Wiki Summary: A Consensus Approach for Keyword Extraction from Multiple Summarization Algorithms

A thesis is submitted in partial fulfillment of the requirements for the degree of

Master of Technology in Computer Technology

in the

Department of Computer Science and Engineering

Submitted By

Ankita Chowdhury

Examination Roll No. - M6TCT23013

University Registration No.- 154194 of 2020-21

Under the Guidance of

Prof. (Dr.) Subhadip Basu

Department of Computer Science and Engineering

Jadavpur University

JADAVPUR UNIVERSITY

188, Raja S.C. Mallick Rd, Kolkata - 700032,West Bengal, India

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

Certificate of Recommendation

This is to certify that the dissertation entitled "**Wiki Summary: A Consensus Approach for Keyword Extraction from Multiple Summarization Algorithms**" has been carried out by **Ankita Chowdhury** (University Roll No: 002010504029, Examination Roll No: M6TCT23013, University Registration No: 154194 of 2020-21), under the guidance and supervision of **Prof. (Dr.) Subhadip Basu**, Department of Computer Science and Technology, Jadavpur University, Kolkata, is being presented for partial fulfillment of the Degree of Master of Computer Technology during the academic year 2022-2023. The research results presented in the thesis have not been included in any other paper submitted for the award of any degree in any other university or institute.

……..
Prof. (Dr.) Subhadip Basu (Thesis Supervisor)
Department of Computer Science and Engineering
Jadavpur University, Kolkata-32

Countersigned

……..
Prof. (Dr.)Nandini Mukhopadhyay
Head of Department,
Computer Science and Engineering
Jadavpur University, Kolkata-32.

……
Prof. (Dr.) Ardhendu Ghoshal
Dean,
Faculty of Instrumentation and
Electronics Engineering
Jadavpur University, Kolkata-32.

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

Certificate of Approval

(Only in case the thesis is approved)

This is to certify that the dissertation entitled "**Wiki Summary: A Consensus Approach for Keyword Extraction from Multiple Summarization Algorithms**" is a bonafide record of work carried out by **Ankita Chowdhury** (University Roll No: 002010504029, Examination Roll No: M6TCT23013, University Registration No: 154194 of 2020-21) in partial fulfillment of the requirement for the Degree of Master of Computer Technology in the Department of Computer Science and Engineering, Jadavpur University during the period of August 2022 to June 2023. It is understood that by this approval the undersigned does not necessarily endorse or approve any statement made, opinion expressed, or conclusion drawn therein but approve the thesis only for the purpose for which it has been submitted.

Signature of Examiner 1 Date:

Signature of Examiner 2 Date:

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

Declaration of Originality and Compliance of Academic Ethics

I hereby declare that this thesis entitled "**Wiki Summary: A Consensus Approach for Keyword Extraction from Multiple Summarization Algorithms**" contains a literature survey and original research work by the undersigned candidate, as part of her degree of Master of Computer Technology.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name (Block Letters): ANKITA CHOWDHURY

University Roll Number: 002010504029
Examination Roll Number: M6TCT23013
University Registration Number: 154194 of 2020-21

Thesis Title: **WIKI SUMMARY: A CONSENSUS APPROACH FOR KEYWORD EXTRACTION FROM MULTIPLE SUMMARIZATION ALGORITHMS**

Signature with Date:

Acknowledgment

I would like to express my deepest gratitude and sincere thanks to my respected mentor and teacher Prof. (Dr.) Subhadip Basu, Department of Computer Science and Engineering, Jadavpur University for his exclusive guidance and undivided support in completing and presenting the thesis successfully. I am indebted to him for the constant encouragement and continuous knowledge sharing that he has given to me. The above words are only a token of my deep respect towards him for all that he has done to make my thesis its present shape.

I would also like to thank my junior Bhaswata Sarkar for the valuable knowledge sharing sessions and guidance that he has provided during the complete duration of this thesis, without whom completing this thesis wouldn't have been possible. I am grateful to him for everything he has done during the work.

I would also like to thank my friends Debam Saha, Amartya Roy, Satarupa Mal, Saba Nadim, Shahid Ahmed Siddique for constantly supporting and guiding me through the research work.

I would also like to thank my colleagues from Infomaticae Technology Pvt. Ltd. for supporting and motivating me throughout the time.

Finally, I express my sense of gratitude and thanks to my parents for unconditional love and support, and without whom the whole work wouldn't have been possible. Last but not least I am grateful to all my friends, teachers, and mentors who have helped me in the work and provided me with key insights.

Regards,

ANKITA CHOWDHURY

University Roll Number: 002010504029
Examination Roll Number: M6TCT23013
University Registration Number: 154194 of 2020-2021

Department of Computer Science and Engineering, Jadavpur University

Signature with Date:

ABSTRACT

Modern times have seen a massive increase in the amount of text data coming from various sources. The fast expansion of the Internet has made it increasingly challenging to access vast volumes of information efficiently. This lengthy article is a valuable source of knowledge and information that must be skillfully summarized in order to draw out the intriguing details that are concealed inside. We require efficient and powerful technologies in order to manage the enormous amount of information.The primary goal of this effort is to condense a given text into fewer sentences while maintaining the key themes of the original text. With minimal context and information loss, text summarization is a potent text mining technique for extracting the useful information from a document. By removing the less important information, it shortens the language and encourages the user to find the needed information right away. Wikipedia is a vast and diverse source of information, with articles covering a wide range of topics and varying in length and complexity. At the same time, Keyword extraction is an important task in text summarization, and various algorithms have been developed to address this challenge.However, the effectiveness of individual algorithms can be limited by the diversity of text genres and the complexity of the language used. In this paper, we propose a consensus-based approach to wikipedia summarization and keyword extraction that combines the results of multiple summarization algorithms. We evaluate our approach on a dataset of Wikipedia pages and show that it outperforms individual algorithms and other state-of-the-art approaches in terms of content coverage, coherence, and readability. Our results suggest that consensus-based approaches can provide a more comprehensive and accurate summary of Wikipedia pages, particularly for long and complex articles and a more robust and accurate solution to keyword extraction, particularly for complex and diverse texts.

Keywords: Extractive summarization, Single-document source, TextRank, LSA, T5 Transformer, GP2 ,BART, ROUGE ,Text summarization, Wikipedia, Text mining, Abstractive summarization, Encoder-Decoder

Table Of Contents

List Of Figures And Tables

CHAPTER 1

INTRODUCTION

We must first understand what a summary is before moving on to the text summarization. A summary is a text that is created from one or more texts, delivers the key ideas from the original text, and is written in a concise manner.

Automatic text summary aims to deliver the source text in a concise, semantically-rich form. The main benefit of adopting a summary is that it shortens the reading process.

There are two types of text summarizing techniques: extractive and abstractive summarization. Selecting significant sentences, paragraphs, etc. from the original content and concatenating them into a shorter version is an extractive summary method. An abstractive summary explains the key ideas in a paper and then communicates those ideas in simple, natural language.

Text summary can be divided into two categories: indicative and informative. Inductive summary merely conveys to the user the text's core point. This kind of summary often lasts 5 to 10 percent of the original material. Contrarily, informative summarizing algorithms provide brief summaries of the primary content.An insightful summary should be between 20 and 30 percent of the main material in length.

1.1 Some key ideas

The primary steps for text summarization and background are described in this section.

1.1.1 Main steps for text summarization

The process of summarizing documents involves three key parts. Topic identification, interpretation and summary generation are these.

1. Topic Identification: The text's most important information is noted. Position, Cue Phrases, and Word Frequency are some of the different topic identification approaches employed. The best techniques for identifying topics are those that depend on the placement of sentences.

2. Interpretation: An interpretation step is required for abstract summaries. In this step, various topics are combined to create a general content.

3. Summary Generation: The system employs a text generating technique in this stage.

1.1.2 Background

The majority of extractive summarizers in the past relied on scoring sentences in the original text. The most popular and contemporary text summarizing methods either employ linguistic or statistical methods. The sentences are weighted using the high frequency words, standard keyword, Cue Method, Title Method, and Location Methods.

The majority of the current automatic text summarization systems create a summary using an extraction strategy. To create extraction summaries, sentence extraction algorithms are frequently utilized. Sentence scoring, which assigns a numerical value to each sentence for the summary, is one technique for finding appropriate sentences. The best sentences are then chosen to make up the document summary based on the compression rate. The compression rate is a crucial component of the extraction procedure that is used to determine the proportion between the length of the summary and the source text. The summary will grow in size and contain more irrelevant content as the compression rate rises. While the summary becomes shorter due to the compression rate, more information is lost. In fact, the quality of the summary is acceptable when the compression rate is between 5 and 30%.

In an abstractive summary, the text is typically processed using Natural

Language Processing (NLP) methods to produce new language that highlights the most important details from the original text. Information extraction, content selection, and surface realization are the three pipelined processes that make up the foundation of abstractive summarization techniques. Utilizing verb or noun phrases, information extraction extracts pertinent information from text. By using query-based extraction, it can also retrieve crucial information. The process of content selection involves choosing a subset of significant text from the extracted text to include in the final summary. In the surface realization task, words or phrases are combined in an orderly sequence utilizing grammatical rules and lexicons (vocabulary plus knowledge of its usage and significance in language). Approaches for abstractive text summarization can be broadly divided into three categories: deep learning-based approaches, semantic-based methods, and structure-based methods. The structure-based approaches function by encoding data from text documents based on logical arrangements, such as templates or other structures like trees, ontology, lead and body, rules (classes and lists), and graphs. By utilizing linguistic/semantic depiction of a text document as an input to the natural language generation system, semantic-based methods identify noun and verb phrases. These systems include semantic text representation, information item-based approaches, multimodal semantic techniques, and semantic graph techniques. Sequence to sequence models are known to constitute the core of the majority of contemporary studies, and they have been used to increase text summarization in a number of recent ways.

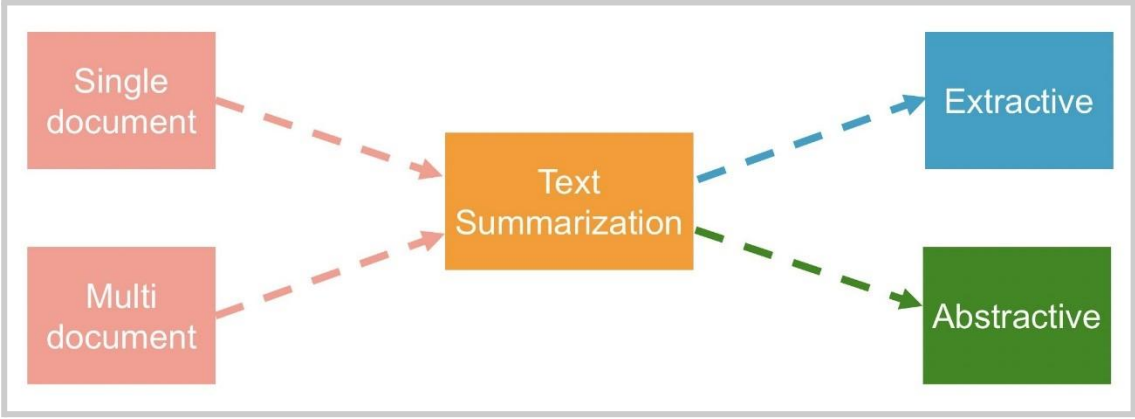


Fig.1: Text Summarization [12]

CHAPTER 2

RELATED WORKS

In essence, over the past ten years, the amount of digital content available online has been growing exponentially at a quicker rate. Information that the users genuinely need has been hindered as a result. Finding the appropriate information from a number of sources is quite difficult for people. Additionally, manually sorting through and summarizing the unstructured data on the internet takes a lot of time. As a result, automatic text summarization is now a necessity of the modern age. The summarizing problem has long piqued the interest of the scholarly community. In general, there are two basic categories of text summary: extractive summarization and abstractive summarization. Up until the early 2000s or the previous decade, the majority of research projects focused heavily on extractive summarization. However, due to the demand for more precise and minimally redundant summaries in recent years, the paradigm has gradually shifted towards abstractive summarization. Some of the earlier efforts in these two techniques are discussed here. [8]

Luhn [56] and Edmundson [57] made the first significant contributions to text summary in the late 1950s and the early 1960s, respectively. The methods were straightforward, concentrating on the placement of sentences and the frequency of words to create a summary made up of quotes from the original text. Since then, numerous initiatives have been made to create fresh methods for text summarising. [3]

The size of the input can be used to categorize text summary. While some strategies, like those in [58], [59], are aimed at single documents, others, like those in [60, 61], are aimed at multi-documents. Another crucial component of text summarization is the summary language. Although the majority of the works are English-focused, some have developed cross-lingual text summarization approaches [64,65] and multilingual approaches [71,62,63]. Depending on the type of summary, such as extractive, abstractive, or hybrid, text summarization can be done. The first strategy entails assigning points to sentences and choosing the subset of high-scored sentences [66]. In the second method, new sentences are created to form the summary from the input document's intermediate representation [67]. The last strategy merely combines the two earlier strategies [70]. When compared to extractive approaches, abstractive approaches have the advantage of being able to produce better summaries using words other than those found in the original document [68]. Many techniques have been created. [3]

In 1958, the idea of using NLP for summarization first emerged. To begin with, each statement was given a score using statistical methods, and the best-scoring sentences were chosen. This score was calculated using a variety of methods, including TF-IDF [24], Bayesian models [25], etc. All of these methods were extractive and merely trimmed the original text, even though they were able to generate a reliable summary through key phrase extraction. The use of machine learning methods [1] for summarization, including Bayesian Learning Models as done in the publication [25], then came into prominence. These machine learning algorithms have been successful at identifying patterns in texts and creating relationships between various terms. In this section, we'll look at the rationale behind and specifics of how summarization techniques were applied. Every text or sentence can be viewed as a sequence of data, and the arrangement of words is crucial for the understanding and production of natural language. The architecture requires information retention with the aid of some memory in order to process sequential data, which is the case for the majority of NLP issues.[1]

The LSTM network [28], one of the RNN variations [26, 27], uses connected nodes to store sequential information by preserving important information and discarding irrelevant information that assisted in producing summaries. The encoder-decoder model [28] was created using this LSTM network technology. The encoder-decoder framework's implementation of Sequ2Seq models for NLP tasks produced excellent results, but parallelization remained a problem. While the sequential information is kept in the encoder-decoder model [29], processing is done by taking one input at a time because LSTM [27] only takes one input at a time. This is a problem because, despite the fact that the model produces better outcomes, it fails in every scenario and defies the goal of establishing machine perception.[1]

As a result, the Attention layer was included [27]. The encoder-decoder model's attention layer [29] analyzes the input sequence at each stage and gives it a weight depending on the previous sequences. Every word in the sentence is taken into account for one input by the attention layer [30], which produces vector matrices. Because of this, the attention layer compels the computer to read the entire text as one input rather than treating different text sequences as separate inputs. This method gained popularity because it was so effective [30] for the abstractive approach. As a starting point for this effort, we used the Google-developed transformer architecture [30][1].

The concept of transformers, described in [72], was a breakthrough that completely changed the NLP industry and helped to deal with long-term dependencies and parallelization during training. The BERT (Bidirectional Encoder Representations from Transformers) representation model, which was developed later, is a pre-trained model that takes into account a word's context from both sides (left and right) for the first time. BERT allows for multi-task learning while also improving knowledge of the context in which a word has been used. To create new models for various languages, BERT can be adjusted. More recently, highly large and diverse French corpora have been used to train models based on BERT, such as CamemBERT [74] and FlauBERT [75]. A variety of NLP tasks, such as text classification, paraphrasing, and word sense disambiguation, have been applied to the developed French models. It is not simple to apply the BERT-based approach to summarization, nevertheless. In actuality, the output of BERT-based models is typically either a class label or a span of the input. It is suggested in [76,77] to use a BERT-based model for text summarizing tasks. The NewYork Times Annotated Corpus, Xsum, and CNN/DailyMail news highlights datasets were used to test the generated models. It has recently been suggested to use the Text-to-Text transfer

transformer (T5) paradigm and framework, which achieves cutting-edge outcomes on a number of benchmarks, including summarization [78]. As far as we are aware, no T5-based effort has addressed text summarizing using the French Wikipedia. Additionally, the abstractive summarization process has not yet taken the character count limitation into account[3].

Numerous studies have been conducted to examine various summarization algorithms, leading to the publication of a number of research articles for the aforementioned reason. We sought to efficiently implement and optimize our models for use in evaluating the performance of summarization techniques [85], coming to a conclusion with specific results. Different summarization methods for both single and multiple documents were taught to us [86]. Through this paper, we learned about some of the most popular approaches, including frequency-driven approaches, topic representation strategies, graph-based approaches, and machine learning techniques [87]. A thorough investigation has shed light on current developments and trends in automatic summarizing methods [88], which outline the state of the art in this field of study. There are primarily two categories of summarizing methods[5].

CHAPTER 3

LITERATURE REVIEW

In the research process, the literature review plays a critical role. It's a place where researchers get their research ideas, which are then refined into concepts and finally theories. It also gives the researcher a bird's eye view of previous research in the field. A researcher will know where his or her research stands based on the findings of the literature review. This chapter includes the literature survey of the methodologies used for text summarizations.

3.1 Text Mining

Text mining, which employs various Artificial Intelligence methods, is also known as text analysis. Text mining is the process of extracting high-quality information from unstructured data by creating patterns and locating crucial keywords. Text mining tasks include text classification—classifying the text according to factors like genre—sentimental analysis—telling us how the author phrased certain sentences and what tone they were written in—document clustering—clustering the documents using unsupervised learning—and text summarization—creating a concise and focused text. Each task differs from the others and has its own methodology and investigation.

The goal is to extract useful numerical indices from the text from the unstructured material. Make the text's information accessible to the various algorithms as a result. The documents' information can be extracted to create summaries. As a result, you can examine individual words and word groups in texts. Text mining, to put it simply, "turns text into numbers." such as the use of unsupervised learning techniques in predictive data mining projects [4].

3.1.1 Information Extraction

This is the most famous text mining technique [79]. Information exchange refers to the process of extracting meaningful information from vast chunks of textual data. This text mining technique focuses on identifying the extraction of entities, attributes, and their relationships from semi-structured or unstructured texts. Whatever information is extracted is then stored in a database for future access and retrieval. The efficacy and relevancy of the outcomes are checked and evaluated using precision and recall processes. In most of the cases, this activity includes processing human language texts by means of NLP [4].

3.1.2 Information Retrieval

The technique of identifying pertinent and connected patterns based on a particular set of words or phrases is known as information retrieval (IR) [79]. IR systems use several algorithms in this text mining technology to follow and monitor user behaviors and find pertinent data as a result. Document retrieval is considered to be an extension of information retrieval. that the returned papers undergo condensing processing. Retrieval of the documents is thus followed by a stage of text summarizing. That concentrates on the user's question. The number of papers that are pertinent to a problem can be reduced with the use of IR technologies. Due to the fact that text mining uses extremely sophisticated algorithms on big document sets. By limiting the quantity of documents, IR can also considerably speed up the analysis. The two most well-known IR systems are Google and Yahoo [4].

3.1.3 Categorization

One of the "supervised" learning methods used in text mining, this method categorizes natural language documents according to their content and assigns them to one of a predefined set of subjects. In text mining, categorization refers to grouping texts. Customer reviews, help requests, and other types of text documents can be automatically categorized based on their contents using a combination of natural language processing (NLP) and machine learning. To

find the appropriate subjects or indexes for each text content, categorization [79] or Natural Language Processing (NLP) is a technique of gathering and processing text documents. In NLP, the co-referencing technique is frequently used to extract pertinent synonyms and abbreviations from textual input. NLP is being employed as an automated process in a variety of applications, including the distribution of personalized advertisements, the detection of spam, the classification of web pages according to hierarchical criteria, and much more [4].

3.1.4 Clustering

One of the most important text mining approaches is clustering [79]. It looks for inherent textual informational structures and classifies them into useful subgroups or 'clusters' for additional examination. The development of meaningful clusters from unlabeled textual data without any prior knowledge of them is a significant difficulty in the clustering process. Standard text mining tools like cluster analysis help distribute data or serve as a pre-processing step for other text mining algorithms that run on identified clusters [4].

3.1.5 Summarization

The practice of automatically creating a compressed version of a certain text that contains useful information for the end-user is referred to as text summarization [79]. This text mining technique aims to browse through numerous text sources to provide succinct summaries of texts that contain a significant amount of information while essentially maintaining the overall content and intent of the original documents. The numerous text classification techniques, including decision trees, neural networks, regression models, and swarm intelligence, are integrated and combined in text summarization [4]. There are broadly two different approaches that are used for text summarization:

Extractive text summarization: The name itself gives what this approach does. This method's purpose is obvious from the name alone. From the original text, we find out the key clauses or words, and we only take those out. That compiled text would serve as our summary.

Abstractive summarization: This is a highly intriguing strategy. Here, we take the original content and create new sentences. This is in contrast to the extractive strategy we previously saw, where we only utilized the sentences that were present. It's possible that the sentences produced by abstractive text summary weren't in the original text.

3.2 Natural Language Processing

The study of how human languages and computers interact is at the heart of the subject of computer science known as natural language processing (NLP) (Chowdhury, 2003). The best method for assisting the system in reading and comprehending the text provided. The automatic software manipulation of natural language, such as speech and text, is known as natural language processing, or NLP for short. Unstructured data can be created from conversations, statements, or even tweets. The majority of data available in the real world is unstructured, which does not cleanly fit into the standard row and column structure of relational databases. It is chaotic and challenging to control. However, a significant revolution in this field is currently taking place as a result of advancements in fields like machine learning. Nowadays, understanding the meaning behind a text's or a speaker's words is more important than attempting to interpret it based on its keywords (the antiquated mechanical method). On this basis, one may recognize rhetorical devices like irony or even carry out sentiment analysis. It is a discipline that focuses on how data science and human language interact and is expanding to many other sectors. Today, NLP is flourishing as a result of the enormous advances in data access and the rise in computing power, which enable practitioners to produce significant results in industries like healthcare, media, finance, and human resources, among others [2].

Pre-trained language models make use of techniques for universally learning the parameters of neural network topologies and language representations, and they can be adjusted for a range of downstream tasks. These models were categorized as first- and second-generation pretrained language models in reference [42]. Word embeddings are pre-trained models of the first generation (words are represented as vectors), and these include word2vec, GloVe, etc. Pre-trained contextual encoders are models of the second generation (or more advanced models), and these include BERT[37], GPT[38], and ULMFiT(Universal Language Modeling fine Tuning), among others[2].

