DOCUMENT SUMMARIZATION

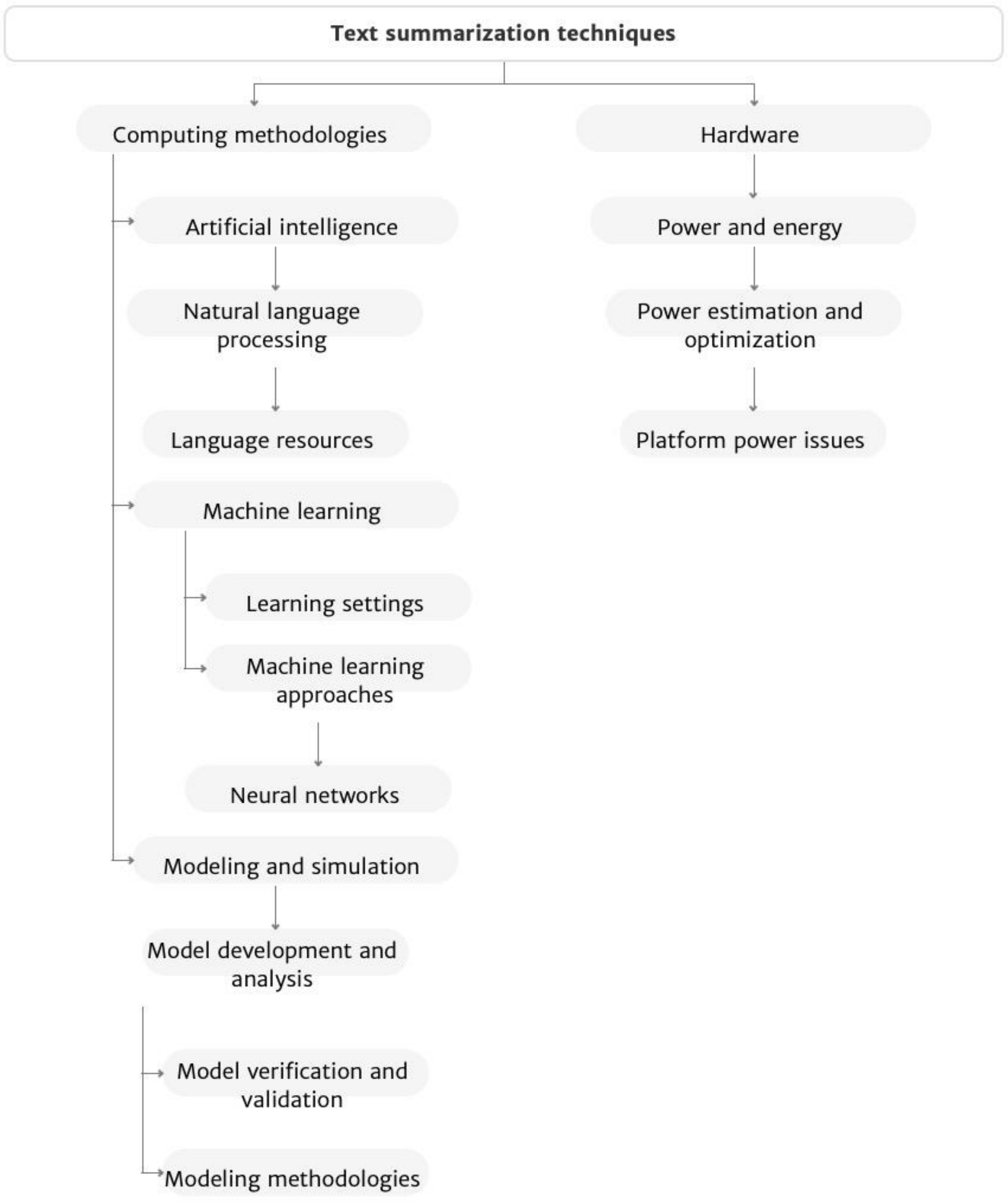
-17CE073,78,82,83

1. Abstract:

In this new age, when great information is available on the web, it is of utmost importance to provide an improved method of extracting information quickly and efficiently. An overview of manuscripts is very difficult for people to retrieve by hand. More text content is available on the web. So, there is the pull of finding the right documents from the number of documents available, and extracting relevant information from them. Automatic text summaries are very important to unlock the above two causes. The interpretation of a text is that the process of explaining and compressing important information relevant to the time of the compilation or collection of documents related to the short version that retains all its meanings.

