Automated Text Summarization using NLP

Harshita Jain, A. P. Shah Institute of Technology, Thane, India, harshitadjain1800@gmail.com
Kevin Khimasia, A. P. Shah Institute of Technology, Thane, India, kevinkhimasia13@gmail.com
Sejal Khedekar, A. P. Shah Institute of Technology, Thane, India, sejalkhedekar2000@gmail.com
Anjali Masur, A. P. Shah Institute of Technology, Thane, India, anjalimasur28s@gmail.com
Dr. Pravin Adivarekar, A. P. Shah Institute of Technology, Thane, India, ppadivarekar@apsit.edu.in

Abstract- Text summarization involves generating a shorter version of any given text in the form of a summary that provides the entire gist without any loss of meaning. Its methods can be divided into two categories: extractive and abstractive summarization. The goal of extractive and abstractive summarizing is to provide a generalized summary that delivers information precisely, which usually necessitates the use of advanced language creation and compression techniques. The extractive model aims at selecting sentences from the passage or text and picking up the main, essential, or relevant information from it as it is without any modification. Thus, it has some limitations. To overcome this and come up with a more concise summary, abstractive text summarization comes into the picture. The summary generated by the abstractive(automated) text summarization approach is more like the summary that a human being would come up with hence, achieving it includes complex algorithms. In a nutshell, abstractive text summarization makes use of neural networks like Recurrent Neural Networks (RNN) and the Long Short Term Memory (LSTM) algorithm which has many layers that are useful in generating meaningful sentences and at the same time shortens the length of text.

Keywords — Abstractive summarization, Attention layer, Encoder-decoder, Extractive summarization, LSTM, NLTK, Seq2Seq.

I. INTRODUCTION

This project aims to achieve automation of generating a summary for the given set of data by generating a summarized text of fixed word length by extractive summarization techniques. The model designed in the project will be trained such that it will choose important words and sentences from the input text and arrange them to formulate meaningful sentences. We will be implementing abstractive summarization as well where the ambiguity of sentences in the summary will be reduced as this approach generates a summary by framing new sentences that serve the purpose. The existing summarization tools have a restriction on the word length for input text so we will be working on this aspect, and try to remove such barriers. Machine learning algorithms are trained to comprehend documents given as input and identify the sections that convey important facts and information before producing the required summarized texts.

Recently, there has been a surge in the amount of text data from various sources. This text volume is a priceless source of information and knowledge that should be effectively summarized in order to be useful. The main goal of this problem is to automate text summarization. Because of the increased availability of documents, extensive research in automatic text summarization has been required. The entire idea is to reduce or minimize the valuable information contained in the documents. Several websites and applications use this to generate news feeds and article summaries. Due to our hectic schedules, it has become critical for us. We prefer short summaries that include all of the important points to reading the entire report and summarizing it ourselves, which is the focus of this paper.

II. LITERATURE SURVEY

Extractive Text Summarization is the method of extracting content from the document and combining it to form a text smaller in size. It can be divided into Structured and Semantic approaches, each of which can be subdivided into subcategories based on various methods [1]. The methods are:

- •Tree-based approach
- •Ontology-based approach
- •Rule-based approach
- •Graph-based approach

The review of the summarization approaches and their important parameters for extracting predominant sentences can also be seen [3]. The method of extraction of sentences gives the idea of the input text in a short form [2]. Sentences are ranked by assigning weights and they are ranked based on their weights. Highly ranked sentences are extracted from

 DOI : 10.35291/2454-9150.2022.0112

the input document so it extracts important sentences which directs to a high-quality summary of the input document and stores the summary as audio. Generating headlines that convey the news and attract the reader's attention is an application of text summarization which can be addressed by two different approaches [4]. The first method is a multi-level pre-training framework which incorporates an unlabeled corpus with different quality vs.-quantity balance at different levels. Second, a self-voting-based article attention layer that extracts salient information shared by multiple articles. The second method of summarization, abstractive summarization focuses on various aspects like Syntactic constraints, Word graph, Text summarization, Attention model, Semantic graph, discourse rules, which are seen across different approaches used to achieve abstractive text summarization [6].

III. Experimental Setup

Text summarization is an extremely important application of Natural Language Processing (NLP). NLP is a subarea of AI that is dedicatedly used to develop interaction between humans and machines with the use of human languages. Natural Language Toolkit (NLTK) is a toolkit of NLP which we have used to achieve extractive summarization of text. Tokenization of sentences and words is done which breaks down the sentences and further, stopwords are eliminated to determine the weightage and importance of words that retain the salient features of the sentences. For abstractive summarization, we have built a model with the use of libraries like TensorFlow, Keras, and BS4. The model includes multiple LSTM encoder-decoder layers in addition to attention layers that help the model generate new, meaningful sentences to form a complete summary [9]. For the ease of users, we have integrated GUI (Figure. 4, 5, 6) in the form of a web application developed in Python. Additionally, we referred to a dataset retrieved from Kaggle for the purpose of training our model to generate a summary [8].

