IMPLEMENTATION OF REST API FOR ARTICLE SUMMARIZATION

By

DHRUVESH RAJODIYA (ID NO:15CEUOG045)

CHANDANI MODI (ID NO:15CEUOS052)

DARSHIT KHANT (ID NO:15CEUOG071)

A project submitted
In

partial fulfilment of the requirements
for the degree of

BACHELOR OF TECHNOLOGY
in
Computer Engineering

Internal Guide	**External Guide**
Dr. Brijesh S. Bhatt,	*Mr. Rajesh Bansal,*
Associate Professor,	*Project Manager, Edgepr,*
Dept. Of Comp. Engg.	*Infosys Technologies Limited*

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University
April 2019

CERTIFICATE

This is to certify that the project work titled

IMPLEMENTATION OF REST API FOR ARTICLE SUMMARIZATION

is the bonafide work of

DHRUVESH RAJODIYA (ID NO:15CEUOG045)

CHANDANI MODI (ID NO:15CEUOS052)

DARSHIT KHANT (ID NO:15CEUOG071)

carried out in the partial fulfillment of the degree of Bachelor of Technology in Computer Engineering at Dharmsinh Desai University in the academic session December 2018 to March 2019.

Dr. Brijesh S. Bhatt,

Associate Professor,

Dept. Of Computer. Engg.

Dr. C. K. Bhensdadia,

Head,

Dept. Of Computer Engg.

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University
April 2019

Company certificate

Acknowledgement

We would like to thank Dr. C.K. Bhensdadia, for providing us with the opportunity to pursue training at Infosys. We would also like to thank Mr. Rajesh Bansal and Mr. Shriram Puttagunta, our external guide at Infosys. He provided us with valuable inputs and understanding of our project during our tenure as student trainees. Our thanks also goes out to Dr. Brijesh S. Bhatt, our internal guide at DDU, for providing useful suggestions and overseeing our project documentation, implementation so as to make it parallel with industry standards. We would also like to thank the Infosys administration for providing us with useful resources and a comfortable workspace during our training period.

Abstract

Automatic text summarization is the task for computers to produce a concise and fluent summary conveying the key information in the input. There are generally two types of automatic texts summarization: extraction and abstraction. An extractive summarization method consists of selecting important sentences, paragraphs etc. from the original document and concatenating them into shorter form. The importance of sentences is decided based on statistical and linguistic features of sentences. An abstractive summarization method consists of understanding the original text and re-telling it in fewer words. It uses linguistic methods to examine and interpret the text and then find the new concepts and expressions to best describe it by generating a new shorter text that conveys the most important information from original text. Many papers has been published on extraction summarization, however, it cannot provide summary close to human language. This project will focus on extractive summarization on long texts. Results have shown that picking shortest-clause from most important sentence choosen by LexRank Algorithm appears to have the best performance.

INDEX

List Of Figures

Chapter 1
Introduction

1.1 Objective

With the growing amount of data in the world, interest in the field of automatic text summarization has been widely increasing so as to reduce manual effort of a person working on it. This thesis focuses on the comparison of various existing algorithms for the summarization of text passages.

1.2 Application

Text summarizer reduces a text file into a passage or paragraph that conveys the main meaning of the file. The searching of important information from a large text file is very difficult job for the users so they can use this summarizer to automatically extract the important information from file. This summarizer saves time of users as instead of reading whole text file they can give this file to summarizer and can get important information from the large document. In today's world to extract information from the World Wide Web is very easy. This extracted information is a huge text repository. With the rapid growth of the World Wide Web (internet), information overload is becoming a problem for many peoples. Automatic summarization can be an indispensable solution to reduce the information overload problem on the web.

1.3 Overview of the Project

Generally, there are two approaches to text summarization:

1. Extractive

Extractive methods work by selecting a subset of existing words, phrases, or sentences in the original text to form the summary. First clean the text file by removing full stop, common words (conjunction, verb, adverb, preposition etc.). Then calculate the frequency of each words and select words which have maximum frequency. This technique retrieves important sentences and it emphasizes on high information richness in the sentence. These generated sentences are clustered together to generate the summary of the document.

Following are some of the abstractive text Summarization algorithms:

1. Lex Rank
2. Luhn
3. LSA
4. Text Rank

2. Abstractive

Abstractive system generates new phrases, possibly rephrasing or using words that were not in the original text. Naturally abstractive approaches are harder than extractive. 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. Has complex capabilities like generalization, paraphrasing and incorporating real-world knowledge.

1.4 Technologies Used

1.4.1 Visual Studio Code

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and MacOS. It includes support for debugging, embedded **Git** control, syntax highlighting, intelligent code completion, snippets, and code **refactoring**

It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences. The source code is free and open source and released under the permissive MIT License. The compiled binaries are freeware and free for private or commercial use.

Chapter 2
About the System

2.1 Survey

Text summarization refers to the technique of shortening long pieces of text. The intention is to create a coherent and fluent summary having only the main points outlined in the document. Automatic text summarization is a common problem in machine learning and natural language processing (NLP).

There is an enormous amount of textual material, and it is only growing every single day. Think of the internet, comprised of web pages, news articles, status updates, blogs and so much more. The data is unstructured and the best that we can do to navigate it is to use search and skim the results.

There is a great need to reduce much of this text data to shorter, focused summaries that capture the salient details, both so we can navigate it more effectively as well as check whether the larger documents contain the information that we are looking for.

Approaches to automatic text summarization involves

- Elimination of redundancy: The sentences in the text which convey the same meaning are said to be redundant and can be eliminated in the summary.

- Identification of significant sentences: Summary being a shorter representation of text requires to include only salient sentences from the original document.

- Generation of coherent summaries: Sentences selected for summarization needs to be ordered and grouped so that coherence and readability is maintained.

- Metrics for evaluating the automatically generated summaries: In most of the cases the quality of the summary is judged by humans and hence automatic evaluation is a desirable feature.

2.2 Need for the Text Summarization

Business leaders, analysts, paralegals, and academic researchers need to comb through huge numbers of documents every day to keep themselves ahead, and a large portion of their time is spent just figuring out what document is relevant and what isn't. By extracting important sentences and creating comprehensive summaries, it's possible to quickly assess whether or not a document is worth reading.

Automatic text summarization is also useful for students and authors. Imagine being able to automatically generate an abstract based for your research paper or chapter in a book in a clear and concise way that is faithful to the original source material!

We cannot possibly create summaries of all of the text manually; there is a great need for automatic methods. There are 6 reasons why we need automatic text summarization tools.

1. Summaries reduce reading time.
2. When researching documents, summaries make the selection process easier.
3. Automatic summarization improves the effectiveness of indexing.
4. Automatic summarization algorithms are less biased than human summarizers.
5. Personalized summaries are useful in question-answering systems as they provide personalized information.
6. Using automatic or semi-automatic summarization systems enables commercial abstract services to increase the number of texts they are able to process.

2.4 How does a text summarization algorithm work?

Usually, text summarization in NLP is treated as a supervised machine learning problem (where future outcomes are predicted based on provided data). Typically, here is how using the extraction-based approach to summarize texts can work:

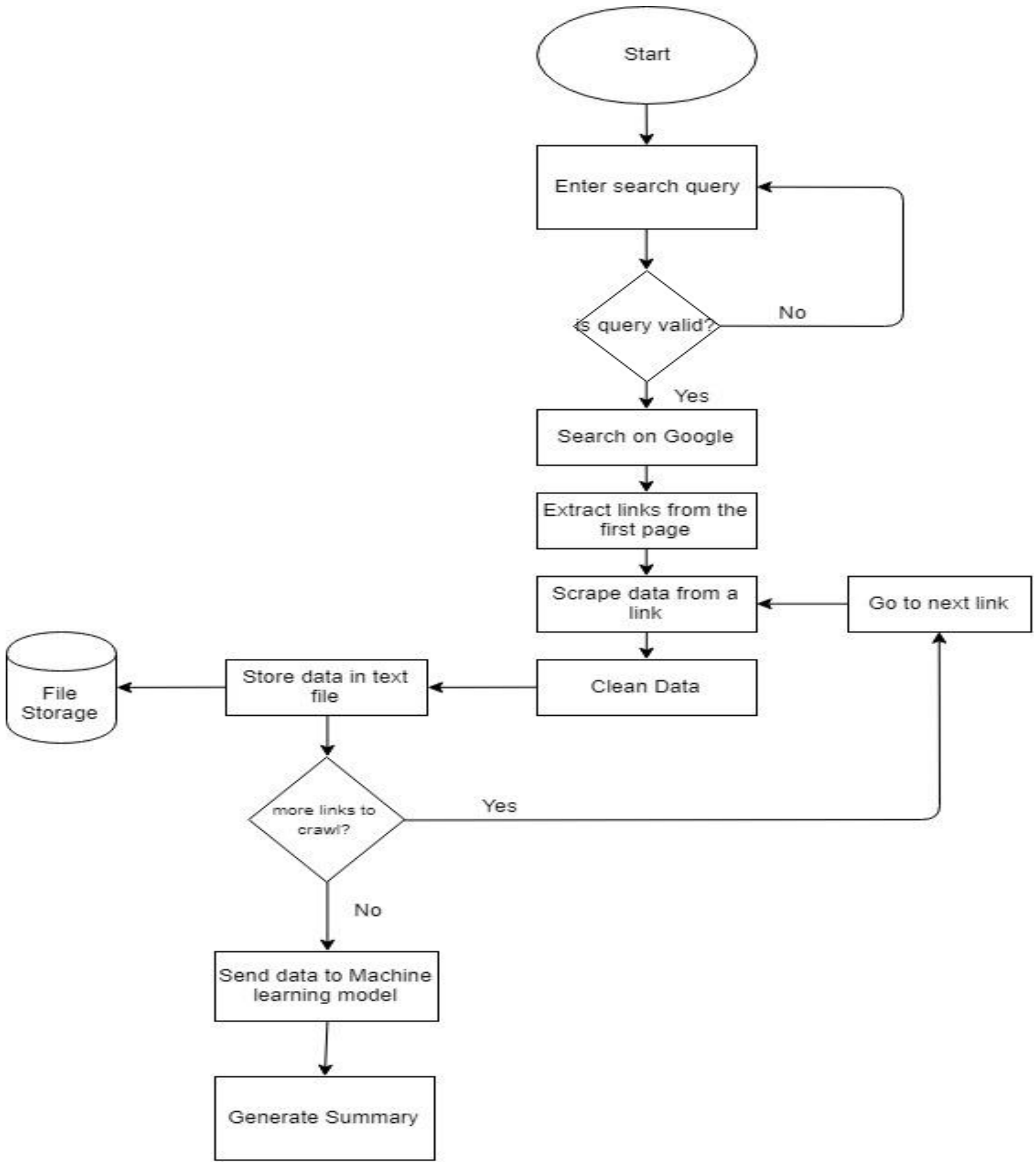
- ✓ Introduce a method to extract the merited keyphrases from the source document. For example, you can use part-of-speech tagging, words sequences, or other linguistic patterns to identify the keyphrases.

- ✓ Gather text documents with positively-labeled keyphrases. The keyphrases should be compatible to the stipulated extraction technique. To increase accuracy, you can also create negatively-labeled keyphrases.