2. Introduction:

It is necessary to understand the summary before we can get to the summarization of the text. A summary can be called a summary of a major role that provides precise information about what a section means in a redesigned design. The main goal after receiving a brief summary i.e. an abbreviated text to obtain a short version of a section that protects its original semantic composition. Summarizing reduces reading time. The summarisation is further divided into two types of summaries viz. Abstract guess and overview. An abbreviated form of summary can be summarized as taking important sentences or paragraphs from the original text and combining them in a consistent way. Surprisingly summarizing can be seen as taking the concept of text and expressing it in natural language.
Text summaries can be divided into two groups: Display and information. The visual summary only reflects the dominant view of the text to the user. This type of summary provides twice as much as 5 to 10 percent of the best text. While, Informative information summary programs provide a brief summary of 20 to 30 percent of the text.


Text summarization techniques
Computing methodologies
Hardware
Artificial intelligence
Natural language processing
Language resources
Machine learning
Learning settings
Machine learning approaches
Neural networks
Modeling and simulation
Model development and analysis
Model verification and validation
Modeling methodologies
Power and energy
Power estimation and optimization
Platform power issues

3. Types of Summarizations

In general, there are two sorts of summarization, abstractive and extractive summarization.

3.1. Abstractive summarization

Abstractive decoding removes words allowed by the semantics of the original text and uses certain words not included in the original text. This method scans the text and translates it using some mysterious language techniques to find text that does not maintain the original meaning of the given text. It can be saved manually to handle how a hu-man might read the text and make sense of the text.

Text to be summarized → Content insight → to maintain strong semantics → create summaries.

3.2. Extractive summarization

Added methods create a subset from the first word and keep the original information intact. The direct method gives weight to the value of the sentences. The algorithm is then used to extract a summary based on the registers and the levels are given based on this weighting value.
Document to summarize → calculate the similarity of sentences → give weight to the sentences → select the top sentences

• Surprising summarization requires a deeper understanding of the text compared to the additional summary leading to limited reading.

• It is often seen that better results are produced with higher summaries compared to default cold summers because abstract summary methods deal with problems such as logical language representations, complex logic and language generation compared to abbreviated summary methods such as sentence output.

• We use an unsupervised method of learning to produce incremental summaries that find sentences to match and put them.

• The similarity between the two non-zero coordinates in the inner product space that gives the cosine of the angle between them can also be the cosine analogy. The symbolism of these sentences is in the form of a carrier group. This will make the search for matches between sentences much easier. Its dimensions are the cosine of the angle between the vectors. Angle will be 0 if the sentences are the same

4. Literature review

Summary programs need to produce a concise and specific summary to send important information to the input. In this chapter we compile our methods for extracting summarized, short-lived summaries and explain how these systems generate summaries. These summaries identify the most important sentences in the input, which can be a single document or a group of related documents, and then combine them to form a summary. The decision of which content is important is largely driven by the summary entries. The option to focus on emerging techniques leaves a huge body of text-to-text methods designed to be tedious, but allows us to focus on some of the best practices that can easily be converted to user account information and working on single and multiple documents. In addition, by looking at the stages in summative extractive operations we are able to show financial consistency and divergence in the summarized methods related to the critical frameworks of the system and which may explain the advantages of some strategies over others. In order to better understand the functionality of summary systems and to emphasize the development of compiler choice they need to do, we distinguish three independent functions performed by almost all data summaries: creating an intermediate keyword presentation; Intermediate Presentation Even the simplest programs get a specific text presentation that should summarize and identify content that is not related to this proposal. Repeated header headings convert the text to the middle format translated as the topic / topics discussed in the text. Some of the most popular summary methods rely on subject representation and this range of methods shows impressive flexibility in the power of complexity and representational power. They include the frequency, the TF.IDF and the title of the methods in which the header return contains a simple table

