IJSAR International Journal of Sciences & Applied Research

International Journal of Sciences & Applied Research

www.ijsar.in

An Extractive Approach for English Text Summarization

Kanchan D. Patil [1], Sandip A. Patil [2], Yogesh S. Deshmukh[1]*

[1]Department of Information Technology, Sanjivani College of Engineering, Kopargaon, India.
[2]Department of Computer Engineering, Sanjivani College of Engineering, Kopargaon, India.

Corresponding author: *Yogesh S. Deshmukh, Department of Information Technology, Sanjivani College of Engineering, Kopargaon, India.

Abstract

Natural-language processing (NLP) is a vast area of computer science, artificial intelligence concerned with the interactions between computers and human languages. The “natural language” means a language that is used for daily communication by humans. The development of NLP applications is challenging because computers traditionally require humans to "speak" to them in a programming language that is precise, unambiguous and highly structure. Text summarization is one of the research area of NLP which gives us meaningful and short description of the vast text documents using different NLP tools and techniques. Nowadays, as we are dealing with huge amount of Digital data it is necessary to have automatic Text Summarization Techniques. Text summarization specifically classified into two major categories as Extractive Text Summarization and Abstractive Text Summarization. This paper focuses on different Extractive Text Summarization Techniques used for Indian Languages.

Keywords: NLP, Text Summerization; Extractive Text Summerization, Abstractive Text Summerization

Introduction

Nowadays we are dealing with the large amount of Digital data on the Internet. If you want to search some information on search engine for example ‘Text’, that web gives you large documents of information that consists of your search word ‘Text’. The information may be relevant to your search are may not be. As well the contents may be duplicate. It is difficult for human to read all the documents completely [1, 2, 3]. So, we require the automatic Text summarization. According to Radef et al. [4] a summary is defined as “a text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually, significantly less than that”. Automatic summarization can be defined as a process of shortening a text document with software, in order to create a summary with the major points of the original document. These Automatic summarization tools and Techniques help human to read and understand the document in short time. Automatic Text summarization is used in various applications such as Search Engine, Articles, Newspapers, Research Abstract etc[5]. Research in the area of Text summarization started in 1950’s but till now no system is available that summarizes the text like human. Some are focusing on

Abstractive Text Summarization and some on Extractive Summarization. Extractive methods work by selecting a subset of existing words, phrases, or sentences in the original text to form the summary. Oppositely in Abstractive Text Summarization, system understands the contents of document and then creates a summery in its own words [3]. As this technique tries to give the generalized summary like human, it needs advanced Natural Language Processing Techniques.

Abstractive text summarization method generates a sentence from a semantic representation and then uses natural language generation technique to create a summary that is closer to what a human might generate. There are summaries containing word sequences that are not present in the original

(Steinberger & Ježek, 2008). It consists of understanding the original text and re-telling it in fewer words. It uses the linguistic approach such as lexical chain, word net, graph theory, and clustering to understands the original text and generate the summary. On the other hand, Extractive text summarization works by selecting a subset of existing words, phrases or sentences from the original text to form summary. Moreover, it is mainly concerned with what the summary content should be. It usually relies on the extraction of sentences (Das & Martins, 2007). This type of summarization uses the statistical approach like title method, location method, Term Frequency-Inverse Document Frequency (TF-IDF) method, and word method for selecting important sentences or keyword from document (Munot & Govilkar, 2014).[16]

Text summarization techniques

Abstractive Text Summarization

In Abstractive Text Summarization systems generate new phrases, possibly rephrasing or using words that were not in the original text. Naturally abstractive approaches are harder. For perfect abstractive summary, the model has to first truly understand the document and then try to express that understanding in short possibly using new words and phrases. It is quite harder than extractive approach. It has complex capabilities like generalization, paraphrasing and incorporating real-world knowledge.

The basic form subject-verb-object of the sentences is considered for abstractive summarization method. Some steps of this method are given below:

Step I: Preprocessing: It is considered for each sentence to create a semantic graph. The actions performed in this step are sentence segmentation, stop word removal and stemming.

