A

Minor Project Report

On

Exploring Different Techniques & Applications of Text Summarization

Submitted for partial fulfillment for the degree of

Bachelor of Technology
(Information Technology)
in the
Department of Information Technology
By

Bhavya Joshi - 179302043

Pawan Kartik - 179302044

Under the Guidance Of
Mr. Virender

June – 2020

SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY
MANIPAL UNIVERSITY JAIPUR

MANIPAL UNIVERSITY
JAIPUR

INSPIRED BY LIFE

CERTIFICATE

Date: 17th June 20

This is to certify that the project titled **EXPLORING DIFFERENT TECHNIQUES & APPLICATIONS OF TEXT SUMMARIZATION** is a record of the bonafide work done by **BHAVYA JOSHI** (179302043) and **PAWAN KARTIK** (179302044) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B. Tech) in **(Information Technology)** of Manipal University Jaipur, during the academic year 2019-20.

Mr. Virender
Project Guide, Dept. of Information Technology
Manipal University Jaipur

Dr. Pankaj Vyas
HOD, Dept. of Information Technology
Manipal University Jaipur

ABSTRACT

With more and more people having access to the internet, lots of information is being created and shared online. This gives us the luxury of having it just a click away from consumption. However, not all of this information is filtered and cleared from the noise. With lots of unimportant events and details, the consumption of this information becomes difficult, a degree that varies from person to person.

One such example is news articles. A person does not need to go through pages of articles for a given topic to understand the gist; a mere summary is more than sufficient in many cases. This has given rise to many apps that crunch through hundreds of articles to generate personalized feed of summaries that a user can go through. Another such example is social media platforms. These platforms can crunch through thousands of posts for a given topic, understand the content that overlaps and then summarize this content. In fact, text summarization can also be used to some extent to answer user queries directly in search results, something that search engines have been doing lately. As more information is shared and consumed, text summarization becomes more relevant.

This project aims to explore different techniques of text summarization and evaluate them on different parameters such as extent of compression/summarization, retention of meaning/gist, grammatical errors made and so on.

The 2 main categories of text summarization were extractive and abstractive. As the names themselves suggest, extractive emphasizes on calculating weights of sentences and picking (top k sentences) them for the summary while abstractive emphasizes on rewriting the sentences to generate the summary.

Extractive method suffers loss in meaning to some extent as the connections between sentences are lost when picking them while abstractive method requires lots of effort in training the model and trying to avoid grammatical and semantic mistakes as sentences are often rewritten. Abstractive is language dependent while extractive can be scaled to certain languages as the core idea remains the same.

LIST OF FIGURES

LIST OF TABLES

Contents

INTRODUCTION

1.1 Overview

As more information is being shared online, text summarization becomes extremely relevant. Of the most cited works in this field dates back to 1958 when Luhn proposed that frequency of words can be used as a statistical measure in this process which still holds true for certain methods. [1]

Consumption of information becomes a costly and time consuming process as the information grows in size and with the presence of irrelevant material or noise. Text summarization can be used as a technique to filter them out. Manual text summarization works best as the meaning of the text can be retained as required while grammatical errors can be avoided. However, this is a time consuming process with varying results. Another option is to use automatic text summarization. Computers can be equipped with algorithms to generate summaries for the provided content. However, the results might vary depending on the content and the algorithm used for this process. Different algorithms give out different results for which the advantages and disadvantages are discussed below.

Automatic text summarization is widely used in different products and services which in return affects the user's experience while engaging with products and services.

1.2 Applications

Notable social media platforms use this process to generate summaries for posts that are grouped based on the content called topics. These topics are used to engage users online. Google's home feed for example generates summaries based on the user's preferences. Search engines today directly answer the provided query rather than just providing links. Text is extracted from ranked and credible websites and the summary is generated for this text which is returned as an answer to the query. The same concept is application for voice based assistants while answering user's queries.

1.3 Advantages & Disadvantages

Table 1.1: Advantages & Disadvantages of using Automatic Text Summarization

	Advantages	**Disadvantages**
1.	Time saving process: Computers are noticeably faster than humans and are capable of generating summaries faster.	Might miss out certain sentences affecting the summary's meaning: Certain sentences that contribute to the summary might be omitted which in return might affect the generated summary.
2.	Scalable: Automatic text summarization can be scaled to different languages with adoption of a proper algorithm whereas humans are limited by the extent of their expertise in a particular language.	Efforts put into training the models might not exactly meet the required standards: Neural Network based models require large resources and time to train. The end results might not exactly meet the required standards or the level of manual text summarization.
3.	Wide usage Automatic text summarization can be used in different fields as discussed in the	Grammatical mistakes - abstractive algorithms are prone to grammatical mistakes:

	overview, thereby enhancing the user’s experience while engaging with a product or a service.	Abstractive methods rewrite certain portions of sentences to generate the summary. There is a chance that these sentences might contain grammatical errors affecting the overall readability.

1.4 Objectives

The main objectives of this project are -

1. Explore different techniques of text summarization: There exist many techniques for automatic text summarization. One of our objectives is to explore and implement widely used such techniques.
2. Compare the generated summaries: Summaries generated vary by algorithm. Goal is to compare different summaries based on parameters like overall meaning retention, time taken to generate the summary, system resources used and to what extent the text was compressed.
3. Identify the optimal parameters (for ex, k in extractive text summarization) for best summary: What optimal values should be used for the parameters to obtain the summaries of required standards and how do the summaries vary as these parameters are fine-tuned?
4. Identify the advantages and disadvantages of different algorithms based on different parameters: Each algorithm is associated with certain advantages and disadvantages by nature. What makes an algorithm better suited than the other?
5. Identify or implement modifications (if possible) to scale an algorithm to different languages.
6. Identify the different applications of automatic text summarization.

Organization of Report

- We’ll first take an overview of the theory, concepts and technology
- Then we’ll look at detailed methodology of each algorithm
- After that the implementation of each method is shown
- Results are shown to compare the performance of all the methods
- Future Work is explored

CONCEPTUAL OVERVIEW

2.1 Concepts and Theory

2.1.1 Theory

Extractive

The main concept used in the extractive text summarization is to focus on important sentences. Each sentence is assigned a weight. The heavier the weight, the more it contributes to the summary. There are different techniques on assigning weights to the sentences. [2]

For example -

1. Word weighted frequency
    a. Word's frequency is calculated as - **freq(word)/max(freq)**

2. Occurrence of important words
    a. A sentence is assigned more weight if more number of important words occur in it. Important words can be picked by using certain filters which ignore stop words and other such common words and collapse adjacent occurring words.

The other concept used is TextRank. TextRank works by building a graph of sentences. Each sentence is considered a node and the connection between 2 sentences is called an edge. This edge is assigned a weight or a score that tells us to what extent 2 sentences are connected. A sentence that is connected or linked to more number of sentences is deemed important and picked up while generating the summary. [3]
Top k sentences are picked based on their scores of weights following the greedy approach.

Abstractive

This method is based on training deep learning models on data so as to help the model learn and understand language. Abstractive text summarization is a complex method that automatically helps the computer learn the grammar and semantics of a language and form new sentences to summarize given text.
These models are typically based on **Recurrent Neural Networks** [4]. RNNs are a special type of neural network where the output from the previous step is fed as input to the current step. In normal neural networks, all the inputs and outputs are independent of each other. We use RNNs here because to predict the next word in a sequence previous words and the context gained from them are required. The most important feature of RNN is Hidden state, which remembers some information about a sequence. RNN are said to have a memory which can remember previously learned information.

