JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Machine Learning Based Keyword Extraction for Improved Text Analysis

Sarvesh Rane[1], Swati Savkare[2], Sumit Sapkal[3], Bhargav Rokade[4]

Department of E&TC, SKNCOE, SPPU, Pune

[1]sarveshrane2000@gmail.com, swati.savkare_skncoe@sinhgad.edu[2], [3]sumitsapkal111@gmail.com, [4]bhargavrokade1034@gmail.com

***Abstract*—The objective of this project is to extract the most important words and expressions by using keyword extraction. Keyword extraction is very useful as it helps us to quickly find out the relevant text from a large amount of data. It is used in the computer science field basically in information retrieval and can be used in text summarization, indexing etc. The input taken will be text-based information in the form of a text file. There are various steps to extract relevant keywords which include pre-processing of data, filtering, POS tagging the data, removal of stop words, then scoring and ranking the information (keywords\key phrases) obtained from aforementioned steps based on text ranking model. Different parameters for the evaluation could be considered such as precision, entropy etc. Further literature of keyword extraction is being done in languages other than English, for instance Hindi. In addition, applications related to keywords are also discussed in this project.**

***Keywords*— Keywords extraction, text summarization, pre-processing, filtering, POS tagging, stop words removal, scoring and ranking.**

I. INTRODUCTION

With the exponential growth of data across industries, it has become increasingly important to find ways to efficiently analyse and summarise large amounts of text-based information. In recent years, text summarization has emerged as a significant area of study, with researchers exploring various methods to automatically condense textual material without sacrificing important objectives. Keyword extraction, the process of identifying the most pertinent words and phrases from text input, is a key component of text summarization. This project focuses on demonstrating the use of keyword extraction to extract relevant keywords and keyphrases from social media sites, books, papers, journals, and other sources. By automating the process, this approach enables efficient search and retrieval of content on specific topics, keywords, and learning objectives, saving valuable time and resources. Through this project, we aim to showcase the importance and relevance of keyword extraction and provide insights into the latest research in this area.

II. LITERATURE SURVEY

In recent years, various techniques and strategies have been proposed to address this task. In this survey, we review five research papers that present different approaches for keyword extraction.

Mingxi Zhang et al.[1] conducted an empirical study on TextRank and tested the effectiveness of the method by changing parameter settings. The authors extracted the word stems using Porter Stemmer and evaluated the effectiveness by Precision, Recall, and Accuracy.

Slobodan Beliga[2] provided a comprehensive survey of techniques and strategies for keyword extraction. The publication systematizes methodologies and analyses previous research on supervised and unsupervised methods. The focus is on graph-based approaches.

Carretero-Campos et al.[3] proposed two methods for improving keyword detection in short texts. The first method is based on entropy, and the second method is based on clustering.

Marina Litvak et al.[4] proposed a language-independent method for automatically extracting keyphrases from text using graph theory. This approach models the relationships between objects in a mathematical framework.

Jo˜ao Ventura1, Joaquim Silva[5] combined two papers that proposed different methods for mining concepts and identifying important keywords. The first paper used natural language processing and machine learning techniques to extract concepts from the text. The second paper used linguistic and statistical methods to identify important keywords based on frequency, part-of-speech tagging, and semantic analysis.

In conclusion, the reviewed papers offer a wide range of approaches for keyword extraction, including graph-based methods, entropy, clustering, and statistical and linguistic methods. These approaches can be used to extract keywords and keyphrases from text, which are essential for many applications, such as document summarization, information retrieval, and topic modelling.

III. PROPOSED MODEL

This section of the chapter gives the description and overview of the diagram\flowchart(process) and along with reasons for each step\block.

Text Input: First the input will be taken in the form of text and the text can be from English or Hindi, as these two languages are supported by the model presented here. Then in the pre-processing step Text Cleaning and Normalization take place where non-printable characters are removed, and the text is converted to lowercase. This is done to ensure that the text is in a consistent and usable format.