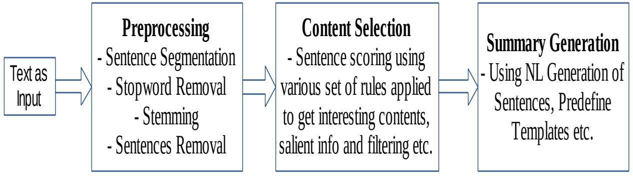
Text as Input
Preprocessing
- Sentence Segmentation
- Stopword Removal
- Stemming
- Sentences Removal
Content Selection
- Sentence scoring using various set of rules applied to get interesting contents, salient info and filtering etc.
Summary Generation
- Using NL Generation of Sentences, Predefine Templates etc.

Figure 1. Generic Abstractive Text Summarization

Demerits:

The biggest challenge for abstractive summary is the representation problem. Systems' capabilities are constrained by the richness of their representations and their ability to generate such structures, systems cannot summarize what their representations cannot capture [9,12].

Extractive Text Summarizaton

The basic process flow of Extractive Text Summarization is given as in figure 1

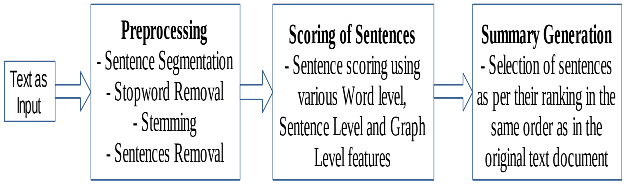
Text as Input
Preprocessing
- Sentence Segmentation
- Stopword Removal
- Stemming
- Sentences Removal
Scoring of Sentences
- Sentence scoring using various Word level, Sentence Level and Graph Level features
Summary Generation
- Selection of sentences as per their ranking in the same order as in the original text document

Figure 2. Generic Extractive Text Summarization

Step I: Preprocessing

In this step, sentences are segmented using some appropriate method. Generally, symbols like '.' , '?', and '!' are used to show the sentence end. Stop words like a, an, at are removed as they do not convey relevant information to the actual topic of summarization. After that stemming is performed. Stemming is the process of reducing derived words to their word stem, base or root form. For example, A English stemmer should reduce the words 'singing', 'sang' to the root word 'sing'. After that the sentences containing unnecessary information for summary like 'diagrams', 'Tables' are removed.

Step II: Scoring of Sentences

After preprocessing, the sentences in the original document are scored using different word level, Sentence Level and Graph level features.

Step III: Summary Generation

The sentences after scoring are selected in the same order as they appear in the original document in this step.

Demerits:

1. Extracted sentences usually tend to be longer than average. Due to this, parts of the segments that are not essential for summary also get included, consuming space.
2. Important or relevant information is usually spread across sentences, and extractive summaries cannot capture this (unless the summary is long enough to hold all those sentences).
3. Conflicting information may not be presented accurately [9, 12, 13].

Extractive text summarization methods

Feature Priority Based Sentence Filtering Method [6]

In this method, The features such as Term Frequency-Inverse Sentence Frequency[TF-ISF], Named Entity presence, Proper Noun Presence are used to select the sentences. The Inverse sentence frequency is variation of inverse document frequency[IDF]. ISF suggest that if the term is less frequent in whole document then it is more important for the sentence. Second feature is Named Entity Presence. Stanford NER is used to identify all the named entities present in the sentences. Stanford NER is a implementation of a Named Entity Recognizer. Named Entity Recognition (NER) labels sequences of words in a text which are the names of things, such as person and company names, or gene and protein names. The sentences are scored based on named entities present in them. Lastly, POS tagger is used to identify Proper Noun Feature. A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads text in some language and assigns parts of speech to each word such as noun, verb, adjective, etc., Sentences which contain proper nouns are the most important and convey maximum information. In this method, The researcher started the summary from the first sentence of the original document and ended with the last sentence of the document. These sentences improve the readability as well they are the important sentences according to the sentence location feature. And the score of intermediate sentences calculated using the following Feature Priority Filtering Algorithm. Although it is very difficult to find an efficient extractive summary using different feature combination, This method works effectively as takes the advantage of sentence location feature.

Algorithm: Feature Priority Filtering Algorithm

1. Compute TF-ISF score for each term.
2. Calculate TF-ISF score of sentences on the basis of terms present in the sentence.
3. Select top 50% sentences on the basis of TF-ISF score
4. Apply named entity recognizer on selected sentences.
5. Select top 50% sentences on the basis of named entity presence.
6. Apply POS tagging on selected sentences.

7. Put the sentences in a list L in decreasing order of their score using proper nouns.
8. To generate a summary of n documents select first sentence of the document and add it to the summary then select n-2 top sentences (other than first and last) from L add them to summary then add the last sentence to the summary.