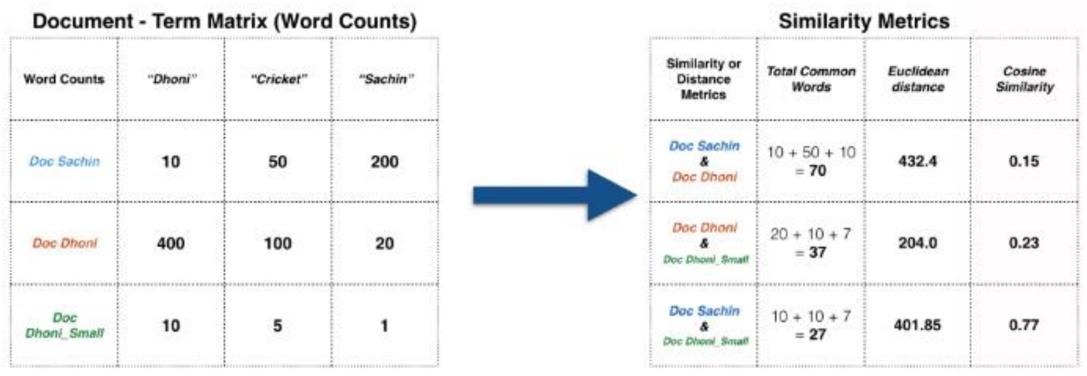
5. Cosine Similarity

The texts are evaluated to be different no matter how large their cosine size is computed. In terms of volume, it measures the cosine of the angle between the two vectors arising in the space of mass [2]. If two identical documents are far from the Euclidean range (due to the size of the document), then it is possible that they may be very close to each other this could be a similarity of cosine similarity. The smaller angle provides higher cosine similarity.

When plotted in an area of multiple dimensions, where a particular size agrees with the word in the text, the cosine similarity entails the orientation (angle) of the text and not the dimension. If you want size, enter the Euclidean range instead.

The cosine analogy is beneficial because even if the two similar texts are very close to the Euclidean range due to size (e.g., the word 'cricket' has appeared 50 times in one text and 10 times in another) they still have a slight angle between them. The smaller the angle, the higher the similarity.

Soft cosines can be a great option if you want to use the exact same metric that can help compile.



Document - Term Matrix (Word Counts)

Word Counts	"Dhoni"	"Cricket"	"Sachin"
Doc Sachin	10	50	200
Doc Dhoni	400	100	20
Doc Dhoni_Small	10	5	1

Similarity Metrics

Similarity or Distance Metrics	Total Common Words	Euclidean distance	Cosine Similarity
Doc Sachin & Doc Dhoni	10 + 50 + 10 = 70	432.4	0.15
Doc Dhoni & Doc Dhoni_Small	20 + 10 + 7 = 37	204.0	0.23
Doc Sachin & Doc Dhoni_Small	10 + 10 + 7 = 27	401.85	0.77

6. TextRank

• TextRank is a general purpose, algorithm based on NLP graph.

• TextRank is an automated comparison process.

• Graph-based algorithms are a way of determining the value of a vertex within a graph, based on the global information that is clearly found throughout the graph.

6.1. TextRank Model -

The basic idea used by the graph-based model is that of voting or recommendation.
When one vertex connects to another, it's basically a vote on that vertex. The higher the number of votes used for a vertex, the higher the sound of that vertex.