- ✓ Train a binary machine learning classifier to make the text summarization. Some of the features you can use include:

- Length of the keyphrase
- Frequency of the keyphrase
- The most recurring word in the keyphrase
- Number of characters in the keyphrase

- ✓ Finally, in the test phrase, create all the keyphrase words and sentences and carry out classification for them.



2.4.1 Flow chart for Text Summarization

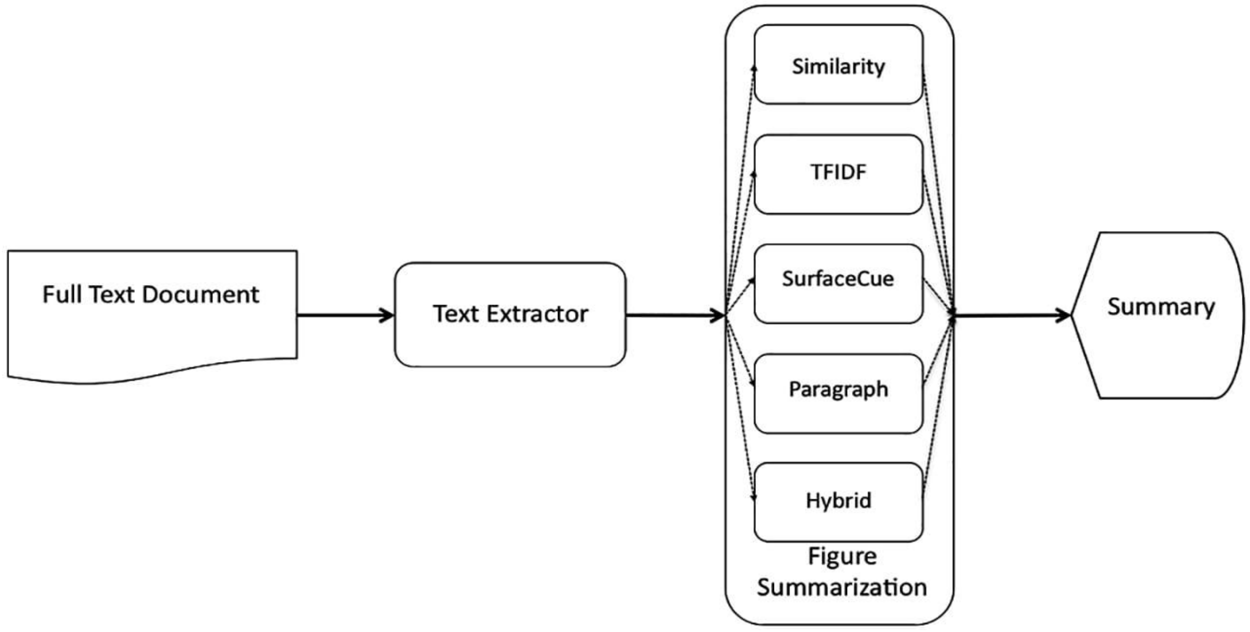
2.5 Text Summarization API

Text Summarization API provides professional text summarizer service which is based on advanced Natural Language Processing and Machine Learning technologies. It can be used to summarize short important text from the URL or document that user provided.

Sometimes, text we are dealing with, is just too long. Too long to read, to analyze and too long to consume. Summarization allows you to take the important, relevant points and topics from a piece of text, making it easier to consume and analyze**.**Submit a piece of text, or a URL and allow API to summarize it for you, extract key phrases and sentences and produce a consumable snapshot of your text.

In knowledge driven economies, it's increasingly common for many jobs to revolve around the consumption and analysis of information. This may entail, for example, reading dozens of articles to draw conclusions about emerging trends. Doable, right? Imagine, though, if you were a financial analyst and had to read through hundreds of articles and reports each day in order to make informed decisions. Suddenly, the task becomes much more daunting. It's difficult to keep track of what's important when you're overloaded with information and your time is limited.

For text analytics problems like this, machine learning can come to the rescue. This Text Summarization API is designed to read through an article and pick out key, "big idea" sentences so you can quickly determine whether the article's content will be useful to you. In short it enables you to better allocate your time towards analysis of the information that is most interesting and relevant.



2.5.1. Diagram for Text Summarization

2.6 SCOPE

The overall scope of the project is to have a deeper knowledge of the techniques in Machine Learning, Deep Learning and Data Analysis in order to generate concise summaries of news articles, which lets a user see a summary given a news article or of the latest news. The scope also involves understanding about how to find a subset of data which contains the "information" of the entire set. The scope of this project can be extended by generating sentiment analysis of those concise summaries.

Chapter 3
Algorithm and Its Working

3.1 Text Summarization can be done in three ways

1. Based on Input Type.
    1. Single Document
    2. Multi Document

2. Based on Output Type.
    1. Extractive
    2. Abstractive

3. Based on Purpose.
    1. Generic
    2. Domain Specific
    3. Query-based

- ➢ In this section we mainly focus on Text Summarization Based on Output Type.

- ➢ There are two ways to do text summarization in this type one is extractive and other one is abstractive approach.

- ➢ Extractive approaches select passages from the source text, then arrange them to form a summary.

- ➢ Abstractive approaches use's natural language generation techniques to write summary. This is more advanced and closer to human-like interpretation.

3.2 Features for Extractive Text Summarization

Some features to be considered for including a sentence in final summary are:

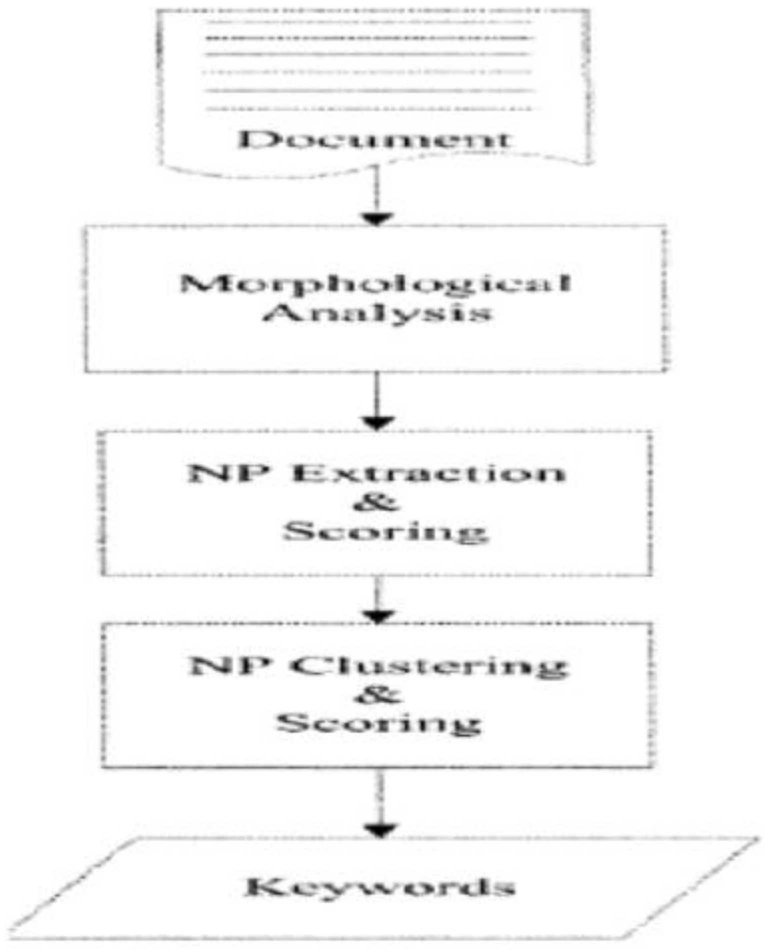
A. Content word (Keyword) feature:

Content words or Keywords are usually nouns and determined using tf × idf measure. Sentences having keywords are of greater chances to be included in summary. Another keyword extraction method is given below, having three modules:

1) Morphological Analysis

2) Noun Phrase (NP) Extraction and Scoring

3) Noun Phrase (NP) Clustering and Scoring



3.2.1 Keyword extraction method

B. 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.

C. Sentence location feature:

Usually first and last sentence of first and last paragraph of a text document are more important and are having greater chances to be included in summary.

D. Sentence Length feature:

Very large and very short sentences are usually not included in summary.

E. Proper Noun feature:

Proper noun is name of a person, place and concept etc. Sentences containing proper nouns are having greater chances for including in summary.

F. Upper-case word feature:

Sentences containing acronyms or proper names are included.

G. Cue-Phrase Feature:

Sentences containing any cue phrase (e.g. “in conclusion”, “this letter”, “this report”, “summary”, “argue”, “purpose”, “develop”, “attempt” etc.) are most likely to be in summaries.

H. Biased Word Feature:

If a word appearing in a sentence is from biased word list, then that sentence is important. Biased word list is previously defined and may contain domain specific words.

I. Font based feature:

Sentences containing words appearing in upper case, bold, italics or Underlined fonts are usually more important.

J. Pronouns:

Pronouns such as “she, they, it” cannot be included in summary unless they are expanded into corresponding nouns.

K. Sentence-to-Sentence Cohesion:

For each sentence s compute the similarity between s and each other sentence s’ of the document, then add up those similarity values, obtaining the raw value of this feature for s. The process is repeated for all sentences.

L. Sentence-to-Centroid Cohesion:

For each sentence s as compute the vector representing the centroid of the document, which is the arithmetic average over the corresponding coordinate values of all the sentences of the document; then compute the similarity between the centroid and each sentence, obtaining the raw value of this feature for each sentence.

M. Occurrence of non-essential information:

Some words are indicators of non-essential information. These words are speech markers such as “because”, “furthermore”, and “additionally”, and typically occur in the beginning of a sentence. This is also a binary feature, taking on the value “true” if the sentence contains at least one of these discourse markers, and “false” otherwise.

N. Discourse analysis:

Discourse level information in a text is one of good feature for text summarization. In order to produce a coherent, fluent summary, and to determine the flow of the author's argument, it is necessary to determine the overall discourse structure of the text and then removing sentences peripheral to the main message of the text.

