ +\ \frac{1}{11} =\ 0.59$$

if it considers all the positions, and weight of

$$\frac{1}{3} =\ 0.333$$

if it considers the first position (f_p) model [6].

V. RESULT ANALYSIS

We experimented TextRank, RAKE and PositionRank algorithm on the research documents. TextRank implementation for research document with large size takes much time to process the document for determining keyphrases.

In research documents which is an example of a structured document, there are specific locations (Title, Abstract, and Introduction) in which keyphrases appear mostly [9]. PositionRank gives more weight to the words which occur in the initial positions of a document, as a result, it is giving relevant keyphrases which best describes the concepts covered in research document as output in less time. The keyphrases extracted by PositionRank are almost similar to that of author

input keyphrases when compared to the TextRank algorithm results which do not consider positional occurrences of word and RAKE algorithm results. In the RAKE algorithm, the highest scoring phrases which we got as an output are quite long, less relevant and the score of relevant phrases is low.

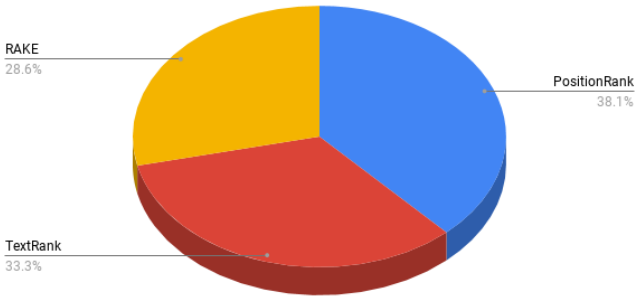

Fig. 3. *A Comparative Study on keyword extraction algorithms*

VI. CONCLUSION

Position Rank, an unsupervised graph-based algorithm, considers both the position of a word and its frequency in a document for keyphrase extraction. Our experiment on research documents proves that PositionRank gives slightly better results when compared to TextRank algorithm which does not consider positional occurrences of words in a document and RAKE algorithm. In RAKE algorithm, if we do not have a complete list of stop words the keyphrases which we get as an output will be quite long and less relevant.

ACKNOWLEDGMENT

We would like to express our gratitude to the Computer Science Department, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Amritapuri for providing support and guidance.

REFERENCES

[1] K Deepa Raj and GP Sajeev. A community based web summarization in near linear time. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 962–968. IEEE, 2018.

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

[3] Yutaka Matsuo and Mitsuru Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(01):157–169, 2004.

[4] Kazi Saidul Hasan and Vincent Ng. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. pages 365–373, 2010.

[5] Shailendra Singh Kathait, Shubhrita Tiwari, Anubha Varshney, and Ajit Sharma. Unsupervised key-phrase extraction using noun phrases. *International Journal of Computer Applications*, 162(1), 2017.

[6] 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)*, volume 1, pages 1105–1115, 2017.

[7] Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 552–559, 2007.

[8] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*, pages 1–20, 2010.

[9] 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)*, volume 1, pages 1262–1273, 2014.

[10] Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. Narrative text classification for automatic key phrase extraction in web document corpora. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pages 51–58. ACM, 2005.

[11] Mark Dredze, Hanna M Wallach, Danny Puller, and Fernando Pereira. Generating summary keywords for emails using topics. In *Proceedings of the 13th international conference on Intelligent user interfaces*, pages 199–206. ACM, 2008.

[12] Mari-Sanna Paukkeri, Ilari T Nieminen, Matti Pöllä, and Timo Honkela. A language-independent approach to keyphrase extraction and evaluation. *Coling 2008: Companion volume: Posters*, pages 83–86, 2008.

[13] Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th international conference on World wide web*, pages 661–670. ACM, 2009.

[14] MG Thushara, SA Sreeremya, and S Smitha. Kea-based document tagging for project recommendation and analysis. In *Recent Findings in Intelligent Computing Techniques*, pages 285–295. Springer, 2018.

[15] MG Thushara and N Dominic. A template based checking and automate tagging algorithm for project documents. *International Journal of Control Theory and Applications*, 9(10):4537–4544, 2016.

[16] Yan Ying, Tan Qingping, Xie Qinzheng, Zeng Ping, and Li Panpan. A graph-based approach of automatic keyphrase extraction. *Procedia Computer Science*, 107:248–255, 2017.

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

[18] Remya RK Menon, Deepthy Joseph, and MR Kaimal. Semantics-based topic inter-relationship extraction. *Journal of Intelligent & Fuzzy Systems*, 32(4):2941–2951, 2017.

[19] Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. World wide web site summarization. *Web Intelligence and Agent Systems: An International Journal*, 2(1):39–53, 2004.

[20] Anette Hulth and Beáta B Megyesi. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 537–544. Association for Computational Linguistics, 2006.

[21] Gábor Berend. Opinion expression mining by exploiting keyphrase extraction. 2011.

[22] Carl Gutwin, Gordon Paynter, Ian Witten, Craig Nevill-Manning, and Eibe Frank. Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, 27(1-2):81–104, 1999.

[23] Available at: https://github.com/jrc1995/textrank-keyword-extraction.

[24] Thushara MG, Krishnapriya MS, and Sangeetha S Nair. Domain classification and tagging of research papers using hybrid keyphrase extraction method. 06 2017.

[25] M. G. Thushara, M. S. Krishnapriya, and S. S. Nair. A model for auto-tagging of research papers based on keyphrase extraction methods. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1695–1700, Sep. 2017.

[26] Corina Florescu and Cornelia Caragea. A position-biased pagerank algorithm for keyphrase extraction. In *AAAI*, pages 4923–4924, 2017.