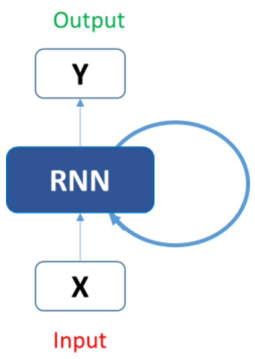


Fig. 2.1 RNN basic architecture

In an RNN, the current state is a function of current input and previous state, each successive input is called as a time step.:

$$h_t = f(h_{t-1}, x_t)$$

The current ht becomes ht-1 for the next time step. We can go as many time steps as the problem demands and combine the information from all the previous states. Once all the time steps are completed the final current state is used to calculate the output yt. The output is then compared to the actual output and the error is generated. The error is then back propagated to the network to update the weights(we shall go into the details of backpropagation in further sections) and the network is trained
The RNN used here is a special type called Long Short Term Memory network which overcomes the problem of long term dependency of RNN

2.1.2 Concepts

- Term frequency - document inverse frequency (tf-idf): It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. tf–idf is one of the most popular term-weighting schemes today.

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

  b = IDF(t) = log_e(Total number of documents / Number of documents with term t in it).

  Required tf-idf value = a * b.

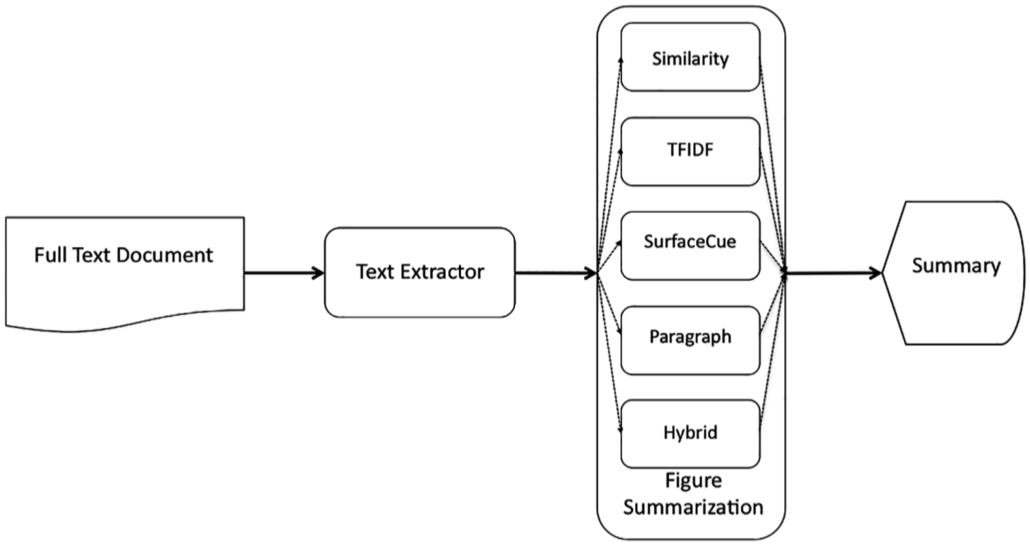


Fig. 2.2 - Text Summarization Process

- Vectors: Because there is no exact standard way for computers to compare strings or sentences, we convert them to vectors and then use vector based operators to compute various values. One such example is cosine similarity.

- Cosine Similarity: Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them.

  Similarity = (A.B) / (||A||.||B||) where A and B are vectors.

- Stop words: Words that do not contribute to any meaning in NLP operations are called stopwords and are removed as part of preprocessing.

- Sequence2Sequence Modeling: This is used for a special class of sequence modeling problems which use RNNs, where the input as well as the output is a sequence. These models involve 2 architecture called Encoder and Decoder. Sequence modeling is used for Speech Recognition and Natural Language Processing in order for computers to understand natural language and predict word sequences. [5]

- Encoder: This is a neural network which has the task of understanding the input sequence and create a smaller dimensional representation of it. Encoder processes the information at every timestep and captures the contextual information present in the input sequence. The hidden state (hi) and cell state (ci) of the last time step are used to initialize the decoder.

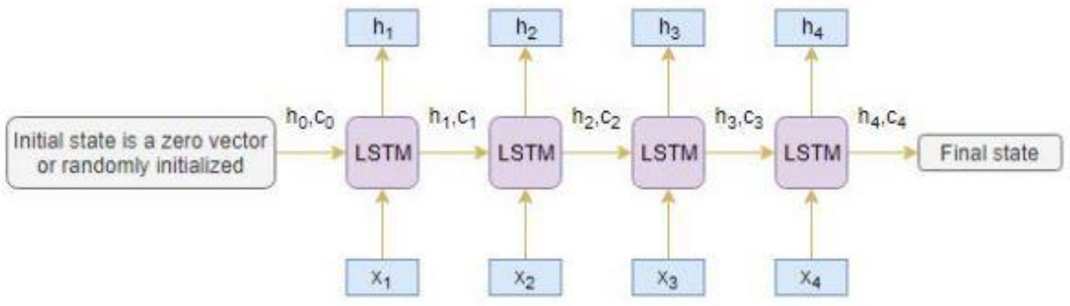


Fig. 2.3 Encoder LSTM

- Decoder: The encoder representation is forwarded to it and it generates a sequence of its own that represents the output. It basically reads the entire target sequence word-by-word and predicts the same sequence offset by one timestep. It predicts the next word in the sequence given the previous word. <start> and <end> are the special tokens which are added to the target sequence before feeding it into the decoder.

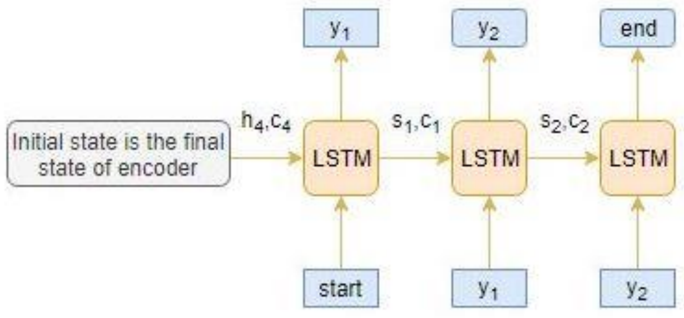


Fig. 2.4 Decoder LSTM

- Inference Process: This is the phase that comes after training of the model and is used to decode new source sequences for which the target is unknown.

- Attention Mechanism: This is used to focus on specific portions of the text to predict the next sequence. To implement attention mechanism, input is taken from each time step of the encoder – with weightage to the timesteps. The weightage depends on the importance of that time step for the decoder to optimally generate the next word in the sequence

2.2 Technologies Used

1. Nltk corpus - for stop words
2. Sklearn's pairwise metrics - for cosine similarity.
3. Tfidf vectorizer
4. Google Colab – for GPU training
5. Keras and Tensorflow
6. Bahdanau attention – to overcome problem of long sentences as performance of a basic encoder–decoder deteriorates rapidly as the length of an input sentence increases
7. Kaggle for data collection

METHODOLOGY

3.1 Extractive Method

There exist many approaches or techniques as part of the extractive method. We shall mainly focus on word weighted frequency and textrank.