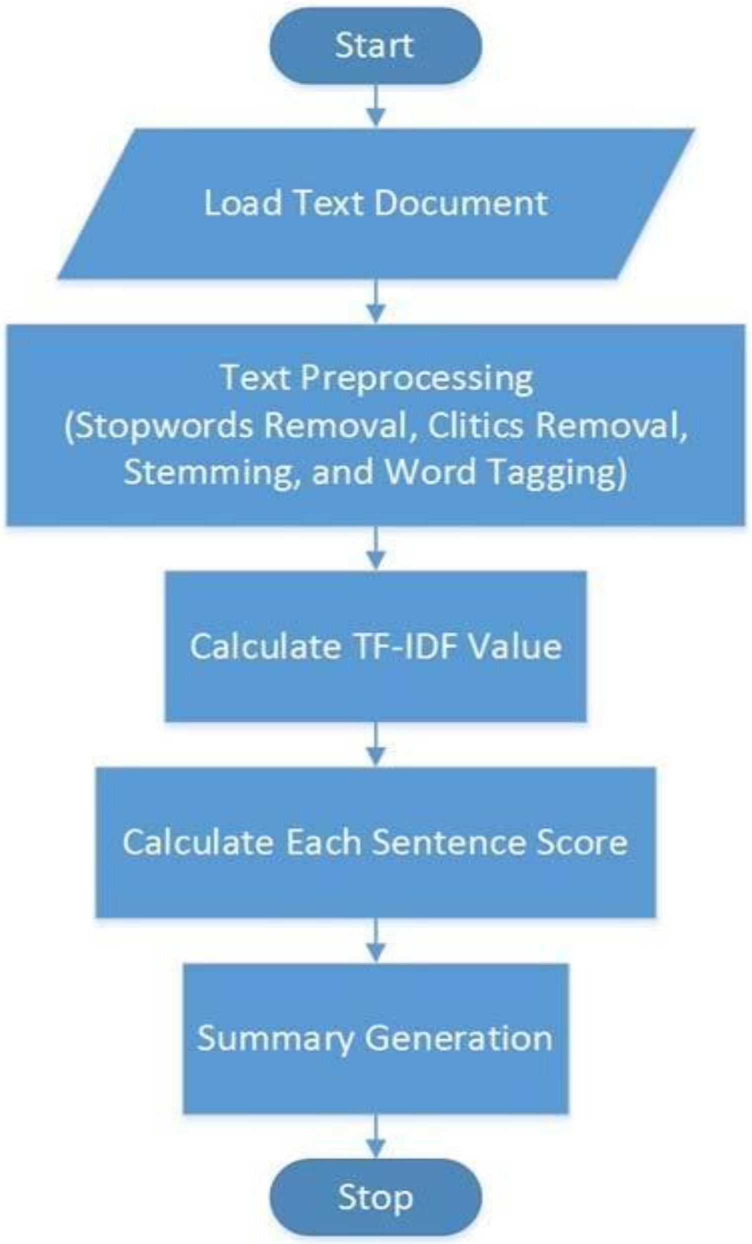
These features are important as, a number of methods of text summarization are using them. These features are covering statistical and linguistic characteristics of a language.
Mainly, our focus is on Extractive text summarization techniques.

3.3 Extractive Text Summarization methods:

3.3.1 Term Frequency-Inverse Document Frequency (TF-IDF(term frequency-inverse document frequency)) method

It is a numerical statistic which reflects how important a word is in a given document. The TF-IDF value increases proportionally to the number of times a word appears in the document. This method mainly works in the weighted term-frequency and inverse sentence frequency paradigm. where sentence-frequency is the number of sentences in the document that contain that term. These sentence vectors are then scored by similarity to the query and the highest scoring sentences are picked to be part of the summary. Summarization is query-specific. The hypothesis assumed by this approach is that if there are ‘‘more specific words’’ in a given sentence, then the sentence is relatively more important. The target words are usually nouns. This method performs a comparison between the term frequency (tf) in a document - in this case each sentence is treated as a document and the document frequency (df), which means the number of times that the word occurs along all documents. The TF/IDF score is calculated as follows:

$$TF/IDF(W) = DN(\log(1+tf)/\log(df))$$


Start
Load Text Document
Text Preprocessing
(Stopwords Removal, Clitics Removal,
Stemming, and Word Tagging)
Calculate TF-IDF Value
Calculate Each Sentence Score
Summary Generation
Stop

3.3.1.1 Flow Chart diagram for TF-IDF

3.3.2 Cluster based method

In this method, the semantic nature of a given document is captured and expressed in natural language by a set of triplets (subjects, verbs, objects related to each sentence).Cluster these triplets using similar information. The triplets statements are considered as the basic unit in the process of summarization. More similar the triplets are there more the useless information is repeated; thus a summary may be constructed using a sequence of sentences related to the computed clusters.

Algorithm for Text Summarization using clustering method:

1. The user selected collection of documents & query is the input to the summarizer.

2. We have maintained a list of maps where each term from document collection is stored in a map with its number of occurrences. A map contains all the synonyms, of the term from the document collection. We have used WordNet dictionary to find synonyms.

3. Query modification technique is used. It works as follows:

    3.1 Split a query into tokens & find the synonym for each token. We will get the synonym from list of maps if the token or synonym exists in a document collection & append the most frequent synonym of the query term to query.

    3.2. We have generated a corpus for strengthening the query. The most frequently occurred words from corpus are selected & those words are appended to the query. So the query is strengthened.

4. Some features are used to calculate sentence score. The features are listed as follows : Noun Feature, Cue Phrase Feature, Sentence Length Feature, Numerical data Feature, Sentence Position, Sentence centrality (similarity with other sentences), Upper Case word feature, Sentence similarity with user query, Term frequency & Inverse Document Frequency is also used to score the sentences.

5. The documents are clustered by using, cosine similarity as a similarity measure to generate the appropriate document clusters.

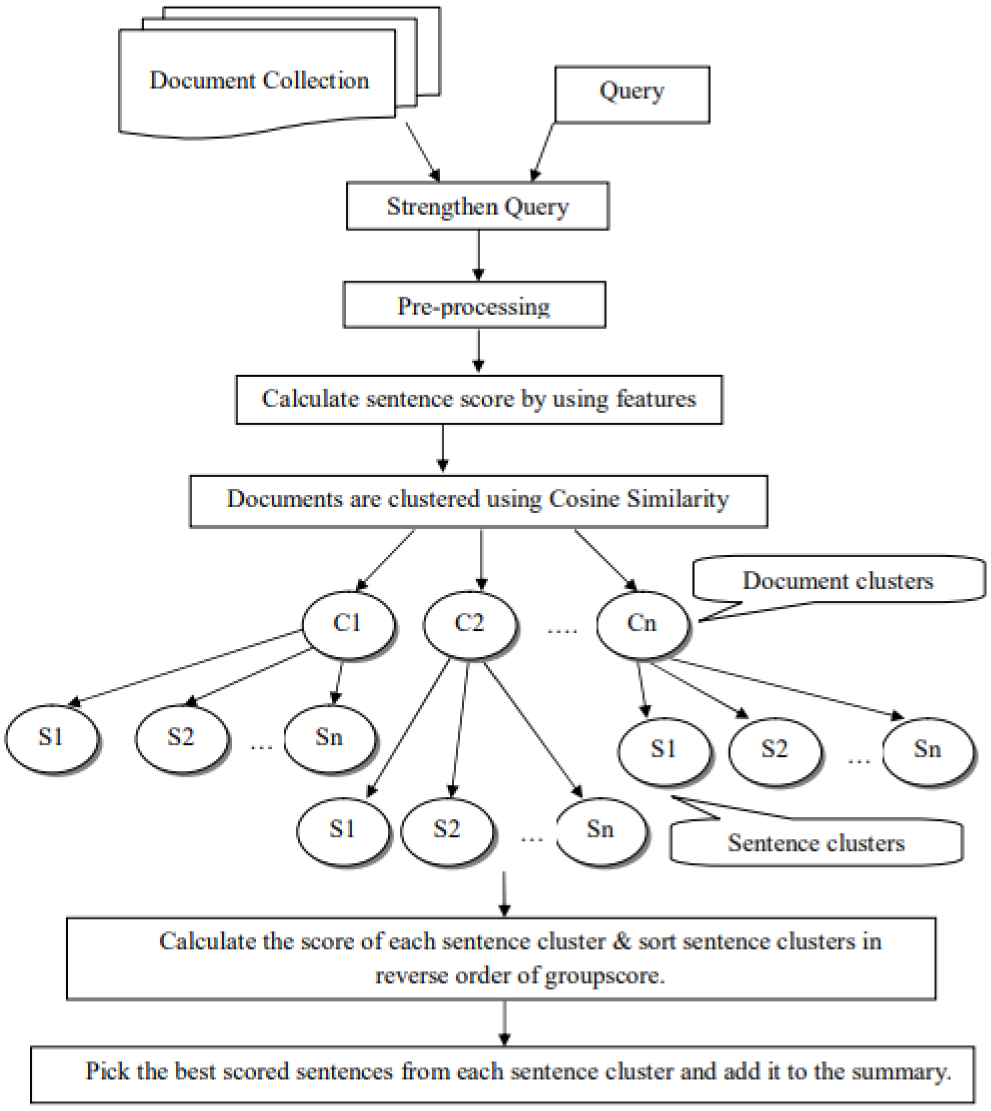
6. Then from every document cluster, sentences are clustered based on their similarity values.

7. Calculate the score of each group (sentence cluster).

8. Sort sentence clusters, in reverse order of group score.

9. Pick the best scored sentences from each sentence cluster and add it to the summary.

10. We have decided the number of sentences to be selected depending on sentence clusters size.



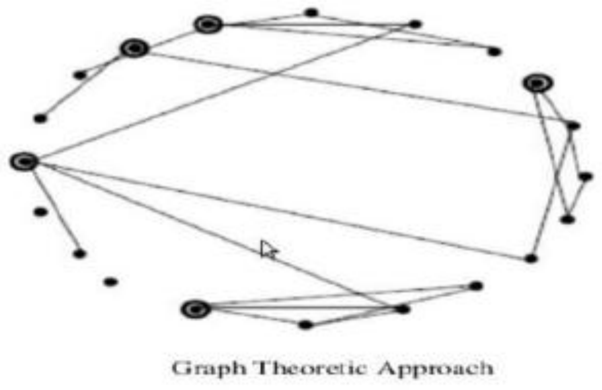
Document Collection
Query
Strengthen Query
Pre-processing
Calculate sentence score by using features
Documents are clustered using Cosine Similarity
Document clusters
C1
C2
Cn
S1
S2
Sn
Sentence clusters
Calculate the score of each sentence cluster & sort sentence clusters in reverse order of groupscore.
Pick the best scored sentences from each sentence cluster and add it to the summary.

3.3.2.1 Figure for cluster based method

3.3.3 Graph theoretic approach

In this technique, there is a node for every sentence . Two sentences are connected with an edge if the two sentences share some common words, in other words, their similarity is above some threshold. This representation gives two results :The partitions contained in the graph (that is those sub-graphs that are unconnected to the other sub graphs), form distinct topics covered in the documents. The second result by the graph-theoretic method is the identification of the important sentences in the document. The nodes with high cardinality (number of edges connected to that node), are the important sentences in the partition, and hence carry higher preference to be included in the summary. Figure shows an example graph for a document. It can be seen that there are about 3-4 topics in the document; the nodes that are encircled can be seen to be informative sentences in the document, since they share information with many other sentences in the document. The graph theoretic method may also be adapted easily for visualization of inter and intra document similarity.

- To build a graph model, from the graph, identify vertices which describe given task as text units
- Draw edges between text units on basis of common match and compute relationship for each edge .
- We may have weighted or un-weighted edges as well as directed or un-directed graphs.
- In the model, apply rank algorithm and repeat until convergence takes place.
- In this graph method, all vertices will be sorted on score of respectively vertex based on last mark of each vertex. And
- Finally, scores will be used for selection purpose.

