IOP Conference Series: Materials Science and Engineering

PAPER • OPEN ACCESS

Text Summarization Techniques and Applications

To cite this article: Virender Dehru *et al* 2021 *IOP Conf. Ser.: Mater. Sci. Eng.* **1099** 012042

Text Summarization Techniques and Applications

Virender Dehru[1], Pradeep Kumar Tiwari[1], Gaurav Aggarwal[1], Bhavya Joshi[1] and Pawan Kartik[1]

[1]Manipal University Jaipur, Dehmi Kalan, Jaipur-Ajmer Expressway, Jaipur, Rajasthan -303007 India

E-mail: pradeeptiwari.mca@gmail.com

Abstract. 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 a personalized feed of summaries that a user can go through. 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. This work aims to explore different techniques of text summarization and evaluate them on different parameters such as the extent of compression/summarization, retention of meaning/gist, and grammatical errors.

1. Introduction

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

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 a personalized feed of summaries that a user can go through [2][3][4]. Another 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. 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. The two main categories of text summarization were extractive and abstractive. As the names themselves suggest, extractive emphasizes calculating weights of sentences and picking (top k sentences) them for the summary while abstractive emphasizes rewriting the sentences to generate the summary [5][6][7].

The extractive method suffers a loss in meaning to some extent as the connections between sentences are lost when picking them while the 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 [8][9][10][11].

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 [12].
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.

1.1 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 an application for voice-based assistants while answering the user's queries.
The objectives of this work are -

Explore different techniques of text summarization, Compare the generated summaries. Identify the optimal parameters (for example, k in extractive text summarization) for the best summary. Identify or implement modifications (if possible) to scale an algorithm to different languages. And also Identify the different applications of automatic text summarization. Table 1 shows the advantages & disadvantages of automatic text summarization.

1.2 Advantages & Disadvantages

Table 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 on 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 the 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 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 overview, thereby enhancing the user's experience while engaging with a product or a service.	Grammatical mistakes - abstractive algorithms are prone to grammatical mistakes: 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.

Organization of work: First take an overview of the theory, concepts, and technology, Then check detailed methodology of each algorithm, compare the results and performance of all the methods.

2. Conceptual View
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 for assigning weights to the sentences [2].
For example:
Word weighted frequency

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

Occurrence of important words

- A sentence is assigned more weight if more number of important words occur in it. Important words can be picked by using certain filters that 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 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 the given text.
These models are typically based on Recurrent Neural Networks [4]. Figure 1 depicts the RNN architecture. 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 of previous words and the context gained from them are required. The most important feature of RNN is the Hidden state, which remembers some information about a sequence. RNN is said to have a memory that can remember previously learned information. Refer figure 1.

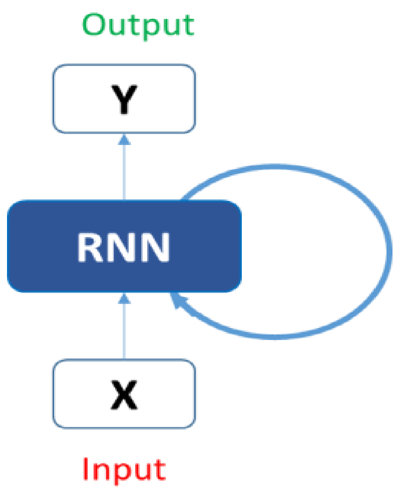

Figure 1. RNN Basic Architecture

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

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

For the next time step, the new ht becomes ht-1. As much time phases as the issue takes, we will go and merge the data from all the previous states. Upon completion of all time steps, the final current state is used to determine the yt output. The output is then correlated with the real output, producing an error. To change the weights, the error is then backpropagated to the network (we shall go through the backpropagation information in further sections) and the network is trained.
A special form called Long Short Term Memory Network is the RNN used here, which overcomes the issue of long-term RNN dependence.

2.1.2 Concepts

- Term frequency - Inverse frequency text (tf-idf): It is also used as a weighting factor in data extraction, text mining and user modelling searches. The meaning of tf-idf increases proportionally to the amount of times a word appears in the text and is offset by the number of documents containing the word in the corpus, which tends to respond to the fact that certain words appear in general more often. One of the most common term-weighting schemes today is tf-idf. Figure 2 illustrates the process of document summarization. Refer figure 2.

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.