The definition of a language model is a mathematical function that applies a probability measure to a set of strings taken from a language's vocabulary. These systems have the ability to analyze language material using an algorithm and learn to anticipate the words of a sentence by calculating the likelihood of a word sequence. In statistical language models for a sequence Shaving N-words, the probabilities are assigned as,

$$P(S) = P(w_1 w_2 w_{3.....} w_N) \qquad (1)$$

$$P(S) = P(w_1)P(w_2 \mid w_1) \ldots P(w_N \mid w_1 w_2 w_{3.....} w_{N-1}) \qquad (2)$$

To reduce the parameters from the above equation, an approximate method was required. The n-gram model estimates the next word from the previous n-1 words.

$$P(w_N \mid w_1 w_2 \ldots\ldots w_{N-1}) \approx P(w_N \mid w_{N-n} \ldots\ldots w_{N-1}) \quad (3)$$

On the other hand, Neural Network Language Models (NNLM) were developed to address the shortcomings of Statistical Language Models. It was able to increase efficiency and get around the drawbacks of traditional language models with NNLM [39]. These models have shown to be more successful at language modeling problems because they can automatically learn features and representation. The first neural network language model was proposed by [40], as illustrated in Fig. 2.

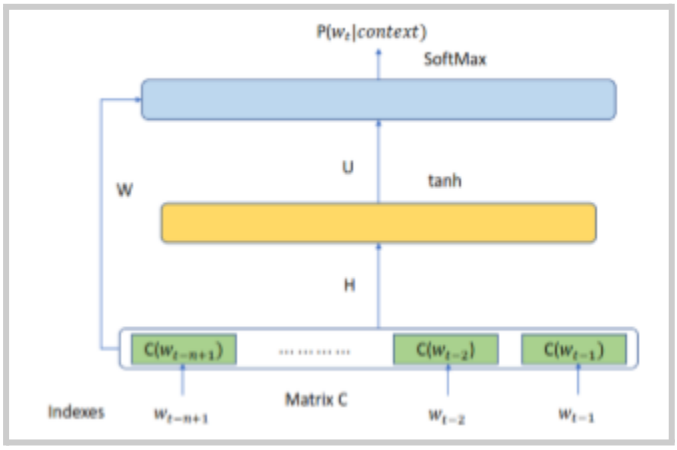


Fig.2: Neural network language model [2]

The model in Fig. 2 can be expressed as,

$$y = b + W_x + U \tanh d + H(x) \quad (4)$$

where x is the feature vector formed by the concatenation of input word features, x = $(C(w_{t-1}), C(w_{t-2}), \cdots, C(W_{t-n+1}))$ and y is the probability of the output word. W, U, and H are weight matrices while b and d are the corresponding biases of the hidden and output layers.

For learning distributed word representations, two models—Continuous Bag-of-Words (CBOW) and Continuous skip-gram shown in Fig.4 and Fig.5 —were proposed by[41]. The CBOW model learns by predicting a word from the context words (past and future)., for example, a word w(t) is predicted from a shared projection of the input words *w(t−1), w(t−2), w(t+1) and w(t+2).* A variety of surrounding words are predicted by the skip-gram model to learn representation, for example, with w(t) as input to a log-linear classifier it is possible to predict *w(t−1), w(t−2), w(t+1) and w(t+2).*

3.2.1 Word Embedding

A group of NLP feature learning algorithms called word embedding map words to vectors of real values. According to Mikolov, Sutskeve, Chen, Corrado, and Dean (2013), it creates a relationship between words and permits calculations among them by allowing similar words to have similar representations. As a typical illustration, the vector representation of the word "queen" would ideally result from the function "king - men + women" once words have been converted to vectors. When working with natural language processing, word embedding has the advantage of capturing more of the word's meaning and frequently enhancing task performance[4].

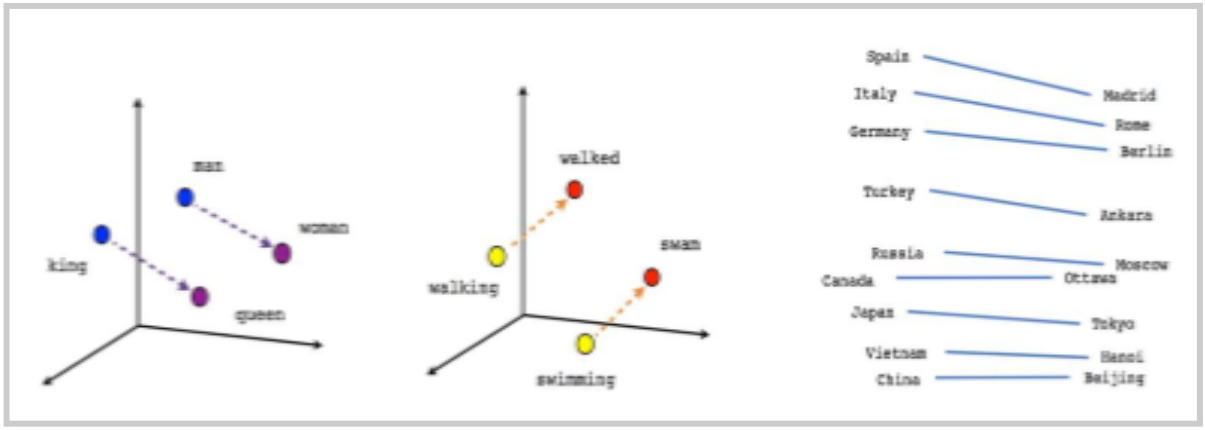


Fig.3: Word Embedding [4]

The most popular names in word embeddings are Word2vec by Google (Mikolov) and GloVe by Stanford (Pennington, Socher and Manning). Let's delve deeper into these word representations.

In Word2vec, Every word in a fixed vocabulary is represented by a vector in the enormous corpus of text . We then go through each position t in the text, which has a centre word c and context words o. Next, we use the similarity of the word vectors for c and o to calculate the probability of o given c (or vice versa). We keep adjusting the word vectors to maximize this probability.

Meaningless (or higher frequency) words like "a," "the," "of," and "then" can be removed from the dataset in order to effectively train Word2vec. This decreases training time and increases model accuracy. In addition, we can use negative sampling for every input by updating the weights for all the correct labels, but only on a small number of incorrect labels. Word2vec has 2 model variants worth mentioning [4].

3.2.1.1 Skip-Gram

One unsupervised learning method used to discover the words that are most connected to a given word is the skip-gram. To determine the context word for a given target word, skip-gram is employed. It is the CBOW algorithm inverted. Here, the target word is input while context words are output. This problem is challenging since there are multiple context words that must be predicted. We take into account a context window with k consecutive words.. Then we skip one of these words and try to learn a neural network that gets all terms except the one skipped and predicts the skipped term. Therefore, if 2 words repeatedly share similar contexts in a large corpus, the embedding vectors of those terms will have close vectors [4].

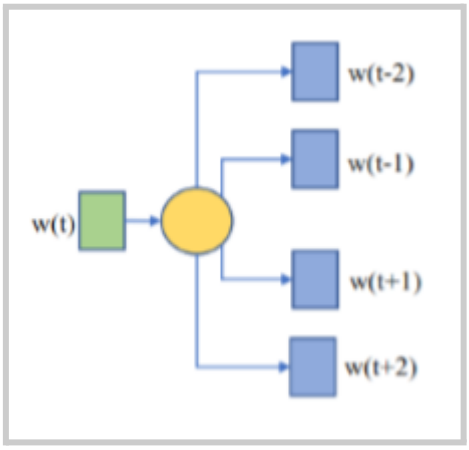


Fig.4: Skip-Gram Model [2]

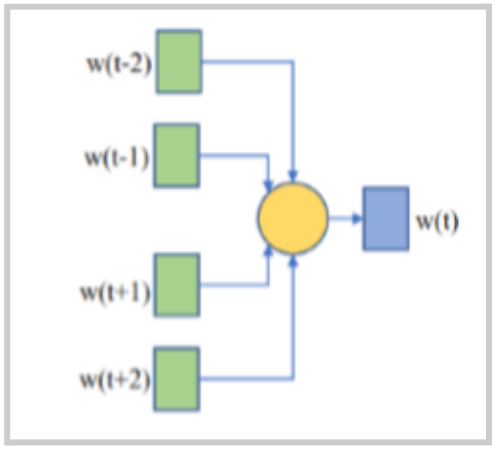


Fig.5: Continuous Bag of Words Model [2]

3.2.1.2 Continuous Bag of words

We extract several sentences in a large corpus. Each time we see a word, we also take the words around it. The word in the centre of this context is then predicted using the context words as input to a neural network. We have one instance of a dataset for the neural network when we have thousands of these context words and the centre word. After training the neural network, the encoded hidden layer output ultimately represents the word embedding. It so happens that words in similar contexts receive similar vectors when we train this over several sentences. One complaint of CBOW and Skip-Gram is that they are both window-based models, which means that the co-occurrence statistics of the corpus are not effectively utilized, resulting in suboptimal embeddings [4].

3.2.2 Neural Machine Translation

The method known as Neural Machine Translation [80] models the entire process using a single, massive artificial neural network called a Recurrent Neural Network (RNN). RNNs are stateful neural networks that have connections across passes and throughout time. In addition to receiving information from the previous layer, neurons also receive information from their own previous pass. This means that the order in which we feed the input and train the network matters: feeding it “Donald” and then “Trump” may yield different results compared to feeding it “Trump” and then “Donald” [4] .

A vast artificial neural network is used in the neural machine translation method to predict the likelihood of word sequences, frequently in the form of complete sentences. Unlike statistical machine translation, which consumes more memory and time, neural machine translation, NMT, trains its parts end-to-end to maximize performance. NMT systems have lately defeated conventional translation systems in the race for machine translation leadership.

Fig.6: A Recurrent neural network for Machine Translation [4]

Machine translation is the task of automatically converting source text in one language to text in another language. There is no one best translation from one language to another for a given sequence of text in the source language. This is due to the ambiguity and adaptability of human language in its natural state. This makes the automatic machine translation task challenging, possibly one of the most challenging in artificial intelligence. The principles for translating text from the source language into the target language are frequently used in traditional machine translation techniques. . The rules are often developed by linguists and may operate at the lexical, syntactic, or semantic level. This focus on rules gives the name to this area of study: Rule-based Machine Translation or RBMT. The complexity of developing the rules and the sheer volume of rules and exceptions necessary are the two main drawbacks of traditional machine translation techniques[4].

3.3 Deep Learning

Artificial neural networks, a class of algorithms inspired by the structure and operation of the brain, are the focus of the machine learning discipline known as deep learning. Deep learning, also known as deep neural learning or deep neural network, is a subset of machine learning in artificial intelligence (AI) that comprises networks capable of learning unsupervised from input that is unstructured or unlabeled. From a biological neural network, it has evolved [4].

A computer model learns to carry out classification tasks directly from images, text, or sound using deep learning. state-of-the-art accuracy can be achieved by deep learning models, sometimes even outperforming human ability. A substantial collection of labeled data and multi-layered neural network architectures are used to train models [4].

There are two main reasons it has only recently become useful:

- A lot of labeled data is necessary for deep learning. For instance, creating a driverless car involves millions of photos and countless hours of video.

- Deep learning calls for a lot of computing power. Deep learning is facilitated by the parallel architecture of high performance GPU. This enables development teams to cut the training period for a deep learning network from weeks to hours or less when paired with clusters or cloud computing.

3.3.1 Artificial Neural Network

Computer programmes that emulate biological neural networks are called artificial neural networks. Such systems learn tasks by considering examples and usually without any prior knowledge. For instance, each email in a dataset is manually classified as "spam" or "not spam" in an email spam detector. Artificial neural networks develop their own set of relevant features between the emails and whether a new email is spam by analyzing this dataset.

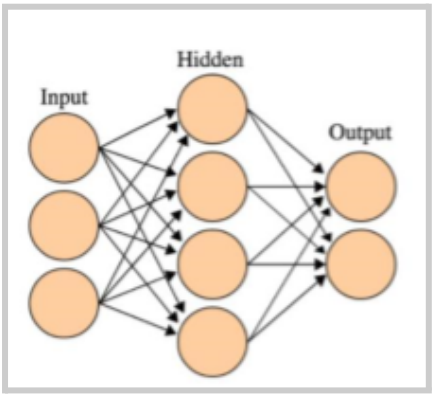


Fig.7: Simple neural network model [6]

The components that make up artificial neural networks, which can extend further, are often arranged in a series of layers. The most typical architecture of a neural network model is shown in Figure 7. It has three different kinds of layers: the input layer has units that typically receive inputs in the form of numbers; the output layer has units that "respond to the input information about how it is learned any task;" and the hidden layer, which has units between the input layer and the output layer and is responsible for converting the inputs into something the output layer can use (Schalkoff, 1997) [6].

Fig. 8. The input layer is the first layer, while the output layer is the last. There are layers that are hidden in between. The data moves sequentially through the layers. The hidden layers are where learning happens. Each layer's nodes are linked to all the nodes in the next layer.. Weights are the variables corresponding to each of these connections [2].

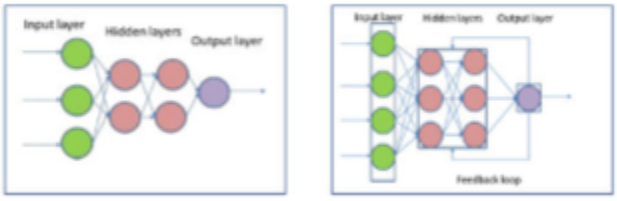


Fig 8: Feedforward and recurrent neural network [2]

InFig.9, node j receives incoming signals from every node i in the preceding layer. Each input X_i is associated with a weight W_{ji}.

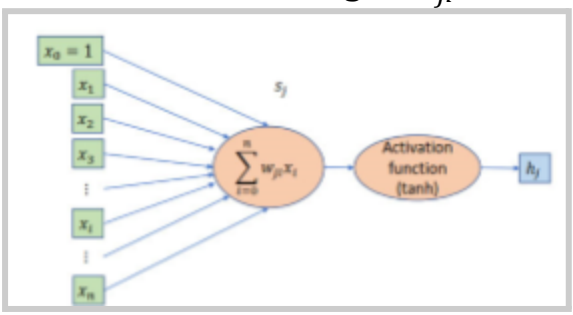


Fig 9: A node of the neural network [2]

The incoming signal to node j is the weighted sum of all incoming signals [2]. The signal then goes through the activation function to produce the output signal h_j. The network can be mathematically expressed as:

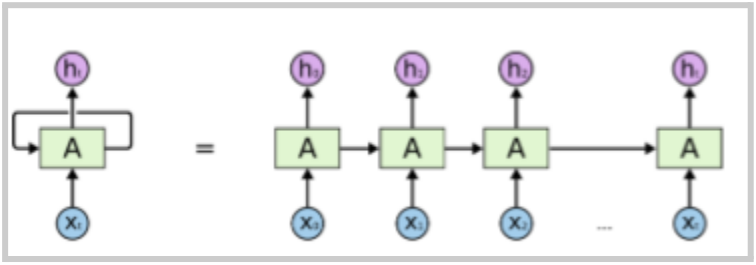
$$S_j = \sum_{i=0}^{n} W_{ji}X_i \qquad (5)$$

$$h_j = \tanh s_j \qquad (6)$$

$$h_j = 1 - (2 / e^{2sj} + 1) \qquad (7)$$

3.3.2 RNN and LSTM

When developing an understanding of the task from the examples provided, traditional neural networks do not remember any prior work. However, the order of words in input documents is crucial for tasks like text summarization. Here, we want the model to keep track of the previous words when it processes the next one. Recurrent neural networks, which feature loops where information can persist in the model, are what we must need to be able to accomplish that (Christopher, 2015). Fig. 10 shows the unrolled appearance of a recurrent neural network (RNN). For the symbols in the figure, "h_t" represents the output units value after each timestamp (if the input is a list of strings, each timestamp can be the processing of one word), "x" represents the input units for each timestamp, and A means a chunk of the neural network [6].



An unrolled recurrent neural network.

(Christopher, 2015)

Fig.10: An unrolled recurrent neural network [6]

Fig.10 shows that the result from the previous timestamp is passed to the next step for part of the calculation that happens in a chunk of the neural network. As a result, the data is obtained from the prior timestamp. The increasing distance between the connected pieces of information, however, makes it difficult for traditional RNNs to recall information well in practice. It is challenging to go back hundreds or thousands of processes to retrieve the information since each activation function is nonlinear [6].

Fortunately, information can be conveyed over large distances using large Short-Term Memory (LSTM) networks. In contrast to the traditional RNN, each LSTM cell has a number of straightforward linear operations that enable data to be conveyed without requiring complicated calculation.

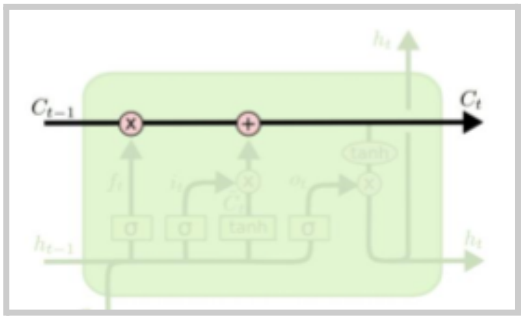


Fig.11: LSTM Cell [6]

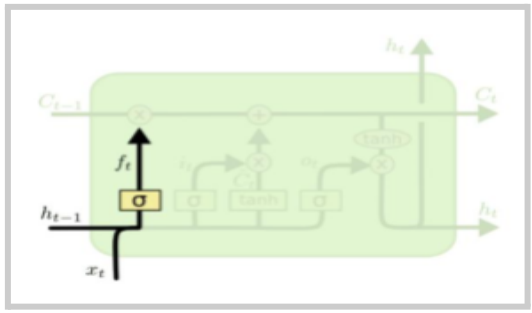


Fig.12: The forget gate layer [6]

As shown in Fig.11, the previous cell state containing all the information so far smoothly goes through an LSTM cell by doing some linear operations. Inside, each LTSM cell makes decisions about what information to keep, and when to allow reads, writes and erasures of information via three gates that open and close [6].

As seen in Fig. 12, the first gate is called the “forget gate layer”, which takes the previous output units value h_{t-1} and the current input x_t , and outputs a number between 0 and 1 to indicate the ratio of passing information. 0 means

do not let any information pass, while 1 means let all information pass. To decide what information needs to be updated, LSTM contains the "input gate layer". It also takes in the previous output units value h_{t-1} and the current input x_t and outputs a number to indicate inside which cells the information should be updated. Then, the previous cell state C_{t-1} is updated to the new state C_t.

The last gate is "output gate layer", which decides what the output should be. Fig. 14 shows that in the output layer, the cell state is going through a tanh function, and then it is multiplied by the weighted output of the sigmoid function. So, the output units value h_t is passed to the next LSTM cell (Christopher, 2015) [6].

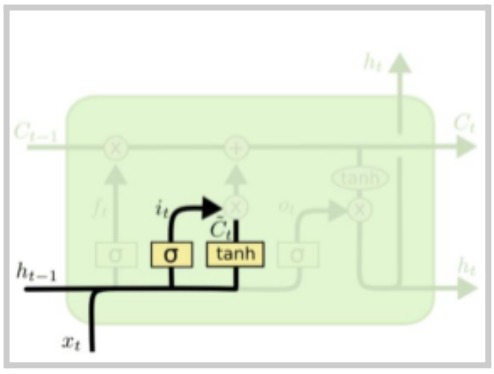


Fig.13: The input gate layer [6]

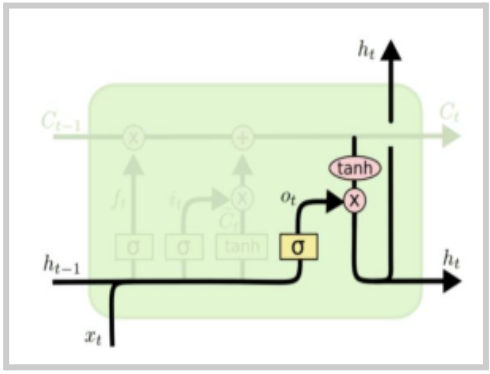


Fig.14: The output gate layer [6]

Simple linear operators connect the three gate layers. The vast LSTM neural network consists of many LSTM cells, and all information is passed through all the cells while the critical information is kept to the end, no matter how many cells the network has [6].

3.3.3 Convolutional Neural Network (CNN)

A CNN is built differently than an ordinary neural network. The neurons in one layer do not have a connection to all neurons in the following layer but only to a specific part. Near the network's termination, there are layers that are fully connected. Some (one or more) layers of the network, some pooling operations, and some of the fully connected neuron layers are used in this type of network as convolution units (where convolution operations are performed). The feature extraction portion of a CNN (where convolutions and pooling operations are carried out) and the classification portion (completely connected layers close to the output) are both separate components. The layers are arranged in the weight, height, and depth dimensions, in contrast to simple neural networks. The result is a vector of probability scores. In the work of [43], abstractive text summarization was accomplished using a CNN-based

deep learning approach. This method was distinct from others in that it searched for significant semantic phrases rather than individual sentences from the text and used those phrases to produce a summary [2].

Multilayer perceptron versions that have been regularized are CNNs. Fully linked networks, or multilayer perceptrons, are typically understood to mean that each neuron in one layer is connected to every neuron in the next layer. These networks are susceptible to overfitting data because of their "fully connectedness". Adding some kind of magnitude measurement of weights to the loss function is a typical method of regularization. CNNs tackle regularization differently; they make use of the data's hierarchical pattern to piece together more complicated patterns out of smaller, simpler ones. CNNs are therefore at the lower end of the connectivity and complexity spectrum. Convolutional networks were influenced by biological processes[81][82][83][84] because of the way that neurons are connected to one another, which is similar to how the visual cortex of an animal is set up. Only in the constrained area of the visual field known as the receptive field do individual cortical neurons respond to inputs. Different neurons' receptive fields partially overlap one another to cover the entire visual field [4].

Comparatively speaking to other image classification algorithms, CNNs employ a minimal amount of pre-processing. This implies that the filters, which were manually designed for traditional techniques, are learned by the network. This feature design's independence from prior knowledge and human effort is a significant benefit [4].

The principal applications of a convolutional neural network (CNN), which comprises one or more convolutional layers, are image processing, classification, segmentation, and other auto correlated data. In essence, a convolution is a filter that is dragged over the input [4].