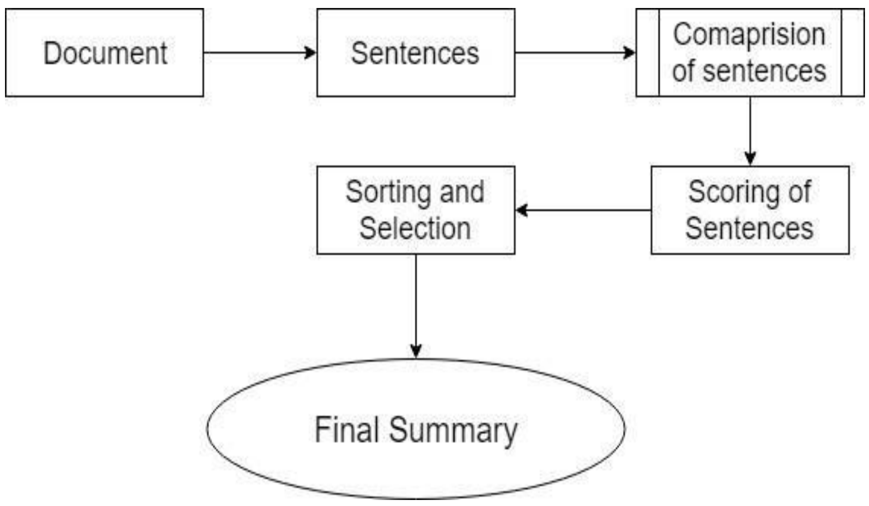


Fig. 3.1 - Flow chart for extractive text summarization

3.1.1 Word Weighted Frequency

The specified paragraph or text is first tokenized into sentences. For each sentence we then remove the stopwords and punctuation. Because this entire model is based on frequency, we need to keep track of each word's frequency and the max frequency. Once we're done with the preprocessing and the frequency calculation, for each sentence, we compute the weight. This is done by adding up the individual scores of all the words in each sentence where the score of a word is defined as - **freq(word)/max(freq).**

Once the scores are calculated, greedy approach is used to pick top k sentences (max k weights) to generate the summary. These sentences are reordered (re-sorted) in the order of their original appearance in the actual text. We also implemented this on Hindi text using Hindi stopwords and got good results.

3.1.2 Word Probability

Another method used is word probability where instead of dividing by **max(freq)**, we divide by **N**, that is the number of all words.

3.1.3 TextRank

TextRank method is based on PageRank, an algorithm that is usually used to rank web pages for search results. It builds a matrix of size n x n and these cells are filled with the probability that the user might visit that site, that is 1/(number of unique links in web page wi). The values are then updated in an interactive fashion.

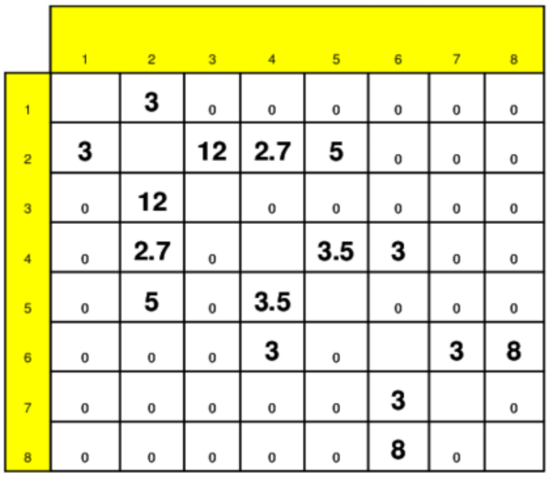
TextRank works similarly. It builds an adjacent matrix of size n x n where n is the number of sentences in the text. For each sentence ni (where i is an index), it is compared with nj (where i != j). This comparison is based on cosine similarity or some other technique through which 2 sentences can be compared.

Entire matrix is filled in such a manner. Then for each sentence ni (where i = 1, 2, 3 and so on), the entire row is added to compute the score for ni.

Top k sentences are picked based on this score through greedy search. These sentences form the required summary.

If the adjacency matrix is used, the time complexity increases to O(n^2) while the adjacency list reduces the complexity to O(v + e) while processing the graph.

TextRank is better at realizing the connection between sentences. If vectors are used it is easy to apply cosine similarity. A connection in the graph between two sentences also tells us that both are required for a meaningful context. Thus TextRank works well.



	1	2	3	4	5	6	7	8
1		3	0	0	0	0	0	0
2	3		12	2.7	5	0	0	0
3	0	12		0	0	0	0	0
4	0	2.7	0		3.5	3	0	0
5	0	5	0	3.5		0	0	0
6	0	0	0	3	0		3	8
7	0	0	0	0	0	3		0
8	0	0	0	0	0	8	0	

Fig. 3.2 - Adjacency Matrix
A typical example of an adjacency matrix with cells filled with values, depicting the connections between 2 nodes.

Table 3.1: Advantages & Disadvantages of using Extractive Methods

	Advantages	**Disadvantages**
1.	Easy to implement: Most of these methods use statistical techniques to compute the scores for the sentences. This makes it extremely easy to implement, debug and compare the models.	Sentences are simply picked up from the text. In some cases, they can be rewritten to further summarize it. E.g. - He picked up the object from the table and threw it out the window This sentence can be written as - He threw the object out of the window. Certain sentences that contribute to the summary might be omitted which in return might affect the generated summary.
2.	Doesn't require much resources: Because these models do not involve complex calculations like gradient descent	Disconnection between the sentences: As sentences are picked from the text and because there might be other sentences that are ignored,

	or back propagation (which are usually found in Neural Networks), calculations are fairly fast and easy to evaluate thereby not requiring high-end hardware to develop, train and run.	the entire summary generated might feel disconnected or less cohesive.
3.	Gives a fairly acceptable summary of larger texts: By adjusting the parameters such as k, the computed summaries are fairly of good standards in general (in terms of time invested and the system resources used).	Can cause some confusion if a sentence uses pronouns to refer to something in another sentence that was ignored.
4.	Doesn't require training: Most of the work done is in calculating the frequencies and sentence scores which are not heavy calculations.	Might give big summaries: To be meaningful more sentences might have to be picked as context would be important
5.	Can be used for other languages without many changes: The core model or idea remains the same and can be scaled to other languages with minor tweaks, for ex, tokenizing and stopwords	

3.2 Abstractive Method

Abstractive methods use deep learning models to predict word sequences. We’ve used Long Short Term Memory networks, a special type of Recurrent Neural Network. These are implemented using Encoder-Decoder architecture set up in 2 phases – training and inference.
To handle long sentences we’ve used Bahdanau attention layer which helps to focus on particular most important parts of the sentence. The methodology used to implement this deep learning model: -

- Two datasets were used:
  News Summary Dataset from Kaggle, contains 2 columns text and headlines
  Food Reviews Amazon from Kaggle, contains multiple columns, most important are Text and Summary
- The model was implemented on Google Colab using Keras. The attention layer file was downloaded from the Internet; it implements Bahdanau attention as written in a published paper [6]
- First, Reviews dataset was read (only top 100,000 rows). Duplicates and NA values were then dropped. The data was cleaned using typical text cleaning operations. Contraction mapping was done to expand English language contractions. (shouldn’t = should not)
- Text was cleaned by removing HTML tags, contractions were expanded, ‘s were removed, any parenthesis text were removed, stopwords were removed and short words were removed
- Same was done to clean the summaries present in both the datasets. Same text preprocessing was applied to the news dataset. Then the start and end tokens were added to the cleaned summary
- The text lengths are analyzed to get the maximum length of the text and summary
- The final data frame was created to contain that data only with text and summary below or equal to the set maximum
- The data was split into train and test set with 90% in train and 10% in test

- The text and summary word sequences were converted into integer sequences using tokenizers and top most common words
- The Encoder model consisting of three LSTM layers stacked on top of each other was made and the Decoder was initialized with encoder states
- A dense layer with softmax activation was added at the end
- This was the setting up of the training phase for both Encoder and Decoder
- The model was compiled using sparse categorical cross-entropy as the loss function
- Early stopping was used to stop training the model if validation loss started increasing
- For reviews the model stopped training at 14 epochs and for news only 5 epochs was used due to time and machine power constraints
- The encoder and decoder inference phase was set up, encoder inputs and outputs from training were supplied as inputs to inference [7]
- Decoder was set up in inference phase and to predict the next word in the sequence, initial states were set to the states from the previous time step
- An inference function to decode input sequence was created which creates target sequence until end token is reached or max summary length is reached
- Then the summaries were generated for the test set.