Graph Based Approach [17]

In every document, the nouns of the text play the most vital role in helping us understand the meaning of the text basis the context it was written in. One of the approach constructs a graph of all the nouns of the text to determine how closely related the nouns of the text are to each other which ultimately helps in weighing the sentences. The sentences are scored based on how significant the nouns present in the sentence are to the entire document. The high scoring sentences are considered the most important sentences in the text and these sentences are chosen for the summary. All the Graph Based methods mainly include the tasks of pre-processing, building graph models, applying ranking algorithms and finally generating summaries.

In preprocessing phase, The text is divided into sentences, and further sentences are decomposed into word. Each word is assigned the most appropriate part of speech tag depending on the form of the word and the tags of its neighboring words. After that, A graph is built with nouns as vertices and the weights of the edges connecting them represent the relevance between the nouns. After this any one of the techniques like weighted graph model using a hybrid approach, Ranking algorithm, Shortest path algorithm are used to score the sentences then the sentences with highest score is used to generate a summary. This method of text summarization works well with news articles, Wikipedia searches and technical documents etc.

Term Frequency-Inverse Document Frequency (TF-IDF) Approach [15,16]

Term Frequency-Inverse Document Frequency is a numerical statistic method which shows how the word is important to a document in the collection. To generate a summary non-stop words that occured more frequently are consider as query word. Then the term frequency and the inverse document frequency is calculated for each non-stop word. The number of times a term occurs in a document is called its term frequency [TF]. TF says that the word occured more frequently better reflect the content of the document than the word occured less frequently. It can be calculated as,

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document). (1)

The inverse document frequency [IDF] is a measure of how much information the word provides, means, whether the term is common or rare across all documents. IDF suggest that if the term is less frequent in whole document then it is more important for the document. It can be calculated as,

IDF (t) = log(Total number of documents in the collection / Number of documents with term t in it). (2)

Thematic words are obtained by comparing the ratio between two frequencies, referred as (TF(t) * IDF(t)) measure. Once (TF(t) * IDF(t)) score has been computed for each word. The next step is to calculate number of such thematic words per sentence. With this value sentences in the input text are ranked and highest scored sentences are picked to be part of summary. Redundancy of information is extremely high in this method.

Text Summerization using Fuzzy Logic [20,21,22]

Fuzzy logic system design usually implicates selecting fuzzy rules and membership function. The selection of fuzzy rules and membership functions directly affect the performance of the fuzzy logic system.

The fuzzy logic system consists of four components: fuzzifier, inference engine, defuzzifier, and the fuzzy knowledge base. In the fuzzifier, crisp inputs are translated into linguistic values using a membership function to be used to the input linguistic variables. After fuzzification, the inference engine refers to the rule base containing fuzzy IF-THEN rules to derive the linguistic values. In the last step, the output linguistic variables from the inference are converted to the final crisp values by the defuzzifier using membership function for representing the final sentence score. In order to implement text summarization based on fuzzy logic, first, the features such as sentence length, term weight, sentence position, sentence to sentence similarity, Title Word etc are used as input to the fuzzifier. Triangular membership functions and fuzzy logic is used to summarize the document.

The input membership function for each feature is divided into five fuzzy set which are composed of unimportant values (low (L) and very low (VL), Median (M)) and important values (high (H) and very high (VH)).

In inference engine, the most important part in this procedure is the definition of fuzzy IF-THEN rules. The important sentences are extracted from these rules according to our features criteria. Sample of IF-THEN rules shows as the following rule.

IF (NoWordInTitle is VH) and (SentenceLength is H) and (TermFreq is VH) and (SentencePosition is H) and (SentenceSimilarity is VH) and (NoProperNoun is H) and (NoThematicWord is VH) and (NumbericalData is H) THEN (Sentence is important) Likewise, the last step in fuzzy logic system is the defuzzification. The output membership function which is divided into three membership functions: Output Unimportant, Average, and Important is used to convert the fuzzy results from the inference engine into a crisp output for the final score of each sentence.

In fuzzy logic method, each sentence of the document is represented by sentence score. Then all document sentences are ranked in a descending order according to their scores. A set of highest score sentences are extracted as document summary based on the compression rate. It has been proven that the extraction of 20 percent of sentences from the source document can be as informative as the full text of a document. Finally, the summary sentences are arranged in the original order.