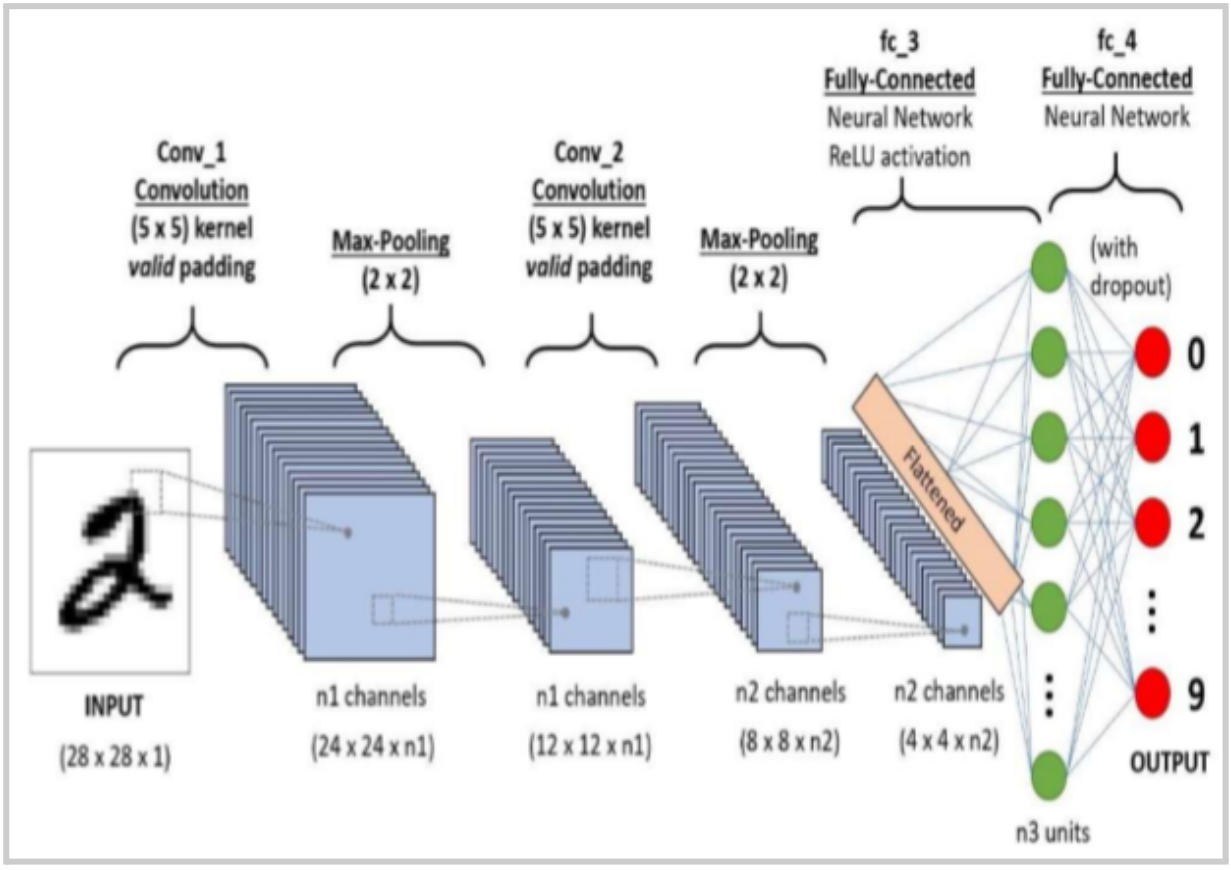


Fig.15:Architecture of Convolutional neural network [4]

A CNN can also be created using a U-Net architecture, which is essentially two nearly mirrored CNNs combined to create a CNN with a U-shaped architecture. U-nets are utilized for tasks like segmentation and image enhancement where the output and input sizes must match [4].

A group of filters known as convolutional kernels are present in each convolutional layer. The filter, which has the same size as the kernel, is an integer matrix applied to a portion of the input pixel values. In order to make things simpler, each pixel is multiplied by the kernel value that corresponds to it, and the result is then summed to create a single value that represents a grid cell—just like a pixel—in the output channel/feature map. These transformations are linear, and each convolution is a particular kind of affine function [4].

3.3.4 Sequence To Sequence Models:

These models have sequences for both the input and the output. Machine translation, image captioning, video captioning, speech recognition, language modeling, question answering systems, text entailment, and text summarization are a few examples of sequence-to-sequence modeling issues. These models are encoder-decoder models. Each of the encoders and the decoder is an independent neural network. A single substantial neural network is created by combining these two networks. The encoder takes an input sequence and builds its representation. The decoder's job is to accept the representation and decode it in order to provide another sequence as output [2].

3.4 ROUGE-N and BLEU Metrics

Recall-Oriented Understudy for Gisting Evaluation is referred to as ROUGE. A machine-generated summary is scored using a set of metrics that are derived from one or more reference summaries written by humans. The examination of N-gram recall across all reference summaries is known as ROUGE-N. According to Lin, Chin-Yew (2004), the recall is computed by dividing the number of overlapping words over the total number of words in the reference summary. In contrast to ROUGE, the BLEU metric is based on N-grams precision. According to (Pineni, Kishore, et al. 2002), it describes the percentage of the words in the machine generated summary overlapping with the reference summaries. The ROUGE-1 score would be 5/8 and the BLEU score would be 5/5, for instance, if the reference summary was "There is a cat and a tall dog" and the generated summary was "There is a tall dog". This is so because there are 5 overlapping words and 5 and 8 words, respectively, in the system summary and reference summary. When working with text summarization, these two metrics are most frequently used [4].

3.5 Libraries

3.5.1 Keras

A popular Python package for machine learning is called Keras, and it was first released in 2015. Numerous implemented activation functions, optimizers, layers, etc. are present in Keras. So it makes it possible to quickly and conveniently build neural networks. François Chollet created and maintains Keras, which is compatible with Python 2.7-3.6 (Keras.io, n.d.).

3.5.2 NLTK

A text processing library called Natural Language Toolkit (NLTK) is widely used in natural language processing (NLP). It offers high-performance tokenization, parsing, categorization, and other processes. It was first made available by the NLTK team in 2001 (Nltk.org, 2018).

3.5.3 Scikit-learn

Python has a machine learning library called Scikit-learn. It carries out simple dimensional reduction techniques like Principal Component Analysis (PCA), clustering techniques like kmeans, regression techniques like logistic regression, and classification techniques like random forests (Scikit-learn.org, 2018).

3.5.4 Pandas

Pandas offers a framework that is adaptable for working with data in a data frame. Many open-source data analysis tools are included that are written in Python, including ways to reshape data structures, merge data frames, and check for missing data ("Pandas", n.d.).

3.5.5 Gensim

The Python module Gensim enables topic modeling. Using several effective algorithms, including Tf-Idf and Latent Dirichlet Allocation, it can analyze raw text data and identify the semantic structure of incoming text data (Rehurek, 2009).

3.5.6 Sumy

Contrary to abstractive techniques, which conceptualize a summary and paraphrase it, extractive text summarization techniques perform summarization by selecting portions of texts and creating a summary. Recently, I came across sumy by miso-belica, which abstracts users from implementing these algorithms on their own.

3.5.7 BART

Bidirectional and Auto-Regressive Transformers (BART) uses a left-to-right decoder (similar to GPT) and a bidirectional encoder (similar to BERT) in a left-to-right machine translation architecture. With the novel in-filling scheme used in the pretraining task, spans of text are replaced with a

single mask token. In BART, input texts are first passed through the bidirectional encoder, or the BERT-like encoder. After that, the texts are looked at from left-to-right and right-to-left. The resulting output is then sent into the autoregressive decoder, which predicts the output based on the input from the encoder and the so far predicted output tokens. Let's first install HuggingFace Transformers to implement BART in the above data. This library, which is built on top of Pytorch and TensorFlow, can be used for a number of linguistic tasks, including question- and answer-based summarization.

3.5.8 GloVe

Global Vectors for Word Representation (GloVe), GloVe is a distributed word representation model that outperforms other models in terms of word similarity, word analogies, and named entity recognition. It was first developed by a group of Stanford students in 2014. (Pennington, Richard and Christopher, 2014).

3.5.9 LXML

The XML toolkit lXML, which is a Python API, is bound to the C libraries libXML2 and libxslt. LXML can parse XML files faster than the ElementTree API, and it also derives the completeness of XML features from libXML2 and libxslt libraries (LXML.de, 2017)

3.5.10 BeautifulSoup

Beautiful Soup is a Python library that is used for web scraping purposes to pull the data out of HTML and XML files. It creates a parse tree from page source code that can be used to extract data in a hierarchical and more readable manner. Beautiful Soup provides a few simple methods and Pythonic idioms for navigating, searching, and modifying a parse tree. It sits atop an HTML or XML parser, providing Pythonic idioms for iterating, searching, and modifying the parse tree.Beautiful Soup is a powerful tool for anyone who needs to extract and process data from HTML or XML files.

3.5.11 PyTorch

PyTorch is defined as an open source machine learning library for Python. It is used for applications such as natural language processing. It is initially developed by Facebook artificial-intelligence research group, and Uber’s Pyro software for probabilistic programming which is built on it.Originally, PyTorch was developed by Hugh Perkins as a Python wrapper for the LusJIT based on

Torch framework.It is easy to debug and understand the code,includes many layers as Torch,lot of loss functions.It can be considered as NumPy extension to GPUs and allows building networks whose structure is dependent on computation itself

3.5.12 Matplotlib

Matplotlib is a Python library used for creating a wide range of charts and plots. It provides customization options and integrates with other Python libraries like NumPy and Pandas. Matplotlib is a powerful tool for data visualization.

3.5.13 Matplotlib_venn

Matplotlib_venn` is a Python library that is used for creating Venn diagrams, as mentioned earlier. It provides a simple and intuitive interface for creating high-quality Venn diagrams, with customizable colors, labels, and titles.

3.5.14 Wordcloud

`Wordcloud` is a Python library for generating word clouds from text data. Word clouds are a popular visualization technique that displays words in different sizes based on their frequency in a given text. The library provides customization options for color, font, and shape of the word cloud.

CHAPTER 4

DISCUSSION

Before beginning a new investigation, a discussion builds familiarity with and comprehension of current research in a certain topic. We should be able to find out what research has already been done and discover what is unknown about our topic by conducting a literature review. The important findings from the previous chapter's literature review are discussed in this chapter.

Connectionist Approach To Generic Text Summarization

Here, automatic large document summarization is the goal. This approach makes use of an adaptive, incremental learning, knowledge representation system that evolves in terms of structure and functionality. This approach proposes using a live website or the url of a Wikipedia Live Article while employing various text summarization models and paradigms that can handle live url data.

In essence, there are two types of summaries: extractive and abstractive. While extractive summaries rely on copying exact sentences from the source document, abstractive summaries use Natural Language Processing (NLP).

TEXT SUMMARIZATION HISTORY:In the past, extractive summarizers have been mostly based on scoring sentences in the source document. The most frequent and latest text summary techniques involve either statistical methodologies, or linguistic techniques. The high frequency words, standard keyword, Cue Method, Title Method, Location Method are utilized for weighing the sentences [12].

Ranking Of Text Units According To Shallow Linguistic Features: This method minimizes the use of weak linking sentences in the resultant text summary by identifying the most important text or phrases using a variety of superficial linguistic factors and taking into account the degree of connectivity among the text units. This method highlights the effect of lexical chain scoring after the nouns and compound nouns are chained by searching for lexically organized relationships between words in the text using WorldNet and using lexicographical relationships such as synonyms and hyponyms. In order to derive a summary, all the sentences are ranked or given preferences based on the total scores of the words in each sentence. Term frequencies, trigger words and phrases, assessing lexical similarities (measuring chain score, word score, and finally sentence score), and other factors are used to determine the scores of words [11].

4.1 Extractive Summarization

Since extractive summarization techniques [89, 90] are statistically oriented, they require less time, resources, and computing power to evaluate and produce a summary, making them computationally more feasible to implement. However, by locating the imperative sentences, the techniques produce a summary. The keywords are detected in a given text based on the frequency of their occurrences. The method might not effectively utilize the provided information or it might exclude certain important details [91] [5].

Extractive summary strategies focus on summarizing a textual source by picking words or sentences that are essential to the context or appear more frequently [92].The techniques for extractive summarization score or weight words or sentences and use significant or equivalent pieces to generate a summary. Various methods and mathematical calculations are used to assign loads or scores for the words/sentences, which are further used to position the sentences/words according to their significance and comparability [105] [5].

Information record → sentences closeness → score sentences → selection of sentences with higher significance.

For extractive summarization, dominant techniques including TF-IDF and Textrank (Hasan, Kazi Saidul, & Vincent Ng, 2010). Textrank was first introduced by Mihalcea, Rada, and Paul Tarau in their paper Textrank: Bringing order to text (2004). The paper proposed the idea of using a graph-based algorithm similar to Google's Pagerank to find the most important sentences. Juan Ramos proposed TF-IDF (2003). He explored the idea of using a word's uniqueness to perform keyword extraction. This kind of extraction can also be applied to an entire sentence by calculating the TF-IDF of each word in the sentence [5].

4.1.1 TextRank

4.1.1.1 Description

TextRank is an unsupervised technique that may be used to automatically summarize texts and extract the most significant keywords from a page. Rada Mihalcea and Paul Tarau first mentioned it in [94]. A PageRank [95] variant is applied by the algorithm to a graph created especially for the goal of summarization. The items in the graph are then ranked according to how well they explain the text; these are the most significant elements. With this method, TextRank can generate summaries without the need for labeling or a training corpus and can apply the algorithm to a variety of languages [7].

4.1.1.2 TF-IDF

The importance of a word in the document is assessed using the Term Frequency-Inverse Document Frequency (TF-IDF) method (Ramos and Juan, 2003). The underlying algorithm determines the frequency of the word in the document (term frequency) and multiplies it by the inverse document frequency, which is the logarithmic function of the number of documents containing that word over the total number of documents in the dataset. One can determine the importance of each phrase by computing the relevance of each word. These sentences can then be used to create a summary of the document, presuming that the most pertinent sentences are also the most significant ones [12].

4.1.1.3 Text as a Graph

Using sentences as nodes in a graph, TextRank models any content for automatic summarization [106]. To create connections between sentences, a function to compute sentence similarity is required. The weight of the graph

edges is determined by this function; the more similar two phrases are, the more significant the graph edge connecting them will be. We can claim that we are more likely to move from one sentence to another if they are quite similar in the context of a Random Walker, as utilized commonly in PageRank [95] [7].

Based on the material that both sentences have, TextRank assesses how similar two sentences are to one another. To prevent encouraging big sentences, this overlap is simply calculated as the number of shared lexical tokens between them divided by the length of each. The original algorithm's function can be formalized as follows: Given S_i, S_j, two sentences represented by a set of n words, that in $S_i = w^i_1, w^i_2, ..., w^i_n$. S_i and S_j's similarity function is represented by the following equation:

$$Sim(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|) + log(|S_j|)} \quad (8)$$

This procedure yields a dense graph that represents the document. PageRank is applied to this graph to determine the significance of each vertex. In order to provide a summary of the document, the most important sentences are chosen and presented [7].

4.1.2 Latent Semantic Analysis (LSA)

The hidden semantic structures of words and sentences are extracted using the algebraic-statistical technique known as latent semantic analysis. It is an unsupervised method that requires neither outside knowledge nor training. The context of the input material is used by LSA to extract information, such as the words that are frequently used together and in various phrases. When sentences share a lot of words, the phrases are likely semantically connected. A sentence's meaning is determined by the words it includes, and a word's meaning is determined by the sentences it appears in. It is possible to determine the relationships between phrases and words using the algebraic technique known as singular value decomposition. Along with the ability to model relationships between words and sentences, SVD also has the ability to reduce noise, which enhances accuracy. The following example illustrates how LSA can represent the meanings of words and sentences [6][10].

Example 1: Three sentences are given as an input to LSA.
d0: 'The man walked the dog'. d1: 'The man took the dog to the park'. d2: 'The dog went to the park'.
After performing the calculations we get the resulting figure, From Fig.16, we can see that d1 is more related to d2 than d0; and the word 'walked' is related

to the word 'man' but not so much related to the word 'park'. These kinds of analysis can be made by using LSA and input data, without any external knowledge. The summarization algorithms that are based on LSA method usually contain three main steps [6][10].

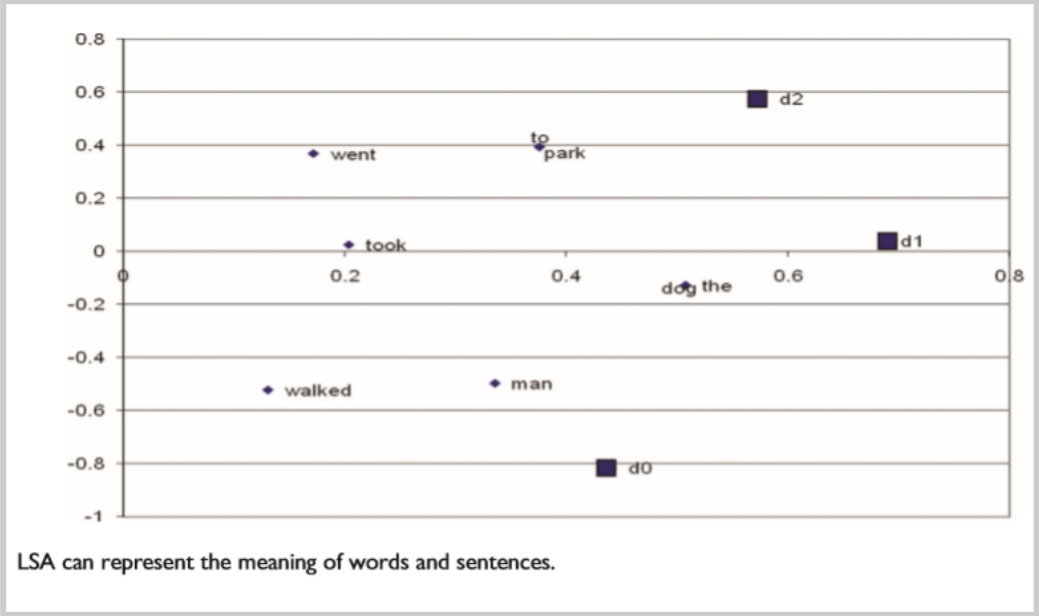


Fig.16: LSA Structure [10]

LSA has a number of limitations: The first is that word order, syntactic relationships, and morphologies are not used. Finding out the meaning of the texts' words and other language may require this kind of information. The second drawback is that it just uses the data in the input document and does not use any outside knowledge. The third restriction has to do with how well the algorithm performs. The performance drops off quickly as the data set grows larger and more homogeneous. The use of the extremely sophisticated algorithm SVD is the cause of the performance decline[6][10].

4.2 Abstractive Summarization

Abstractive summarization techniques [93] use Natural language processing to analyze the data, and then provide a summary by reformatting the provided information succinctly around the key concept. A summary produced using abstractive summarization technique is more equivalent to a summary created by a human, a condition that an extractive summarization technique EST-produced summary might not meet. In order to effectively integrate multiple machine learning algorithms with huge datasets that include good variety and conditional elements, abstract summarization methods need to be used. Abstractive summarization approaches are computationally expensive

and take time to develop well since they depend on machine learning algorithms. The size of the data being summarized has an exponential effect on implementation costs [5].

Abstractive methods can be compared to how people read and analyze texts. It chooses words that are pertinent to the text in terms of semantics. Since abstractive summarization analyzes the content using natural language processing techniques and produces concise data that constitutes the most important idea and key contents of the textual data provided for summarization, the summary produced may contain words that were not even present in the given data [5].

Information record → get setting → semantics → make own run down.

Deep learning models are used most frequently to do abstractive summarization. Sequence to sequence modeling is one such model that has been gaining popularity (Nallapati, Zhou, Santos, Gulçehre, & Xiang 2016). In speech recognition and machine translation, sequence to sequence models have been effective (Sutskever, I., Vinyals, O., & Le, Q. V., 2014). Sequence to sequence models utilizing encoders and decoders outperform more conventional methods of text summarization, according to recent studies on abstractive summarization. The encoder component converts the input document into a vector of a fixed length. The decoder component then uses the fixed-length vector to produce the desired result (Bahdanau, Cho, & Bengio, 2014) [5].

Convolution network model for encoder and feedforward neural network model for decoder were employed in the model developed by Rush et al., a team from Facebook AI Research (for more information, please read Appendix A: Extended Technical Terms). Only the first sentence of each article's content is used to create the headline in their model (2015).Long Short Term Memory (LSTM) was used in the model created by Nallapati et al., a team from IBM Watson, in both the encoder and decoder. They utilized the identical dataset of news articles that the Facebook AI Research team did. Additionally, the IBM Watson group generated the headline by using the first two to five sentences of the article content (2016). In specific datasets, Rush et al.'s models couldn't compete with Nallapati et al [5].

The performance of the encoder-decoder model is improved by the attention mechanism, which is discussed in Konstantin Lopyrev's article (2015). The model includes four LSTM layers. The news article dataset was also employed by Loprev, and the model was able to predict the headlines of the stories based on their initial paragraphs [5].

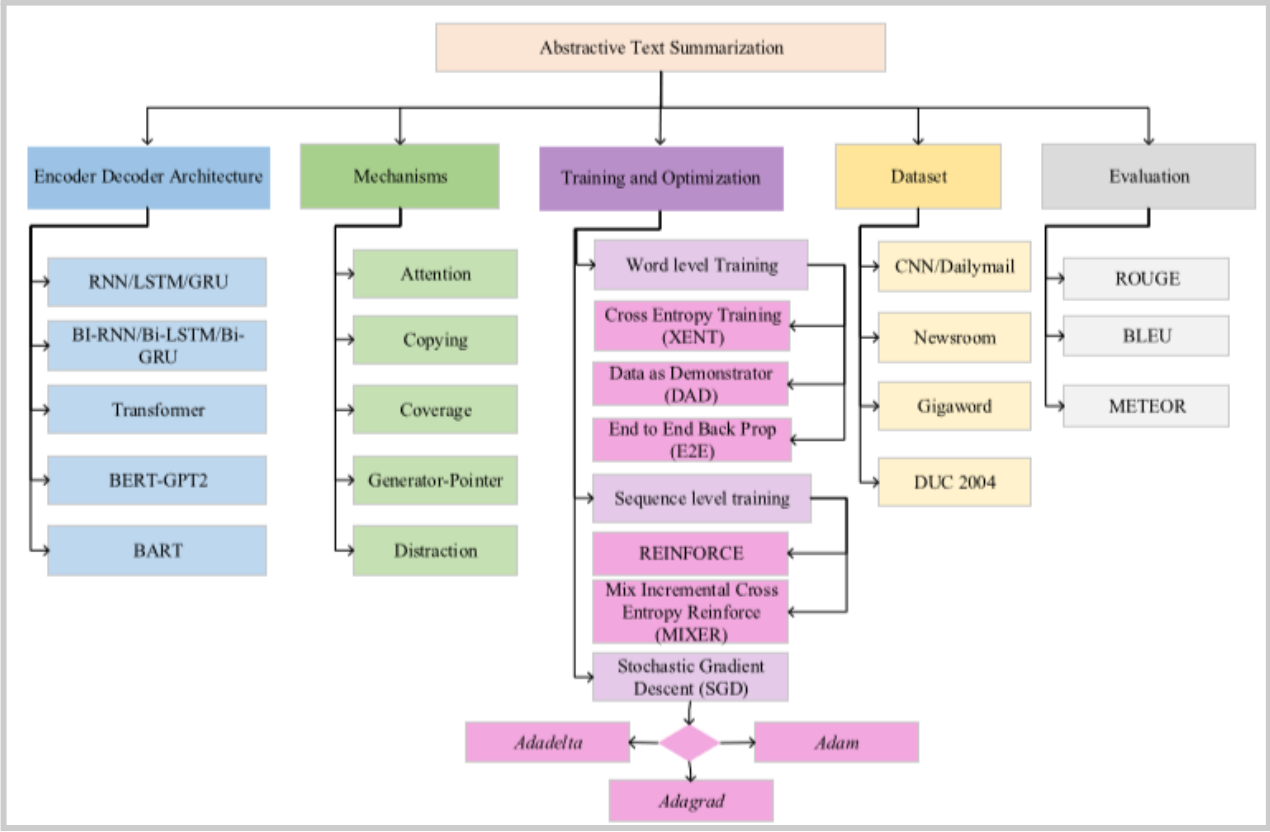


Fig:17: Abstractive text summarization research framework position [2]

The research framework presented in Fig.17 begins with the identification of the crucial components and procedures involved in the development of models for abstractive summarization based on neural networks. Encoder Decoder Architecture, Mechanisms, Training and Optimisation, Dataset, and Evaluation Metric have been identified as the key components. These components are further divided into subclasses. The following paragraphs give a short introduction to these components [5].