Table 3.2: Advantages & Disadvantages of using Abstractive Methods

	Advantages	**Disadvantages**
1.	Coherent: It produce coherent, less redundant and information rich summary. Sophisticated methods produce perfect summaries.	Requires Computational Power: To train our very simple model, it took hours. To train better models powerful machines are required.
2.	Powerful: This method basically allows for the machine to understand language. This has huge potential.	New Field: Methods are relatively new and thus there is a lack of documentation and experts
3.	Used in Real Time: Abstractive methods are used in various real life scenarios like Video Captioning, Headlines generation	Semantic Errors: Simple models generally are unable to grasp the language and this leads to repetition errors, context errors and short meaningless summaries

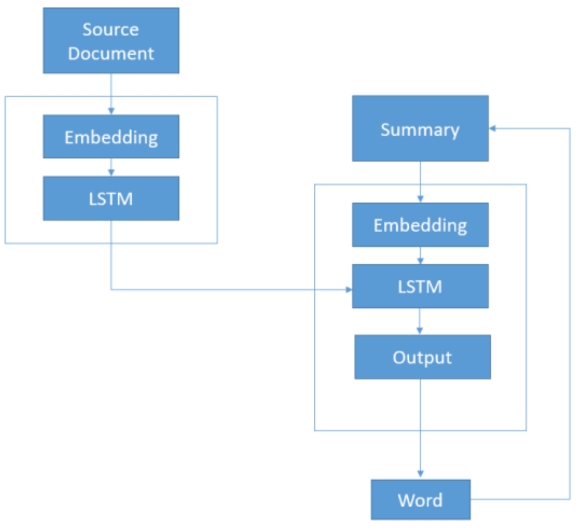


Fig. 3.3 – Abstractive Method

IMPLEMENTATION AND RESULTS

4.1 Modules & Implementation

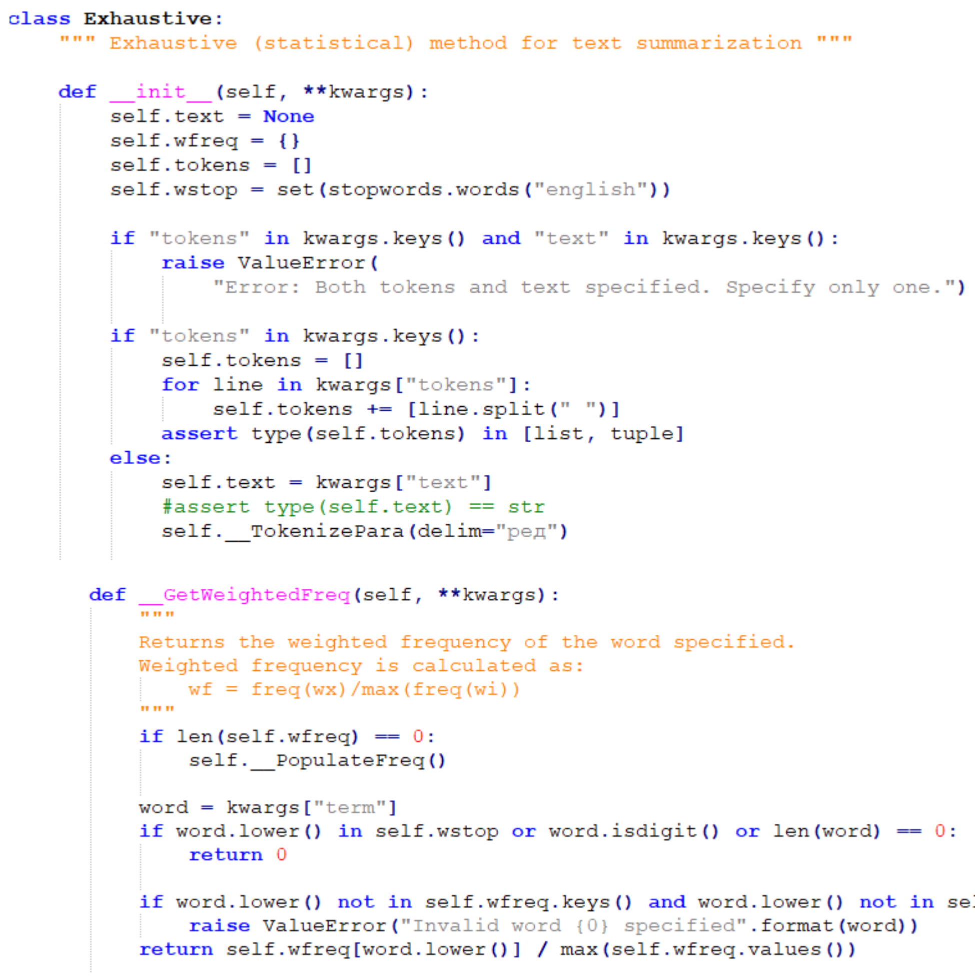
4.1.1 Extractive

The source code for the framework for text summarization (extractive based) can be viewed here.

Weighted Frequency

Modules and methods -

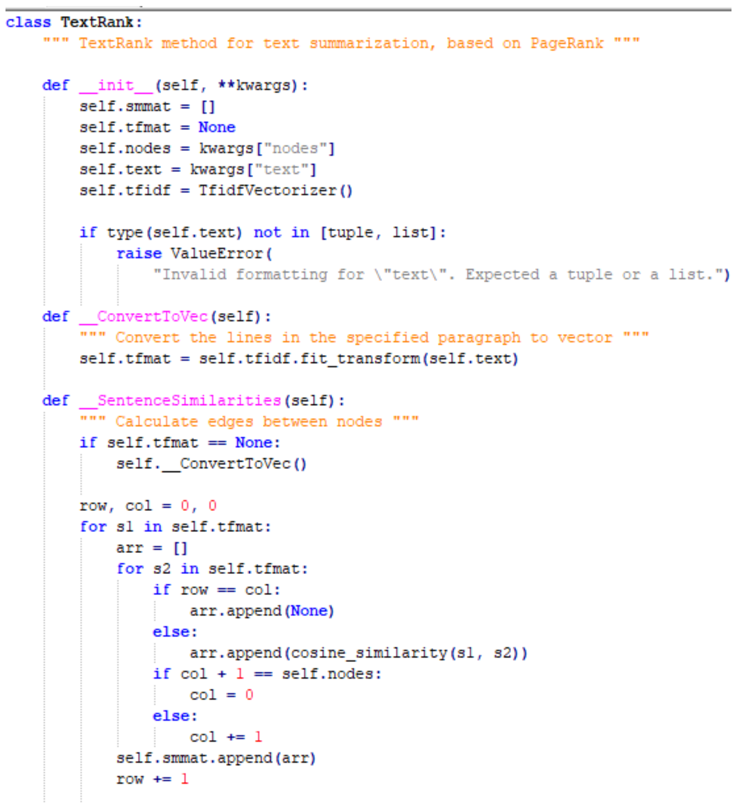
- Python/TextSummarizer.py/Exhaustive
    - __GetWeightedFreq()
        - Calculates the score for the specified word based on either word probability or word frequency method.
    - __PopulateFreq()
        - Compute the frequency table.
    - __TokenizePara()
        - Tokenizes the paragraph based on the delimiter.
    - KTopRanks()
        - Return K top words for summary formation.



```python
class Exhaustive:
    """ Exhaustive (statistical) method for text summarization """

    def __init__(self, **kwargs):
        self.text = None
        self.wfreq = {}
        self.tokens = []
        self.wstop = set(stopwords.words("english"))

        if "tokens" in kwargs.keys() and "text" in kwargs.keys():
            raise ValueError(
                "Error: Both tokens and text specified. Specify only one.")

        if "tokens" in kwargs.keys():
            self.tokens = []
            for line in kwargs["tokens"]:
                self.tokens += [line.split(" ")]
            assert type(self.tokens) in [list, tuple]
        else:
            self.text = kwargs["text"]
            #assert type(self.text) == str
            self.__TokenizePara(delim="ред")

    def __GetWeightedFreq(self, **kwargs):
        """
        Returns the weighted frequency of the word specified.
        Weighted frequency is calculated as:
            wf = freq(wx)/max(freq(wi))
        """
        if len(self.wfreq) == 0:
            self.__PopulateFreq()

        word = kwargs["term"]
        if word.lower() in self.wstop or word.isdigit() or len(word) == 0:
            return 0

        if word.lower() not in self.wfreq.keys() and word.lower() not in self.wstop:
            raise ValueError("Invalid word {0} specified".format(word))
        return self.wfreq[word.lower()] / max(self.wfreq.values())
```