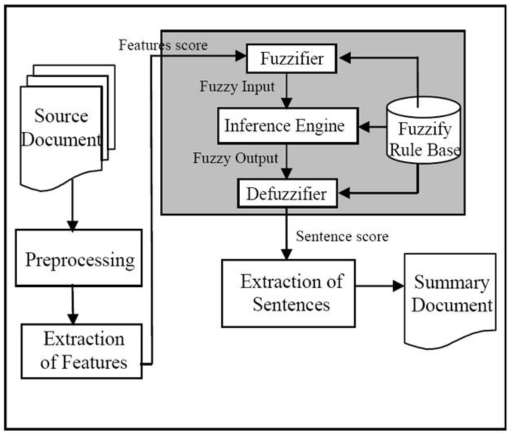

Figure 3. Fuzzy Inference Engine

Clustering Techniques[6,10]

Different approaches to clustering data can be described with the help of the hierarchy shown in Figure 4 (other taxonometric representations of clustering methodology are possible; ours is based on the discussion in Jain and Dubes [1988]). At the top level, there is a distinction between hierarchical and partitional approaches (hierarchical methods produce a nested series of partitions, while partitional methods produce only one).

The taxonomy shown in Figure 2 must be supplemented by a discussion of crosscutting issues that may (in principle) affect all of the different approaches regardless of their placement in the taxonomy.

Agglomerative vs. divisive: This aspect relates to algorithmic structure and operation. An agglomerative approach begins with each pattern in a distinct (singleton) cluster, and successively merges clusters together until a stopping criterion is satisfied. A divisive method begins with all patterns in a single cluster and performs splitting until a stopping criterion is met.

Monothetic vs. polythetic: This aspect relates to the sequential or simultaneous use of features in the clustering process. Most algorithms are polythetic; that is, all features enter into the computation of distances between patterns, and decisions are based on those distances. A simple monothetic algorithm reported in Anderberg [1973] considers features sequentially to divide the given collection of patterns.

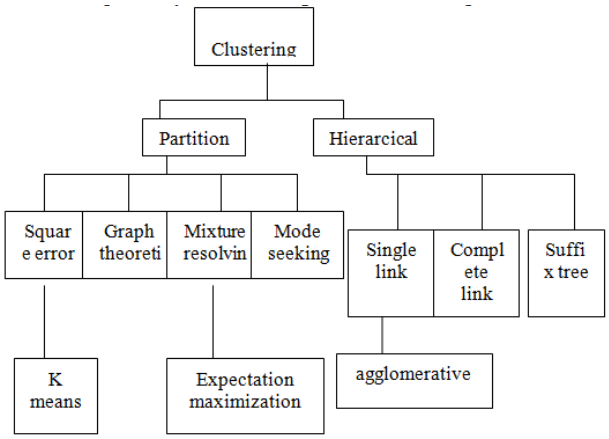

Figure 4 Clustering Techniques.

Hard vs. fuzzy: A hard clustering algorithm allocates each pattern to a single cluster during its operation and in its output. A fuzzy clustering method assigns degrees of membership in several clusters to each input pattern. A fuzzy clustering can be converted to a hard clustering by assigning each pattern to the cluster with the largest measure of membership.

Deterministic vs. stochastic: This issue is most relevant to partitional approaches designed to optimize a squared error function. This optimization can be accomplished using traditional techniques or through a random search of the state space consisting of all possible labelling.

Incremental vs. non-incremental: This issue arises when the pattern set to be clustered is large, and constraints on execution time or memory space affect the architecture of the algorithm. The early history of clustering methodology does not contain many examples of clustering algorithms designed to work with large data sets, but the advent of data mining has fostered the development of clustering algorithms that minimize the number of scans through the pattern set, reduce the number of patterns examined during execution, or reduce the size of data structures used in the algorithm's operations.

Related works in Indian languages

The extractive summarization research works in Indian Languages are not up to date as compared to the other languages like English, German, and Spanish etc. It is mainly due to the diversity in the Indian Languages and the lack of resources such as raw data, various NLP tools etc. This section explains the extractive summarization works in Indian Languages like Malayalam, Hindi, and Bengali etc.