4.2.1 Encoder Decoder Architecture

Our choice of encoder-decoder architecture gives us the option to design our encoder and decoder using standard RNN/LSTM/GRU, bidirectional RNN/LSTM/GRU, Transformer, BERT/GPT-2 architecture, or the very recent BART model [2].

Encoder: A neural network transforms a list of words into a list of dense vector representations. Both the word and its context are intended to be captured by these complex representations. Most often, word embedding layers followed by recurrent neural networks (RNNs), which include LSTM components (Hochreiter and Schmidhuber, 1997) or gated recurrent units (GRUs) (Chung et al., 2014), are used to create encoders [13][2].

Decoder: A neural network generates the subsequent word in the summary, based on the representation of the created text's prefix and a dense context vector that represents the input sequence. Commonly, an RNN, a fully connected layer whose output dimension matches the size of the vocabulary, and a softmax layer that transforms a vector into a distribution over the vocabulary are used to implement the decoder [13][2].

1) RECURRENT NEURAL NETWORK (RNN)

RNN [44] has a different architecture than a typical/feedforward neural network. In these networks, information can endure due to feedback loops, as seen in Fig. 6. They introduce the concept of memory in neural networks. These networks can learn information based on the context because of their feedback-based nature. The RNN's input in Fig. 18 is x_t, and the previous hidden state is h_{t-1}. There is data from the hidden state right now, as well as input from the present. Both serve as a vector when combined, passing via tanh activation to create the output h_t, which turns into the network's memory (a new hidden state). RNN functionality can be described mathematically as.

$$h_t = \tanh(\omega_h h_t - 1 + \omega_x x_t) \qquad (9)$$

Here, h_t is the current state, h_{t-1} is the previous hidden state and, ω_h is its weight. x_t is the current input, ω_x is the weight and, tanh is the activation function that determines the output of the neural network [2].

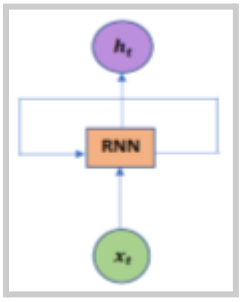


Fig. 18. Recurrent neural network [2]

The architecture of RNN is ideal for problems involving sequential data. For the purpose of statistical machine translation, two RNNs acting as an encoder and decoder were first proposed by [45]. This model was subsequently enhanced and expanded by adding an attention mechanism that automatically searches for parts of the source text that demonstrate significance in predicting a target word for neural machine translation [46]. Even though machine translation is a more general function than abstractive summarization, the two are closely related because they both involve sequence to sequence learning. Numerous deep learning techniques are currently used to generate concise summaries

and are inspired by the success of neural machine translation [2].

2) LONG SHORT-TERM MEMORY (LSTM)

LSTM, introduced by [47], is a very special variant of RNN [44] because it can tackle the problem of long-term dependencies. RNN is not able to retain information for a long period of time, for instance, if the next word in a sequence needs to be anticipated and the correctly predicted word depends on knowledge from a few sentences earlier (past information). This is where LSTMs for (long-term gaps/dependencies) are put to use. Long-term storage of information is a capability of LSTM. With enough training data, researchers [48] contend that LSTM-based approaches can be successfully applied to a variety of sequence learning problems. Fig. 19 shows the architecture of an LSTM cell [2].

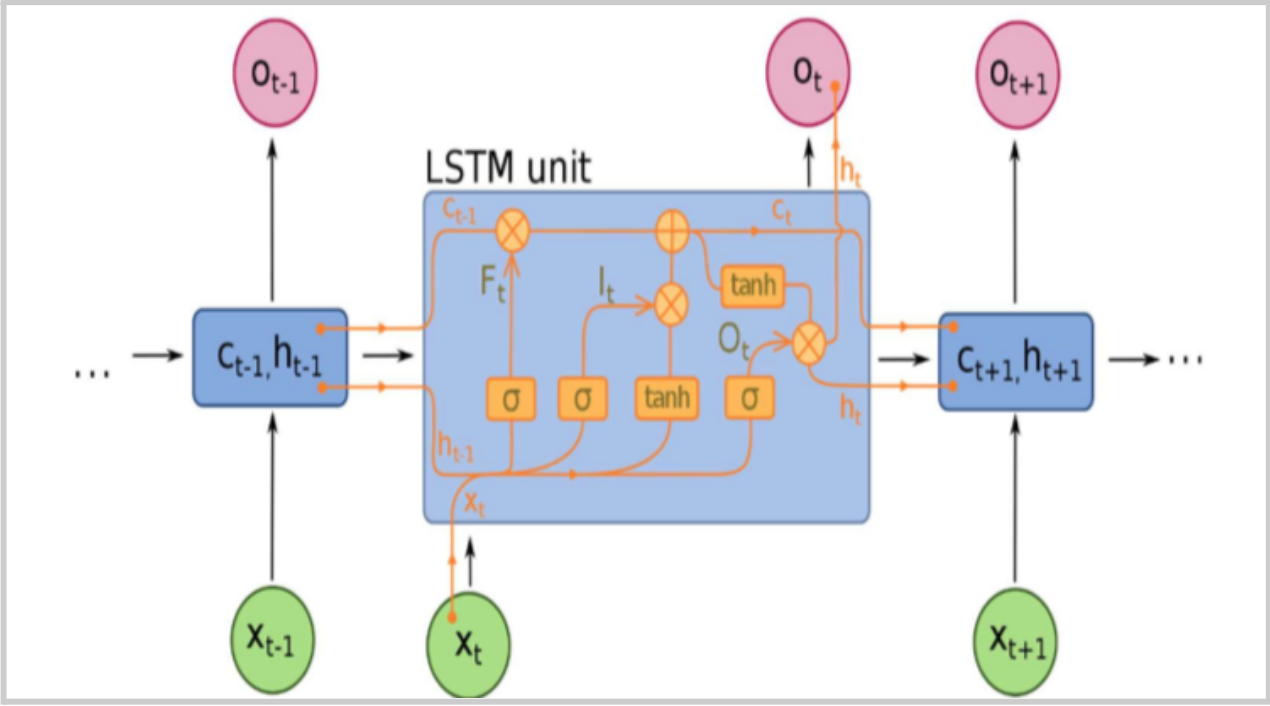


Fig. 19: Architecture of LSTM [4]

There is a forget gate, an input gate, and an output gate in LSTM which work using the sigmoid activation function. The forget gate decides which information to keep or which information to forget. The input gate updates the state of the cell and the output gate decides what would be the next hidden state. The working of an LSTM cell is given by (9) to (14) as follows:

$$f_t = \sigma(\omega_f \cdot [h_{t-1}, x_t] + b_f) \quad (10)$$

$$i_t = \sigma\,(\omega_i \cdot [h_{t-1}, x_t] + b_i) \quad (11)$$

$$\acute{C}_t = tanh(\omega_C \cdot [h_{t-1}, x_t] + b_C) \quad (12)$$

$$C_t = f_t * C_{t-1} + i_t * \acute{C}_t \quad (13)$$

$$o_t = \sigma(\omega_o\,[h_{t-1}, x_t] + b_o) \quad (14)$$

$$h_t = o_t * tanh(C_t) \quad (15)$$

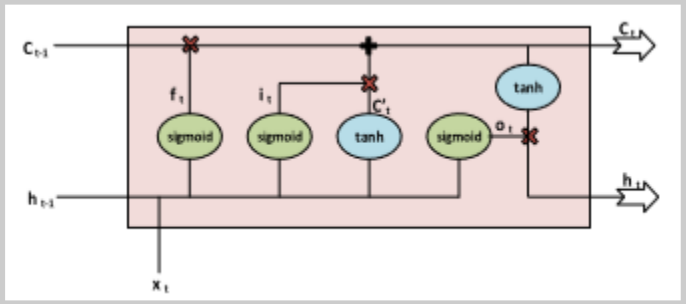


Fig. 20. The architecture of an LSTM cell [2]

In (10) to (15) and Fig. 20, x_t is the input, $\acute{C}_t$ is the candidate (holds possible values to add to the cell state), C_t is the new cell state, ft is the output of forget gate, it is the output of the input gate, C_{t-1} is the previous state of the cell, h_{t-1} is the previous hidden state, h_t is the new hidden state while ω and b represent the corresponding weights and biases. Vanishing Gradient is a very common problem in multi-layered neural networks. A neural network's capacity increases with the number of layers it possesses, which means it can learn from big training datasets more effectively and map more complex functions from input to output. The gradient progressively loses value as it is back-propagated through the network, which is when the issue develops. The gradient significantly lessens as it gets closer to the initial layers. As a result, the initial layers' weights and biases are not correctly updated. The disappearing gradient causes the entire network to be inaccurate because the initial layers are crucial in identifying the elements of input data [2].

3) GATED RECURRENT UNIT (GRU)

GRU [49] is a variation of LSTM [47] due to similarities in their designs. The issue of vanishing gradient in recurrent neural networks is addressed. An update gate and a reset gate are features of the GRU design. The reset gate

deals with information that flows out of the memory and helps the model identify past information that the network needs to forget. The update gate deals with information that flows into the memory and aids the model in determining which of the past information needs to be passed on. These are the two vectors that determine what data should be sent to the output. Fig. 21 presents the architecture and functionality of GRU. It can be expressed as follows:

$$z_t = \sigma\,(\omega_z \cdot [h_{t-1}, x_t]) \quad (16)$$

$$r_t = \sigma\,(\omega_r \cdot [h_{t-1}, x_t]) \quad (17)$$

$$h'_t = tanh(\omega \cdot [r_t * h_{t-1}, x_t]) \quad (18)$$

$$h_t = (1-z_t) * h_{t-1} + z_t * h'_t \quad (19)$$

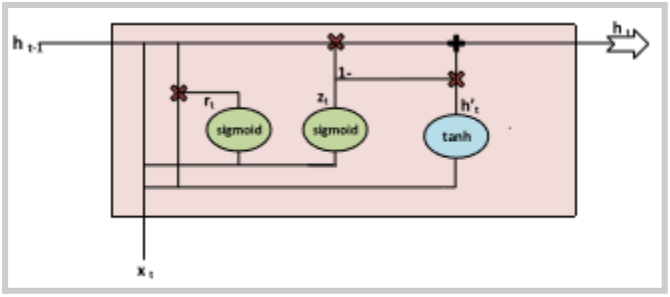


Fig. 21. The architecture of gated recurrent unit (GRU) [2]

In (16) to (19), x_t is the input, z_t is the update gate vector, r_t is the reset gate vector, h'_t is the tanh activation vector and h_t is the output. The ω, ω_r, and ω_z are the corresponding weight vectors [2].

4) BI-DIRECTIONAL RNN/LSTM/GRU

Bidirectional neural networks [50] consider two sequences for predicting the output, one in the forward direction and the other in the reverse direction. It suggests that we can use data from previous time steps as well as data from later time steps to create predictions about the current state utilizing bidirectional networks. As a result, the network is able to capture a broader context and solve issues more successfully [2].

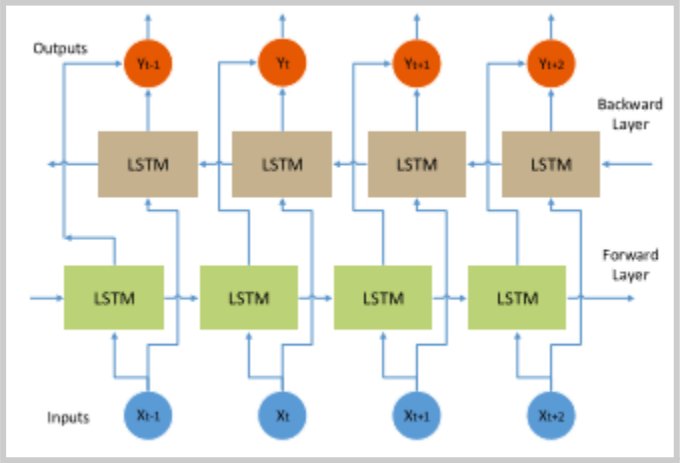


Fig.22: Bidirectional LSTM network [2]

Fig.22 presents a bidirectional LSTM network. There are two LSTM layers: a forward layer and a backward layer. Both the forward layer and the backward layer receive the input. Concatenating the output from the forward and backward layers, respectively, is the output [2].

5) TRANSFORMER

In 2017, Google unveiled Transformers [51], a breakthrough for sequence learning challenges. Transformers don't require recurrent and convolution units because they are purely reliant on attention mechanisms. Its architecture has an encoder and a decoder that are stacked multiple times. Attention units and feed-forward units make up the encoder and decoder blocks. Six identical encoder units make up the encoder portion, and six identical decoder units make up the decoder portion. Both a feed forward unit and a multi-head attention unit are included in each encoder unit. In addition to the feed forward unit and the multi-head attention unit, each decoder unit also features a separate masked multi-head attention unit [2].

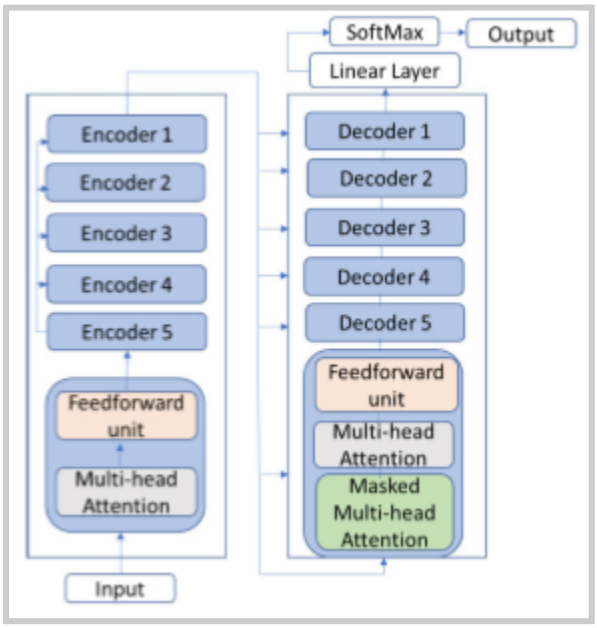


Fig.23: Transformer Architecture [2]

Fig. 23 presents transformer architecture. The functioning of the transformer starts with the word embeddings of the input sequence. The word embeddings are propagated to the first encoder which is then transformed and passed on to the following encoder. This process continues and the output of the last encoder unit is transferred to the decoder unit (all decoders in the stack) [2].

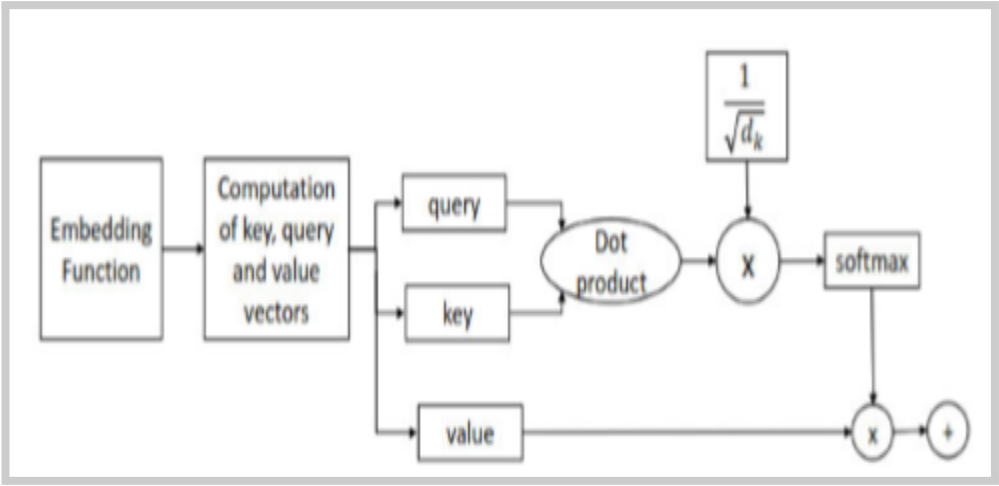


Fig.24: Attention mechanism in the Transformer [2]

The attention mechanism in the transformer as shown in Fig. 24, is very interesting. The architecture is dependent on it for its functionality. First, the input embedding is used to compute the three vectors ,the key, query, and value vectors. To determine the attention weight, a dot product operation

between key and query vectors is used. The attention score is scaled by $1/\sqrt{dk}$ and passed on to the softmax function which is then multiplied with the value vector.Following that, the feedforward layer receives this output. The transformer's self-attention is also referred to as multi-head attention because it is computed repeatedly in parallel rather than just once [2].

6) BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMER-GENERATIVE PRE-TRAINED TRANSFORMER (BERT-GPT)

A pre-trained language model can be applied to a variety of NLP tasks using BERT [37] developed by Google in 2019, whereas GPT [38] introduced by Open AI in 2018 pre-trains a language model on a vast body of text that can then be fine-tuned on a variety of specific tasks. The main distinction between the two is that BERT supports bidirectional training while GPT only supports unidirectional training. Both systems use transformer-based architecture. BERT is a multilayer transformer encoder while GPT is a multilayer transformer decoder. While BERT is not autoregressive and employs all surrounding context at once, GPT-2 is autoregressive, meaning that each token has a context of the previous words. For the first layer of GPT-2, Byte Pair Encodings (BPE) are employed as word vectors.

The Cloze task (Taylor, 1953), which served as the basis for the BERT model in [108], relieves the unidirectionality requirement by employing a "Masked Language Model" (MLM) pre-training objective. The MLM only randomly masks a small portion of the input tokens; the remaining tokens' original vocabulary id is exclusively anticipated based on context. It also makes use of “next sentence prediction” task in addition to mask language model to jointly pretrain textpair representations [16] as shown in Fig. 25.

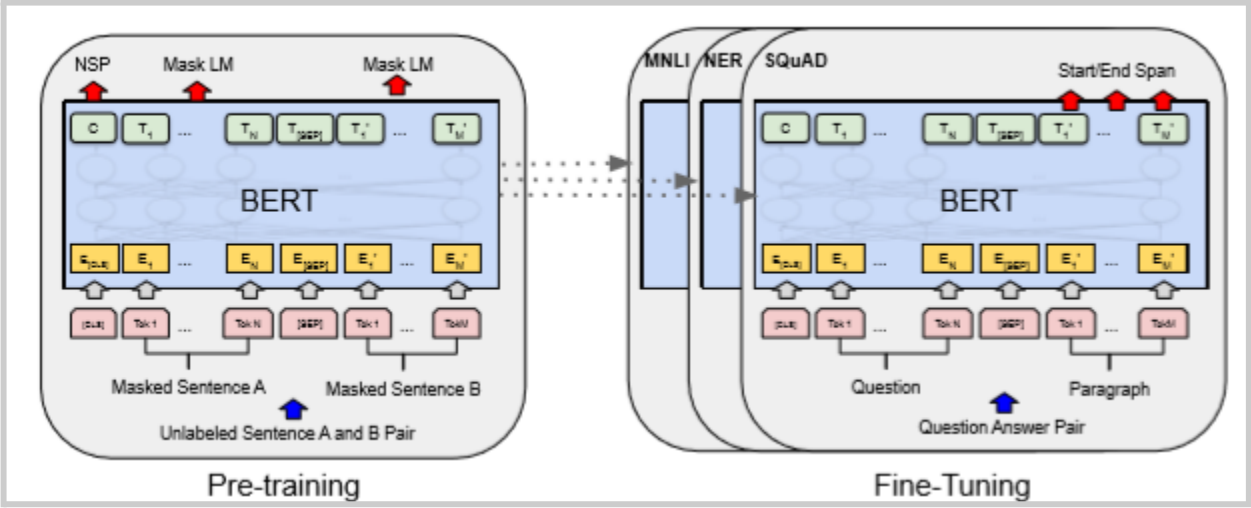


Fig.25: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers). [16]

BERT consists of the pretraining and fine-tuning modules. Masked Language Model (MLM) for bidirectional prediction and Next Sentence Prediction (NSP) for sentence-level understanding are the two tasks used to train BERT. For BERT, input embedding is the sum of token embeddings, segment embeddings, and position embeddings. In the MLM task, a mask token is used to substitute 15% of the words in the input sequence. The model then predicts the masked word based on the context that the other words in the sequence offer. A classification layer is added which receives the output of the encoder. The embedding matrix is then multiplied by the output vectors. The likelihood of each word in the vocabulary is then calculated using SoftMax. In the NSP task, the model is given a pair of sentences as input, and it works to predict whether the second sentence appears later in the original document. A [SEP] token is added at the conclusion of each sequence and a [CLS] token is added at the beginning of the first sentence. A sentence identifying embedding is added to each token. Every token additionally receives a positional embedding to identify where it belongs in the sequence. BERT can be fine-tuned for a variety of natural language tasks adding a layer to the main model [2].