TextRank

- Python/TextSummarizer.py/TextRank
  - __ConvertToVec()
    - Convert all the sentences to vector representation.
  - __SentenceSimilarities()
    - Build the graph with sentences as nodes and compute the node edges (sentence similarities)
  - KTopRanks()
    - Return K top words for summary formation.



```python
class TextRank:
    """ TextRank method for text summarization, based on PageRank """

    def __init__(self, **kwargs):
        self.smmat = []
        self.tfmat = None
        self.nodes = kwargs["nodes"]
        self.text = kwargs["text"]
        self.tfidf = TfidfVectorizer()

        if type(self.text) not in [tuple, list]:
            raise ValueError(
                "Invalid formatting for \"text\". Expected a tuple or a list.")

    def __ConvertToVec(self):
        """ Convert the lines in the specified paragraph to vector """
        self.tfmat = self.tfidf.fit_transform(self.text)

    def __SentenceSimilarities(self):
        """ Calculate edges between nodes """
        if self.tfmat == None:
            self.__ConvertToVec()

        row, col = 0, 0
        for s1 in self.tfmat:
            arr = []
            for s2 in self.tfmat:
                if row == col:
                    arr.append(None)
                else:
                    arr.append(cosine_similarity(s1, s2))
                if col + 1 == self.nodes:
                    col = 0
                else:
                    col += 1
            self.smmat.append(arr)
            row += 1
```

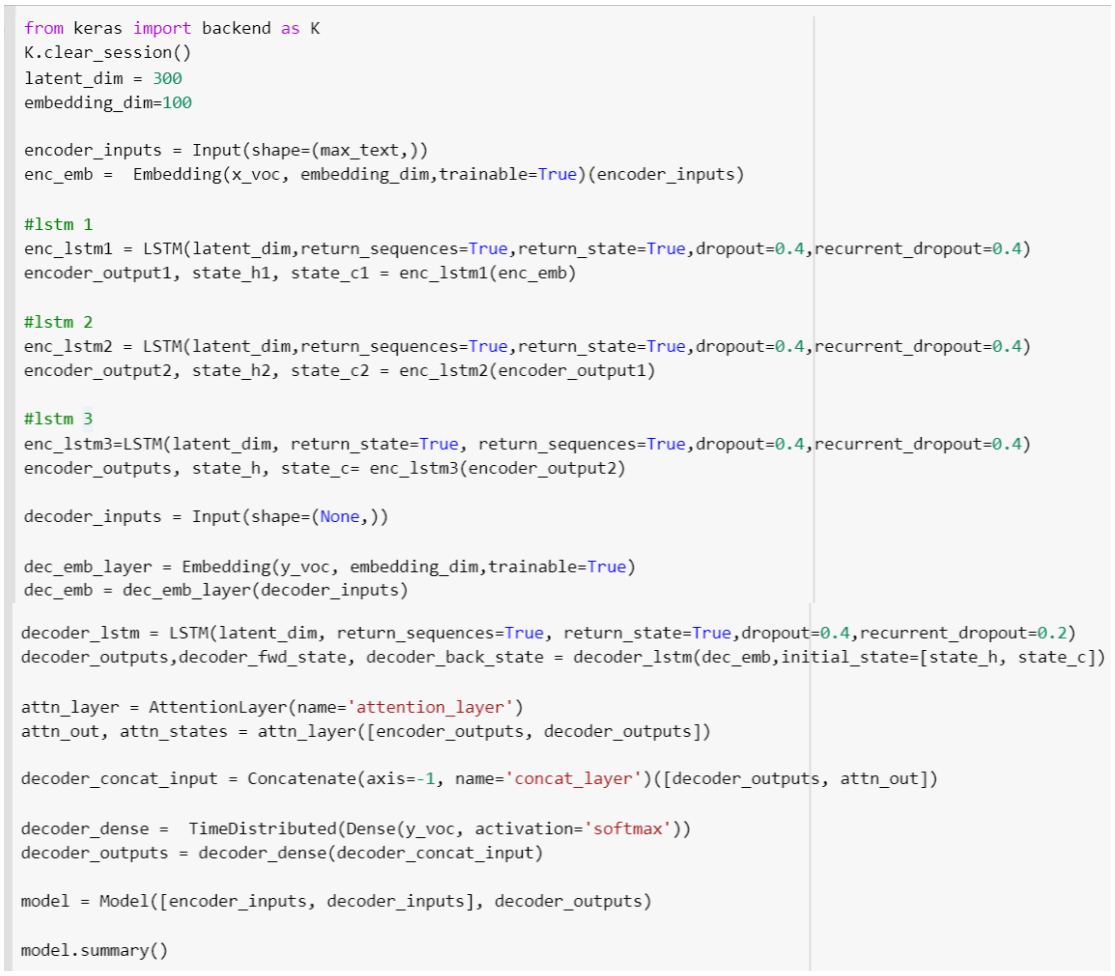
4.1.2 Abstractive

The source python notebook for model on Reviews dataset can be found here (open in Google drive with Google Colab or download)
The source python notebook for model on News dataset can be found here

Main modules and methods –

- clean_text()
  - Used to clean the data text as well as summary
- decode_sequence()
  - Used to predict the test sequences using the created models
- Model
  - It is based on Encoder Decoder LSTM layers that are trained on the dataset and then set up for inference of the test set

For both the datasets the same exact modules and models are used, giving varying result.



```
from keras import backend as K
K.clear_session()
latent_dim = 300
embedding_dim=100

encoder_inputs = Input(shape=(max_text,))
enc_emb =  Embedding(x_voc, embedding_dim,trainable=True)(encoder_inputs)

#lstm 1
enc_lstm1 = LSTM(latent_dim,return_sequences=True,return_state=True,dropout=0.4,recurrent_dropout=0.4)
encoder_output1, state_h1, state_c1 = enc_lstm1(enc_emb)

#lstm 2
enc_lstm2 = LSTM(latent_dim,return_sequences=True,return_state=True,dropout=0.4,recurrent_dropout=0.4)
encoder_output2, state_h2, state_c2 = enc_lstm2(encoder_output1)

#lstm 3
enc_lstm3=LSTM(latent_dim, return_state=True, return_sequences=True,dropout=0.4,recurrent_dropout=0.4)
encoder_outputs, state_h, state_c= enc_lstm3(encoder_output2)

decoder_inputs = Input(shape=(None,))

dec_emb_layer = Embedding(y_voc, embedding_dim,trainable=True)
dec_emb = dec_emb_layer(decoder_inputs)

decoder_lstm = LSTM(latent_dim, return_sequences=True, return_state=True,dropout=0.4,recurrent_dropout=0.2)
decoder_outputs,decoder_fwd_state, decoder_back_state = decoder_lstm(dec_emb,initial_state=[state_h, state_c])

attn_layer = AttentionLayer(name='attention_layer')
attn_out, attn_states = attn_layer([encoder_outputs, decoder_outputs])

decoder_concat_input = Concatenate(axis=-1, name='concat_layer')([decoder_outputs, attn_out])

decoder_dense =  TimeDistributed(Dense(y_voc, activation='softmax'))
decoder_outputs = decoder_dense(decoder_concat_input)

model = Model([encoder_inputs, decoder_inputs], decoder_outputs)

model.summary()
```