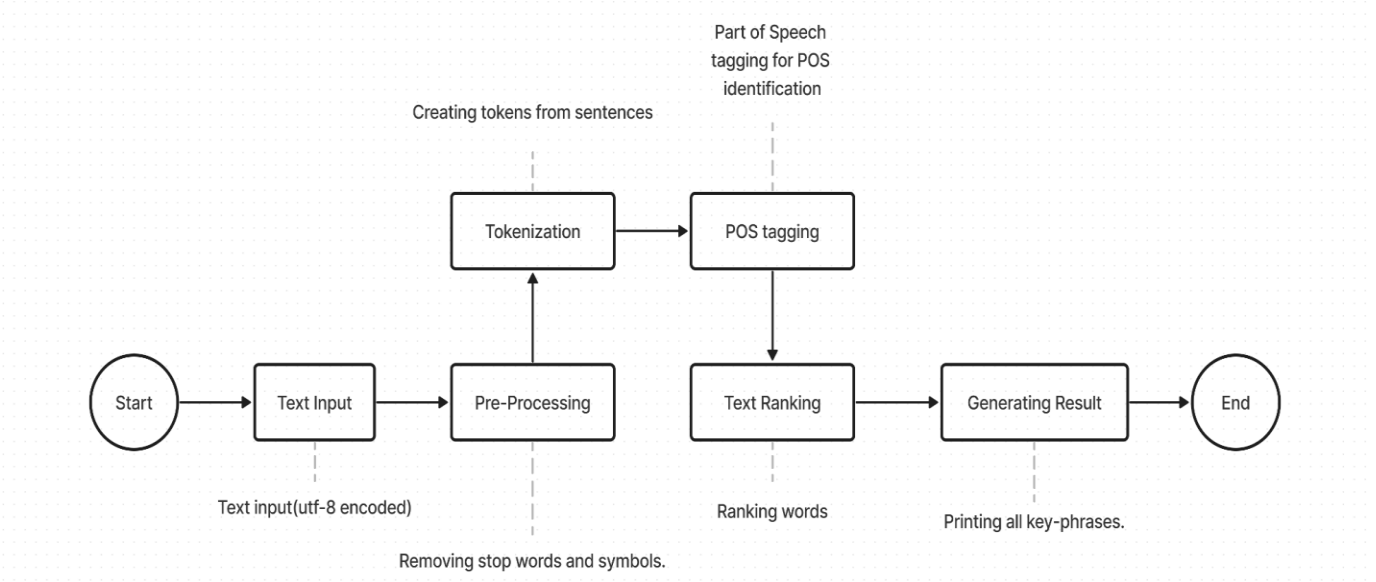

Fig. 1 Text Rank Model methodology for keyword extraction.

Tokenization: The text is split into individual words or tokens. This is necessary because TextRank works with individual words as vertices in a graph. And it further makes it easy to tag each of these words which will be further used for filtering purposes.

POS Tagging and Lemmatization: The words are tagged with their part-of-speech (POS) and lemmatized to reduce inflectional forms of words to a common base form.

Stop word Removal: Stop words, which are common words like "the" and "a" that do not add significant meaning to the text, are removed along with words which are not adjective, noun, or gerund. And the punctuations are also considered here as stop words.

Vocabulary Creation: A vocabulary is created by selecting unique words from the processed text. Each word in the vocabulary will be represented by a vertex in the graph.

Text Rank: A graph is built by connecting vertices (words) that co-occur within a specified window size. The weight of the edge between vertices is determined by the frequency of their co-occurrence. Then, the score of each vertex is initialized to one and then iteratively updated based on the scores of its neighbouring vertices. Phrase Partitioning: The text is partitioned into phrases using the stop words as delimiters. This is done to create keyphrase candidates that consist of multiple words.

Single-word keyphrase candidates that are part of multi-word keyphrase candidates are removed.

Generating Result: Each keyphrase candidate is scored by adding up the scores of its constituent words. Like for phrase cutoff threshold, the score will summation of cutoff score and threshold score. The keyphrases are ranked based on their scores, and the top keywords are selected for output.

IV.IMPLEMENTATION

The implementation consists of roughly 16 steps, which are further discussed below and comprises various methods and processes to obtain the desired output of relevant keywords.