7) BIDIRECTIONAL AND AUTOREGRESSIVE TRANSFORMERS (BART)

BART, very recently introduced by [52], has two major components, a bidirectional encoder, and an autoregressive decoder. Both components are implemented as a sequence-to-sequence model and have a transformer-based architecture. A noising function is used to add noise to the text during pre-training, and the system learns to reform the real text from the distorted text. When optimized for text production, it performs admirably in a variety of tasks such as abstractive dialogue, question-answering, and text summarization [52]. The encoder and decoder in the BART base model each have six layers, whereas there are twelve layers in the big model. Several methods are used to pretrain BART, such as token masking, in which [MASK] is substituted for random tokens. Token deletion involves deleting specific tokens and replacing them with new tokens. Some text areas in text-in filling are changed to [MASK] tokens. Sentence permutation involves randomly mixing up the sentences in the text. During document rotation, a random token is selected to serve as the document's beginning. The part of the document before the random token is inserted at the end. BART can be fine-tuned such that other applications like sequence classification, token classification, sequence synthesis, and machine translation can use the representations it generates. The researchers further compared the pre-training objectives with those of other models like the GPT language model, Permuted Language model, masked language model, multitask masked language model, and masked sequence to sequence. Token masking was deemed to be extremely significant, left-to-right pre-training enhanced Natural Language Generation (NLG) tasks, and bidirectionality was deemed to be crucial for question answering systems [2].

4.2.2 Mechanism

Mechanisms are functionalities added to the basic neural encoder-decoder architecture to address certain issues of abstractive summarization systems and to improve the generated summaries.

1) ATTENTION

The idea of the attention mechanism is to direct more focus and attention on certain chunks of input data as compared to others. It was first presented for neural machine translation in [46]. Later, it was used for a variety of different tasks, such as abstractive summarization. Context vectors are produced by using the encoder's intermediate states when attention is applied. The system looks for context vectors with the most important information when it

constructs the output sequence.. General attention refers to attention between input and output elements, while self-attention refers to attention between input elements.. Fig. 26 presents the attention model introduced by [46]. The model consists of a bidirectional RNN encoder and the RNN decoder [2].

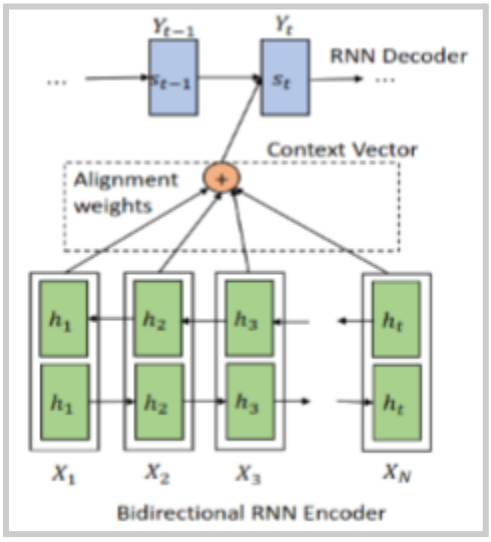


Fig.26: Attention mechanism proposed by [2]

As observed in Fig. 26, the encoder generates the hidden states h_1 to h_t. The context vector is computed as:

$$e_{ij} = a(s_{i-1}, h_j) \tag{20}$$

In (20), a is the alignment model.

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{Tx} exp(e_{ik})} \tag{21}$$

The softMax function is used to normalize the alignment scores. The context vector is a weighted sum of αij and hj.

$$c_i = \sum_{j=1}^{Tx} \alpha_{ij} h_j \tag{22}$$

2) COPYING

The copying mechanism described in [53] can be employed when certain input sequence parts need to be copied in output sequence components. The foundation of this approach is a bidirectional encoder, that encodes the input sequence and a decoder that predicts the output sequence. To predict the output sequence from the copy mode or the produce mode, a probabilistic model is used. Considering a set of vocabulary, V = $\{v_1,v_2,...,v_n\}$, and an input sequence, X = $\{x_1,x_2,....,x_{Ts}\}$.There might be words in the input sequence X, that are not in V. Here, the copy mode can copy words from X that are not in V. If M is the representation of the input sequence from the encoder, st is the decoder state at time t and, C_t is the context, the probability of output word y_t is given by combined probabilities as

$$p(y_t \mid s_t, y_{t-1}, c_t, M) = p(y_t, g \mid s_t, y_{t-1}, c_t, M) + p(y_t, c \mid s_t, y_{t-1}, c_t, M) \quad (23)$$

where c and g are the copy and generate modes respectively [2].

3) COVERAGE

It was introduced to solve the output sequence's repeat issue and aid in reducing or eliminating it [54]. This concept includes the ability to track previously studied vocabulary. To do this, attention distribution is used to keep the system aware of the covered sequence and to penalize the network if it attends to the same sequence repeatedly [2].

Mathematically, at time stept, the coverage vector c^t is represented as

$$c^t = \sum_{t'=0}^{t-1} \alpha t' \quad (24)$$

The loss term that penalizes overlap between c^t and the new attention distribution α^t is expressed as

$$loss_t = \sum_i min(\alpha^t{}_i, c^t{}_i) \quad (25)$$

4) POINTER-GENERATOR

This mechanism makes an attempt to suggest a solution to eliminate the issue of factual details and terms that are OOV. While still having the ability to create new words using a generator, it can replicate words or facts using pointers [54]. Together with computing an attention distribution α and a vocabulary

distribution p_{vocab}, the model also calculates a generation probability denoted by p_{gen}. The generation probability represents the probability of predicting the final word either from the vocabulary or copying it from the input [2].

Mathematically, the final probability of generating output word is expressed as,

$$p_{final}(w) = p_{gen} p_{vocab}(w) + (1 - p_{gen}) \sum_{i:wi=w} \alpha i \qquad (26)$$

5) DISTRACTION

The distraction mechanism proposed by [55] involves diversion in order to move between several parts of a publication in order to comprehend the overall meaning for summarizing a document rather than continuously concentrating on a single part. The first distraction task is carried out during the training process and the second distraction task is carried out during the decoding phase in this model, which was developed by the researchers. By training the model to avoid focusing on the same area again during training, the distraction was applied to the content vector.. The already viewed vector was stored as a history content vector and merged with the currently computed vector. Formally,

$$c_t = \tanh\left(W_c c't - U_c \sum_{j=1}^{t-1} c_j\right) \qquad (27)$$

$$c'_t = \sum_{i=1}^{Tx} \alpha_{t,i} h_i \qquad (28)$$

Where, c_j is the history content vector, c' t is the input content vector, $\alpha_{t,i}$ is the attention weight at time t and h_i is the hidden state. Also, the distraction was put directly on the attention weight vectors. For this purpose, the previous attention weights were stored in a history attention weight vector and then cumulated with the currently computed attention weights.

$$a'_{t,i} = v^T{}_a \tanh\left(W_a s'_t + U_a h_i - b_a \sum_{j=1}^{t-1} \alpha_{j,i}\right) \qquad (29)$$

And,

$$\alpha_{t,i} = \frac{exp(\alpha'_{t,i})}{\sum_{j=1}^{Tx} exp(\alpha'_{t,j})} \quad (30)$$

Where, υ_a, W_a, U_a and, b_a are the weight matrices.

These studies demonstrate the promise of the encoder-decoder concept as a text summarizing technique. More information from the original article material may be extracted with LSTM layers in encoder-decoders than with conventional RNNs [2].

CHAPTER 5

PROPOSED METHODOLOGY

Text Processing is one of the most common tasks in many NLP applications. These algorithms help the computers to analyze, understand and derive meaningful summary in a smart and useful way. With this paper, out of all the Text Processing Algorithms, we have tried to present Text Rank, Latent Semantic Analysis (LSA) to generate the extractive text summary, Transformer based models like T5, Improved GPT-2 model, A seq2seq based BART model to generate the abstractive text summary that provides the interesting results when tested with Wikipedia live urls or articles. We will outline our data preprocessing, and then describe various models that we used for our study.

Major Steps of summarization

The goal of this project is to explore automatic text summarization and analyze its applications on different Wikipedia sites datasets. To achieve the goal, we completed the following steps:

Step 1: Choose and clean datasets
Step 2: Build the extractive summarization model
Step 3: Build the abstractive summarization model
Step 4: Test and compare models on different datasets
Step 5: Tune the abstractive summarization model
Step 6: Build an end-to-end application

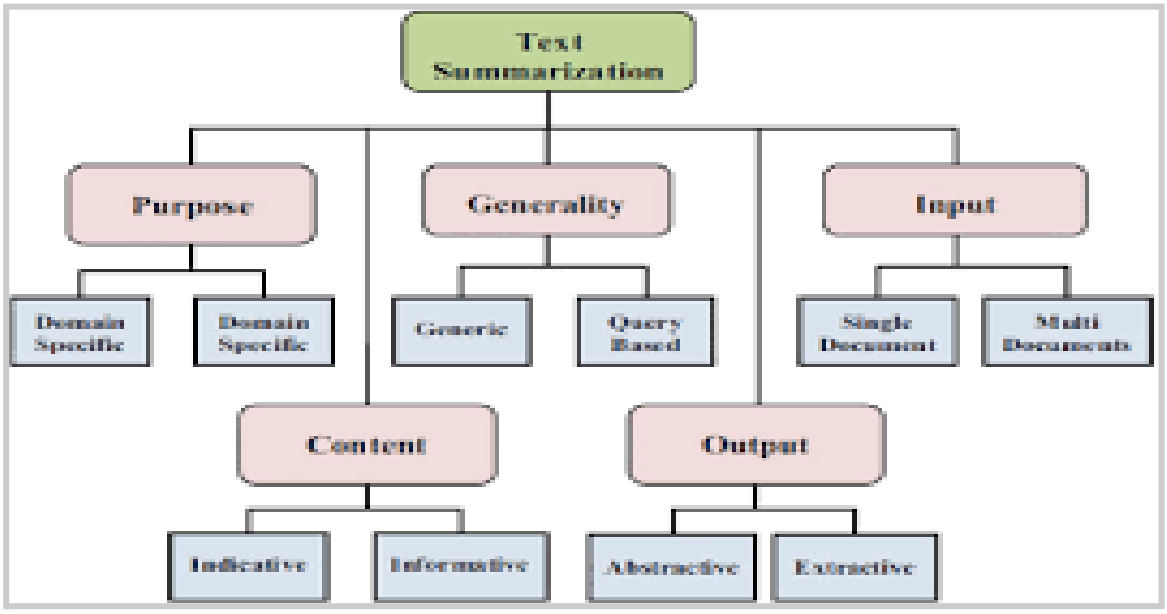


Fig.27: Text Summarization Category

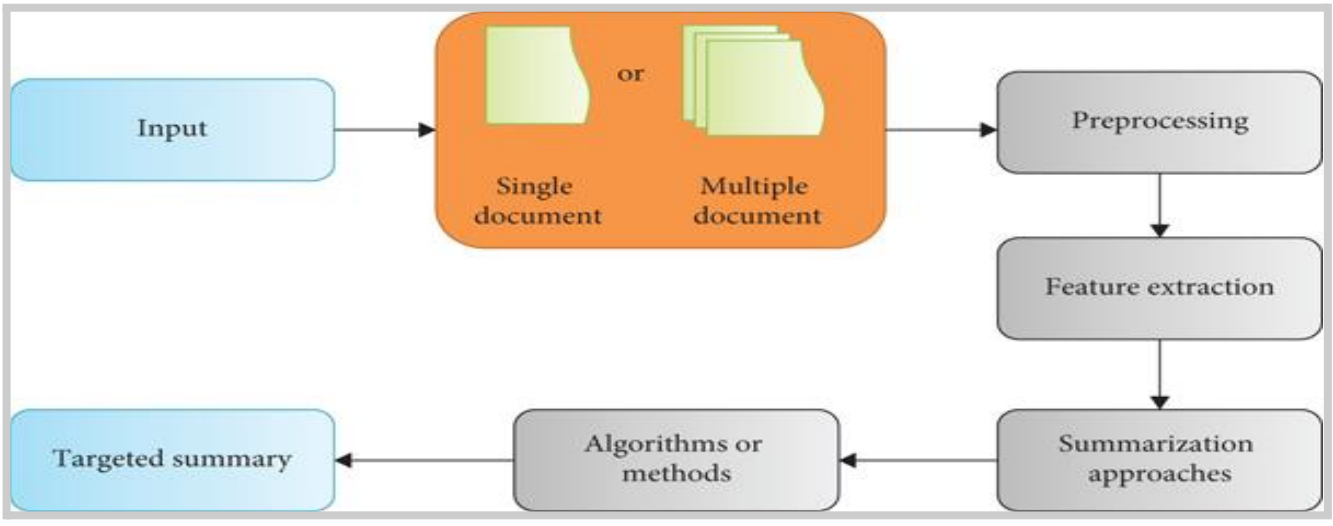


Fig.28: Text Summarization Steps with Single or Multiple document

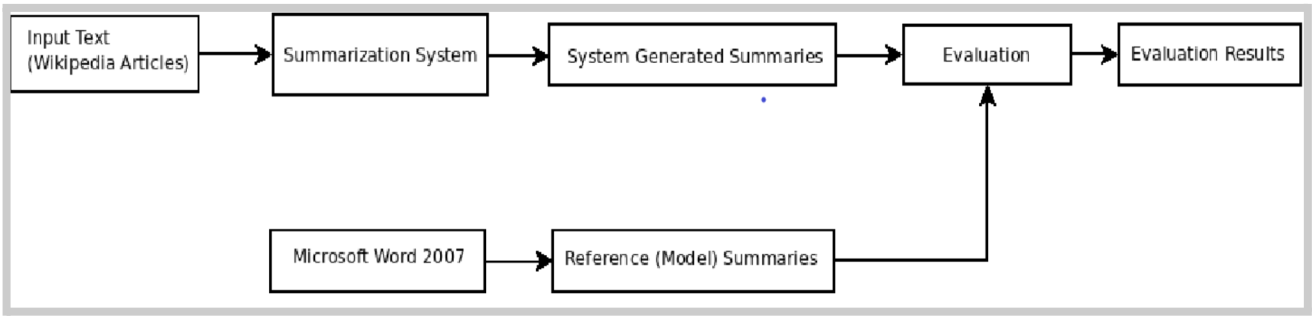


Fig.29: System Overview

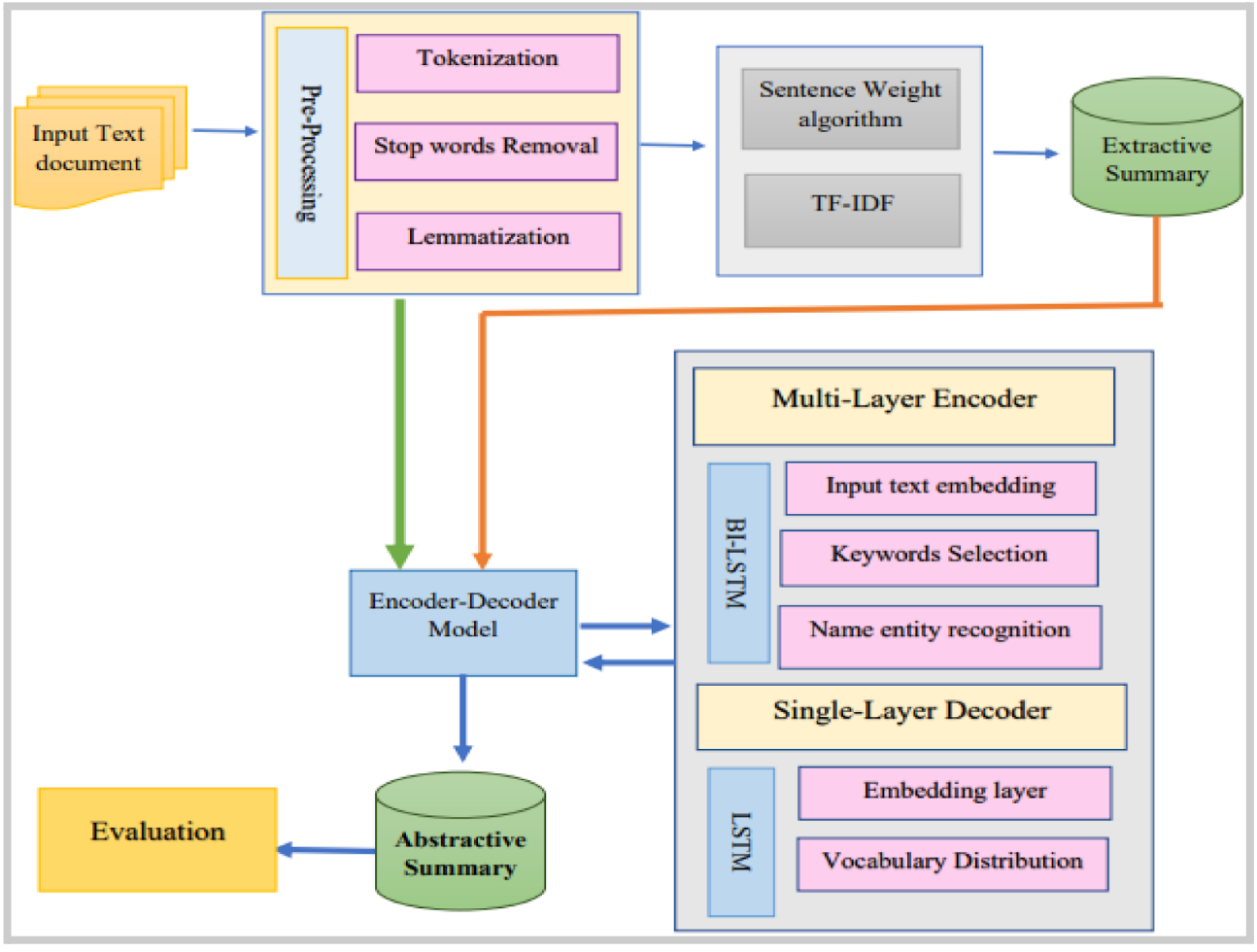


Fig. 30. Extractive and Abstractive Text Summarization

5.1 Data Preprocessing

Text preprocessing , task of cleaning and removing ambiguity from data is one of the main tasks of natural language processing, which is necessary for the effective operation of an NLP algorithm. When given the proper data, an algorithm can learn weights more precisely [6].

5.1.1 Generic Proposed Approach

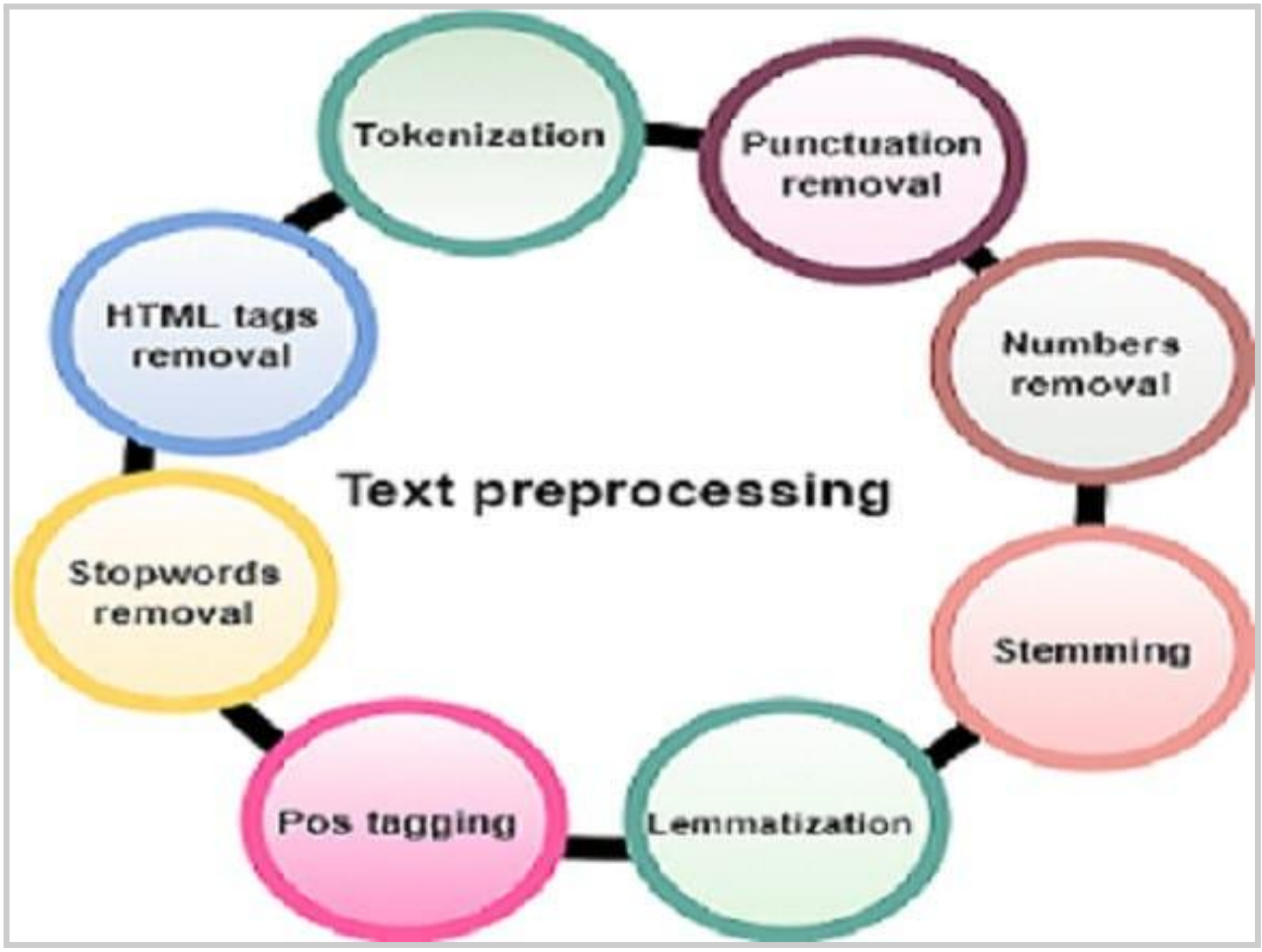


Fig.31: Data Preprocessing

In the generic model, the text document is split into a list of sentences, and this list of sentences is then passed to the text preprocessing [14, 15]. Text preprocessing involves 4 stages:

1) **Normalization:**One of the fundamental steps in any text cleaning work is to normalize the data such that all the text is in the same order, meaning that either all the letters are in lower case or all the letters are in upper case. In general, all letters are normalized to lower case for the benefit of the model.

2) **Punctuation removal**:Eliminating punctuation is one of the tasks required for an algorithm to learn weight appropriately and to capture semantic components of the text. Removed punctuation includes '?', '!', '/', and other similar symbols.
3) **Stop words removal**:Stop words include the letters 'a', 'an', 'in', 'is', etc. These are the meaningless pieces of information that any text document has that don't add to its syntactic or semantic structure. These are eliminated by tokenizing the sentence after the punctuation has been removed, and then eliminating stop words by examining and eliminating each word that appears on the predefined list of stop words. The stop words are removed using nltk (natural language tool kit) library.
4) **Lemmatization**:Lemmatizing the text is the final step in the text preprocessing process. This produces the effective result of the NLP algorithm rather than being a step that must be taken. In order to exclude the endings of words with the same meaning, morphological analysis of the words is used.