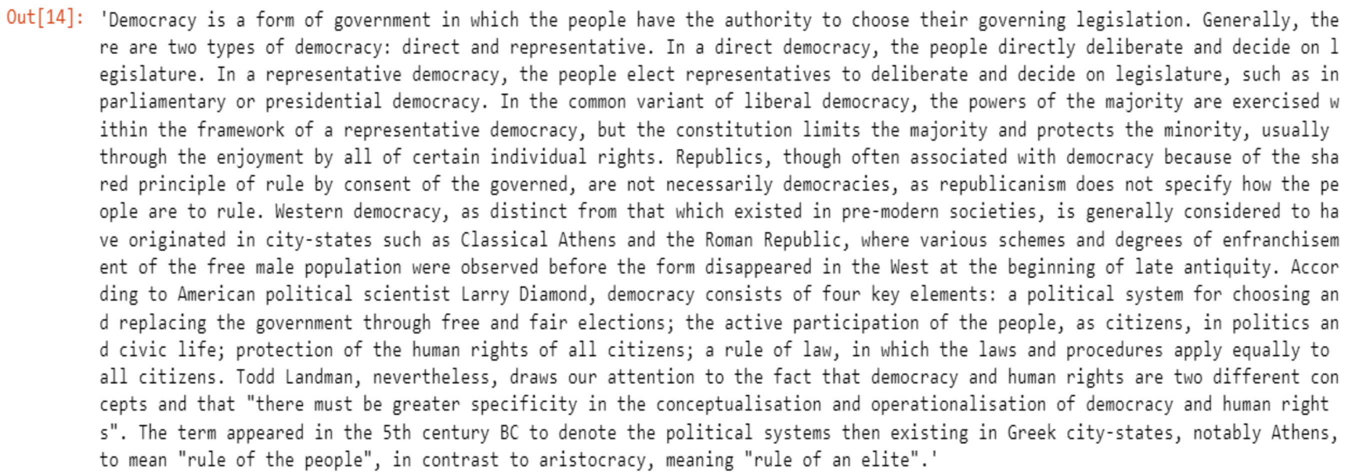
4.2 Results

We mainly evaluated news articles and general text for the models. We also implemented Weighted Frequency on Hindi text.
For general text comprehensions, TextRank was slightly faster (about 10%) and performed slightly better amongst other extractive models as the sentences in the summaries generated by other models seemed disconnected. Results for a sample test case can be viewed here.

Original text -
Democracy is a form of government in which the people have the authority to choose their governing legislation. Who people are and how authority is shared among them are core issues for democratic theory, development and constitution. Some cornerstones of these issues are freedom of assembly and speech, inclusiveness and equality, membership, consent, voting, right to life and minority rights. Generally, there are two types of democracy: direct and representative. In a direct democracy, the people directly deliberate and decide on legislature. In a representative democracy, the people elect representatives to deliberate and decide on legislature, such as in parliamentary or presidential democracy. Liquid democracy combines elements of these two basic types. However, the noun "democracy" has, over time, been modified by more than 3,500 adjectives which suggests that it may have types that can elude and elide this duality. The most common day-to-day decision making approach of democracies has been the majority rule, though other decision making approaches like supermajority and consensus have been equally integral to democracies. They serve the crucial purpose of inclusiveness and broader legitimacy on sensitive issues, counterbalancing majoritarianism, and therefore mostly take precedence on a constitutional level. In the common variant of liberal democracy,

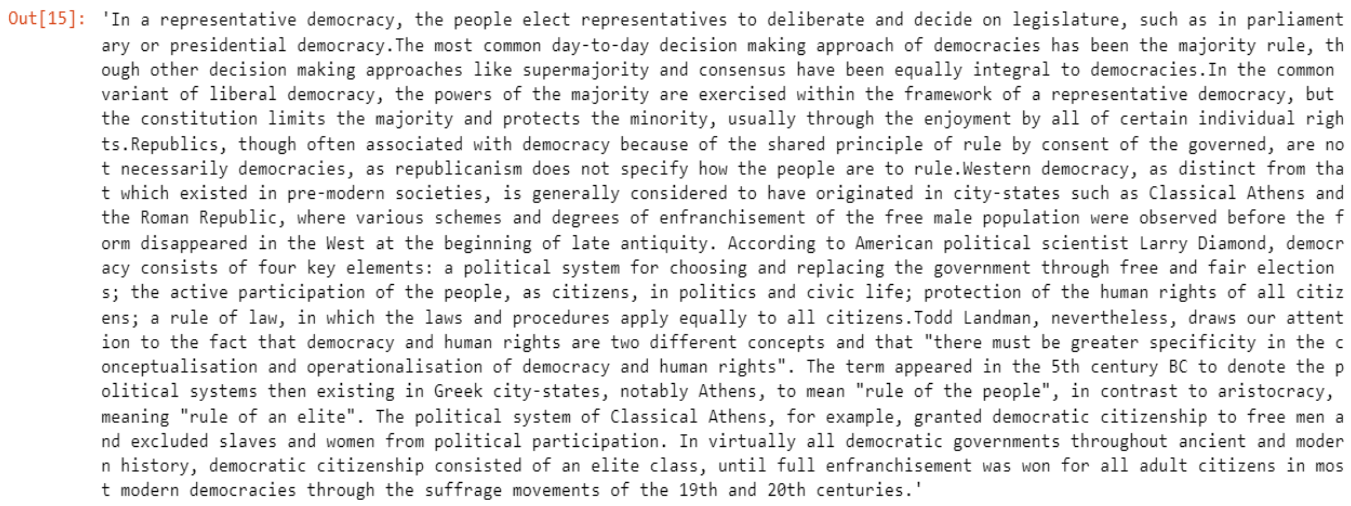
the powers of the majority are exercised within the framework of a representative democracy, but the constitution limits the majority and protects the minority, usually through the enjoyment by all of certain individual rights, e.g. freedom of speech, or freedom of association. Besides these general types of democracy, there have been a wealth of further types (see below). Republics, though often associated with democracy because of the shared principle of rule by consent of the governed, are not necessarily democracies, as republicanism does not specify how the people are to rule. Democracy is a system of processing conflicts in which outcomes depend on what participants do, but no single force controls what occurs and its outcomes. The uncertainty of outcomes is inherent in democracy. Democracy makes all forces struggle repeatedly to realize their interests and devolves power from groups of people to sets of rules. Western democracy, as distinct from that which existed in pre-modern societies, is generally considered to have originated in city-states such as Classical Athens and the Roman Republic, where various schemes and degrees of enfranchisement of the free male population were observed before the form disappeared in the West at the beginning of late antiquity. The English word dates back to the 16th century, from the older Middle French and Middle Latin equivalents. According to American political scientist Larry Diamond, democracy consists of four key elements: a political system for choosing and replacing the government through free and fair elections; the active participation of the people, as citizens, in politics and civic life; protection of the human rights of all citizens; a rule of law, in which the laws and procedures apply equally to all citizens. Todd Landman, nevertheless, draws our attention to the fact that democracy and human rights are two different concepts and that "there must be greater specificity in the conceptualisation and operationalisation of democracy and human rights". The term appeared in the 5th century BC to denote the political systems then existing in Greek city-states, notably Athens, to mean "rule of the people", in contrast to aristocracy, meaning "rule of an elite". While theoretically, these definitions are in opposition, in practice the distinction has been blurred historically. The political system of Classical Athens, for example, granted democratic citizenship to free men and excluded slaves and women from political participation. In virtually all democratic governments throughout ancient and modern history, democratic citizenship consisted of an elite class, until full enfranchisement was won for all adult citizens in most modern democracies through the suffrage movements of the 19th and 20th centuries.