Malayalam Text Summarization [3]

Krishnaprasad P, Sooryanarayanan A and Ajeesh Ramanujan uses abstractive approach to summarize the text in Malayalam language. They generated the summary from the given document by recombining the extracted important sentences from the text. In order to identify the important sentences in the text they follow the content word method. Content word is extracted from the frequency distribution of except stop words. The proposed system comprises of two components, Text analyzing component and the summary generation component. The Text analyzing component is used to identify the features associated with the sentences and based upon the features it assign the score to each sentence. The main tasks involved are sentence marking, feature extraction and sentence ranking. Summary generation component uses the sentence score to generate the summary and it involves two main tasks Sentence Selection and Summary generation.

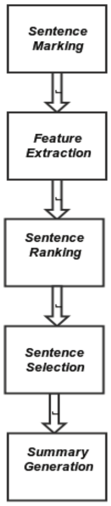

Figure 5. The Extractive Summarization System for Malayalam

Figure 5 shows the architecture of proposed summarization system. Firstly text normalization process is performed and it involves the splitting the text into sentences and further split the sentences into words. Then the normalized text will be used for feature extraction process. Feature extraction process extracts the features associated with the sentences and the features associated with words. The features used are frequency of word and number of characters in the word etc. After that the next process is scoring of sentences. System will calculate the score of each sentence based on word frequency and average number of characters per sentence. In the sentence ranking task system will rank each and every sentence in the given text based on the frequency words in the text and average number of characters in each word. After the sentence ranking the next task is Sentence Selection. In this phase, top N scored sentences may be used to generate the summary. But this generate the coherence. So, After selecting the sentences, the sentences are recombined in the chronological order present in the original input text for getting readable summary.

The proposed system for Malayalam provides faster method to generate the summary. For each news article 4 summaries have been generated based on the condensation rate of 10,15,20,25 percentages and the generated summaries are evaluated with the reference summaries by using standard metric ROUGH. The performance of the given system may be improved by adding the stemming process, improvement in the sentence splitting criteria and adding more number of features.

Hindi Text Summarization

Nikita Desai and Prachi shah in their paper proposed automatic text summarization using Supervised Machine Learning Technique for Hindi Language. They represented Each sentence in the document by a set of various features namely- sentence paragraph position, sentence overall position, numeric data, presence of inverted commas, sentence length and keywords in sentences. The sentences are classified into one of four classes namely- most important, important, less important and not important. The classes are in turn having ranks from 4 to 1 respectively with “4”indicating most important sentence and “1” being least relevant sentence. Next a supervised machine learning tool SVMrank is used to train the summarizer to extract important sentences, based

on the feature vector. The sentences are ordered according to the ranking of classes. Then based on the required compression ratio, sentences are included in the final summary. The experiment was performed on news articles of different category such as bollywood, politics and sports. The performance of the technique is compared with the human generated summaries. The average result of experiments indicates 72% accuracy at 50% compression ratio and 60% accuracy at 25% compression ratio.

The proposed technique is grouped into 3 major Blocks - Pre-processing, Processing and Extraction The general outline of the methodology used is as follows.

```
Input: A text file (original -Og).
Output: A summarized text(S) of original (Og), as per compression ratio.
1. Read Input text File -Og
2. Pre-process the file Og . //Preprocessing step
2.1 Segment text file into sentences.
2.2 Tokenize each sentence into words.
2.3 Remove stop-words.
3. //Processing step
3.1 Extract following features from Og file
Sentence Paragraph Position (f1),
Sentence Overall Position (f2),
Numerical Data in Sentence (f3),
Presence of Inverted Commas (f4),
Sentence Length (f5),
Keywords in Sentence (f6)
3.2 Apply SVM model to rank sentences in range from4 to1,with “4” indicating most important sentence and “1” indicating not important sentence.
4. Generate summary . //Extraction step
While (lines in summary file (S) does not exceed maximum limit as per given by compression ratio)
Do
4.1Extract all lines from Og with rank4
4.2 Extract all lines from Og with rank3
4.3 Extract all lines from Og with rank2
5. Display summary file (S).
```

In this system the performance in each of the subtasks directly affects the ability to generate high quality summaries. Also it is noteworthy that the summarization is more difficult if need more compression. More features like named entity recognition, cue words, context information, world knowledge etc, can be added to improve the technique. Also it would be interesting to find other suitable machine learning classifiers other than SVM for the task.