5.1.2 Proposed Approach for web scraping

Information can be extracted from the Web using a variety of techniques. It is preferable to use an API to extract structured data rather than web scraping. Unfortunately not all website offers an API because they do not want users to extract data in a structured way. To convert unstructured data in HTML or XML format into structured data (database or spreadsheet), we can execute web scraping. In this paper, we scraped a Web page using Beautiful Soup [96]. Beautiful Soup is a Python [97] library for extracting the body of HTML and XML files. It also defines a class for auto detecting the encoding of an HTML or XML document, and converting it to Unicode. We first remove stop words from the raw data after extracting the structured data. Next, the provided text is then tokenized into sentences, with each sentence being given a unique index based on the input sequences of the sentences in the document. Finally, tokenize each sentence into the group of words needed for graphical representation [9].

5.2 Model Description

5.2.1 Text Rank-Based Summarization

In the proposed method, we have considered the document's sentences to be equivalent to Web sites in the PageRank system [98]. The similarity between the two sentences determines the probability of moving from sentence A to sentence B. The improved TextRank described in this paper makes use of the PageRank's conceptual framework[9].

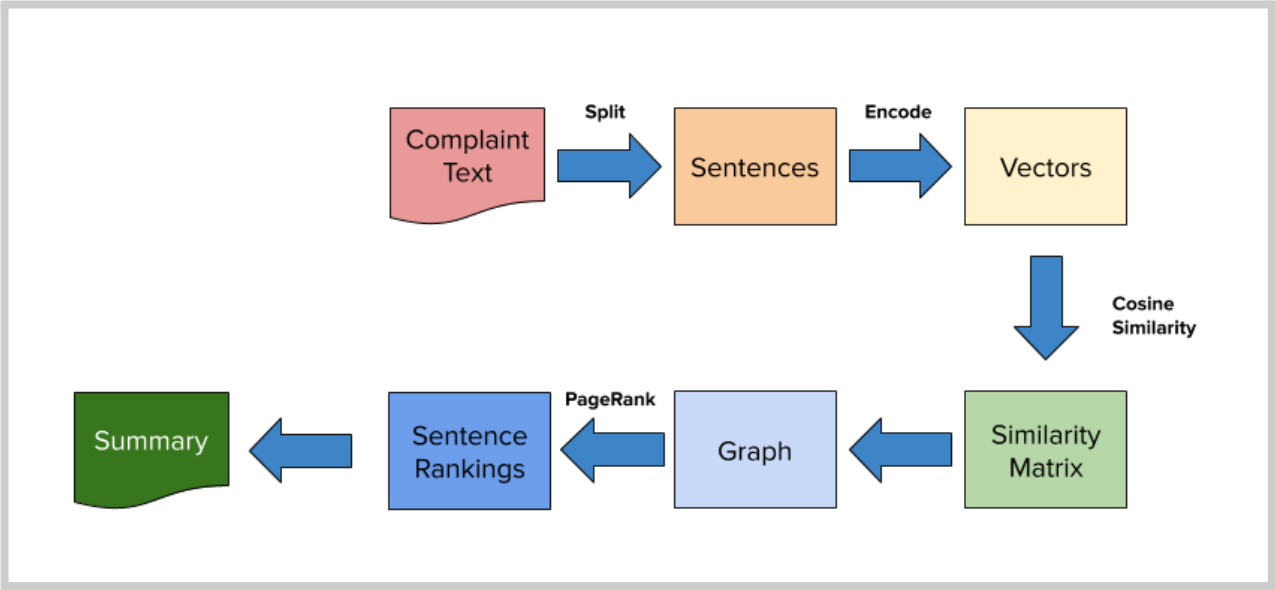


Fig.32: Auto Text Summarization using Text Rank Algorithm

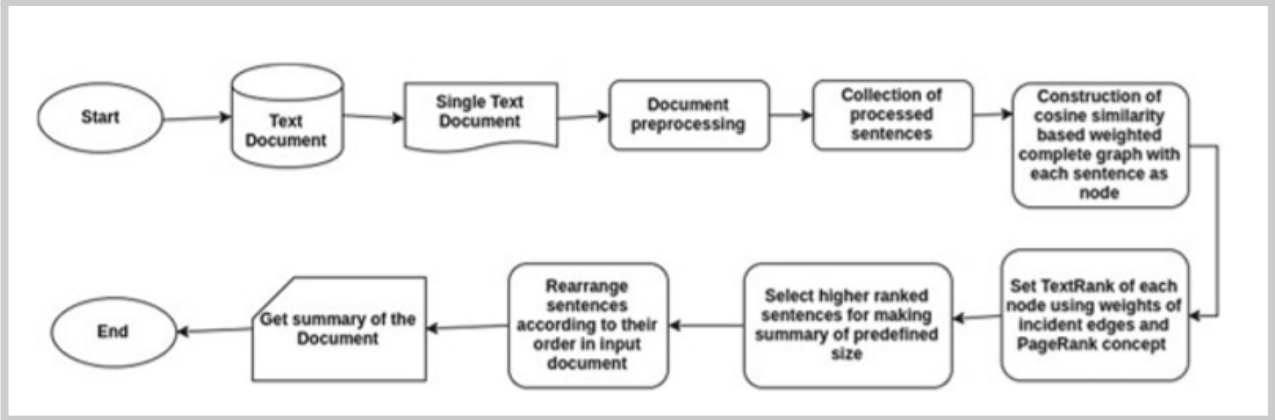


Fig.33: System Architecture of Graph-Based Text Summarization Using Modified TextRank [9]

Using a sentence ranking algorithm, we may choose the most significant sentences from the input text document. In PageRank, significant web pages are linked to other significant web pages. Similarly, in our approach ,we are assuming that the significant sentences are connected to other relevant sentences in the input document[9].In this model, at first to identify the sentences, we have transferred the input document ***D*** into sentences ***{s_1, s_2, … , s_n}*** using NLTK package, i.e., ***D ={ s_1, s_2, … , s_n}***. We have assigned index for each of the sentence according to the input sequence of the sentences in the document. In order to properly order the summarized phrases and provide a coherent summary, index values are widely utilized. In the next step, each sentence is tokenized into a set of words. To define similarity among the sentences, we represent each sentence of the given text document as a word vector. Next, a complete weighted graph ***G =(V, E, W)*** of the input document is built, where ***V*** is the set of vertices, each of which corresponds to a sentence that the word vector is supposed to represent. ***E*** is the set of edges between every pair of vertices. ***W*** is the set of weights assign to the edges of the graph ***G***. The weight ***w*** associated to ***edge (u, v)*** ∈ ***E*** is assigned using following logic [9]:

(i) Suppose $S(u) = \{w^1_u, w^2_u, \dots, w^s_u\}$ is the sentence corresponds to the vertex u and $S(v) = \{w^1_v, w^2_v, \dots, w^t_v\}$ is the sentence corresponds to the vertex ***v***. Here, we have considered term frequency (*tf*) and inverse sentence frequency (*isf*) for each word in the sentence. For an example, $tf_w(S(u))$ is the term frequency of word w in $S(u)$ which gives the number of occurrences of the word w in the sentence. Similarly, $isf_w(D)$ is the inverse sentence frequency of the word w in the input document D which is defined by Eq.(31).

$$isf(w, D) = log\frac{\|D\|}{\|\{S \in D : w \in S\}\|} \quad (31)$$

where, ‖{S ∈ D : w ∈ S}‖ is the number of sentences in which word w appears, and ‖D‖ is the number of sentences present in the input document [9].

(ii) Now, the isf-modified-Cosine similarity is used to measure the similarity between every pair of sentences and it is used as weight w of *edge (u, v)* using Eq.(32)

$$w(u, v) = \frac{\sum_{\forall x \in S(u) \cup S(v)} tf_x(S(u)) tf_x(S(v)) (isf_x(D))^2}{\sqrt{\sum_{\forall y \in S(u)} (tf_y(S(u)) isf_y(D))^2} \times \sqrt{\sum_{\forall z \in S(v)} (tf_z(S(v)) isf_z(D))^2}} \quad (32)$$

Traditional cosine similarity only considers the term frequency of the corresponding words in the text document, and in case of cosine similarity, the dimension should be the same for all sentence vectors. In contrast, isf-modified-cosine similarity also takes into account the different levels of importance for the corresponding words in the sentences as well as the various length of the sentence in the text document [9].

Thus, we get a complete weighted graph which is made sparse by removing the edges having weight less than the threshold (t), set as the average weight of all the edges in the graph. This graph represents the similarity graph of the input document. For weight assignment, score is assigned to every sentence corresponding to every node of the graph G. In this proposed method, we initialize the TextRank score of each node of the graph by the average weight of the edges incident to it for giving importance to the weights associated with the edges(E), whereas in traditional PageRank the initial PageRank value of each node is set to $\frac{1}{T}$, where T is the total number of nodes in the graph. Next, the TextRank score is updated via modified TextRank as defined in Eq.(33) [9].

$$TR(S(u)) = \frac{d}{T} + (1-d) * \Sigma_{v \in adj(u)} \frac{TR(S(v))}{deg(S(v))} \quad (33)$$

where $TR(S(u))$ is the text rank of sentence S corresponds to the node u, $TR(S(v))$ is the text rank of the sentence S corresponds to the node v such as $v \in adj(u)$,$deg(S(v))$ is the degree of the node v corresponding to the sentence S, T is the total number of nodes present in the graph, and d is a "damping factor." Here, we have considered the value of d as 0.15. Finally for summarization, we have selected top n scored sentences and rearranged those n sentences according to the sentence index which we have assigned at first step. The n output sentences construct the summary of our proposed model. Using the pseudocode of the proposed Algorithm (2.1), we can extract important sentences from a given input document [9].

Algorithm2.1: MODIFIED TEXTRANK SUMMARY(D, n) [9]

Input:Extracted text from the target document D and size of summary = n (say).
Output:Summary of the input document.
1.Encode D into D' using UTF-8 format;
2.D' is tokenized into individual sentences (S_i) using the NLTK library;
3.Initialize *index_value* to zero for the sentence index;
4.**for each** $S_i \in D'$ in order of appearance in encoded D **do**
4.1. Remove the stop words from the sentence;
4.2. Increase the *index_value* by 1;
4.3. Assign the *index_value* to the sentence;
5.**for each** processed tokenized sentence $\dot{Si} \in D'$ **do**
Take word count vector $v_i = \{w_1, w_2, \dots, w_{ti}\}$;
6.Build a graph $G = (V, E, W)$ where V = set of vertices corresponding to sentences represented by word count vector, E = set of edges and W = set of weights associated with each edge $(u, v) \in E$ computed by Eq.(32);
7.Remove edges with weight less than the average weight considering all edges of G;
8.$\forall v \in V$ Initialize TextRank by the average weight of the edges incident to it;
9.Modify TextRank of every node $v \in V$ using Eq.(33) based on the initial TextRank;
10.Arrange the vertices of the graph in descending order of their TextRank score;
11.Take first n vertices from the sorted list;
12.Put in summary the sentences associated to the selected n vertices in order of the *index_value*;
return(*summary*)

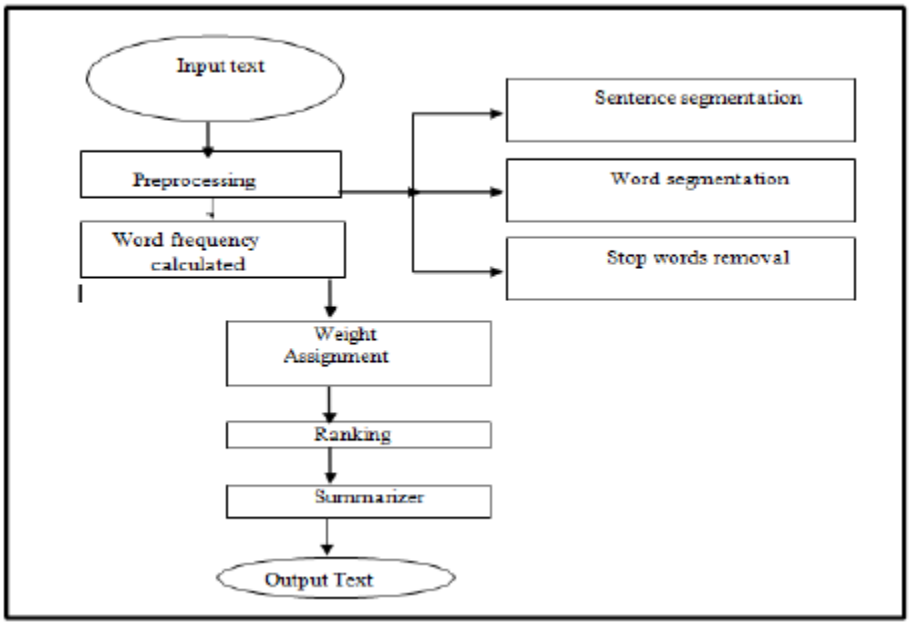


Fig.34: Frequency summarizer

5.2.2 Latent Semantic Analysis (LSA) Summarization

The algorithms in the literature that use LSA for text summarization perform differently. In this section, information on LSA will be given and these approaches will then be discussed in more detail.

5.2.2.1 Step 1: Input matrix creation:

A document's input must be provided in a fashion that a computer can comprehend and use for computation. Typically, this representation takes the form of a matrix, where the rows represent words/phrases and the columns represent sentences. The significance of words in sentences is represented by the cells. The values for the cell can be entered in different approaches. Since not all words are used in every sentence, the resulting matrix is frequently sparse. For summarization, the method used to build an input matrix is crucial since it has an impact on the matrices that SVD uses to calculate the results. As was already established, the SVD algorithm has a high level of complexity, which worsens with the size of the input matrix. Rows of the matrix, or the words, can be shrunk by methods like eliminating stop words, using only the roots of words, substituting phrases for words, and so on in order to reduce the matrix size. Additionally, the matrix's cell values can alter the SVD results [6,10].

There are different approaches to filling out the cell values. These approaches are as follows [6,10]:

• Frequency *of word:* the cell is filled in with the frequency of the word in the sentence.

• *Binary representation*: the cell is filled in with 0/1 depending on the existence of a word in the sentence.

• *Tf-idf (Term Frequency-Inverse Document Frequency)*: the cell is filled in with the tf-idf value of the word. A higher tf-idf value means that the word is more frequent in the sentence but less frequent in the whole document. A higher value also indicates that the word is much more representative for that sentence than others.

• *Log entropy*: the cell is filled in with the log-entropy value of the word, which gives information on how informative the word is in the sentence.

• *Root type*: the cell is filled in with the frequency of the word if its root type is a noun, otherwise the cell value is set to 0.

• *Modified Tf-idf*: this approach is proposed in Ozsoy et al. [99], in order to eliminate noise from the input matrix. The cell values are set to tf-idf scores first, and then the words that have scores less than or equal to the average of the row are set to 0.

5.2.2.2 Step 2 : Singular Value Decomposition:

SVD is an algebraic method that can model relationships among words/phrases and sentences. In this method, the given input matrix A is decomposed into three new matrices as follows:

$$A = U \sum V^T$$

where A is the input matrix $(m \times n)$; U is words × extracted concepts $(m \times n)$; $\sum$ represents scaling values, diagonal descending matrix $(n \times n)$; and V is sentences × extracted concepts $(n \times n)$ [6,10].

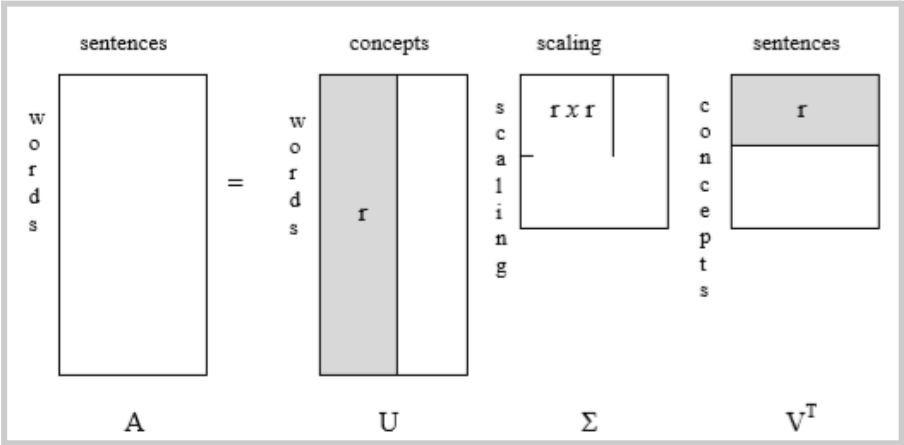


Fig.35: Singular Value Decomposition[10]

SVD has the capability of mapping m-dimensional term vector space into r dimensional singular vector space. The mapping reveals latent semantic structure of the input document. r linearly independent vectors represent the topics (concepts) of the input document. r value is the rank and is equal to or less than the value of min (m, n) [6,10].

The co-occurrence of the words is the basis for choosing the singular vectors. The words are regarded as considered if they co-occur in distinct sections of the document. The size of singular vectors provides insight into the significance of the idea. By using words that are related to the concepts, sentences that are related to those concepts are projected along singular vectors. The sentence with the highest index value is the most representative sentence about that singular vector (concept) [6,10].

The reduction in dimension, r value, is important; since it can affect the later computation performance. As stated in (Deerwester, et al. 1990), the value of r should be able to fit the real structure of the input data, while not having noisy data, such as containing unimportant information. The proper way of deciding the r-value is still an open question in literature [6,10].

The well-known problem with SVD is that it takes a lot of time. The SVD calculation must be repeated whenever new terms or sentences are added to the starting matrix. Deerwester, et al. (1990) advise placing new terms and sentences in the centroid of the terms and sentences, respectively, to solve this issue. Polysemy-related issues with SVD are another problem. Each word is represented as a point in space after the SVD calculations. If a word has several completely distinct meanings, it is represented in the single vectors space as the average of those meanings. In the later stages of the document analysis, this could be problematic. (Deerwester, et al. 1990) advises classifying the words in the space according to where they fall under each of their several meanings [6,10].

5.2.2.3 Step 3: Sentence Selections:

Different algorithms are used to choose significant sentences using the results of SVD.

A. Sentence selection approaches

There are various approaches for selection of the sentences while creating summaries using LSA. In this section, five of them will be explained.

A.1. Gong and Liu (2001). One of the key studies in LSA-based text summarizing is the algorithm of Gong and Liu [100]. After representing the input document in the matrix and calculating SVD values, V^T matrix, the matrix

V^T matrix ($k = 2$)			
	Sent0	Sent1	Sent2
Con0	0.457	0.728	0.510
Con1	-0.770	0.037	0.637

Fig.36: VT matrix. From each row, the sentence with highest score is chosen until a predefined number of sentences have been collected [10]

of extracted *concepts* × *sentences* is used for selecting the important sentences. In V^T matrix, row order indicates the importance of the concepts, such that first row in a V^T matrix denotes the most significant notion that was extracted, and subsequent rows are ranked according to importance. This matrix's cell values depict the relationship between the sentence and the notion. The more closely the language relates to the concept, the higher the cell value [10].

Gong and Liu's method involves selecting one sentence from the most crucial concept, followed by another from the second-most crucial concept, and so on until a certain number of sentences have been gathered. A parameter specifies how many sentences must be gathered [10].

In Example1,as mentioned in previous chapter Discussion three sentences were given, and the SVD calculations were performed accordingly. The resulting V^T matrix having rank set to two is given in Fig, 36. In this figure, first, the concept *con0* is chosen, and then the sentence *sent1* is chosen, since it has the highest cell value in that row [10].

There are some drawbacks to Gong and Liu's strategy, which are listed by Steinberger and Jezek [101]. The first drawback is collecting the same number of phrases using the smaller dimension. The specified number is given, and if it is large, statements from less important ideas are chosen. Choosing just one sentence from each concept has a second drawback. Sentences that are closely related to a concept but do not have the highest cell value can be found in some ideas, especially important ones. The final drawback is that it's expected that all of the concepts are important at the same level, even though this may not be the case [10].

A.2. Steinberger and Jezek (2004). The development of an input matrix and an SVD computation are the first steps in Steinberger and Jezek's methodology [101]. The next step is sentence selection, which is different from Gong and Liu's method. The approach of Steinberger and Jezek uses both V and matrices for sentence selection. This method chooses sentences based on the length of

each sentence vector, which is represented by the row of the *V* matrix. The ideas whose indexes are less than or equal to the specified dimension are used to determine the length of the sentence i. In order to emphasize the key ideas more, matrix is employed as a multiplication parameter. The sentence that has the longest length value is picked to be in the final summary. Using the results of Example 1, calculated length values are given in Fig. 37. The dimension size is two for this example. Since the sentence *sent1* has the highest length, it is extracted first as a part of the summary. By eliminating the drawbacks of the Gong and Liu summarization algorithm, the major goal of this algorithm is to produce a better summary. The Steinberger and Jezek approach allows for the collecting of multiple sentences from a single major concept while choosing related sentences to all key concepts [10].

A.3 Murray et al. (2005). As with the earlier algorithms, the first two steps of the LSA algorithm are carried out before the sentence selection step. V^T and matrices are employed in this method [107] to choose sentences. More than one sentence can be collected from the topmost important concepts, placed in the first rows of the V^T matrix. Using a matrix, it is decided how many sentences from each concept will be collected. For each concept, the value is decided by getting the percentage of the related singular value over the sum of all singular values. In Fig 38, the V^T matrix of Example 1 is given. From the calculations of the matrix, it is observed that one sentence collection from the first row is enough, but for demonstration purposes two sentences will be collected from Fig 38. So, from *con0* the sentences *sent1* and *sent2* are selected as a part of the summary. Gong and Liu's issue with just choosing one sentence from each *concept* is resolved by Murray et al.'s strategy, especially when the concept is crucial. This method allows for the selection of multiple sentences, even if none of them have the highest cell value in the row of the connected idea. Additionally, the number of sentences in the final summary need not match the reduced dimension [10].