<u>TextRank:</u>



```
Out[14]: 'Democracy is a form of government in which the people have the authority to choose their governing legislation. Generally, the
         re are two types of democracy: direct and representative. In a direct democracy, the people directly deliberate and decide on l
         egislature. In a representative democracy, the people elect representatives to deliberate and decide on legislature, such as in
         parliamentary or presidential democracy. In the common variant of liberal democracy, the powers of the majority are exercised w
         ithin the framework of a representative democracy, but the constitution limits the majority and protects the minority, usually
         through the enjoyment by all of certain individual rights. Republics, though often associated with democracy because of the sha
         red principle of rule by consent of the governed, are not necessarily democracies, as republicanism does not specify how the pe
         ople are to rule. Western democracy, as distinct from that which existed in pre-modern societies, is generally considered to ha
         ve originated in city-states such as Classical Athens and the Roman Republic, where various schemes and degrees of enfranchisem
         ent of the free male population were observed before the form disappeared in the West at the beginning of late antiquity. Accor
         ding to American political scientist Larry Diamond, democracy consists of four key elements: a political system for choosing an
         d replacing the government through free and fair elections; the active participation of the people, as citizens, in politics an
         d civic life; protection of the human rights of all citizens; a rule of law, in which the laws and procedures apply equally to
         all citizens. Todd Landman, nevertheless, draws our attention to the fact that democracy and human rights are two different con
         cepts and that "there must be greater specificity in the conceptualisation and operationalisation of democracy and human right
         s". The term appeared in the 5th century BC to denote the political systems then existing in Greek city-states, notably Athens,
         to mean "rule of the people", in contrast to aristocracy, meaning "rule of an elite".'
```

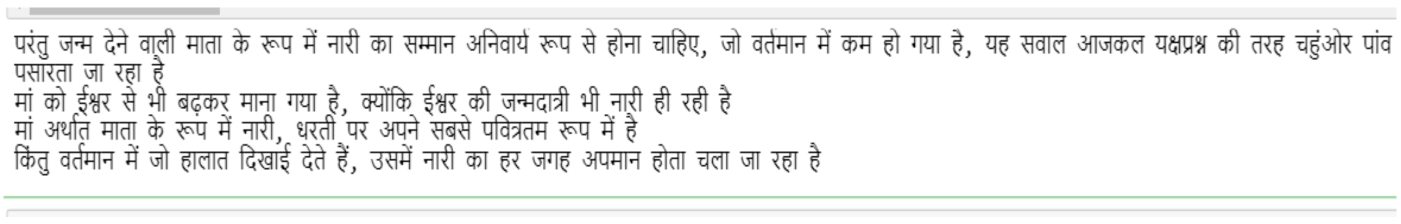
Word Weighted Frequency



```
Out[15]: 'In a representative democracy, the people elect representatives to deliberate and decide on legislature, such as in parliament
         ary or presidential democracy.The most common day-to-day decision making approach of democracies has been the majority rule, th
         ough other decision making approaches like supermajority and consensus have been equally integral to democracies.In the common
         variant of liberal democracy, the powers of the majority are exercised within the framework of a representative democracy, but
         the constitution limits the majority and protects the minority, usually through the enjoyment by all of certain individual righ
         ts.Republics, though often associated with democracy because of the shared principle of rule by consent of the governed, are no
         t necessarily democracies, as republicanism does not specify how the people are to rule.Western democracy, as distinct from tha
         t which existed in pre-modern societies, is generally considered to have originated in city-states such as Classical Athens and
         the Roman Republic, where various schemes and degrees of enfranchisement of the free male population were observed before the f
         orm disappeared in the West at the beginning of late antiquity. According to American political scientist Larry Diamond, democr
         acy consists of four key elements: a political system for choosing and replacing the government through free and fair election
         s; the active participation of the people, as citizens, in politics and civic life; protection of the human rights of all citiz
         ens; a rule of law, in which the laws and procedures apply equally to all citizens.Todd Landman, nevertheless, draws our attent
         ion to the fact that democracy and human rights are two different concepts and that "there must be greater specificity in the c
         onceptualisation and operationalisation of democracy and human rights". The term appeared in the 5th century BC to denote the p
         olitical systems then existing in Greek city-states, notably Athens, to mean "rule of the people", in contrast to aristocracy,
         meaning "rule of an elite". The political system of Classical Athens, for example, granted democratic citizenship to free men a
         nd excluded slaves and women from political participation. In virtually all democratic governments throughout ancient and moder
         n history, democratic citizenship consisted of an elite class, until full enfranchisement was won for all adult citizens in mos
         t modern democracies through the suffrage movements of the 19th and 20th centuries.'
```