Step 1: Take input that is fed in the form of text of a document file in a text format. So the input was taken from various sources like articles or scientific documents in both the English and Hindi language.

Step 2: Data pre-processing, which includes data cleaning and date transformation, where non-printable characters are employed to design information. The most frequent non-printable characters are eliminated, and the text is then changed to lowercase before being tokenized using the word_tokenize function of the nltk.tokenize module, which is a part of the nltk library.

Step 3: Using the pos_tag() function from nltk, the input text is then POS tagged so that the words can be lemmatized based on their POS tags (like JJ for adjective, SYM for symbols, and RB for adverb) from the Penn Treebank.

Step 4: The tokenized text is lemmatized using the WordNetLemmatizer() function from the nltk.stem package. Documents will often use variants of a term, taking the word "glass" instead of, say, "glasses." Lemmatization's fundamental purpose is to reduce

a word's inflectional forms and occasionally derivationally related forms to a single, basic form. Step 5: The lemmatized text is once more POS-tagged.
Step 6: Filtering will be done using POS data at this stage. This filtering is carried out by treating every word from the lemmatized text that isn't a noun, adjective, or gerund as a stop word. Punctuation marks will also be included on the stop word list.
Step 7: Here an external file with a list of stop words is added to the previous stop words to create a final list which contains the sum total of all stop words and potential phrase delimiters.
Step 8: Using the final list, remove stop words from the lemmatized text, and create a vocabulary from the processed text that solely comprises unique words.
Step 9: Every word in the vocabulary will act as a graph vertex. By their index in the vocabulary list, the words will be represented in the vertices. The word vertex represented by vocabulary index i and the word vertex represented by vocabulary are connected by an edge that is weighted. There is no edge connection between the words represented by index and j if weighted edge[i][j] is equal to zero. If two words co-occur in a window with a given window size' in the processed_text, then there is a connection between those words (and the words that represent them, i and j).
Step 10: The weighted sum of a vertex's connections is determined.
Step 11: Scoring of vertices(words) is determined by using a formula:
score[i] = (1-d)+d*[□(j)((weighted_edge[i][j]/inout[j])*score(j))], where j belongs to the list of vertices that has a connection with i, d is a damping factor and the score is iteratively updated until convergence.
Step 12: In addition to the keywords, the phrases are potential candidates for keyphrase extraction. Lemmatized material is therefore divided into phrases using stop words as delimiters.
Step 13: Next, a list of original phrases is generated because the repeating phrases and keyphrases candidates are irrelevant in this situation.
Step 14: To reduce the number of potential keyphrases, the single-word candidates that are present in multi-word alternatives are dropped.
Step 15: Score the phrases (candidate keyphrases) and create a list of keyphrases and keywords by listing the phrases that aren't tokenized.
Step 16: The final step involves ranking the keyphrases according to their scores, after which the top keyphrases are shown.

V. RESULT ANALYSIS

```
input_text = """
This paper discusses the comprehensive performance profiling, improvement and benchmarking of a
Computational Fluid Dynamics code, one of the Grand Challenge applications, on three popular multiprocessors.
In the process of analyzing performance we considered language, compiler, architecture, and algorithmic changes and quantified
each of them and their incremental contribution to bottom-line performance.
We demonstrate that parallelization alone cannot result in significant gains if the granularity of parallel threads
and the effect of parallelization on data locality are not taken into account.
Unlike benchmarking studies that often focus on the performance or effectiveness of parallelizing compilers
on specific loop kernels, we used the entire CFD code to measure the global effectiveness of
compilers and parallel architectures. We probed the performance bottlenecks in each case
and derived solutions which eliminate or neutralize the performance inhibiting factors.
The major conclusion of our work is that overall performance is extremely sensitive to the
synergetic effects of compiler optimizations, algorithmic and code tuning, and architectural idiosyncrasies.
"""
```

Fig. 2 Input taken from Krapivin2009 dataset.[1]

```
Keywords: (35)

performance inhibiting factor
comprehensive performance profiling
analyzing performance
bottom-line performance
performance bottleneck
computational fluid dynamic code
entire cfd code
compiler optimization
parallelizing compiler
code tuning
```