6.2. Text as a graph -

We need to create a graph that represents the text, linking specific words or structures of technology with logical relationships.
TextRank includes two NLP-

• Keyword removal function

• Sentencing word

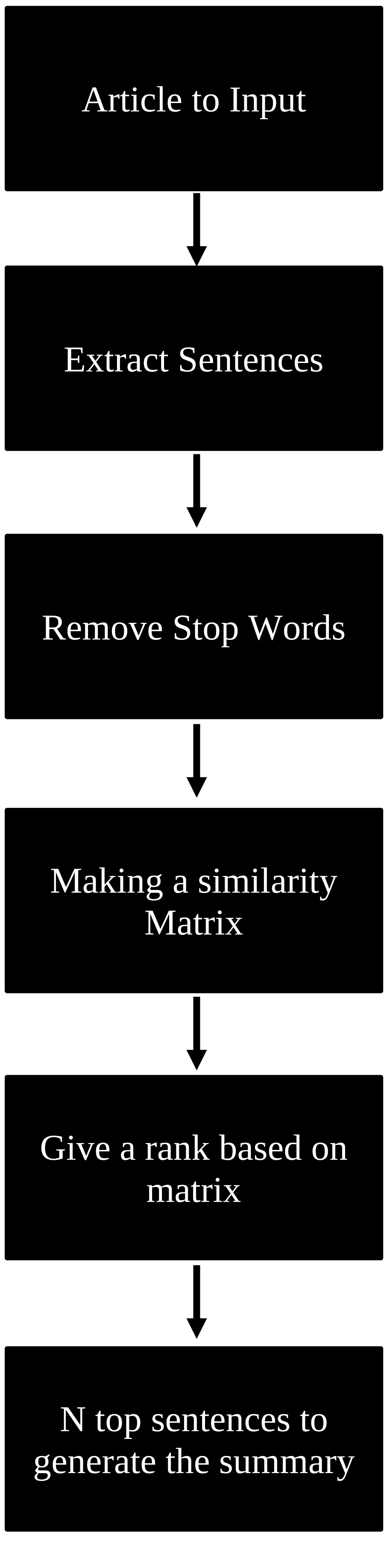
6.3. Keyword picker -

The keyword extraction task is to automatically identify in a text a set of words that best describes a document.
The easiest way is to use the frequency monitoring method.
However, this leads to serious consequences.
The TextRank keyword domain algorithm is not fully monitored. It is not necessary to install the train.

6.4. Release Domain -

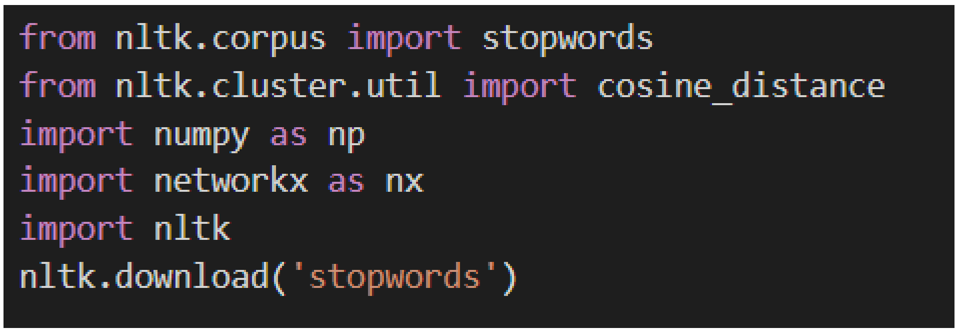
TextRank is well suited for applications that include all sentences, since it allows the position of the text over a computer script that can be re-calculated based on information based on the entire text.