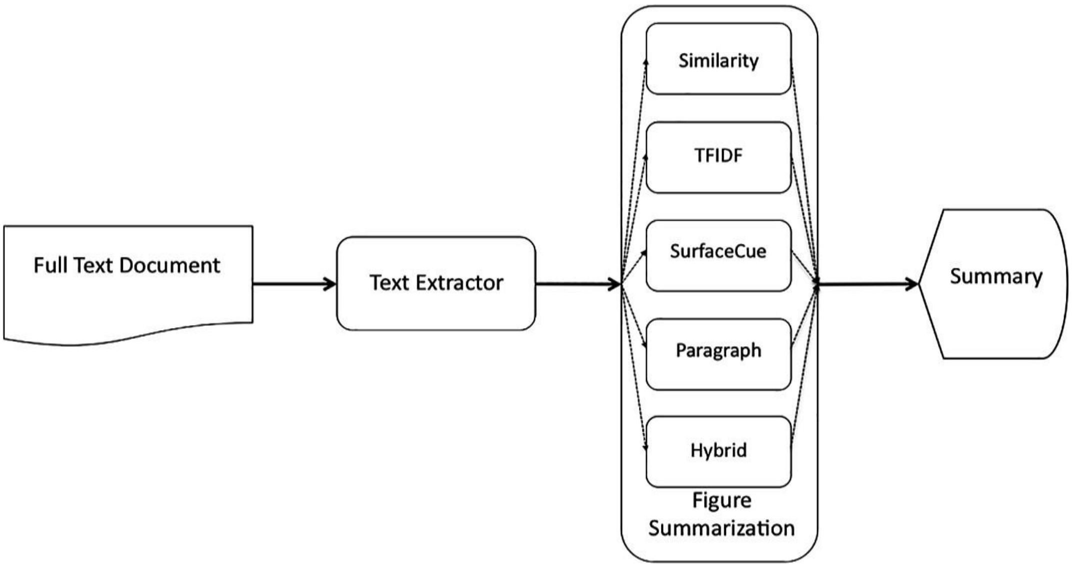

Figure 2. Text Summarization Process

- 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 that 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 for computers to understand natural language and predict word sequences [5].

- Encoder: This is a neural network with the purpose of interpreting and constructing a smaller dimensional representation of the input set. At any point, the encoder processes the information and collects the contextual information present in the input sequence. In order to initialise the decoder, the hidden state (hi) and cell state (ci) of the last time stage are used. Figure 3 displays the LSTM process for the encoder.

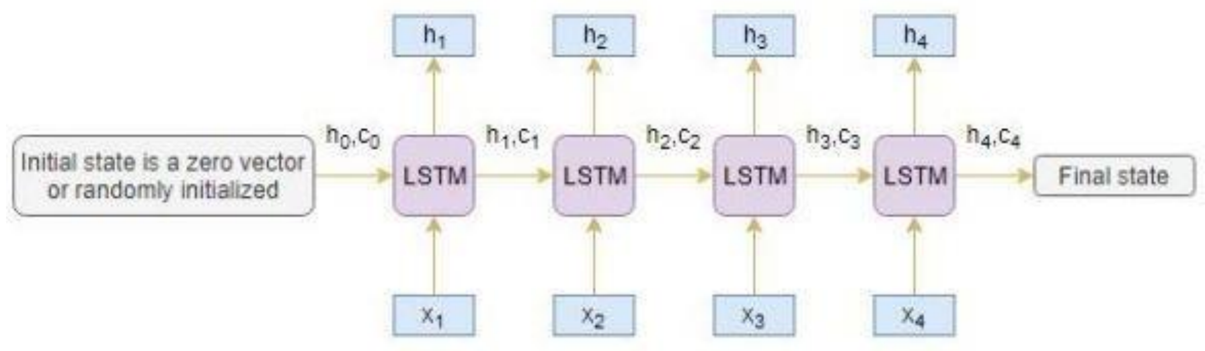

Figure 3. Encoder LSTM

- Decoder: The representation of the encoder is redirected to it and a sequence of its own is created to represent the output. It reads the whole word-by-word and predicts one time-step offset of the same sequence. In the sequence given the prior word, it predicts the next word. The special tokens that are applied to the target sequence prior to feeding it into the decoder are <start> and <end>. The decoder function is illustrated in Figure 4.

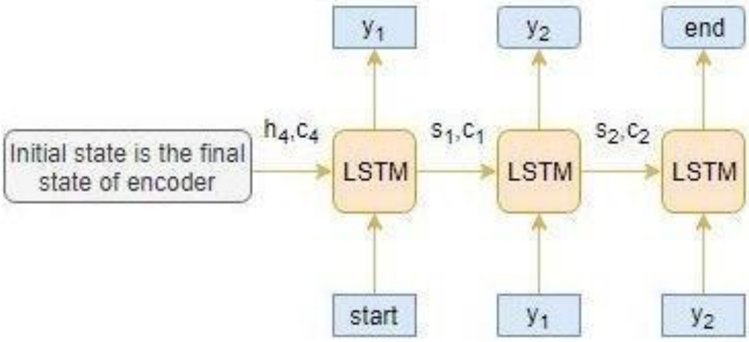

Figure 4. Decoder LSTM

- 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 the attention mechanism, the 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