IV. Methodology

To achieve extractive summarization, we have first performed some pre-processing steps on the input data which is taken in the form of text or via a file. After performing stemming and removal of stop words [3], we have followed an approach like sentence ranking [2]. The filtered input text is tokenized, and every sentence is assigned a weightage based on word frequency which is then considered to determine the which sentences will provide the summary of the text without losing the meaning.

In order to generate a summary, especially an abstractive summary, it is necessary that the model remembers the previous words so that it can create meaningful sentences. This is addressed by RNN which uses the output of previous hidden layers to generate the next word. But in RNN, only a small amount of data is remembered from the previous layers because of which, generation of long sentences becomes ambiguous [9]. LSTM is a simple extension of the RNN model that overcomes the limitation of ambiguity. They are good at learning long-term dependency for a required period of time. LSTM is a chain-like structure having four neural network layers. The cell state displayed at the top of the image is crucial to LSTM. The cell state is controlled by 3 gates namely: forget gate, input gate and output gate, as shown in Fig.1. The first gate i.e., the forget gate decides which information should be thrown away from the cell state using the sigmoid layer. Further, new information is stored in the cell using the input gates sigmoid layer and tanh layer. At the end the output layer gives the output that we want by using sigmoid layer and tanh layer.

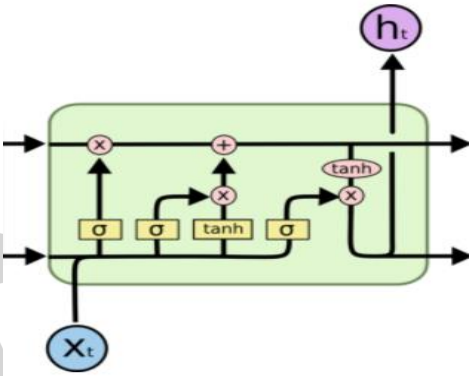

Fig.1: LSTM Layer block [9].

LSTMs have long been used to process and understand sequential data. For seq2seq jobs, the encoder-decoder architecture is commonly used. The fact that the encoding stage must represent the full input sequence x1, x2, x3, x4 as a single vector c, which might cause information loss because all information must be compressed into c, is a fundamental disadvantage of this architecture [9]. Furthermore, the decoder must decipher the past information just from this one vector, which is a difficult process in and of itself. And this is where the upper bound on the model's knowledge and performance comes into play, as this may be too much to ask of basic encoders and decoders for sequential processing of long input sequences. As shown in Fig. 2, the attention mechanism is beneficial as it allows us to look at all hidden states from the encoder sequence in order to make predictions. In a simplified encoder-decoder design, the decoder is expected to begin making predictions by merely glancing at the encoder step's final output, which has condensed information. Attention-based architecture, on the other hand, pays attention to every hidden state from each encoder node at every time step and then generates predictions depending on which one is more informative. In Fig. 2, we can see that the input sequence is given as input to the encoder in the form of x_1, x_2, x_3, and x_4 vectors and outputs for each hidden layer are generated as h_1, h_2, h_3, and h_4. The attention mechanism then determines the amount of attention to be given to each input vector as α_1, α_2, α_3, α_4, and then generates the context vectors c_1, c_2, c_3, c_4 which are fed

 DOI : 10.35291/2454-9150.2022.0112

to the decoder. The decoder then generates the output denoted as y_1, y_2, y_3, and y_4 in Fig. 2.

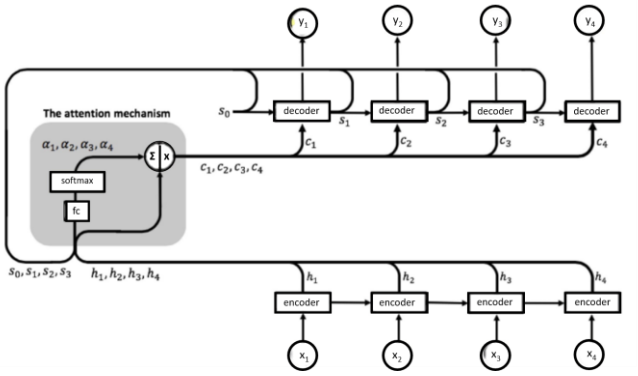

Fig. 2: Encoder-decoder using attention mechanism [9].

V. ALGORITHM

The algorithm for extractive and abstractive approaches differ to some extent and is accordingly distinguished ahead.