7. Flow of System to generate a summary of given text


Article to Input
Extract Sentences
Remove Stop Words
Making a similarity Matrix
Give a rank based on matrix
N top sentences to generate the summary

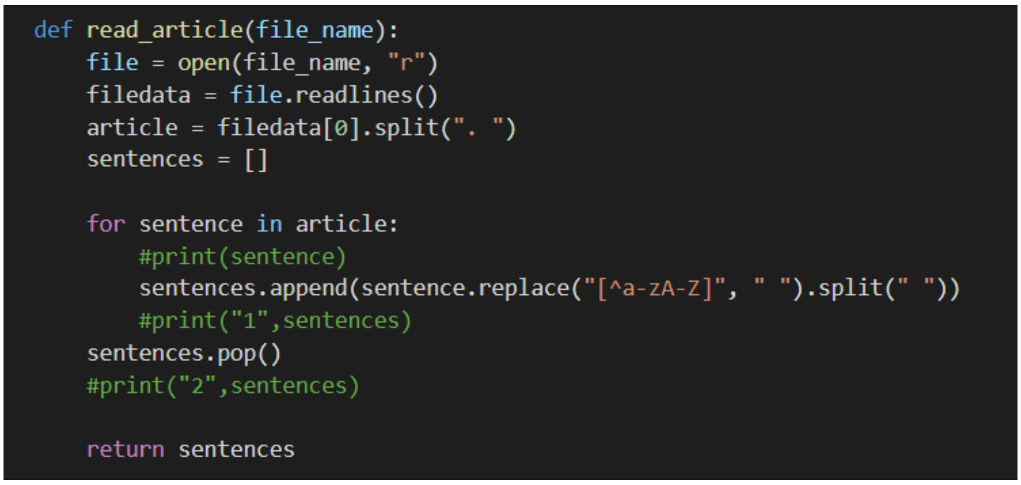
8. Code constitution of the system

1. Libraries to be imported



```python
from nltk.corpus import stopwords
from nltk.cluster.util import cosine_distance
import numpy as np
import networkx as nx
import nltk
nltk.download('stopwords')
```

2. Clean sentence generation



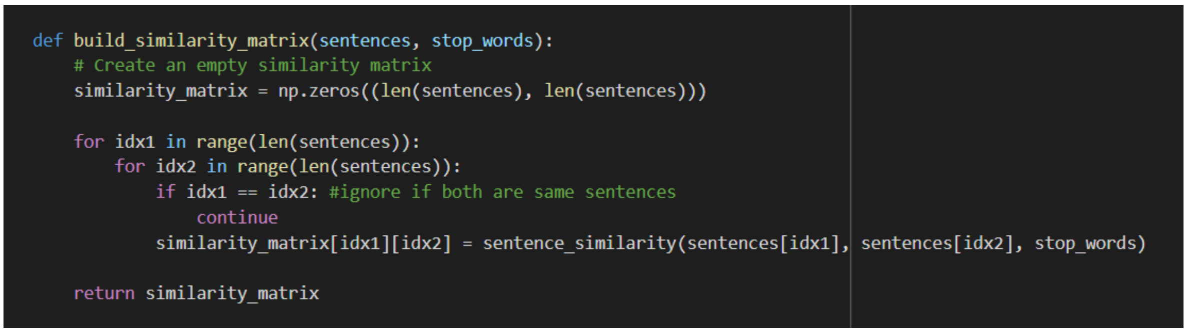
```python
def read_article(file_name):
    file = open(file_name, "r")
    filedata = file.readlines()
    article = filedata[0].split(". ")
    sentences = []

    for sentence in article:
        #print(sentence)
        sentences.append(sentence.replace("[^a-zA-Z]", " ").split(" "))
        #print("1",sentences)
    sentences.pop()
    #print("2",sentences)

    return sentences
```

3. Matrix of similarity

Here the use of cosine similarity gives the similarity between sentences.