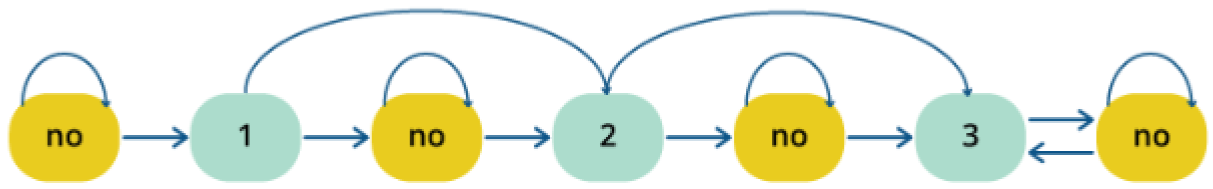


3.3.3.1 Figure of Graph Theoretic approch

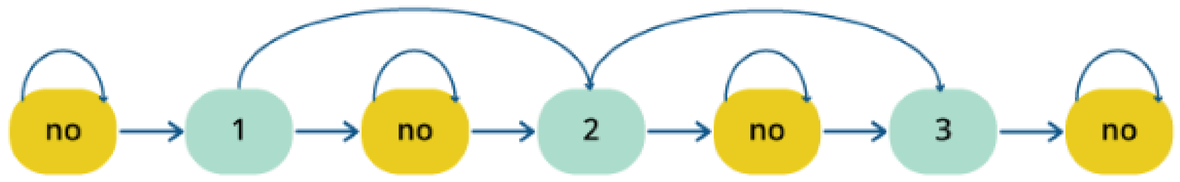
3.3.4 Machine Learning approach

In this method, the training dataset is used for reference and the summarization process is modeled as a classification problem: sentences are classified as summary sentences and non-summary sentences based on the features that they possess. The classification probabilities are learnt statistically from the training data, using Bayes' rule:

$$P\,(s \in < S \mid F1, F2, \ldots, FN) = P\,(F1, F2, \ldots, FN \mid s \in S) \;*\; P\,(s \in S) \,/\, P\,(F1, F2, \ldots, FN)$$



3.3.4.1 Summary Extraction Markov Model to Extract 2 Lead Sentences



3.3.4.2 Summary Extraction Markov Model to Extract 3 Sentences

3.3.5 Text summarization with neural networks

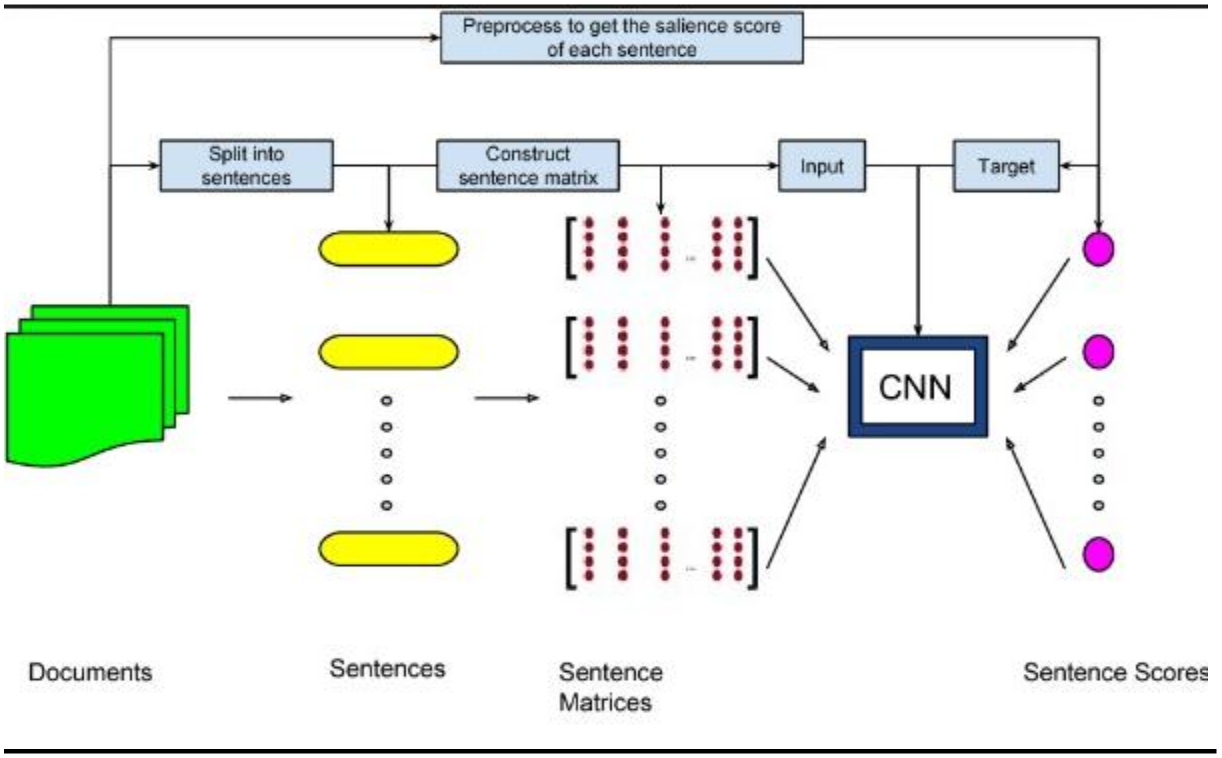
In this method, each document is converted into a list of sentences. Each sentence is represented as a vector [f1,f2,...,f7], composed of 7 features.

Seven Features of a Document

1) f1 Paragraph follows title
2) f2 Paragraph location in document
3) f3 Sentence location in paragraph
4) f4 First sentence in paragraph
5) f5 Sentence length
6) f6 Number of thematic words in the sentence
7) f7 Number of title words in the sentence.

The first phase of the process involves training the neural networks to learn the types of sentences that should be included in the summary. Once the network has learned the features that must exist in summary sentences, we need to discover the trends and relationships among the features that are inherent in the majority of sentences. This is accomplished by the feature fusion phase, which consists of two steps:

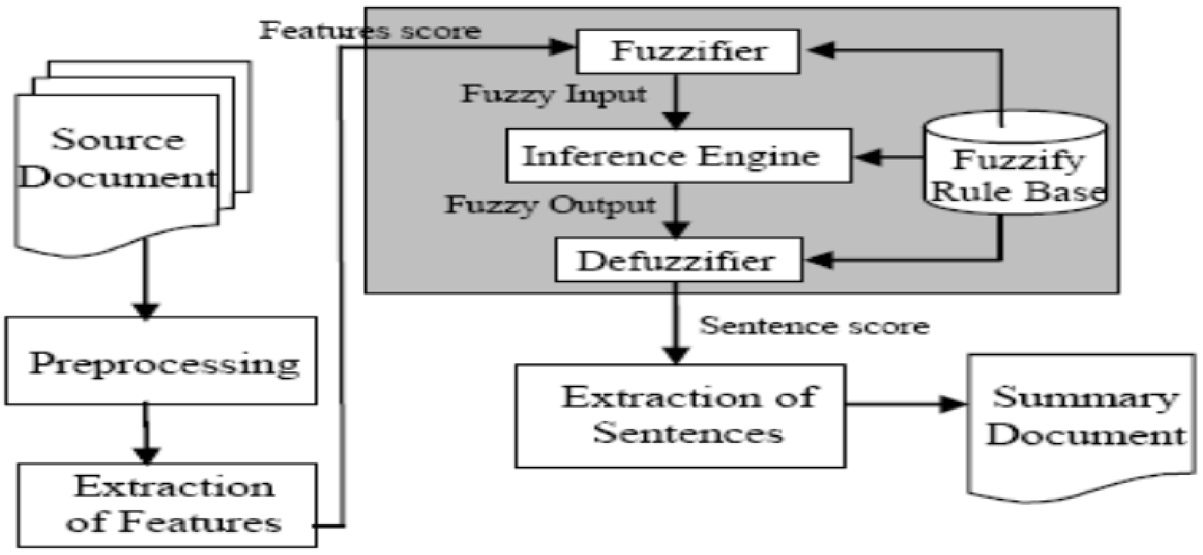
1) eliminating uncommon features; and
2) collapsing the effects of common features.



3.3.5.1 Diagram for Text Summarization with neural Networks

3.3.6 Automatic text summarization based on fuzzy logic

This method considers each characteristic of a text such as sentence length, similarity to little, similarity to key word and etc. as the input of fuzzy system .Then, it enters all the rules needed for summarization, in the knowledge base of system. Afterward, a value from zero to one is obtained for each sentence in the output based on sentence characteristics and the available rules in the knowledge base. The obtained value in the output determines the degree of the importance of the sentence in the final summary. The input membership function for each feature is divided into three membership functions which are composed of insignificant values (low L), very low (VL), medium (M), significant values (High h) and very high (VH). The important sentences are extracted using IF-THEN rules according to the feature criteria.



3.3.6.1 Diagram of Automatic text summarization based on fuzzy logic

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 IFTHEN 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.

- There are some libraries/packages present. With help of that libraries/packages Extractive text summarization can be performed.
  - Beautiful Soup
  - Urllib
  - nltk (Natural Language ToolKit)
  - sumy
  - Gensim
- In this system we have used sumy. This module is widely used in automatic text summarization which is part of the field of natural language processing.

- There are some methods to do text summarization using sumy library.

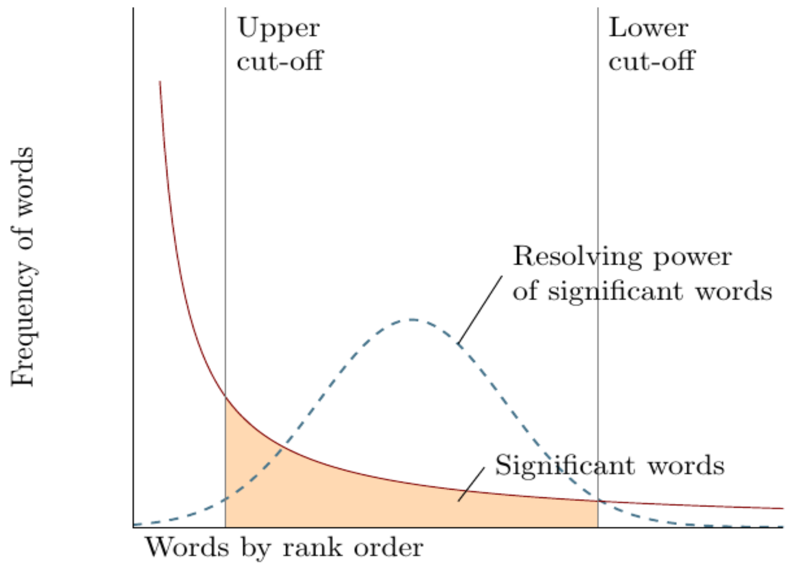
3.4 Sumy summarization methods:

- Luhn– heurestic method
- LexRank Algorithm
- LSA Algorithm
- TextRank Algorithm
- SumBasic
- KL-Sum

3.4.1 Luhn– heurestic method

Luhn's algorithm was first proposed in a 1958 paper written by Hans Peter Luhn. As stated before, Luhn's algorithm is based on the fact that humans are creatures of habit and will repeat keywords throughout a document. More importantly, he believes that the keywords an author
uses is well defined and represents a single concept or notion. Even if an author tries to use reasonable synonyms for his or her keyword, they will eventually run out and fall back to using the best word that defines the notion, which will be the keyword that is repeated the most.