A. *Extractive Summarization:*

1. Pre-process the input_data by applying stemming and stopwords removal methods [3].
2. Create a list of tokenized words and record frequency in a table.
3. Perform sentence tokenization on the pre-processed data.
4. On the basis of the values in the frequency table, check for the presence of the word in each sentence token and add the corresponding frequency value for that sentence.
5. Calculate the average of the values assigned to each sentence.
6. Based on the length of input text, consider the following multiplication factor required to determine the rank of the sentence:
    a. 10% for a range of length between 100 and 400.
    b. 20% for length greater than 400.

B. *Abstractive Summarization:*

The dataset used to train the model for abstractive text summarization is based on the reviews collected by Amazon for food items, available on Kaggle [8].

1. Pre-process the dataset [3][8] and store the input text and their corresponding summaries in separate dataframes.
2. For the input text, perform pre-processing by expanding contractions, removing stopwords, and performing stemming.
3. Separate the unique words for input and target data and define the maximum length of input text and summary to train the model with.
4. Split the dataset for training and testing purposes, and apply encoder-decoder model.
5. Create stacked LSTM layers and initialize the decoder's LSTM layer with the output states of the encoder.
6. Implement attention layer mechanism and concatenate the attention layer's output with the decoder output.
7. Compile a model and save it as a Seq2Seq model.
8. Use the Rouge-Score technique to check the accuracy of the model.

VI. PROPOSED SYSTEM

A. *Explanation of Proposed System*

In this project, we will be using an abstractive approach to summarize the text. We would create our model and integrate it with the website to provide a user-friendly interface. Users can generate the summary by adding text or by uploading files. Non-registered users can summarize the text with specific words and usage limits. Whereas, registered users would have the benefit of summarizing the text without any usage limit and their summary would be saved in the database for a week so that they can revisit and access their summarized text.

B. *Block Diagram*

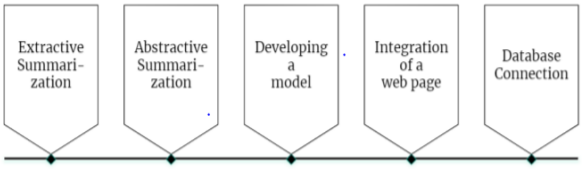

Fig. 3: Flow of Modules

As shown in Fig. 3, both extractive and abstractive summarization approaches have been implemented in this project. Abstractive summarization will be used for actual text summarization whereas extractive text summarization will be used for the generation of notes. An ML model will be designed on the successful completion of abstractive summarization. To provide a User Interface (UI) we will integrate a web page with the ML model and add database connectivity to store the user credentials and their activity on the website.

1. User State-Web Page: Whenever the application will be accessed, login status will be checked. If a login hasn't been done then the user will be able to access only a few of the features of the application. Upon logging in, the registered user can access additional features, helpful for the summarization of text.

2. Non-Registered User: Upon logging in, the registered user can access additional features, helpful for If any user hasn't logged in or hasn't registered, they will be able to use the application but with few restrictions. A limit will be set in terms of a number of words for

 DOI : 10.35291/2454-9150.2022.0112

adding text for which summarization is required. Similarly, a limit for summarized text will also be applied. For non-registered users, there will also be a usage limit i.e., they can use the tool only thrice after which they will have to register if they wish to continue using the application.

3. Account: If users decide upon creating an account they have to go through registering activity. A SQL database will be connected to the web page which will store the user's credentials upon generation of an account. Then the user can log in through their credentials and access the account. With the help of an account, the user can track their previous work easily.

4. Registered User: Once logging activity is done the user can access additional features and summarize more text. The limit for input text and summarized text will be increased. There will be no limit to the number of times a user can use the tool for summarizing text passages. We also wish to add the feature of storing the summarized text for 7 days after which it will be erased from the database. The user can revisit the application and access the summarized text generated within the last 7 days, as shown in Fig. 6.

C. *Outcome*

The end goal is to generate summaries in both extractive and abstractive formats. The summary generated in extractive is a set of important sentences chosen from the input text which can be useful when the user requires to go through the content of a textbook and can use this summary as notes. This outcome is categorized as 'Notes Generation'. Abstractive summarization attempts to create a summary similar to a summary generated by humans, which is the primary focus of this project. The user has the liberty to choose the length of the summary that is to be generated by selecting from a predefined range of the percentage of information that the user needs in the summary, from the text. Additionally, login and sign-up options will be available so that users can access the previously generated summary that will be stored in a database for a period of one week.