Panjabi Text Summarization

In their paper, the authors Vishal Gupta and Gurpreet Singh Nehal proposed Automatic Punjabi Text Extractive Summarization System in which it comprises of two main phases: 1) Pre Processing 2) Processing. Pre Processing is structured representation of original Punjabi text. Preprocessing phase includes Punjabi words boundary identification, Punjabi sentences boundary identification, Punjabi stop words elimination, Punjabi language stemmer for Nouns and proper names, applying input restrictions and elimination of duplicate sentences. In processing phase, sentence features are calculated and final score of each sentence is determined using feature-weight equation. Top ranked sentences in proper order are selected for final summary. This demo paper concentrates on Automatic Punjabi Text Extractive Summarization System.

PROPOSED SYSTEM

In proposed system, we developed a system which will generate the summary of document by extractive approach. The single document is given as input and then it goes through four phases. The first phase includes sentence segmentation, tokenization, stemming and stop word removal. For stemming purpose, porter’s algorithm is used. Then it goes through word and sentence scoring. It takes the features like title feature, Upper case word, Sentence length and word frequency. After this the sentence ranking is done on the basis of frequencies of word and sentences. Then the summary is generated.

The detailed System phases are as follows.

Phase 1:

In this phase, the proposed system is going to accept the single text document and it contains the following phases:

1) Sentence segmentation: From the set of input in the text documents, each individual document D is segmented separately as D = S1, S2, . . ., Sn, where n is the number of sentences in document.

2) Tokenization: Terms of each sentence are tokenized as T = t1, t2,. . ., tm, where m is the number of terms.

3) Stemming: Stemming performs the elimination of ’ed’ and ’ing’ suffixes from the given word. For that purpose Porter’s algorithm is used.

4) Stop word removal: Commonly used words in English language such as ’a’, ’an’, and ’the’ which has less important significance with respect to the document are removed.

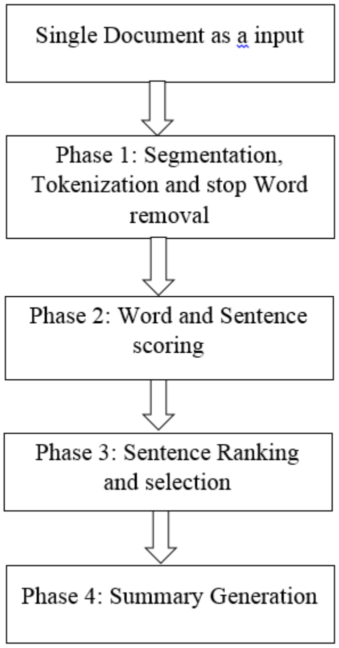
Single Document as a input
Phase 1: Segmentation, Tokenization and stop Word removal
Phase 2: Word and Sentence scoring
Phase 3: Sentence Ranking and selection
Phase 4: Summary Generation

Figure 6 Proposed Architecture

Phase 2:

In this phase, the frequencies are get allocated to each word. the frequency is depends on how many times that particular term occurs in that document. For that purpose standard formula is used as follows:

1. Formula to calculate word frequency:

2. Formula to calculate sentence frequency:

Following features are used for determining the sentence weights:

1) Title word feature: Sentences containing words that appear in the title are also indicative of the theme of the document. These sentences are having greater chances for including in summary.

2) Term frequency Method: This method is based on the frequency of the term. The word having more frequency is taken into the summary.

3) Sentence Length feature: Very large and very short sentences are usually not included in summary.

4) Upper-case word feature: upper-case words (with certain obvious exceptions) are treated as thematic words, as well. Sentences containing acronyms or proper names are included.

Phase 3:

In this phase, depending on the frequency of each sentence the ranking is done, which means the sentences are get assigned with some frequency and after that they are ranked in descending order. This became the input to next process.

Phase 4:

To select the sentences which we got in phase 3 for summary generation, we set threshold value. Depending upon this threshold value high score sentences are used to generate summary.

Conclusion

The main aim of this research work is to combine the both approaches of query dependent summarization and clustering of the document. The proposed work will be mainly focused on summarization of text files (i.e. .txt).The proposed work will be limited to clustering of text files of Standard files related to the topic popular amongst researchers will be used. Standard performance evaluation metrics will be used to validate performance.

Acknowledgment

We would like to thank Mr. Amit Kolhe, Managing Trustee of Sanjivani College of Engineering, Kopargaon, India and Principal of Sanjivani College of Engineering, Kopargaon, India for providing the resources needed to carry out the proposed work.