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

3. Methodology

3.1 Extractive Method

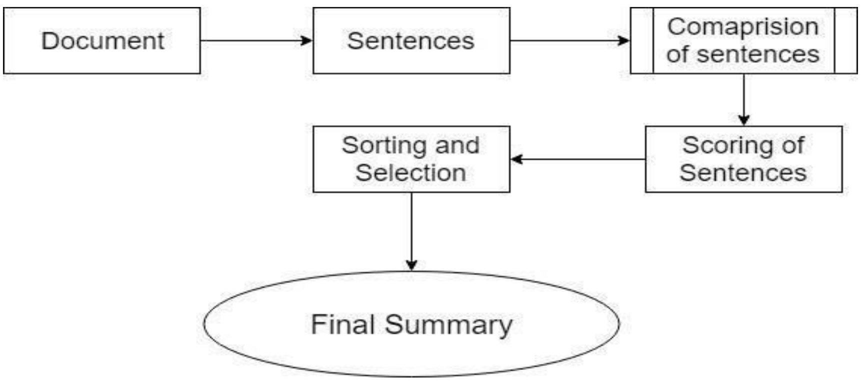

Figure 5. Flow Chart for Extractive Text Summarization

There exist many approaches or techniques as part of the extractive method. Mainly focus on word weighted frequency and text rank. Figure 5 depict the 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, then remove the stop words and punctuation. Because this entire model is based on frequency, 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, the 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**, which 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 interactively.
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. The 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 a greedy search. These sentences from 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. Figure 6 showing the adjacency matrix.

	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	

Figure 6. Adjacency Matrix

A typical example of an adjacency matrix with cells filled with values, depicting the connections between 2 nodes. Refer figure 6.

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 the 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 of text and headlines
Food Reviews Amazon from Kaggle, contains multiple columns, most important are Text and Summary.

3.2 Working Mechanism Procedure

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, review 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). The text was cleaned by removing HTML tags, contractions were expanded, 's were removed, any parenthesis text was removed, stopwords were removed and short words were removed. The same was done to clean the summaries present in both 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 a test. The text and summary word sequences were converted into integer sequences using tokenizers and topmost 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 were 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]. The decoder was set up in the 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.

4. Implementation and Results

4.1 Modules & Implementation

4.1.1 Extractive

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.

<u>TextRank</u>

- 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.

4.1.2 Abstractive

The source python notebook for a model on Reviews dataset can be found (open in Google drive with Google Colab or download).
The source python notebook for a model on News dataset can be found
Main modules and methods –

- clean_text()
    - Used to clean the data text as well as the 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 modules and models are used, giving varying results.

4.2 Results

We mainly evaluated news articles and general text for the models. We also implemented the Weighted Frequency on Hindi text.
For general text comprehension, 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. In this work, we used test paragraph for test summarization. Text paragraph 1 is in English and second text paragraph in Hindi.

4.2.1 Original Text

- Test Paragraph: 1

Democracy is a form of the government where people get to choose their leaders. While there are many democratic nations in the world, the process of electing leaders of their choice and the formation of the government varies. While some countries elect Presidents, others elect Prime Ministers. Who gets to vote and how people vote is a major factor in democracy. Separation of powers and checks and balances exist to make sure that every institution controlled by a democratic government functions freely and fairly. It is widely considered that there are 4 key aspects of a democratic government - choosing and replacing a government through free and fair elections; participation of people in this process through voting; provision and protection of fundamental rights; and rule of law. There have been instances in history, where people fought for their right; right to vote and freely elect their leaders. Today, a healthy democracy not only lets people vote but also lets them hold their leaders accountable.

- Results: Depict the three-sentence summery 1, 2 and 3
  - 1 Sentence Summary:
    - It is widely considered that there are 4 key aspects of a democratic government - choosing and replacing a government through free and fair elections; participation of people in this process through voting; provision and protection of fundamental rights; and rule of law.
  - 2 Sentence Summary:
    - It is widely considered that there are 4 key aspects of a democratic government - choosing and replacing a government through free and fair elections; participation of people in this process through voting; provision and protection of fundamental rights; and rule of law.
    - Today, a healthy democracy not only lets people vote but also lets them hold their leaders accountable.
  - 3 Sentence Summary:
    - It is widely considered that there are 4 key aspects of a democratic government - choosing and replacing a government through free and fair elections; participation of people in this process through voting; provision and protection of fundamental rights; and rule of law.
    - There have been instances in history, where people fought for their right; right to vote and freely elect their leaders.
    - Today, a healthy democracy not only lets people vote but also lets them hold their leaders accountable.