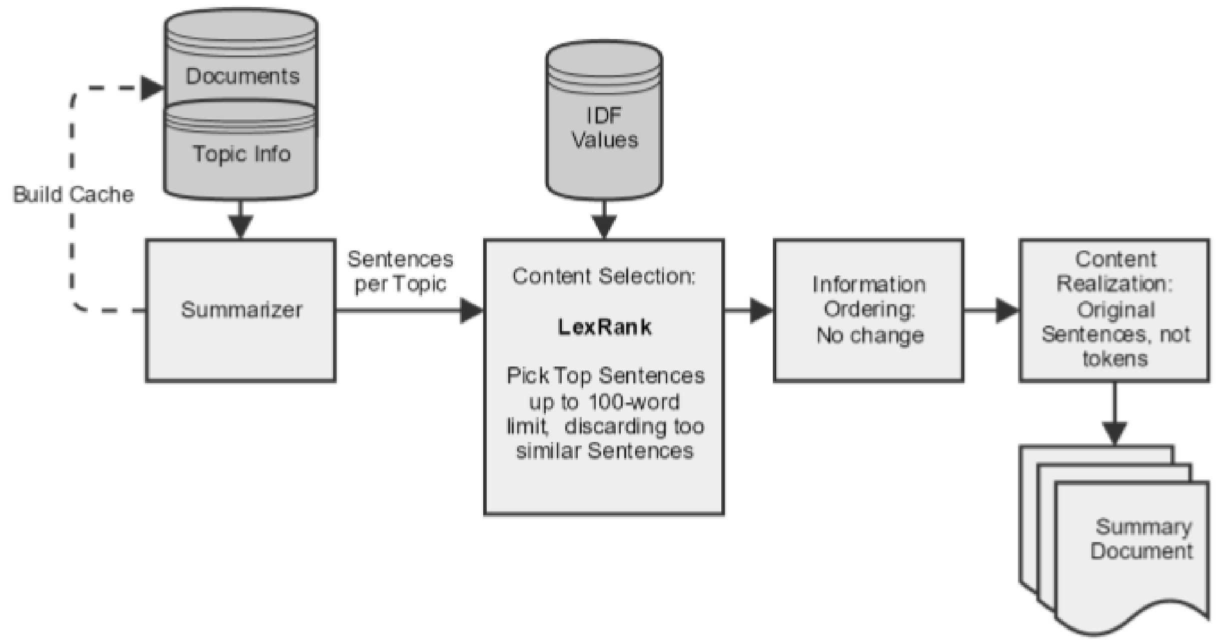
Running with the notion that an author will be repetitive with using a limited number of keywords to convey meaning, we can begin to rank sentences based on keyword frequency and proximity within a sentence. To determine sentence weight, we first look for significant words in a sentence, then take a subset of words in the sentence with the first and last word in the subset being a significant word. A subset is closed when four or five insignificant words are present before the next use of a significant word. Within the subset, we now count the number of times the significant word is present then divide by the number of total words in the subset. This number will be the weight given to that sentence. If a given sentence is long enough to contain multiple such subsets of significant words, we simply take the higher subset score as the weight of the sentence. To generate the auto-extraction, we only need to take the highest x sentences where x is a user defined number of sentences for summary length and putting the sentences back in the order they first appear. Besides just taking the highest rated sentences, it is also possible to break the text down into paragraphs and take the highest y sentences of each paragraph where y is x divided by the number of paragraphs. We could use this system since paragraphs are logical divisions of information specified by the author of the text.



3.4.1.1 Diagram for Luhn Algorithm

3.4.2 LexRank Algorithm

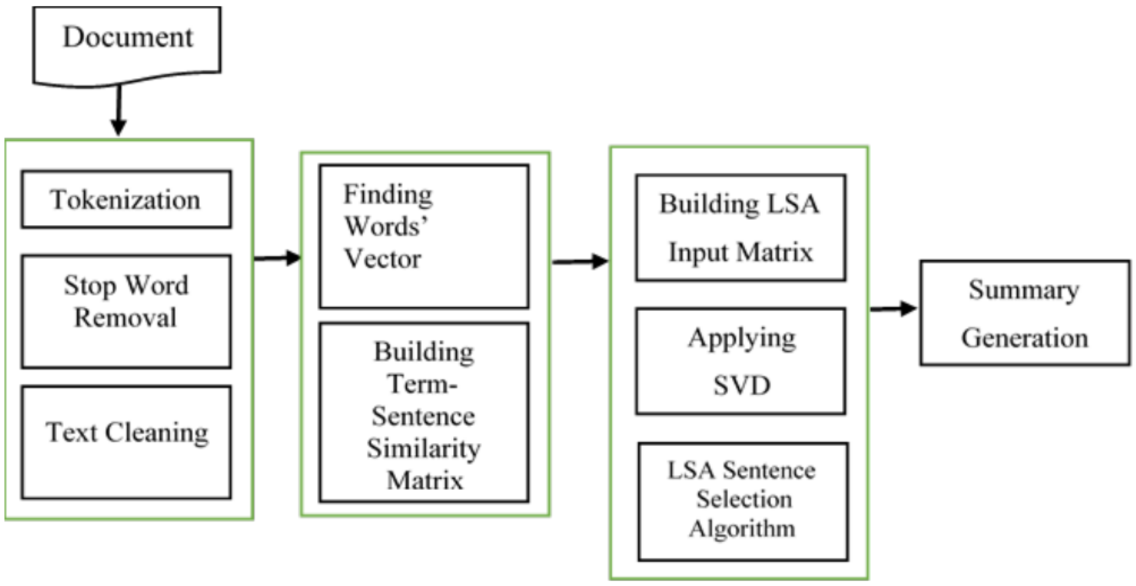
LexRank is graph-based lexical centrality as salience method for calculating relative importance of textual units. In extractive Text Summarization (TS), it is find most suitable sentence or concept from document. Salience is a term which check the present of important words or topic in sentence with similarity to a centroid pseudo sentence. In LexRank, for Calculate the importance of the sentence based on the concept of centrality of the eigenvector in a graphic representation of sentences. Also in this model, Intra-sentences Cosine similarity is used for adjacency matrix in sentence representation for connectivity matrix. LexRank using heuristic approach to build summary via inserting sentences in rank order and it discards sentences which are too similar and placed in summary.


Documents
Topic Info
Build Cache
IDF Values
Summarizer
Sentences per Topic
Content Selection:
LexRank
Pick Top Sentences up to 100-word limit, discarding too similar Sentences
Information Ordering: No change
Content Realization: Original Sentences, not tokens
Summary Document

3.4.2.1 Diagram for Lexrank Algorithm

3.4.3 LSA Algorithm

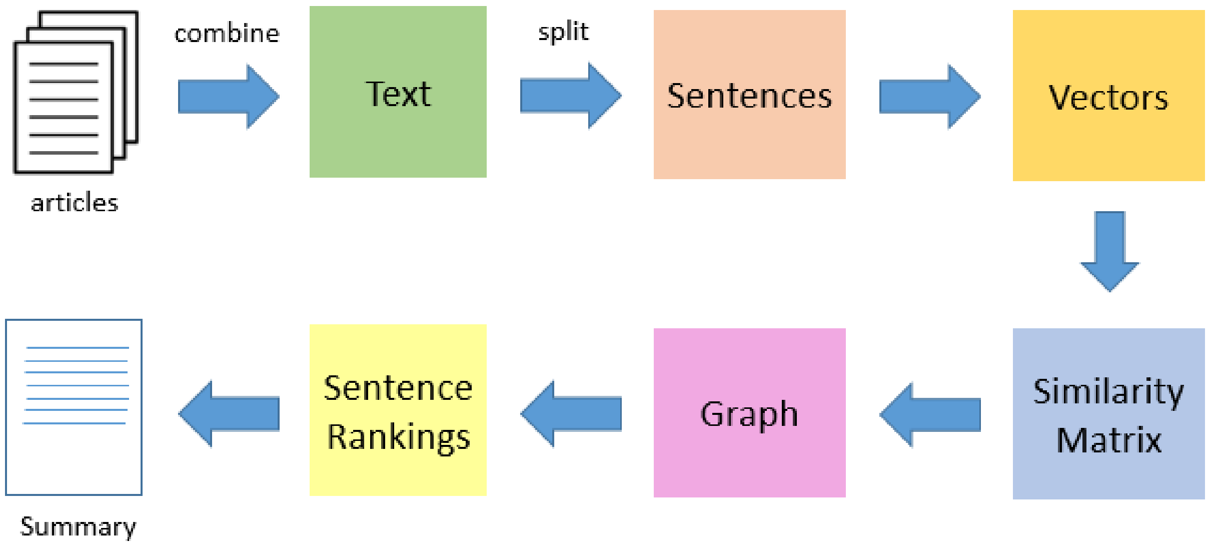
LSA where in sentence taking out summarizer evaluates a set of summary sentences based on its prediction similarity to that of the full sentences set on the top latent singular vector. There are few steps required to build summary with the help of Latent semantic analysis. First step is applying singular value decomposition (SVD) to document. Second is choosing sentence by its capacity of projection similarity. And finally, LSA-based forward sentence selection algorithm is applied to build summary. Here they have used centroid-based MEAD and MMR (Maximal Marginal Relevance) methods.



3.4.3.1 Diagram for LSA Algorithm

3.4.4 TextRank Algorithm

TextRank is unsupervised approach. TextRank method on Graph based method which takes into consideration local vertex-specific information as well as full graph global statistics repeatedly for determining significance of vertex. Let's take a look at the flow of the TextRank algorithm that we will be following.



3.4.4.1 Diagram for TextRank Algorithm

- The first step would be to concatenate all the text contained in the articles.
- Then split the text into individual sentences.
- In the next step, we will find vector representation (word embeddings) for each and every sentence.
- Similarities between sentence vectors are then calculated and stored in a matrix.
- The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation.
- Finally, a certain number of top-ranked sentences form the final summary.

There exists some similarity between Page rank and TextRank algorithm:

- In place of web pages, we use sentences.
- Similarity between any two sentences is used as an equivalent to the web page transition probability.
- The similarity scores are stored in a square matrix, similar to the matrix M used for PageRank.

3.4.5 SumBasic

SumBasic is an algorithm that generate multi-document text summaries. Its design is motivated by the observation that words that occur frequently in the group of documents they occur more likely in human abstracts than words that occur less frequently. It works in following ways:

- It calculates the probability distribution over words w appearing in the input $p(w_i)$ for every I ; $p(w_i) = n/N$ where n is the number of times the word appeared in the input, and N is the total number of content word tokens in the input.
- For sentence S_j in input, assign a weight equal to the average probability of the words in the sentence.
- Pick the best scoring sentence that contains the highest probability word.
- For each word w_i in the sentence chosen at step 3, update their probability.
- If desired summary length is not generated, repeat a process from step 2

To understand deep about the algorithms here is the difference between the different algorithms.

❖ Differences between LexRank and TextRank

- ➢ TextRank was applied to summarization with frequency of sentences score, while LexRank combines the LexRank score with other features like sentence position and length.
- ➢ TextRank was used for single document summarization, while LexRank has been applied to multi-document summarization.
- ➢ TextRank uses a very similar measure based on the number of words two sentences have in common. While LexRank uses cosine similarity of TF-IDF vectors.