Length scores	
Sent0	1.043
Sent1	1.929
Sent2	1.889

Fig.37: Length scores. The sentence with the highest length score is chosen[10]

V^T matrix ($k = 2$)			
	Sent0	Sent1	Sent2
Con0	0.457	0.728	0.510
Con1	-0.770	0.037	0.637

Fig.38: The VT matrix. From each row, the sentence with the highest score is chosen until all predefined sentences have been collected[10]

V^T matrix ($k = 2$)				
	Sent0	Sent1	Sent2	Avg.
Con0	~~0.457~~ 0	0.728	~~0.510~~ 0	0.565
Con1	~~-0.770~~ 0	0.037	0.637	-0.021
Length	0	0.765	0.637	

Fig.39: The VT matrix after pre-processing [10]

A.4. Cross method. The Steinberger Jezek [101] methodology, as forth in Ozsoy et al. [99], is extended by the cross method. In this approach, input matrix creation and SVD calculation steps are executed as in the other approaches and the V^T matrix is used for sentence selection purposes. There is a pre-processing stage that comes before the SVD calculation and sentence selection steps. The pre-processing step aims to eliminate the overall impact of sentences that are peripherally related to the concept but do not constitute the concept's primary sentence. The average sentence score is computed for each concept, which is represented by the rows of the V^T matrix. Then the cell values which are less than or equal to the average score are set to zero. Less related sentences are removed while more related ones are retained for that concept after setting the cell values to zero for those whose scores are below average. Pre-processing is followed by a version of Steinberger and Jezek's approach's phases. In our cross approach, the total length of each sentence vector, which is represented by a column of the V^T matrix, is determined. The user specifies the number of concepts to be used while calculating the length score; otherwise, all of the extracted concepts are used. The resulting summary is then assembled using the longest sentence vectors. In Fig 39, an example V^T matrix is given after the pre-processing is executed. The average score for each concept is first calculated for the pre-processing step, and any cell values below this average are subsequently set to zero. After pre-processing, the concept scores are added with values to determine the length scores. In this example matrix, *sen1* has the highest length score, so it has been chosen to be part of the summary [10].

A.5. Topic method. Ozsoy et al. [99] have proposed the topic method. It first performs a pre-processing stage, then chooses a sentence. Both of these steps make use of the V^T matrix. Finding the key concepts and sub-concepts are the major goal of the topic technique. The themes in the input document are known to be the concepts that come from SVD calculations. These topics, however, may also be subtopics of other extracted topics. In this approach, after deciding the main topics that may be a group of subtopics, the sentences are collected from the main topics as a part of the summary. This approach's pre-processing step begins similarly to the pre-processing step of the cross approach. First, using the row of the V^T matrix, the average sentence score for

each concept is determined. The cell values below this score are then set to zero. Only the most crucial sentences that are directly related to the concept are left after this step, which removes sentences that are not strongly related to it [10].In Fig. 40, an example V^T matrix after pre-processing is given.

V^T matrix ($k = 2$)				
	Sent0	Sent1	Sent2	Avg.
Con0	~~0.457~~ 0	0.728	~~0.510~~ 0	0.565
Con1	~~-0.770~~ 0	0.037	0.637	-0.021

Fig.40: The VT matrix after pre-processing [10]

	Con0	Con1	Strength
Con0	1.456	0.765	2.221
Con1	0.765	1.348	2.113

Fig.41: New concept×concept matrix [10]

After the first step of pre-processing comes the step of finding out the main topics. For this step, a *concept × concept* matrix is created by finding out the concepts that have common sentences. The common sentences are the ones that have cell values other than zero in both concepts that are considered. Then the new cell values of the *concept × concept* matrix are set to the total of common sentence scores. In Fig. 41, the *concept × concept* matrix based on the V^T is given. After the creation of the *concept × concept* matrix, the strength of each concept is calculated. For each concept, the strength value is computed by getting the cumulative cell values for each row of the *concept × concept* matrix. The concept with the highest strength value is chosen as the main topic of the input document. A higher strength value indicates that the concept is much more related to the other concepts, and it is one of the main topics of the input text. In Fig. 41, calculated strength values can be seen. Since *con0* has the highest strength value, it is chosen to be the main topic [10].

Following these steps, the sentences are extracted from the pre-processed V^T matrix using Gong and Liu's methodology. As previously stated, until a pre-defined numbers of sentences are gathered, one single sentence is collected from each concept. The approach uses the selected primary concepts for sentence selection rather than the topmost concepts from the V^T matrix. In Fig.40, *sen1* is chosen from *con0*, since that sentence has the highest cell value [10].

5.2.3 T5 Transformer Model –Based Summarization

Fundamental Concepts of the Transformer Model -The Transformer Network [30] is built only on multiple attention layers. It relies on attention layers and positional encoding to remember the order of the words in the input sequence rather than using RNN. Multiple attention layers' ability to create global dependencies facilitates input processing in parallel [1].

The transformer model [11] contains encoder and decoder layers, where each is connected to a multi-head attention layer and feed forward network layers. The model remembers the position and sequence of words with the help of cosine and sine functions that creates positional encoding. The multi-head attention layer [11] in the encoder and decoder layer applies a mechanism called self-attention. The input is fed into three connected layers to create query (Q), key (K), and value (V) vectors [11]. These vectors are split into n vectors.

$$Attention = softmax\left(\frac{QK^T}{\sqrt{dk}}\right)V \qquad (34)$$

Self-attention is applied on n separate vectors to create multi-head attention [30].

$$MultiHead(Q,K,V) = Concat(head_1,\dots,head_h)W^O \qquad (35)$$

where, $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$ Where the projections are parameter matrices [30]

$$W_i^Q \in \mathbb{R}^{d_{model} \times d_k},\ W_i^K \in \mathbb{R}^{d_{model} \times d_k},\ W_i^V \in \mathbb{R}^{d_{model} \times d_V} \text{ and } W^O \in \mathbb{R}^{hd_V \times d_{model}}$$

Pretrained Models based on Transformers -Hugging Face [31] is an open source project that offers a variety of beneficial NLP libraries and datasets. The Transformer library is the most well-known one there. The transformer library includes a variety of pre-trained text summarization models that may be customized for any dataset. Here, we'll talk about some pre-trained models that were optimized and put into use for the BBC news dataset to produce reasonably accurate summaries [1]. The models we used are as follows:

T5 Transformer Framework

With the text-to-text framework described in the paper [102], NLP tasks such as document summarization, machine translation, and question answering regression tasks can be trained to predict the string representation of a number rather than the actual number itself using the same model for loss function and hyperparameters [16]. The input and output of the T5 model is always purely text to text format i.e., text string as shown in Fig. 42.with fig 1 for T5 framework

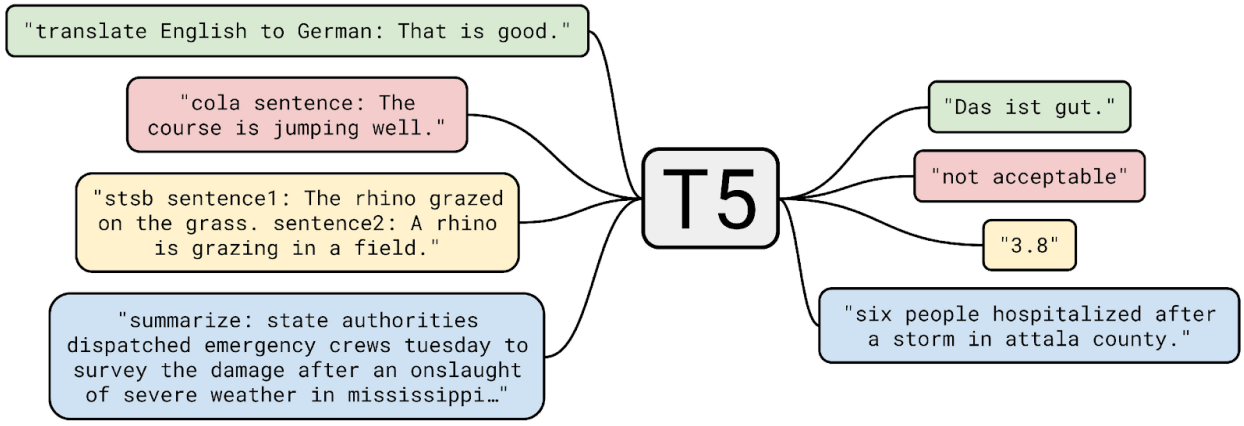


Fig.42: T5 Text-to-Text Framework [16]

"Text-to-Text Transfer Transformer" is referred to as T5 [32]. Transfer learning is the guiding principle of the T5 model [33]. The model was initially trained using Transfer Learning on a task with a large text before being fine-tuned on a downstream task so that the model acquires general-purpose abilities and knowledge to be used to tasks like summarization. T5 [32] employs a sequence-to-sequence generation technique that feeds the encoded input via cross-attention layers to the decoder and generates the decoder output autoregressive. We have improved a T5 model [32] in which the encoder receives a set of tokens that are translated into a set of embeddings as input. The encoder block has a block with two subcomponents, a self attention layer and a feed forward network. The only structural difference between the encoder and decoder is that every personal attention layer is followed by a generalized attention mechanism. This enables the model to only work with the previous outputs. An output from the last decoder block is passed into the following layer. The activation function in this last dense layer, is softmax. The input embedding matrix receives the weights from this layer's output [1,3,17].

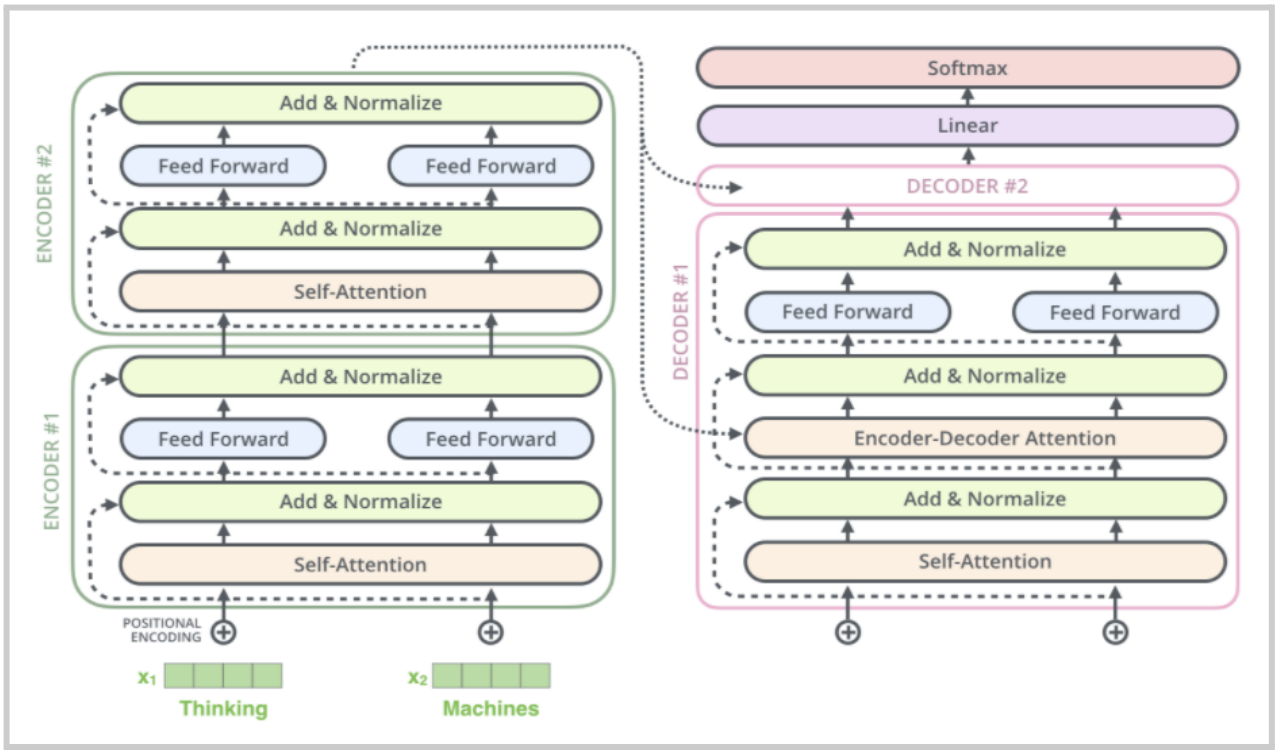


Fig.43: T5 architecture [17]

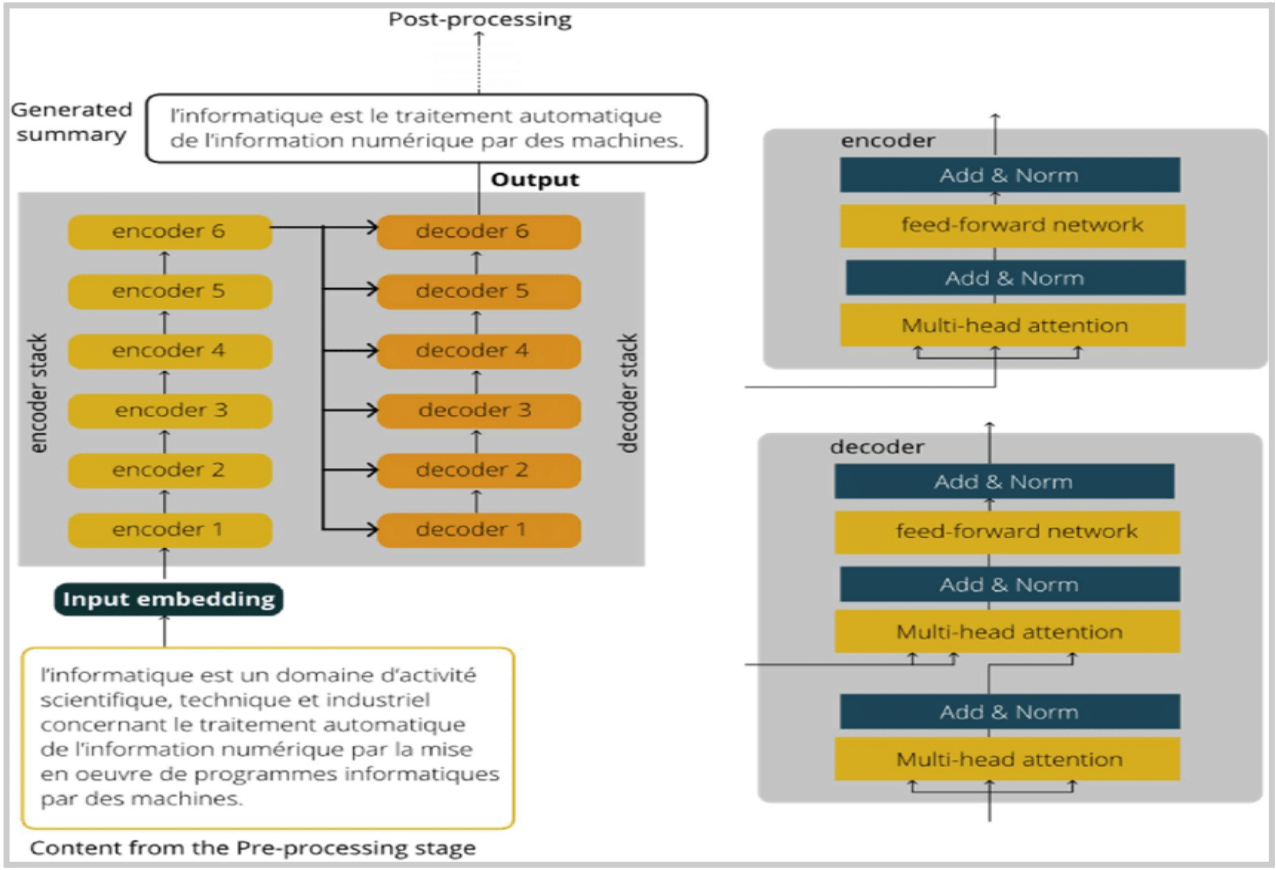


Fig.44: Data flow in transformer [3]

T5 Transformer Model Implementation [18]

- The following libraries were imported: Transformers, T5 Model, Pandas, Pytorch, Pytorch Utils for Dataset and Data loader, and Pytorch.
- Purge the dataset of unnecessary columns.
- For test and validation, the data are split in an 80/20 ratio.
- To generate train and validation data loaders, train and validation parameters are specified and provided to the pytorch Data loader construct.
- Specify the model and optimizer that will be used for training and to update the weights of the network.
- Train the model with the required input parameters.
- Produce the summaries.

5.2.4 Generative Pre-trained Transformer-2 (GPT2) Model–Based Summarization

Generative Pre-trained Transformer-2 model [103] is employed for text summarization on our own dataset that contains contents from different wikipedia sites. The decoder section of the transformer network, which has a multi-headed masked self-attention block to process all the tokens concurrently, is included in this model. Given the previous tokens in the text, it is trained to recognise the following token. The model is enhanced by fine-tuning all the pretrained parameters. It uses a deep neural network, in which the inputs are handled by many layers of neurons that operate successive layers to produce the output at the last layer [19]. The architecture of GPT-2 model is shown in Fig.45.

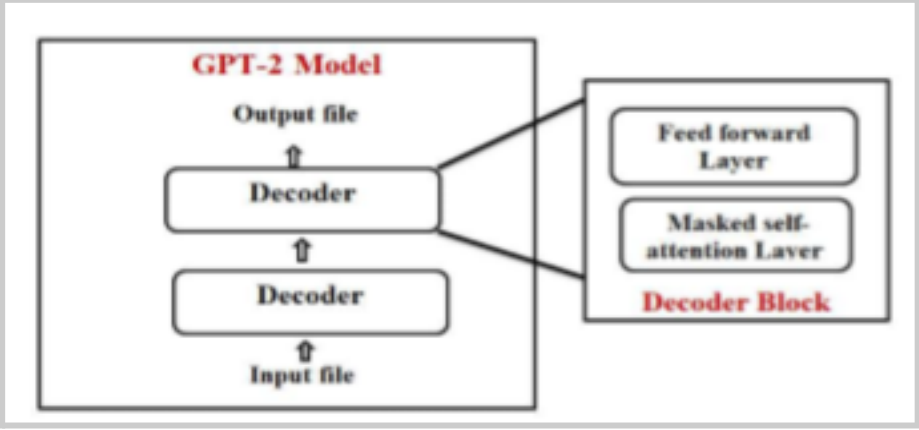


Fig.45: GPT-2 Model Architecture [19]

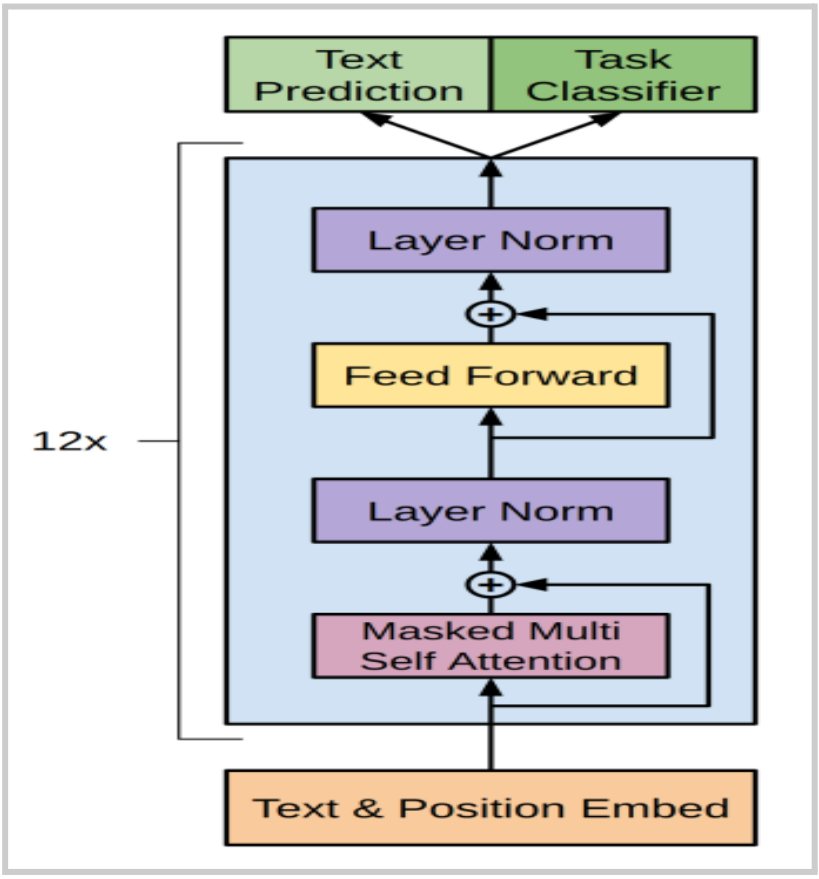


Fig.46: **GPT-2 Model with Prediction and Classifier**

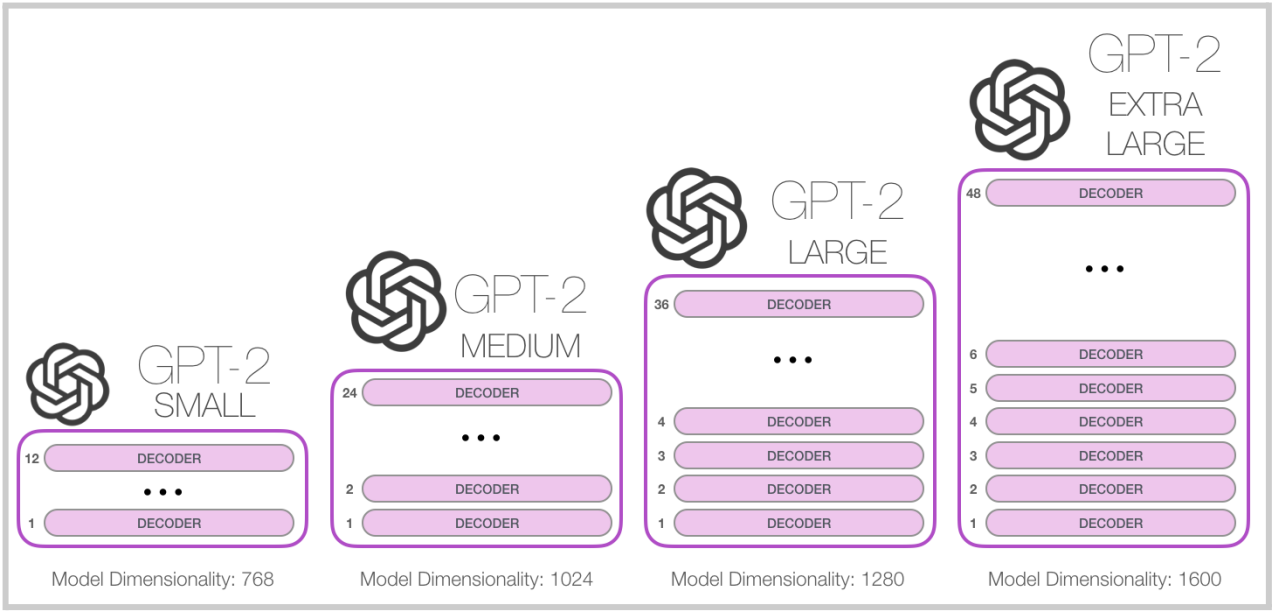


Fig.47: **Different size of GPT-2 Model**

The smallest variant of the trained GPT-2, takes up 500MBs of storage to store all of its parameters. The largest GPT-2 variant is 13 times the size so it could take up more than 6.5 GBs of storage space.