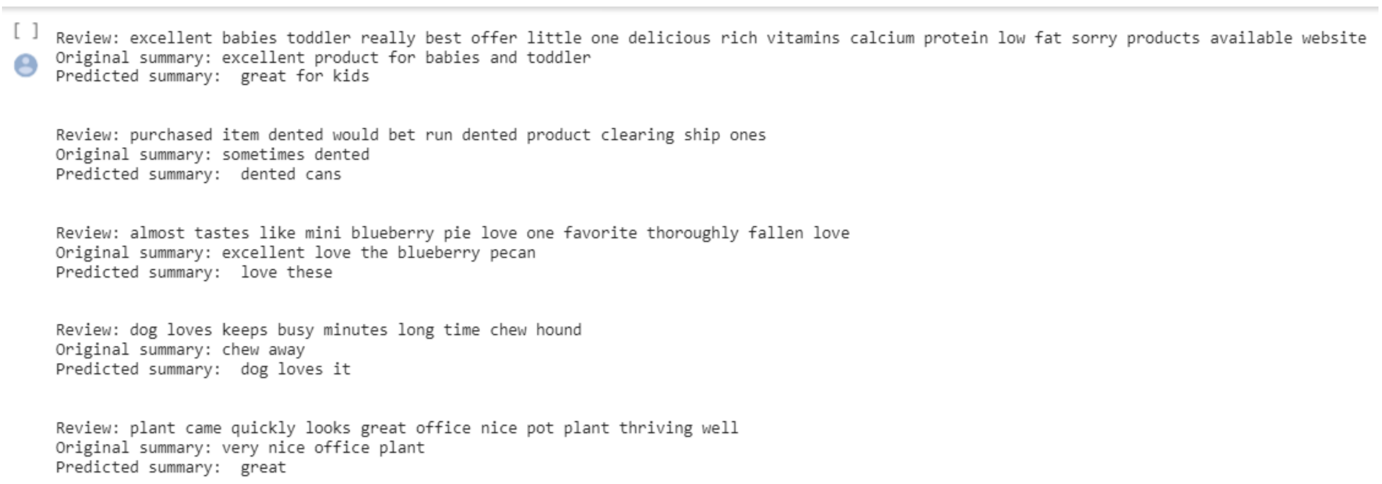
Hindi Text

text = "संस्कृत में एक श्लोक है- 'यस्य पूज्यंते नार्यस्तु तत्र रमन्ते देवता:। अर्थात्, जहां नारी की पूजा होती है, वहां देवता निवास करते हैं। भारतीय संस्कृति में नारी के सम्मान को बहुत महत्व दिया गया है। किंतु वर्तमान में जो हालात दिखाई देते हैं, उसमें नारी का हर जगह अपमान होता चला जा रहा है। उसे 'भोग की वस्तु' समझकर आदमी 'अपने तरीके' से 'इस्तेमाल' कर रहा है। यह बेहद चिंताजनक बात है। लेकिन हमारी संस्कृति को बनाए रखते हुए नारी का सम्मान कैसे किया जाए, इस पर विचार करना आवश्यक है।मां अर्थात माता के रूप में नारी, धरती पर अपने सबसे पवित्रतम रूप में है। माता यानी जननी। मां को ईश्वर से भी बढ़कर माना गया है, क्योंकि ईश्वर की जन्मदात्री भी नारी ही रही है। मां देवकी (कृष्ण) तथा मां पार्वती (गणपति/कार्तिकेय) के संदर्भ में हम देख सकते हैं इसे। किंतु बदलते समय के हिसाब से संतानों ने अपनी मां को महत्व देना कम कर दिया है। यह चिंताजनक पहलू है। सब धन-लिप्सा व अपने स्वार्थ में डूबते जा रहे हैं। परंतु जन्म देने वाली माता के रूप में नारी का सम्मान अनिवार्य रूप से होना चाहिए, जो वर्तमान में कम हो गया है, यह सवाल आजकल यक्षप्रश्न की तरह चहुंओर पांव पसारता जा रहा है। इस बारे में नई पीढ़ी को आत्मावलोकन करना चाहिए।"



```
परंतु जन्म देने वाली माता के रूप में नारी का सम्मान अनिवार्य रूप से होना चाहिए, जो वर्तमान में कम हो गया है, यह सवाल आजकल यक्षप्रश्न की तरह चहुंओर पांव
पसारता जा रहा है
मां को ईश्वर से भी बढ़कर माना गया है, क्योंकि ईश्वर की जन्मदात्री भी नारी ही रही है
मां अर्थात माता के रूप में नारी, धरती पर अपने सबसे पवित्रतम रूप में है
किंतु वर्तमान में जो हालात दिखाई देते हैं, उसमें नारी का हर जगह अपमान होता चला जा रहा है
```

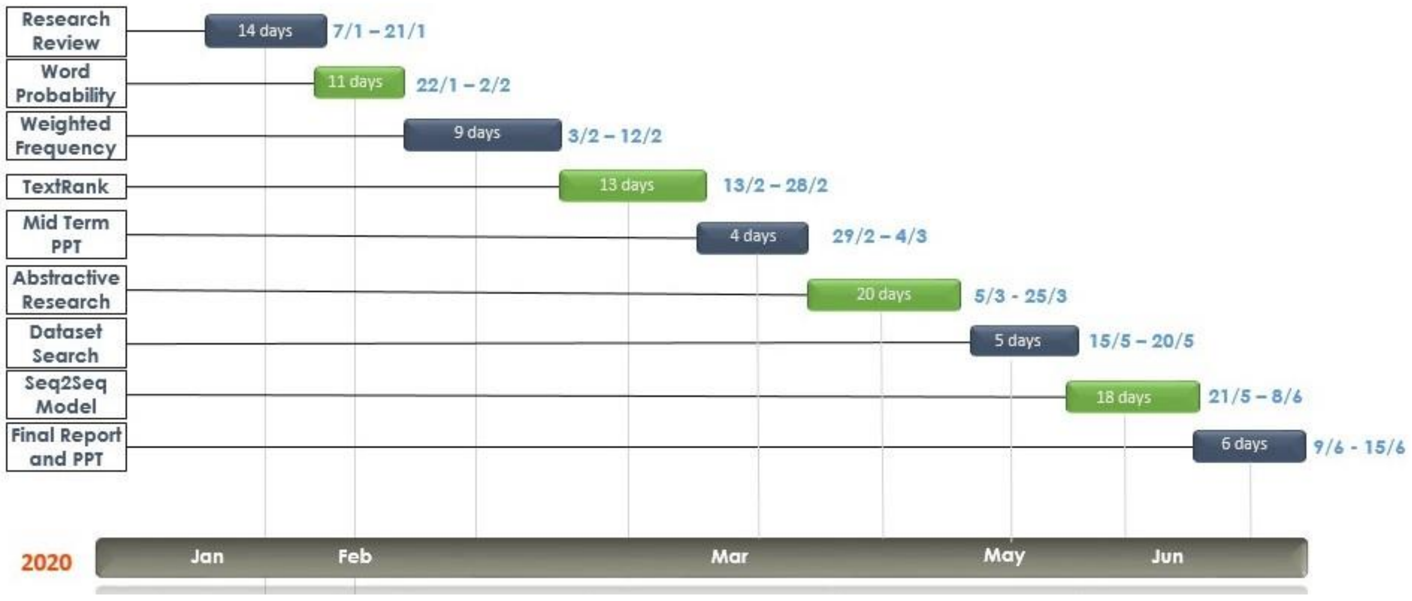
Abstractive



```
[ ]  Review: excellent babies toddler really best offer little one delicious rich vitamins calcium protein low fat sorry products available website
     Original summary: excellent product for babies and toddler
     Predicted summary:  great for kids


     Review: purchased item dented would bet run dented product clearing ship ones
     Original summary: sometimes dented
     Predicted summary:  dented cans


     Review: almost tastes like mini blueberry pie love one favorite thoroughly fallen love
     Original summary: excellent love the blueberry pecan
     Predicted summary:  love these


     Review: dog loves keeps busy minutes long time chew hound
     Original summary: chew away
     Predicted summary:  dog loves it


     Review: plant came quickly looks great office nice pot plant thriving well
     Original summary: very nice office plant
     Predicted summary:  great
```

FUTURE WORK AND CONCLUSION

4.1 Time Line



4.2 Future Work

We've just seen how simple statistical based algorithms can be used to generate fast and decent summaries. As more breakthrough research papers are published in the field of neural networks and in the field of NLP and with hardware improvements (CPU + GPU), text summarization shall get more and more reliable. As the information that is being shared online increases every year and with more people spending their time on the internet, text summarization will be widely used to enhance both the user experience and the data delivery.

There is ever increasing research and better methods in the field of Natural Language Processing, and in the future more complex work can be done using models with more layers or using completely new architectures, like Pointer Generator networks etc. to help computers understand Natural Language like never before and use it in various fields.

References

Journals/Papers

[1] H. Luhn, "The Automatic Creation of Literature Abstracts," IBM Journal, no. April, 1958.

[2] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, B. J. Gutierrez and K. Kochut, "Text Summarization Techniques: A Brief Survey," arXiv.org, 2017.

[3] R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Texts," UNT Digital Library, 2004.

[4] P. Romain, X. Caiming and S. Richard, "A DEEP REINFORCED MODEL FOR ABSTRACTIVE SUMMARIZATION," arXiv.org, no. arXiv:1705.04304v3 [cs.CL], 2017.

[5] S. Tian, K. Yaser, R. Naren and R. Chandan, "Neural Abstractive Text Summarization with Sequence-to-Sequence Models: A Survey," arXiv.org, no. arXiv:1812.02303v3 [cs.CL], 2018.

[6] B. Dzmitry and C. KyungHyun, "Neural Machine Translation by Jointly Learning To Align And Translate," in ICLR, 2015.

[7] N. Ramesh, Z. Bowen, d. s. Cicero Nogueira, G. Caglar and X. Bing, "Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond," arXiv.org, no. arXiv:1602.06023v5 [cs.CL], 2016.

Web

[1] Introduction to Recurrent Neural Networks, www.analyticsvidhya.com
[2] Comparing Text Summarization Techniques, https://towardsdatascience.com
[3] Understanding LSTM Networks, https://colah.github.io/posts/2015-08-Understanding-LSTMs/
[4] Text Summarization in Python: Extractive vs. Abstractive techniques revisited, https://rare-technologies.com/