❖ Difference between LexRank and Luhn

- ➢ LexRank uses cosine similarity for ranking the sentences where Luhn uses clustering of sentences base on the portion of importance words in the sentence.
- ➢ Lex Rank represents the sentences and words using Graph, while Luhn uses clusters and chunks for representation of sentences.
- ➢ Both algorithm uses predefined Important sets

Chapter 4
Implementation

1.5 Implementation Environment

1. Django Framework

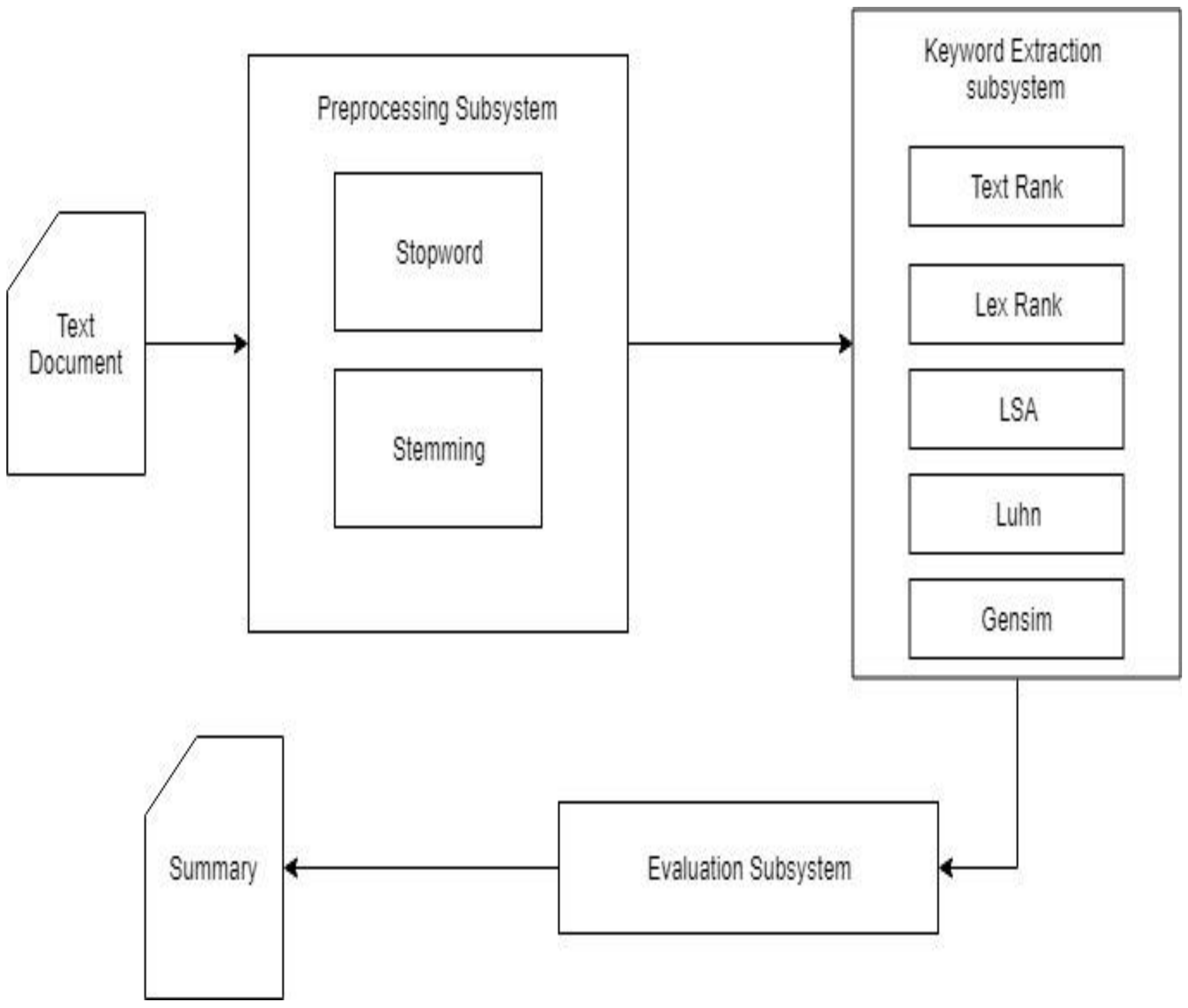
Django is a high-level Python Web framework that encourages rapid development And clean, pragmatic design. It assists in building and maintaining quality web applications. Django helps eliminate repetitive tasks making the development process easy. It aims to make each element of its stack independent of the others.

- Advantages of Django Framework:
  - ➢ Object Relational Mapping support
  - ➢ Multilingual support
  - ➢ Framework support
  - ➢ Administration GUI
  - ➢ Development Environment
  - ➢ Model-View-Template support

2. Django Rest Framework(DRF)

Django REST framework is a powerful and flexible toolkit for building Web APIs.
Some reasons we might want to use REST framework:

- ➢ Web browse able API
- ➢ Authentication policies
- ➢ Serialization for ORM & Non-ORM
- ➢ Regular function-based Views
- ➢ Extensive documentation
- ➢ Relationships and hyperlink APIs
- ➢ View sets and Routers



4.1.1 Figure of system working

4.2 Module Description

➢ Views

It has a Csrf_Exempt summary method which takes Json data as Input and return summary as Json Result. This module contains following two methods.

1. Summary Method:

It encodes the input Json data using UTF-8 decoder. Then it extracts the required parameters from decoded data. It finds type of data whether it is URL or
Simple text data. Then it uses the Scrapper module to fetch and clean the data which is present in the URL Page. Then it gets the appropriate summarizer object by calling the get_summarizer method. Now, using this summarizer it gets the summarize text for given input data.

2. Get_Summary Method:

This Methods is used to create the appropriate Summarizer Object. It takes algorithm name and the language for which user wants the summary. It initializes the following summarizer objects as per required algorithm.

- Text Rank Summarizer
- Lex Rank Summarizer
- Luhn Summarizer
- LSA Summarizer
- Gensim Summarizer
- NLTK Summarizer

This method gets the stop words for required language and does the stemming of the data. It uses the PlainTextParser to calculate the sentences and paragraphs of the document. At last it creates summarizer object with all parameters and returns that object.

➢ Urls

It contains all the paths and url for the Api classes and Api views. In this module all url patterns is created for class based views. This module is at Api level for neviagtion purpose.

➢ Scrapper

This module does the scrapping work. It Fetches the data from the given URL using Requests library of python. Then it parse the Html content using Beatifulsoup library using 'lxml' parser. It has following method for the cleaning of the Text data as well as Html Data.

1. Text_clean Method:

This method takes Text data as input and gives cleaned data for text summarization as output. In this method 're' library of python is used for cleaning of text data. This method removes all the spaces from the text data. It also removes any non-important special character from text data. It generates the paragraph list and sentences list after the cleaning is done. It returns the generated lists.

2. Html_clean Method:

This method takes Html data as input and returns clean data for summarization as output. At First, it creates Parsing Object using Beautiful library of python and 'lxml' parser. Then using Html Tag name and its attribute it finds all the article title and links for article description. Then it fetches the text data from that link which contains only paragraph text. Then it removes all the Html tags from that text. Then it removes all the extra spaces and special characters. At last it returns this cleaned text.

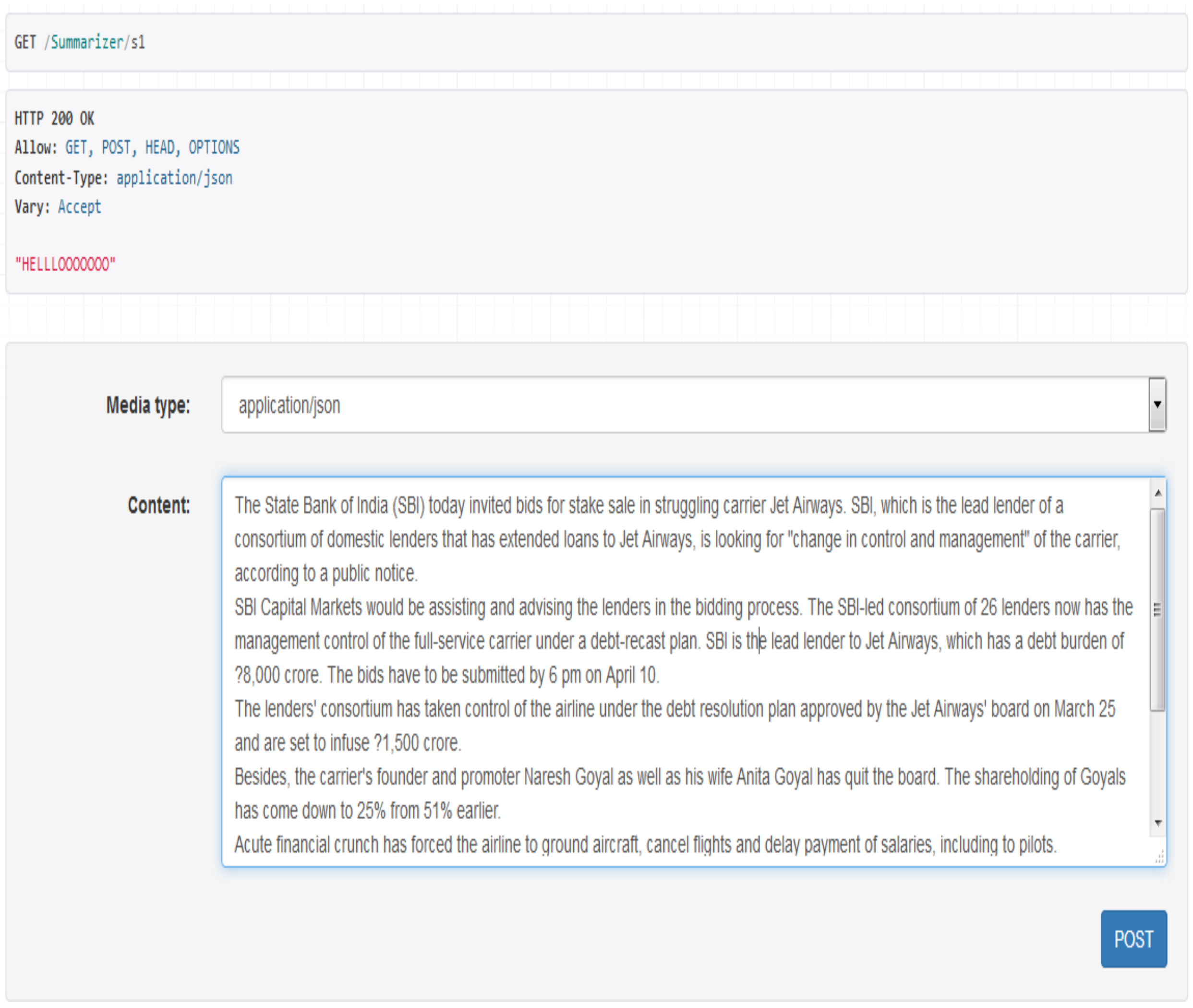
Chapter 5
Test Case Design and snapshots
Of screen layouts