References

1. Elena Lloret, “Text Summerization: An overview”.
2. Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, Krys Kochut, “Text Summarization Techniques: A Brief Survey”, In Proceedings of arXiv, USA, July 2017.
3. Krishnaprasad P, Sooryanarayanan A, Ajeesh Ramanujan, “Malayalam Text Summarization: An Extractive Approach”, IEEE, International Conference on Next Generation Intelligent Systems (ICNGIS), 2016
4. Dragomir R Radev, Eduard Hovy, and Kathleen McKeown. 2002. “Introduction to the special issue on summarization”. Computational linguistics 28, 4 (2002), 399–408.
5. Sunitha C., Dr. A. Jaya, Amal Ganesh, “A Study on Abstractive Text Summarization Techniques in Indian Languages”, Elsevier, Fourth International Conference on Recent Trends in Computer Science and Engineering, 2016.
6. Yogesh Kumar Meena, Dinesh Gopalani, “Feature Priority Based Sentence Filtering Method for Extractive Automatic Text Summarization”, Elsevier, International Conference on Intelligent Computing, Communication & Convergence (ICCC), 2015.
7. Sabina Yeasmin, Priyanka Basak Tumpa, Adiba Mahjabin Nitu, Md. Palash Uddin, Emran Ali, Masud Ibn Afjal, “Study of Abstractive Text Summarization Techniques”, American Journal of Engineering Research, 2017, Volume-6, Issue-8, pp-253-260.
8. Nikita desai, prachi shah, “Automatic Text Summarization Using Supervised Machine Learning Technique For Hindi Langauge”, International Journal of Research in Engineering and Technology, 2016, Vol. 5, Issue. 6.
9. Vishal Gupta, Gurpreet Singh Lehal, “A Survey of Text Summarization Extractive Techniques”, Journal of Emerging Technologies in Web Intelligence, August 2010, Vol. 2, No. 3.
10. Sheetal Shimpikar, Sharvari Govilkar, “A Survey of Text Summarization Techniques for Indian Regional Languages”, International Journal of Computer Applications, Volume 165, No.11, May 2017.
11. Yogesh Kumar Meena, Dinesh Gopalani, “Domain Independent Framework for Automatic Text Summarization”, Elsevier, International Conference on Intelligent Computing, Communication & Convergence (ICCC), 2015.
12. Jimmy Lin., “Summarization.”, Encyclopedia of Database Systems. Heidelberg, Germany: Springer-Verlag, 2009.
13. Jackie CK Cheung, “Comparing Abstractive and Extractive Summarization of Evaluative Text: Controversiality and Content Selection”, B. Sc. (Hons.) Thesis in the Department of Computer Science of the Faculty of Science, University of British Columbia, 2008.
14. Soumye Singhal, Arnab Bhattacharya, “Abstractive Text Summarization”
15. Rene Arnulfo Garcia-Herandez and Yulia Ledeneva, “Word Sequence Models for Single Text Summarization”, IEEE, 44-48, 2009.
16. Hans christian, mikhael pramodana agus, derwin suhartono, “Single Document Automatic Text Summarization Using

Term Frequency-Inverse Document Frequency (tf-idf)", ComTech Vol. 7 No. 4, December 2016,285-294

17. Akash Ajampura Natesh, Somaiah Thimmaiah Balekuttira, Annapurna P Patil, "Graph Based Approach for Automatic Text Summarization", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, Special Issue 2, October 2016.
18. Khushboo S. Thakkar, Dr. R. V. Dharaskar, M. B. Chandak, "Graph-Based Algorithms for Text Summarization", IEEE, Third International Conference on Emerging Trends in Engineering and Technology, 2010.
19. Rasim ALGULIEV, Ramiz ALIGULIYEV, "Evolutionary Algorithm for Extractive Text Summarization", Scientific Research, Intelligent Information Management, November 2009, 1, 128-138.
20. Mr. S. A. Babar, Prof. S. A. Thora, "Improving Text Summarization using Fuzzy Logic & Latent Semantic Analysis", International Journal of Innovative Research in Advanced Engineering (IJIRAE), Volume 1, Issue 4, May 2014.
21. L. Suanmali , N. Salim and M.S. Binwahlan,"Fuzzy Logic Based Method for Improving Text Summarization" , International Journal of Computer Science and Information Security, 2009, Vol. 2, No. 1,pp. 4-10.
22. S. A. Babar, Pallavi D. Patil, "Improving Performance of Text Summarization", Elsevier, International Conference on Information and Communication Technologies (ICICT), 2014.