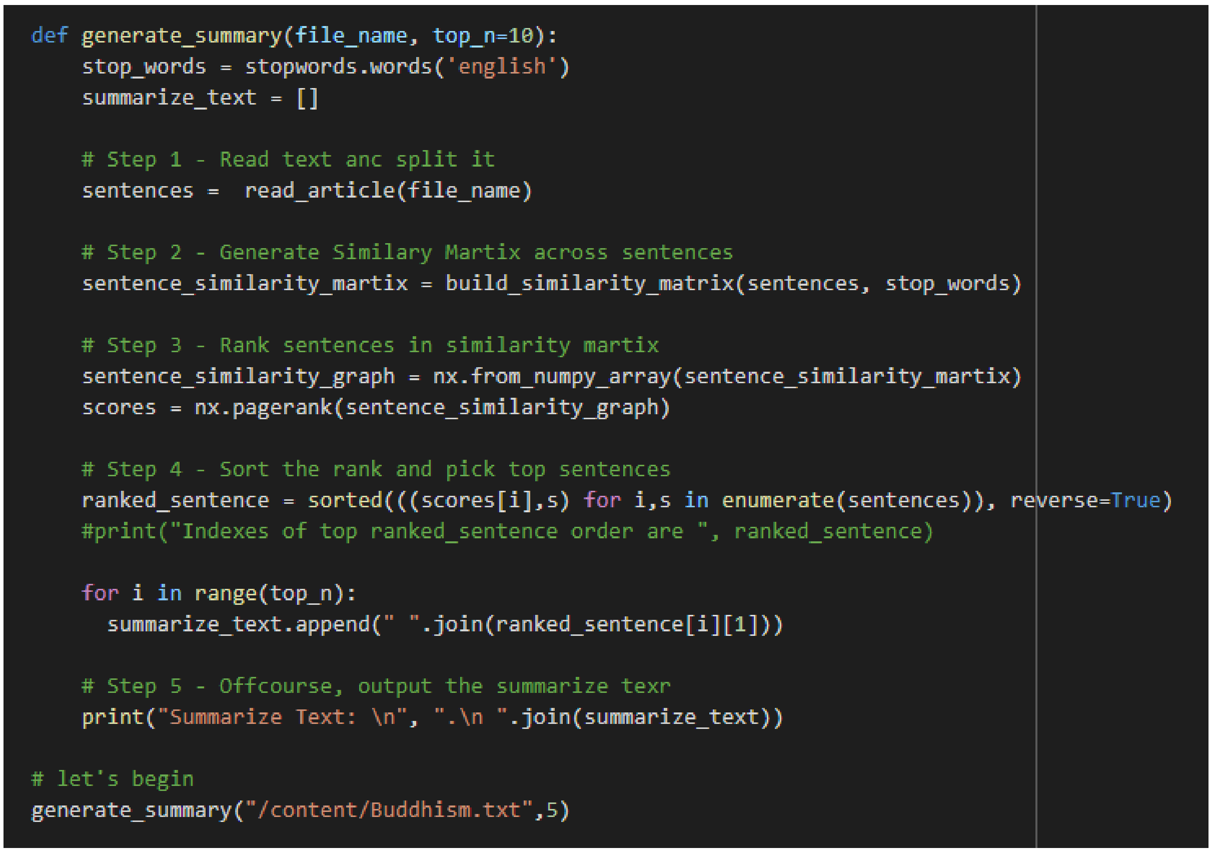
```python
def build_similarity_matrix(sentences, stop_words):
    # Create an empty similarity matrix
    similarity_matrix = np.zeros((len(sentences), len(sentences)))

    for idx1 in range(len(sentences)):
        for idx2 in range(len(sentences)):
            if idx1 == idx2: #ignore if both are same sentences
                continue
            similarity_matrix[idx1][idx2] = sentence_similarity(sentences[idx1], sentences[idx2], stop_words)

    return similarity_matrix
```