Test Case	Test Case Scenario	Expected Output	Actual Output	Status (Pass/Fail)
TC01	Check Request Type	Request Type Should Be POST	Other Than POST type is not accepted	Pass
TC02	Check the module dependency	All Required library should be imported and Corpus should be downloaded	As Expected	Pass
TC03	Check Request Body Media Type	Input Media Type Should Be Json Otherwise Exception Should be Thrown	As Expected	Pass
TC04	Check All Required Parameters in Body are preset or not	Any one parameter is missing then Proper Exception Should be thrown	As Expected	Pass
TC05	Check whether datatype of Data parameter is valid or not	Data Type must be URL or String otherwise Throw Exception	As Expected	Pass
TC06	Check value of algo parameter	Algo value should be supported algorithm name	As Expected	Pass
TC07	Check whether URL is Valid or Not	Url Should Be valid otherwise Exception should be Thrown	As Excepted	Pass
TC08	Whether string data is decodable or not	String data should be decodable In UTF-8 format or not	AS Expected	Pass
TC09	Sentence parameter value should be valid	Sentences value should not be more than the sentences in text data otherwise Exception is thrown	As Expected	Pass

TC10	Check type of return value of Scrapper module	Return value should be list or string otherwise Exception should be thrown	As Expected	Pass
TC11	Check URL status for all the article on given page	All URL's of all article should be valid otherwise article should not be considered	As expected	Pass
TC12	Check the return value of clean method whether it contains only text or not	Return value should only contain text not the html tags	As Expected	Pass
TC13	Check the Type of Summarizer	Summarizer should be initialized with the required algo summarizer	As Expected	Pass
TC14	Check The parameters of HTTPResponse	All parameters of HTTPResponse should be set to their respective Value	As Expected	Pass
TC15	Check the Status of JSONResponse	JsonResponse should be initialized with appropriate Dictionary otherwise Exception should be thrown	As Expected	Pass

ScreenShots

Input Article



```
GET /Summarizer/s1
```

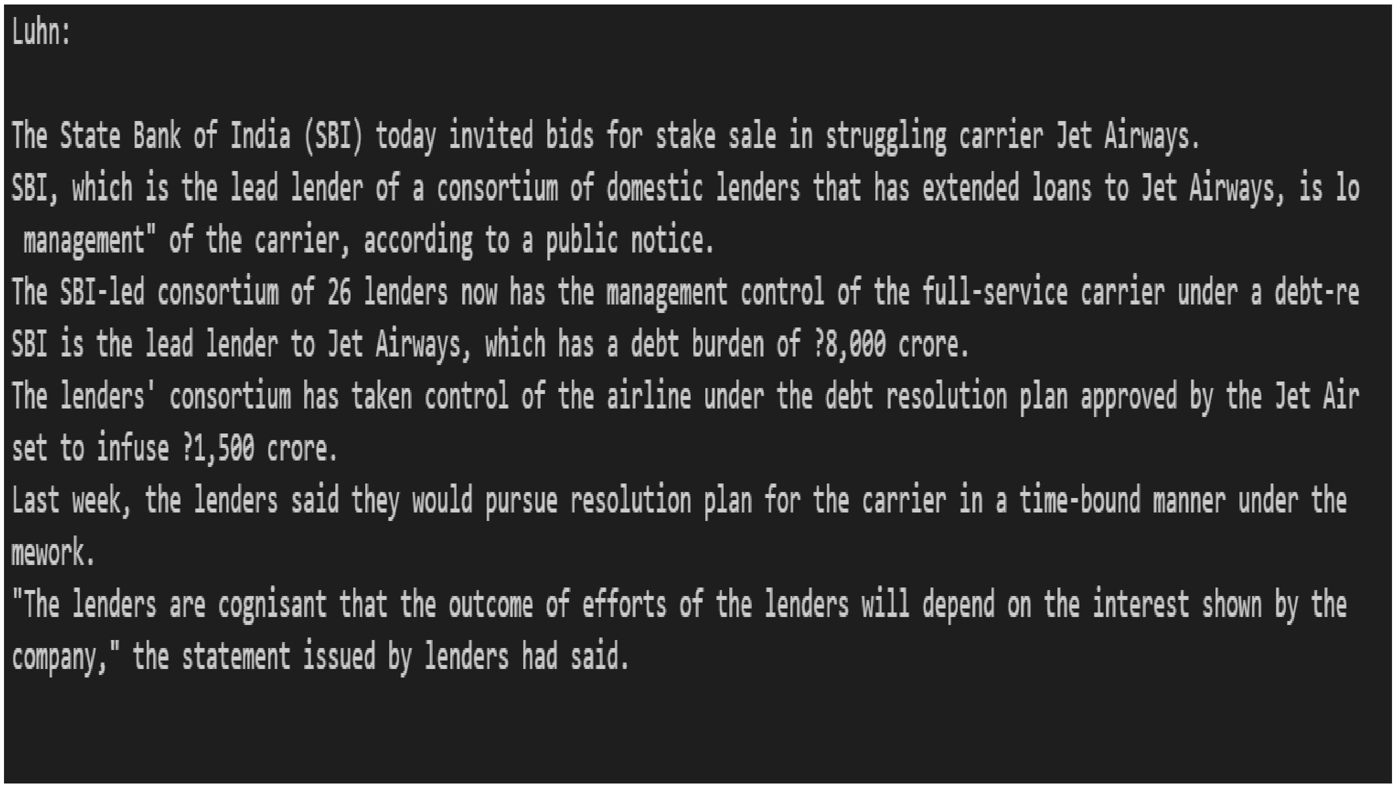
```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

"HELLOOOOOOO"
```

Media type: application/json

Content: The State Bank of India (SBI) today invited bids for stake sale in struggling carrier Jet Airways. SBI, which is the lead lender of a consortium of domestic lenders that has extended loans to Jet Airways, is looking for "change in control and management" of the carrier, according to a public notice.
SBI Capital Markets would be assisting and advising the lenders in the bidding process. The SBI-led consortium of 26 lenders now has the management control of the full-service carrier under a debt-recast plan. SBI is the lead lender to Jet Airways, which has a debt burden of ?8,000 crore. The bids have to be submitted by 6 pm on April 10.
The lenders' consortium has taken control of the airline under the debt resolution plan approved by the Jet Airways' board on March 25 and are set to infuse ?1,500 crore.
Besides, the carrier's founder and promoter Naresh Goyal as well as his wife Anita Goyal has quit the board. The shareholding of Goyals has come down to 25% from 51% earlier.
Acute financial crunch has forced the airline to ground aircraft, cancel flights and delay payment of salaries, including to pilots.

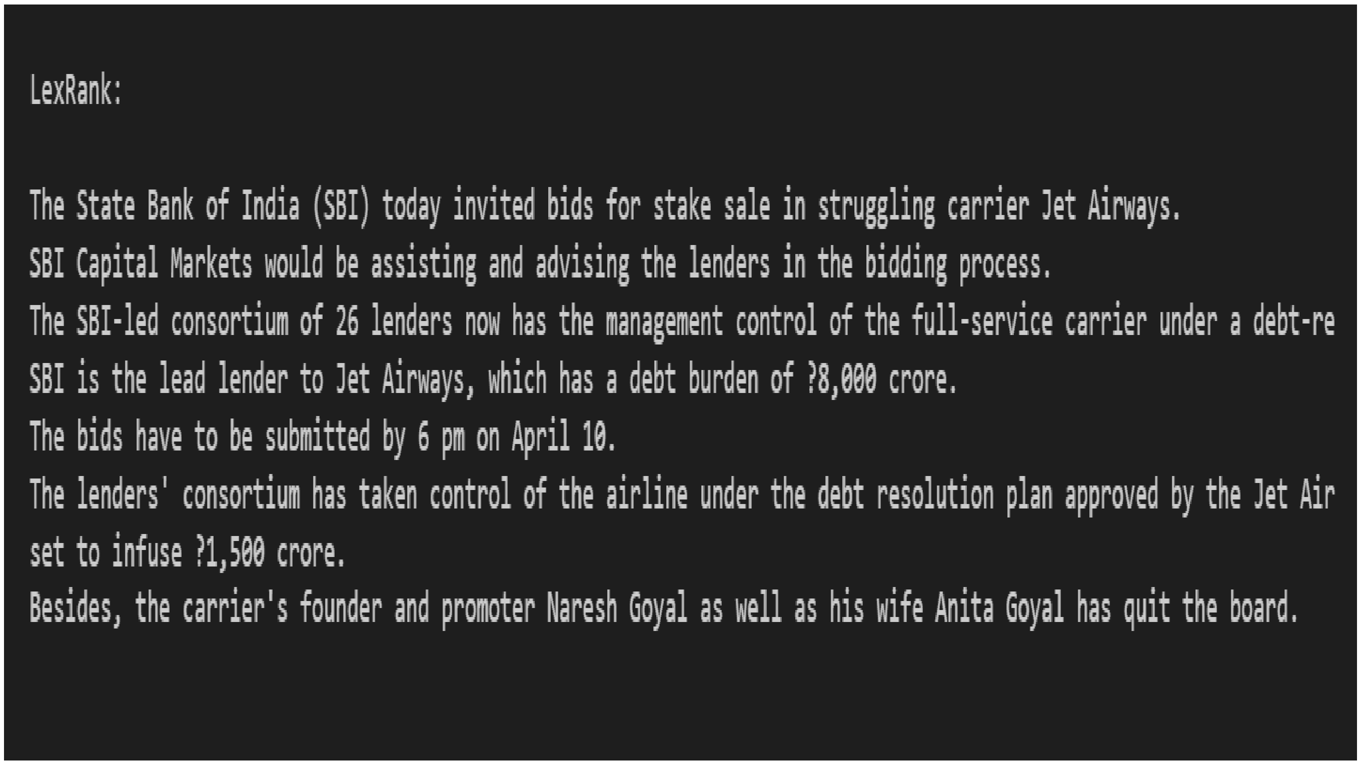
Output using Luhn algorithm



```
Luhn:


The State Bank of India (SBI) today invited bids for stake sale in struggling carrier Jet Airways.
SBI, which is the lead lender of a consortium of domestic lenders that has extended loans to Jet Airways, is lo
 management" of the carrier, according to a public notice.
The SBI-led consortium of 26 lenders now has the management control of the full-service carrier under a debt-re
SBI is the lead lender to Jet Airways, which has a debt burden of ?8,000 crore.
The lenders' consortium has taken control of the airline under the debt resolution plan approved by the Jet Air
set to infuse ?1,500 crore.
Last week, the lenders said they would pursue resolution plan for the carrier in a time-bound manner under the
mework.
"The lenders are cognisant that the outcome of efforts of the lenders will depend on the interest shown by the
company," the statement issued by lenders had said.
```