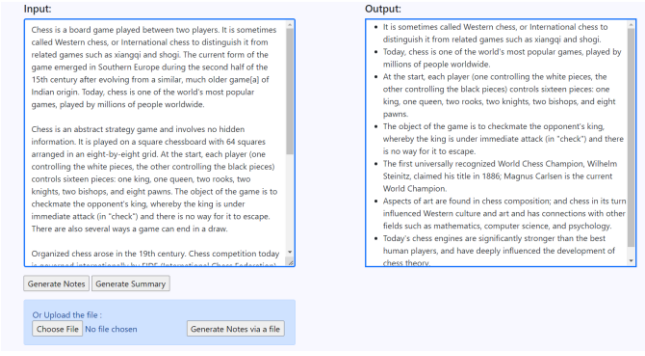

Fig. 4: Home page (Non-registered User)

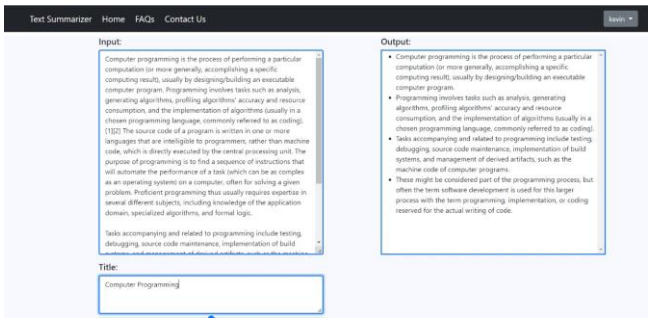

Fig. 5: Home page (Registered User)

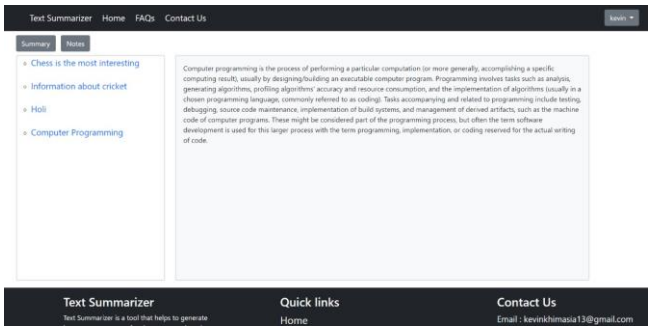

Fig. 6: Profile page

VII. CONCLUSION

Summarization may seem like a basic concept but a powerful tool can make an extensive impact in any research field. Smart work is always preferable compared to hard work and summarization tools aid in reducing the tedious hard work required when researching a topic. Summarization is a meagre application of NLP, and it can be explored even further so that the accuracy of such applications can be increased, and machines can generate content like the ones generate by a human mind. In this project, we have chosen the LSTM model to implement text summarization as it is the model that is efficient in generating meaningful sentences based on the comparison we made with the traditional RNN models that can also be used for the same purpose. A language is a very profound method of communication that is constantly evolving, which is why teaching a machine to understand any language and to train it to work as a human mind is a tedious task. We understood while working on this project that achieving higher accuracy is bit difficult as any spoken language has its own variation depending on person to person.

ACKNOWLEDGMENT

Research work is an integral part of any project and researching for our project is what inspired us to create a summarization tool that highlight precise and useful information from the given input data and eliminate the irrelevant information. We would like to extend our gratitude towards our project guide who helped us throughout all the cycles of development of this project, as well as the constant support of our college's faculty.

 DOI : 10.35291/2454-9150.2022.0112

REFERENCES

[1] Anish Jadhav, Rajat Jain, Steve Fernandes, Sana Shaikh, "Text summarization using neural networks", IEEE International Conference on Advances in Computing, Communication and Control (ICAC3), 2019.

[2] J.N.Madhuri, R.Ganesh Kumar, "Extractive text summarization using sentence ranking", IEEE International Conference on Data Science and Communication (IconDSC), 2019.

[3] Shohreh Rad Ramini, Ali Toofanzahdeh Mozhdehi, "An overview on extractive text summarization", IEEE International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2017.

[4] Xiaotao Gu, Yuning Mao, Jiawei Han, Jialu Liu, Hongkun Yu, You Wu, Cong Yu, Daniel Finnie, Jiaqi Zhai, Nicholas Zukoski, "Generating Representative Headlines for News Stories", The Web Conference '20, April 20 - 24, 2020, Taipei.

[5] Sumit Chopra, Michael Auli, and Alexander M Rush. 2016, "Abstractive sentence summarization with attentive recurrent neural networks", in Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 93–98.

[6] Md Ashraful Islam Talukder, Sheikh Abujar, Abu Kaisar Mohammad Masum, Sharmin Akter, Syed Akhter Hossain, "Comparative Study on Abstractive Text Summarization", 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)

[7] Pooja Batra, Kavya Bhatt, Sarika Chaudhary, Saloni Varshney, Srashti Verma, "Abstractive Text Summarization using NLP", 2020 International Conference on Advances in Computing, Communication & Materials.

[8] https://www.kaggle.com/datasets/snap/amazon-fine-food-reviews?resource=download

[9] https://medium.com/swlh/a-simple-overview-of-rnn-lstm-and-attention-mechanism-9e844763d07b

 DOI : 10.35291/2454-9150.2022.0112