- Test Paragraph: 2

संस्कृत में एक श्लोक है- 'यस्य पूज्यंते नार्यस्तु तत्र रमन्ते देवता:। इसका मतलब है कि जहां भी लोग महिलाओं की पूजा करते हैं, भगवान मौजूद होते हैं। लेकिन वास्तव में महिलाओं को जहां भी आप देखते हैं, उनके साथ बुरा व्यवहार किया जाता है। लोग महिलाओं को एक दायित्व मानते हैं और उनका उपयोग किसी बुरे उद्देश्य के लिए करते हैं। यह बहुत चिंताजनक है। महिलाओं का सम्मान करने और उन्हें समाज में एक समान स्थान दिलाने के लिए यह सुनिश्चित करना बहुत महत्वपूर्ण है। इस दुनिया की पहली नारीवादी चिंतक सिमोन द बोउआर ने कहा था, 'पहले औरत के पंखों को काट दिया जाता है और फिर उस पर इल्जाम लगाया जाता है कि उसे उड़ना नहीं आता।' यह आज भी उतना ही सत्य है जितना पुराने समय में था। समान अवसर दिए जाने पर महिलाएं अधिक खुशहाल, अधिक शिक्षित समाज का नेतृत्व करेंगी। यह राष्ट्र में बेहतर घरों और कम गरीबी को जन्म देगा।

- Results: Depict the three-sentence summery 1, 2 and 3
  - 1 Sentence Summary:
    - इस दुनिया की पहली नारीवादी चिंतक सिमोन द बोउआर ने कहा था, 'पहले औरत के पंखों को काट दिया जाता है और फिर उस पर इल्जाम लगाया जाता है कि उसे उड़ना नहीं आता।
  - 2 Sentence Summary:
    - इस दुनिया की पहली नारीवादी चिंतक सिमोन द बोउआर ने कहा था, 'पहले औरत के पंखों को काट दिया जाता है और फिर उस पर इल्जाम लगाया जाता है कि उसे उड़ना नहीं आता।'

        - समान अवसर दिए जाने पर महिलाएं अधिक खुशहाल, अधिक शिक्षित समाज का नेतृत्व करेंगी।
    - 3 Sentence Summary:
        - लेकिन वास्तव में महिलाओं को जहां भी आप देखते हैं, उनके साथ बुरा व्यवहार किया जाता है।
        - इस दुनिया की पहली नारीवादी चिंतक सिमोन द बोउआर ने कहा था, 'पहले औरत के पंखों को काट दिया जाता है और फिर उस पर इल्जाम लगाया जाता है कि उसे उड़ना नहीं आता।'
        - समान अवसर दिए जाने पर महिलाएं अधिक खुशहाल, अधिक शिक्षित समाज का नेतृत्व करेंगी।

5. Conclusion

This research shows the 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 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 are 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.

6. References

[1] Gupta, V., & Lehal, G. S. (2009). A survey of text mining techniques and applications. *Journal of emerging technologies in web intelligence*, *1*(1), 60-76.

[2] Tas, O., & Kiyani, F. (2007). A survey automatic text summarization. *Press Academia Procedia*, *5*(1), 205-213.

[3] Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). Text summarization techniques: a brief survey. *arXiv preprint arXiv:1707.02268*

[4] Nenkova, A., & McKeown, K. (2012). A survey of text summarization techniques. In *Mining text data* (pp. 43-76). Springer, Boston, MA.

[5] Kanapala, A., Pal, S., & Pamula, R. (2019). Text summarization from legal documents: a survey. *Artificial Intelligence Review*, *51*(3), 371-402.

[6] Wang, L., Yao, J., Tao, Y., Zhong, L., Liu, W., & Du, Q. (2018). A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. *arXiv preprint arXiv:1805.03616*.

[7] Kryscinski, W., McCann, B., Xiong, C., & Socher, R. (2020, November). Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 9332-9346).

[8] Song, S., Huang, H., & Ruan, T. (2019). Abstractive text summarization using LSTM-CNN based deep learning. *Multimedia Tools and Applications*, *78*(1), 857-875.

[9] Cai, H., Zheng, V. W., & Chang, K. C. C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering*, *30*(9), 1616-1637.

[10] Kobayashi, V. B., Mol, S. T., Berkers, H. A., Kismihók, G., & Den Hartog, D. N. (2018). Text mining in organizational research. *Organizational research methods*, *21*(3), 733-765.

[11] Salloum, S. A., Al-Emran, M., Monem, A. A., & Shaalan, K. (2018). Using text mining techniques for extracting information from research articles. In *Intelligent natural language processing: trends and applications* (pp. 373-397). Springer, Cham.

[12] Qiang, J., Qian, Z., Li, Y., Yuan, Y., & Wu, X. (2020). Short text topic modeling techniques, applications, and performance: a survey. *IEEE Transactions on Knowledge and Data Engineering*.