4. Summary generation method

This method calls all other functions in order to run the summarization pipeline going smoothly.



```python
def generate_summary(file_name, top_n=10):
    stop_words = stopwords.words('english')
    summarize_text = []

    # Step 1 - Read text anc split it
    sentences =  read_article(file_name)

    # Step 2 - Generate Similary Martix across sentences
    sentence_similarity_martix = build_similarity_matrix(sentences, stop_words)

    # Step 3 - Rank sentences in similarity martix
    sentence_similarity_graph = nx.from_numpy_array(sentence_similarity_martix)
    scores = nx.pagerank(sentence_similarity_graph)

    # Step 4 - Sort the rank and pick top sentences
    ranked_sentence = sorted(((scores[i],s) for i,s in enumerate(sentences)), reverse=True)
    #print("Indexes of top ranked_sentence order are ", ranked_sentence)

    for i in range(top_n):
      summarize_text.append(" ".join(ranked_sentence[i][1]))

    # Step 5 - Offcourse, output the summarize texr
    print("Summarize Text: \n", ".\n ".join(summarize_text))

# let's begin
generate_summary("/content/Buddhism.txt",5)
```

9. Conclusion

Thus, this document proposes and demonstrates the work put down by us in creating a Document summarization system. The system has been implemented by us using the concepts of Natural Language processing, Cosine similarity and TextRank model. The system developed by us aims at achieving high accuracy in achieving a summary of any given data set by [i] Taking the document and tokenizing it into sentences[ii]Creating the tokenized sentences to tokenized words and removing the stop words [iii] Using the textRank to rank the sentences.[iv] Developing a similarity matrix based on cosine similarity[v]Generating an accurate summary of the given passage in a limited but precise manner and accurate to the corresponding text. Thus, the document summarisation system developed by us gives satisfactory results in the summarization and can be used at many places like getting headlines of the news articles and making the person know if it is important or not. Thus, it proves a significant development

10. References

1. AMINI, M. R.AND GALLINARI, P. 2002. The use of unlabeled data to improve supervised learning fortextsummarization. In Proceedingsofthe25th Annual International ACMSIGIR Conference on Research and Development in Information Retrieval(SIGIR). 105–112.
2. BAEZA-YATES, R.AND RIBEIRO-NETO, B. 1999. Modern Information Retrival. ACM Press/Addison Wesley.
3. BALABANOVI´ C,M.ANDSHOHAM,Y. 1997. Fab:content-based,collaborativerecommendation.Comm. ACM 40, 3, 66–72.
4. BARKER, K.AND CORNACCHIA, N. 2000. Using nounphrase heads to extract document keyphrases. InProceedingsofthe13thBiennialConferenceoftheCanadianSocietyonComputationalStudies of Intelligence: Advances in Artificial Intelligence. 40–52.
5. BARZILAY, R.AND ELHADAD, M. 1997. Using lexical chains for text summarization. In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization. 10–17.

6. BERGER, A.AND MITTAL, V. 2000. OCELOT: A system for summarizing Web Pages. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and development in Information Retrieval (SIGIR). 144–151.
7. B¨OHM, C.AND BERCHTOLD, S. 2001. Searching in high-dimensional spaces-index structures for improving the performance of multimedia databases. ACM Comput. Surv. 33, 3, 322–373.
8. CARBONELL, J.AND GOLDSTEIN, J. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR). 335–336.
9. CARENINI, G., N G, R. T.,AND ZHOU, X. 2007. Summarizing email conversations with clue words. In Proceedings of the 16th International Conference on World Wide Web. 91–100.
10. CONROY, J. M.AND O'LEARY, D. P. 2001. Text summarization via Hidden Markov Models. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR). 406–407.
11. DAUM´ E, H.AND MARCU, D. 2006. Bayesian query-focused summarization. In Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL). 305–312.
12. EDMUNDSON, H. P. 1969. New methods in automatic abstracting. J. ACM 16, 2, 264–285. ERKAN, G.AND RADEV, D. R. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. J. Artif. Intell. Res. 22, 457–479.
13. FRANK, E., PAYNTER, G. W., WITTEN, I. H., GUTWIN, C.,AND NEVILL-MANNING, C. G. 1999. Domainspecific keyphrase extraction. In Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI). 668–673.
14. GONG, Y. H.AND LIU, X. 2001. Generic text summarization using Relevance Measure and Latent Semantic Analysis. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR). 19–25.