Fig. 3 Extracted key phrases (Output)

In this chapter we will discuss results obtained from the aforementioned process and by analysing, comparing the input and the output, along with errors or exceptions, advantages and disadvantages that occurred during the process. In summary, the TextRank algorithm is an effective method for keyword extraction from text. The performance of the algorithm can vary depending on factors such as the quality and size of the dataset, the specific parameters and settings used, and the complexity of the text being analysed. The study evaluated the TextRank algorithm's performance against other state-of-the-art keyword extraction methods and found that it achieved moderate to good performance for keyword extraction. The accuracy was determined using the predefined accuracy function and the "golden standard words" defined in the nltk toolkit. The input text was collected from Hulth2003 and Krapivin2009[1], and the system was able to recognise and rank significant keyphrases that conveyed its core. The output of [1] and the outcome from our model were similar and in accord. Window size, w = 3, damping factor, d = 0.85, and maximum iterations, 50, are important parameters in an empirical study of TextRank for keyword extraction. However, the algorithm has its limitations and can have trouble recognizing keyphrases that contain numerous words but do not co-occur within the designated window size. Overall, the TextRank algorithm is a useful tool for keyword extraction, but it is important to carefully consider the specific task and dataset when using it.

VI. CONCLUSIONS

Text mining is the area that is now advancing the quickest. Its significance will grow over time as more data becomes available online as a result of digitization, necessitating a way to locate the most crucial and pertinent data. The project demonstrates how information can be gleaned from text that is useful. This study offers important data that might be used to generate future material summaries. The method can be applied to text data including customer reviews, poll results, social media comments, emails, and chat conversations. In this study, we have evaluated the performance of two models - an English model and a Hindi model. Our findings reveal that the English model performs with an accuracy of approximately 95%, while the Hindi model lags behind with an accuracy of about 72%. This disparity in accuracy is primarily attributable to the variation in training data and the prevalence of Unknown POS (Parts of Speech) tags observed in the Hindi language.

REFERENCES

[1] S. M[1] Mingxi Zhang, Xuemin Li, Shuibo Yue, and Liu Qian Yang. (2020). An Empirical Study of TextRank for Keyword Extraction.
[2] Beliga, S. (2014). Keyword extraction: a review of methods and approaches. University of Rijeka, Department of Informatics, Rijeka.
[3] Carretero-Campos, C., Bernaola-Galván, P.,Coronado, A. V., & Carpena, P. (2013). Improving statistical keyword detection in short texts: Entropic and clustering approaches. Physica A: Statistical Mechanics and its Applications, 392(6), 1481-1492.
[4] M. Litvak, M. Last, H. Aizenman, I. Gobits, A. Kandel, “DegExt – A Language-Independent Graph-Based keyphrase Extractor” in Proc. of the 7th AWIC 2011, pp. 121-130, Switzerland, 2011. M. Litvak, M. Last, H. Aizenman, I. Gobits, A. Kandel, “DegExt – A Language-Independent Graph-Based Keyphrase Extractor” in Proc. of the 7th AWIC 2011, pp. 121-130, Switzerland, 2011.
[5] Ventura, J., & Silva, J. (2012). Mining concepts from texts. Procedia Computer Science, 9, 27-36. 17.Hong, B., & Zhen, D. (2012). An extended keyword extraction method. Physics Procedia, 24, 1120-1127.
[6] P. D. Turney, ``Learning to extract keyphrases from text," CoRR,vol. cs.LG/0212013, pp. 12, Oct. 2002.
[7] C.-C. Huang, M. Eskenazi, J. Carbonell, L.-W. Ku, and P.-C. Yang,``Cross-lingual information to the rescue in keyword extraction," in Proc.52nd Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations,Baltimore, MD, USA, 2014, pp. 16.
[8] G. Salton and C. Buckley, ``Term-weighting approaches in automatic text retrieval," Inf. Process. Manage., vol. 24, no. 5, pp. 513523, Jan. 1988.
[9] Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing Order into Text. Conference on Empirical Methods in Natural Language Processing.