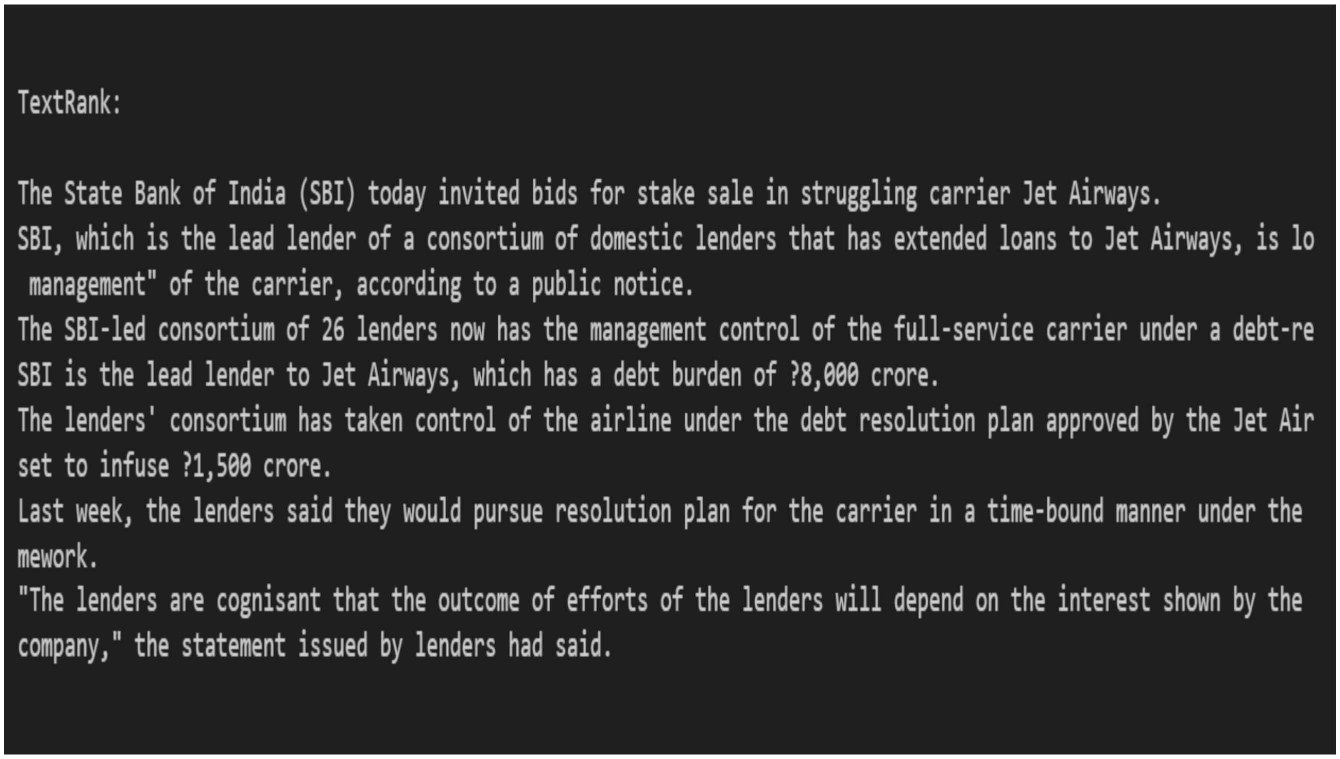
Output using Lex Rank algorithm



```
LexRank:

The State Bank of India (SBI) today invited bids for stake sale in struggling carrier Jet Airways.
SBI Capital Markets would be assisting and advising the lenders in the bidding process.
The SBI-led consortium of 26 lenders now has the management control of the full-service carrier under a debt-re
SBI is the lead lender to Jet Airways, which has a debt burden of ?8,000 crore.
The bids have to be submitted by 6 pm on April 10.
The lenders' consortium has taken control of the airline under the debt resolution plan approved by the Jet Air
set to infuse ?1,500 crore.
Besides, the carrier's founder and promoter Naresh Goyal as well as his wife Anita Goyal has quit the board.
```

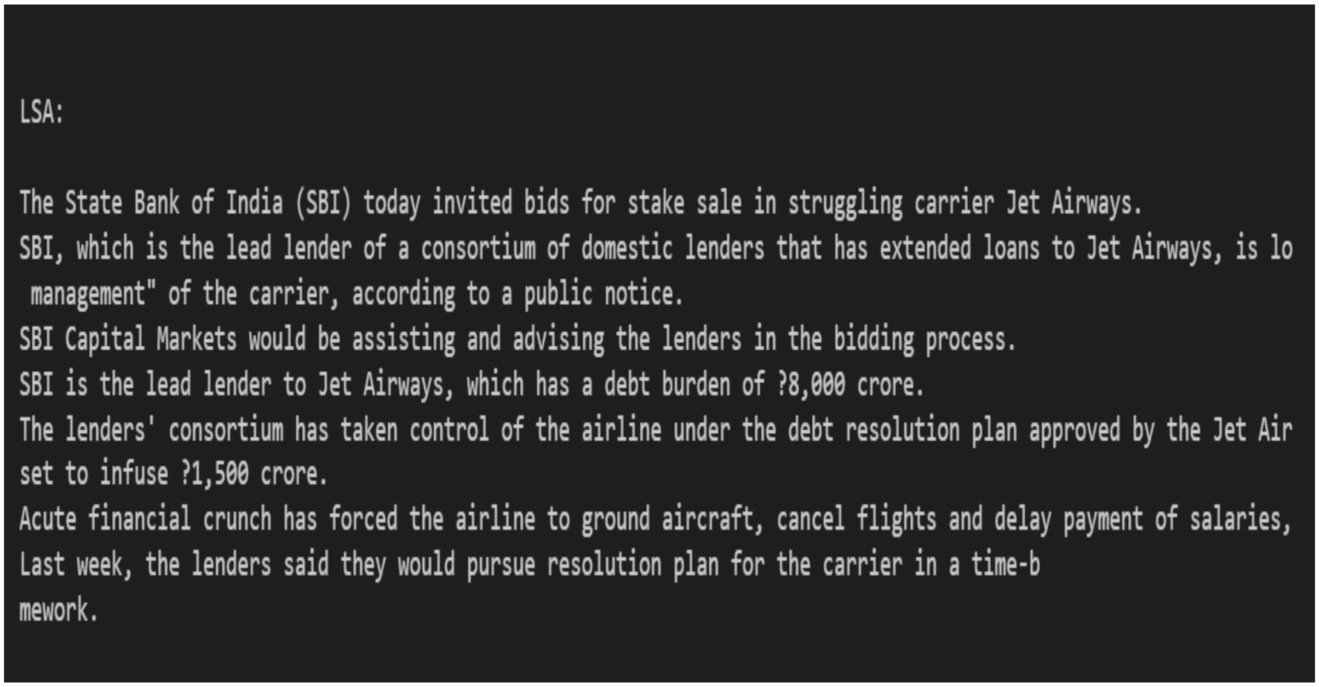
Output using Text Rank algorithm



```
TextRank:

The State Bank of India (SBI) today invited bids for stake sale in struggling carrier Jet Airways.
SBI, which is the lead lender of a consortium of domestic lenders that has extended loans to Jet Airways, is lo
 management" of the carrier, according to a public notice.
The SBI-led consortium of 26 lenders now has the management control of the full-service carrier under a debt-re
SBI is the lead lender to Jet Airways, which has a debt burden of ?8,000 crore.
The lenders' consortium has taken control of the airline under the debt resolution plan approved by the Jet Air
set to infuse ?1,500 crore.
Last week, the lenders said they would pursue resolution plan for the carrier in a time-bound manner under the
mework.
"The lenders are cognisant that the outcome of efforts of the lenders will depend on the interest shown by the
company," the statement issued by lenders had said.
```

Output using LSA algorithm



```
LSA:

The State Bank of India (SBI) today invited bids for stake sale in struggling carrier Jet Airways.
SBI, which is the lead lender of a consortium of domestic lenders that has extended loans to Jet Airways, is lo
 management" of the carrier, according to a public notice.
SBI Capital Markets would be assisting and advising the lenders in the bidding process.
SBI is the lead lender to Jet Airways, which has a debt burden of ?8,000 crore.
The lenders' consortium has taken control of the airline under the debt resolution plan approved by the Jet Air
set to infuse ?1,500 crore.
Acute financial crunch has forced the airline to ground aircraft, cancel flights and delay payment of salaries,
Last week, the lenders said they would pursue resolution plan for the carrier in a time-b
mework.
```

Chapter 6
Conclusion and Future Extensions

Conclusion

The document summarization problem is a very important problem due to its impact on the information retrieval methods as well as on the efficiency of the decision making processes, and particularly in the age of Big Data Analysis. Though a good kind of text summarization techniques and algorithms are developed there's a requirement for developing new approaches to supply precise and reliable document summaries that may tolerate variations in document characteristics.

Automatic text summarization is an old challenge but the current research direction diverts towards emerging trends in biomedicine, product review, education domains, emails and blogs. This is due to the fact that there is information overload in these areas, especially on the World Wide Web. Automated summarization is an important area in NLP (Natural Language Processing) research. It consists of automatically creating a summary of one text. The purpose of extractive document summarization is to automatically select a number of indicative sentences, passages, or paragraphs from the original document ..Here extractive method have been researched. Most summarization techniques are based on extractive methods. Abstractive method is similar to summaries made by humans. Abstractive summarization as of now requires heavy machinery for language generation and is difficult to replicate into the domain specific areas.

Future Research

In this section, we mention some of the possible future extensions of this research. In this thesis, we focused on summarization of news articles belonging to sports and technical domain. The techniques proposed here are adaptable across other domains.

One of the future plans may be to apply the topic-focused summarization framework to news articles or blogs and to extend the work in the machine leaning approaches. The work presented by the thesis can also be applicable to multi document summarization by using minimal extensions which can be our other future research topic.

The state of the art summarization systems are all extractive in nature, but the community is gradually progressing towards abstractive summarization. Although a complete abstractive summarization would require deeper natural language understanding and processing, a hybrid or shallow abstractive summarization can be achieved through sentence compression and textual entailment techniques. Textual entailment helps in detecting shorter versions of text that entail with same meaning as original text. With textual entailment we can produce more concise and shorter summaries. Research in summarization continues to enhance the diversity and information richness, and strive to produce coherent and focused answers to users information need.

Bibliography

References:

- https://www.kdnuggets.com/2019/01/approaches-text-summarization-overview.html
- https://en.wikipedia.org/wiki/Automatic_summarization#TextRank_and_LexRank
- http://calvinfo.blogspot.com/2010/07/luhns-auto-abstract-algorithm.html
- https://www.kdnuggets.com/2017/12/general-approach-preprocessing-text-data.html
- https://medium.com/sciforce/towards-automatic-text-summarization-extractive-methods-e8439cd54715
- https://www.researchgate.net/publication/228619779_A_Survey_of_Text_Summarization_Extractive_Techniques
- https://dzone.com/articles/create-a-simple-api-using-django-rest-framework-in
- http://mccormickml.com/2016/03/25/lsa-for-text-classification-tutorial/
- https://www.django-rest-framework.org/tutorial/2-requests-and-responses/#tutorial-2-requests-and-responses
- https://stackabuse.com/text-summarization-with-nltk-in-python/
- https://towardsdatascience.com/write-a-simple-summarizer-in-python-e9ca6138a08e
- https://medium.com/jatana/unsupervised-text-summarization-using-sentence-embeddings-adb15ce83db1