5.2.5 Bidirectional and Autoregressive Transformers (BART) Model –Based Summarization

Bidirectional and Auto Regressive Transformers is what they are known as [36]. It is constructed using a seq2seq model that underwent pre-training denoising. It makes use of the common seq2seq model architecture, which combines an encoder that resembles BERT [34] and a GPT-like decoder [35].

It is trained by corrupting text with random noise function thus, model learns to restructure the original text as shown in Fig. 48. A standard Transformer with simple neural machine translation architecture is used in BART.It assesses several noising techniques. It uses a novel in-filling approach where a single mask token is placed in text spans to discover the best performance by randomly rearranging the original sentence order [1] [20].

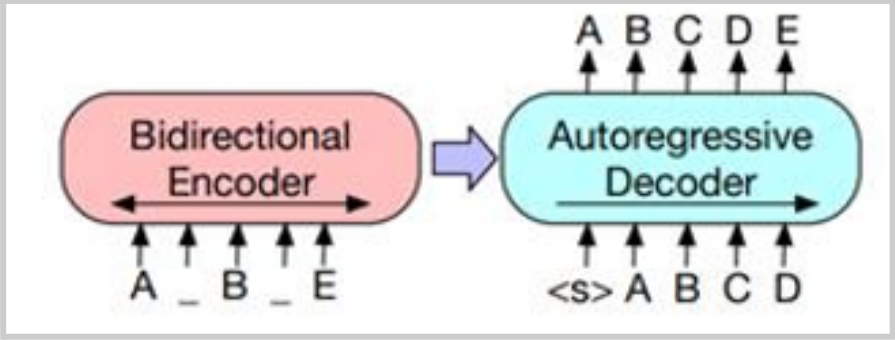


Fig.48: A Schematic Diagram of BART [16]

In the pre-training job, the original phrases' order is randomly changed, and a new technique is used to switch text ranges with a single mask token. There are twice as many layers in the BART big model [36] as there are in the base model. Although they are quite similar, BART [34] has about 10% more features than a BERT model of comparable size. The autoregressive BART decoder is controlled to produce sequential NLP tasks like text summarization. The denoising pre-training objective is strongly tied to the fact that the data is taken from the input but altered. As a result, the encoder's input is the input sequence embedding, and the decoder's output is produced autoregressively. The pre-trained model "facebook/bart-large-cnn'' and the Bart tokenizer, which is made from the GPT-2 tokenizer, were both used. Because of this, words are encoded differently based on where they are in a phrase [1] [21].

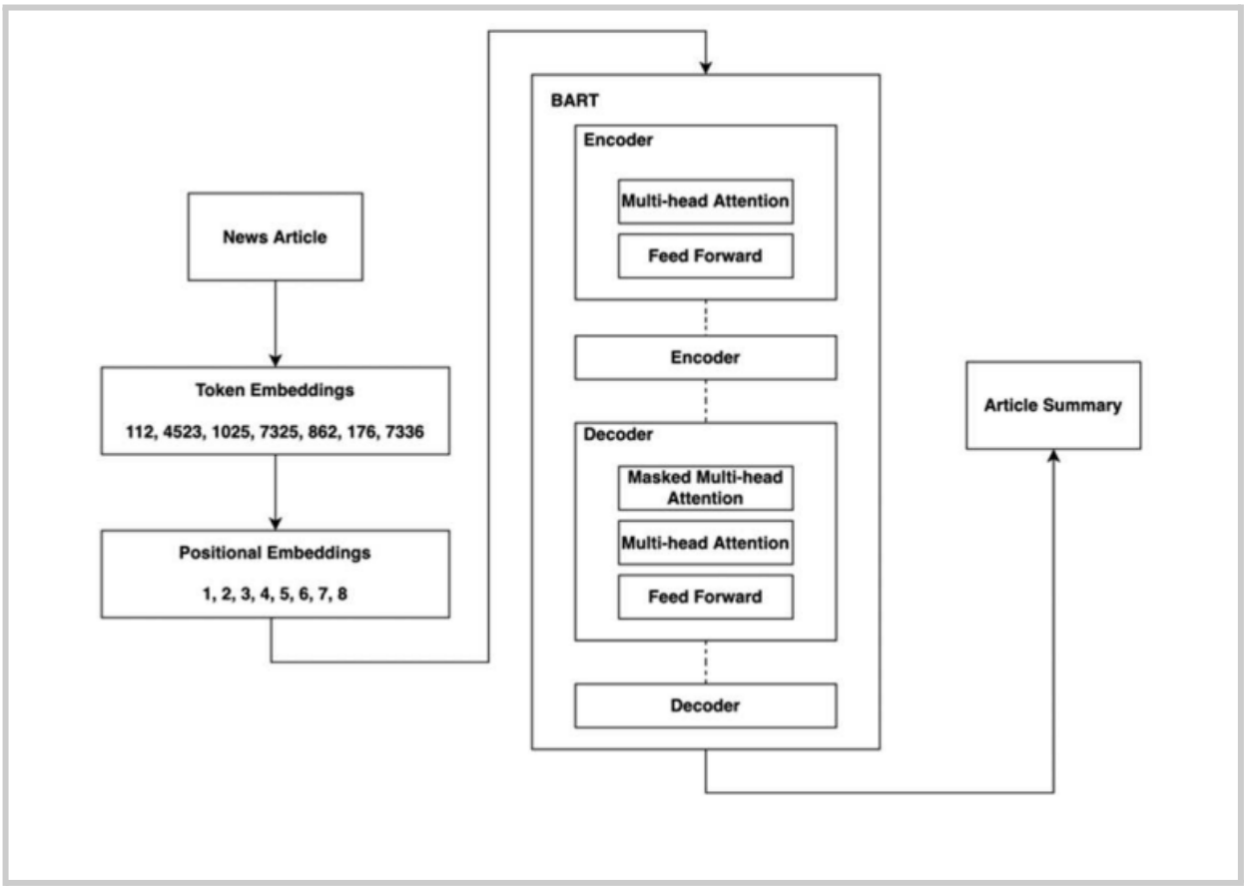


Fig.49: BART Model Workflow [22]

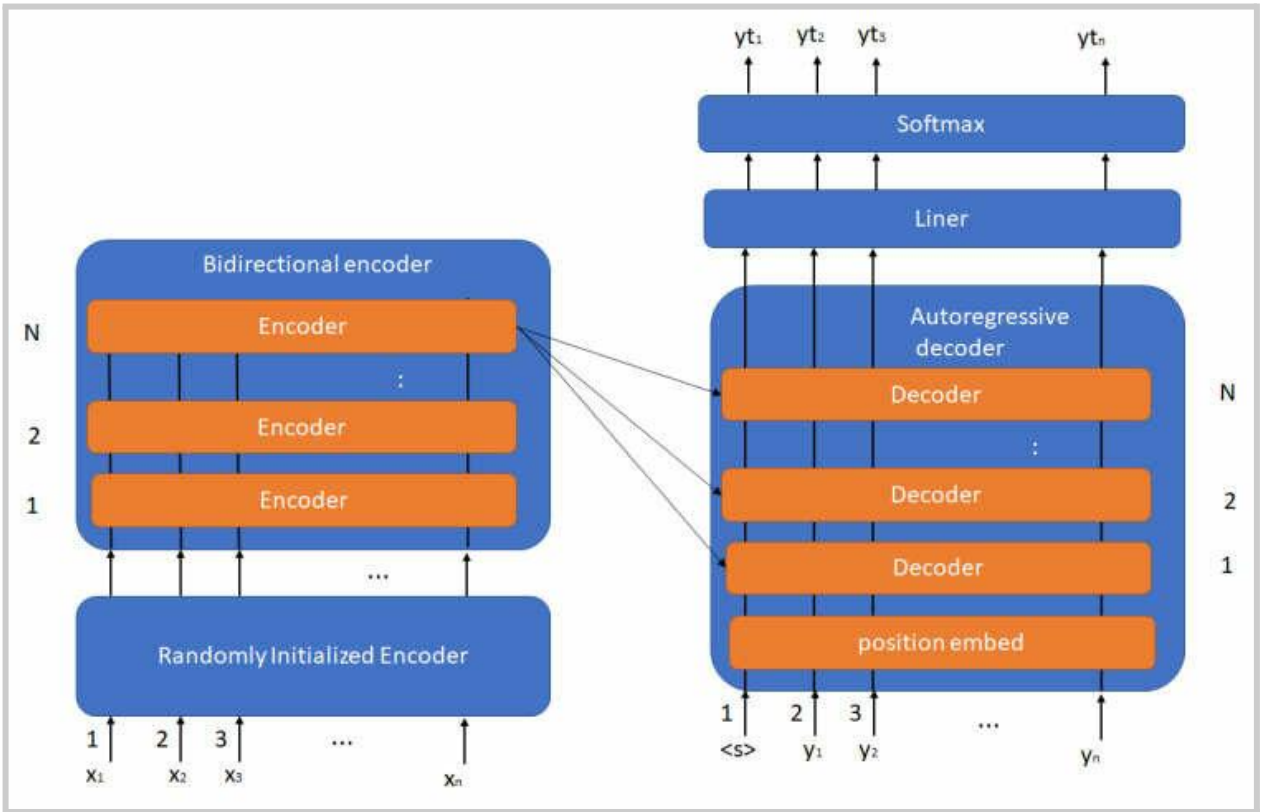


Fig.50: BART Model Architecture

BART, or the Bidirectional and Auto-Regressive Transformer, will be used for text summarization [104]. The article's content will first be abstractedly summarized using BART before being prepared for presentation. The BART model employed in this study is precisely known as bart-large-cnn (BART model that has been optimized on CNN/DailyMail dataset to perform abstractive summarization task) [22].

The BART model employed in this study has 16 attention heads, 1024 hidden units, 12 encoder levels, and 12 decoder layers. Text-based article content will be provided to the BART model as input. Then, embeddings will be used to represent article content. The embeddings will receive positional encoding in order to include positional information about the words in a phrase. Embeddings will then be passed into the encoding layers and eventually the decoding layers [22].

The goal of this research is to build an algorithm to separate article content into smaller portions so that BART can process the document without truncating the input text. This technique will be used to overcome the BART model's limitation while processing long documents (documents longer than 1024 tokens). The text will then be broken up into parts, with each component being processed separately before being concatenated. The final summary will then be created by summarizing the combined material a second time [22].

Input embeddings will pass through a number of encoder layers during the encoding process. There are 12 encoder layers in the BART model used in this study. The multi-head attention layer and the feed forward layer are two of the many sub-layers that make up each encoder. With 16 attention heads working simultaneously, the multi-head attention layer will determine the self-attention for each word in the input sequence. The attention vector, which contains the attention value for each word in the input sequence, will be the layer's output. The feed forward layer will then be traversed by the attention vector. The attention vector will undergo a linear transformation in the feed forward layer so that it can be handled by the following encoder or decoder block [22].

The embeddings will be sent to the decoder after passing through the 12 layers of the encoder. Input embeddings will pass through 12 decoder layers during the decoding process, just like during the encoding process. The encoder-decoder attention layer is a new sub-layer that is sandwiched between the encoder-decoder layers, each of which contains the same sub-layers as the encoder. A target sequence and the embeddings from the encoder output will be input to the decoder layer. First, the input to the multi-head attention layer's first sub-layer, the target sequence, will be used. The decoder block's

multi-head attention layer operates differently than the encoder block. In order for the decoder to only consider words in the current position and words in the previous position while calculating the attention values, it calculates the self-attention by masking the future position. The second sub-layer, the encoder-decoder attention layer, will then receive the input. The query from the previous layer and the key and value from the encoder output were the initial two inputs to the encoder-decoder attention layer. Based on the provided Query, Key, and Value, this layer will then determine the attention for each word in the input and target sequence [22].

The feed forward layer will receive an attention vector as its output from this layer. The attention vector will undergo a linear transformation in the feed forward layer, much like in the encoder, so that the subsequent decoder block can process it. The output from the final decoder will be sent into a linear layer and softmax layer after traveling through all of the decoder blocks. A probability distribution of the words in the input sequence will be generated by the softmax layer. The word included in the cell with the highest probability will be chosen as the predicted word for the current timestep [22].

5.3 Algorithm for using the different model to perform text summarization of a Wikipedia site

Step 1: Identify the Wikipedia page that you want to summarize.

Step 2: Scrape the content of the page using a web scraping tool or the Wikipedia API.

Step 3: Preprocess the text to remove any unwanted characters, stop words, or special characters that may interfere with the model's performance.

Step 4: Split the text into sentences using a sentence tokenizer.

Step 5: Initialize the different ML models and load its weights and configuration.

Step 6: Encode the text using the ML Models tokenizer to convert the text into tokens that can be understood by the model.

Step 7: Set a target length for the summary.

Step 8: Use the Different ML model (T5 Transformer, GPT2 , BART Models) to generate a summary by setting the input as the encoded text and passing the target length as a parameter to the model.

Step 9: Decode the summary using the BART tokenizer to convert the model output from tokens back into human-readable text.

Step 10: Return the summarized text to the user.

5.4 Web Application to perform text summarization of a Wikipedia site

This web-application will show the summary of a wikipedia page. Loads of python modules are used to create the summaries.

5.4.1 Description

The text is extracted from the wikipedia page from the user input URL. We generate a corresponding URL if it's a query and if it's an URL, the validity of the URL is checked first. The image and title of the corresponding wikipedia title is also fetched. We get the summary according to the option chosen and clean it to remove unnecessary elements.

For the analysis part first punctuations are removed and the whole summary is converted to small cases. We get the summaries from different algorithms and generate venn diagrams, word clouds and graphs as shown in the picture. We consider the most occurring words in the summary (top 24).

Word Cloud:

A word cloud in NLP data visualization serves as a powerful tool to visually summarize and communicate textual information in a concise and visually appealing manner.With the word frequency information, a word cloud can be generated. The most frequent words are typically displayed with larger font sizes, while less frequent words are represented with smaller font sizes. The positioning of words within the cloud is often randomized to create an aesthetically pleasing arrangement.To enhance the visual impact and interpretability of the word cloud, various customization options can be applied. These include choosing appropriate colors, fonts, and background shapes.Additionally, it is often possible to filter out certain words or add custom stopwords to remove irrelevant or noisy terms from the word cloud.Once the word cloud is generated, it can be analyzed to gain insights into the textual data. Frequently occurring words that are prominently displayed in the word cloud indicate important or significant terms within the dataset. These words can provide a quick summary or overview of the main themes, topics, or sentiments present in the analyzed text.It helps researchers, analysts, and users to quickly grasp the key aspects of a text corpus and identify important patterns or trends within the data.

Venn Diagrams:

Venn diagrams are useful data visualization tools in Natural Language Processing (NLP) that can help analyze the relationships between sets of words or concepts within a text corpus. A Venn diagram visually represents the overlapping and non-overlapping areas between multiple sets, providing insights into the shared characteristics or intersections among different groups of words.If we have multiple text documents or categories, wecan create a Venn diagram to understand the common words shared between them. Each set represents a specific document or category, and the overlapping regions indicate the words that appear in multiple sets. This can help identify common themes, topics, or vocabulary across different documents or categories.If you have manually assigned categories or labels to words in your NLP analysis, a Venn diagram can illustrate the relationships between these categories.

When creating a Venn diagram for NLP data visualization, it is important to carefully select and preprocess the words or concepts to be included in the diagram. This may involve removing stopwords, normalizing word forms, or applying other text processing techniques to ensure meaningful and accurate representations.

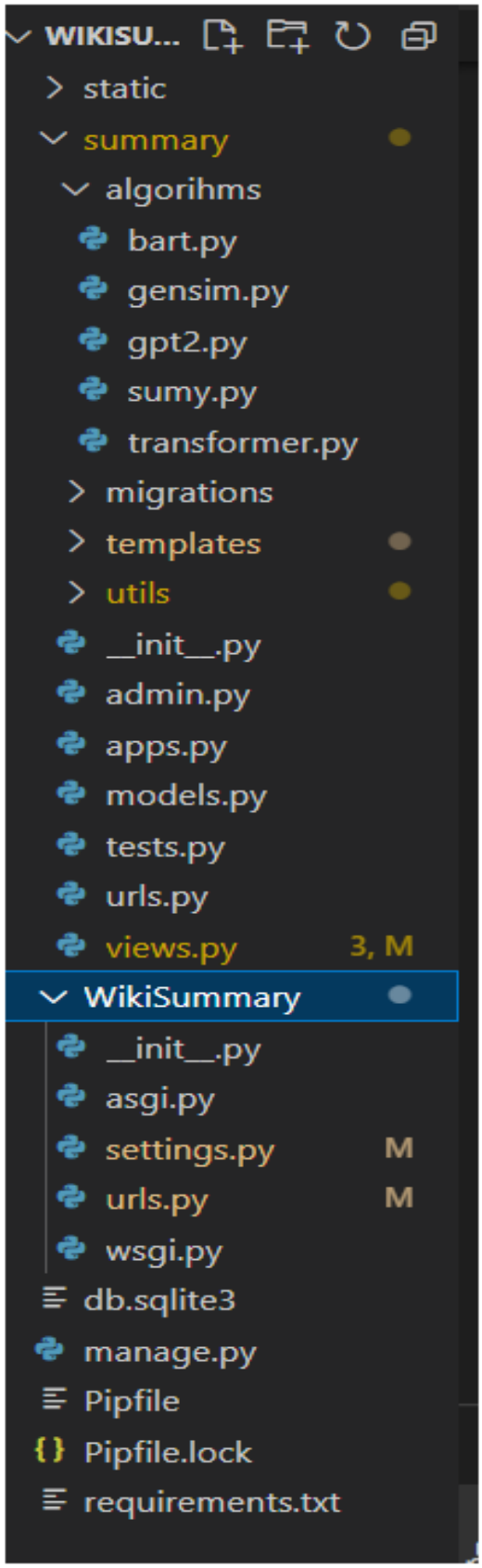
By using Venn diagrams in NLP data visualization, researchers, analysts, and users can gain a deeper understanding of the relationships and overlaps between different sets of words or concepts. These visual representations can aid in identifying patterns, exploring connections, and extracting valuable insights from textual data.

In the home page we take an input of a wikipedia URL (eg.https://en.wikipedia.org/wiki/History_of_India) or a search query (history of India).
The home page will display the summary according to the given options (shown in picture). Along with the summary the wordcloud and count of words (top 24 in count) is also shown.

In the analysis page we will take input similarly. Here some detailed analysis corresponding to the summary involving different algorithms is shown.

Wordcloud with the abstract algorithms and other algorithms are shown differently.

Venn diagrams of the word counts are also shown representing the words used in different algorithms.



5.4.2 Framework

This web application uses the Django framework. Django is a free and open-source web framework written in Python that follows the Model-View-Controller (MVC) architectural pattern. It is designed to help developers build high-level, scalable, and maintainable web applications quickly and easily. Django includes a wide range of built-in features such as an ORM (Object Relational Mapping) system, a templating engine, a URL routing system, an admin interface, and many more. It also comes with a robust security system and built-in support for multiple databases, making it a popular choice among developers for building complex web applications. The framework was initially developed for the web developers working on newspaper sites, but it has now grown to become one of the most widely used frameworks for web development.

5.4.3 Module and API used

Validators Module:

The `validators` module is a Python library that offers a wide range of functions for data validation, including email validation, URL validation, IP address validation, credit card number validation, and more. These functions are helpful in ensuring that user inputs are in the correct format and type, and the module also includes some helper functions for checking the length and range of numbers, and validating dates and times. Overall, the `validators` module is a useful tool for data validation in Python.

torch Module:

The `torch` module is the primary package for building and training deep learning models in Python using the PyTorch library. It provides a wide range of functionality, including tensors for representing numerical data, pre-defined neural network modules, optimization algorithms for training models, loss functions for evaluating model performance, and data loading utilities for processing input data. Additionally, `torch` offers distributed computing capabilities for training models across multiple devices. PyTorch has gained popularity among machine learning researchers and practitioners due to its ease of use, flexibility, and dynamic computational graph.

Wikipedia api:

The Wikipedia API is a set of web APIs that allow developers to programmatically access Wikipedia data and metadata. It provides functionality for searching, retrieving, and querying Wikipedia pages, and can be used for natural language processing, data analysis, and building bots and applications that interact with Wikipedia content.

5.4.4 Methodology of Web application

Set up the development environment:

Install Python: Ensure that Python is installed on our system.
Install Django: Use pip (Python package installer) to install Django by running the command: pip install django.

Create a new Django project:

Open a command prompt or terminal and navigate to the directory where we want to create our project.
Run the command: django-admin startproject project_name. Replace "project_name" with the desired name for our project.This creates a new Django project with a default directory structure.

Create a Django app:

Navigate into the project directory: cd project_name.
Run the command: python manage.py startapp app_name. Replace "app_name" with the desired name for your app.This creates a new Django app within our project.

Create views and templates:

Define the views in the views.py file of our app.
Create HTML templates in the templates directory to define the structure and layout of our web pages.
Implement the necessary logic in views to handle user requests, retrieve Wikipedia articles, and display the summarization results.

Install the required libraries for text summarization:

Install libraries such as nltk or gensim , sumy, torch, transformer for text processing and summarization.
Use pip to install the required packages, for example: pip install nltk.

Implement the summarization logic:

Import the necessary libraries in our Django views or create separate utility functions.
Implement the logic to fetch Wikipedia articles using the Wikipedia API or a web scraping library like BeautifulSoup.
Extract relevant content from the Wikipedia articles using techniques like HTML parsing or regex.
Apply text processing techniques such as tokenization, stop word removal, and stemming/lemmatization.
Utilize text summarization algorithms, such as the TextRank , LSA , T5 algorithm, GPT2 , Bart Models to generate summaries.
Display the summaries directly to the user.

Wire up the URLs:

Define the URL patterns in the urls.py file of our app.
Map the URLs to the corresponding views.

Test and run the application:

Start the development server by running the command: python manage.py runserver.
Open a web browser and visit the specified URL to access our application.
Test the application by creating, retrieving, and summarizing Wikipedia articles.
Ensure that the summarization results are displayed correctly.

Enhance the application:

Improve the user interface by adding CSS styling and JavaScript interactions.
Implement features such as search functionality, pagination, and user authentication.
Handle exceptions and error cases gracefully.
Conduct thorough testing to ensure the application's reliability and performance.

Deploy the application:

Choose a hosting platform or server to deploy our Django application.
Configure the necessary settings for deployment, such as static files, connections, etc.
Follow the deployment instructions provided by our hosting platform or server.

5.4.5 Workflow of Web application

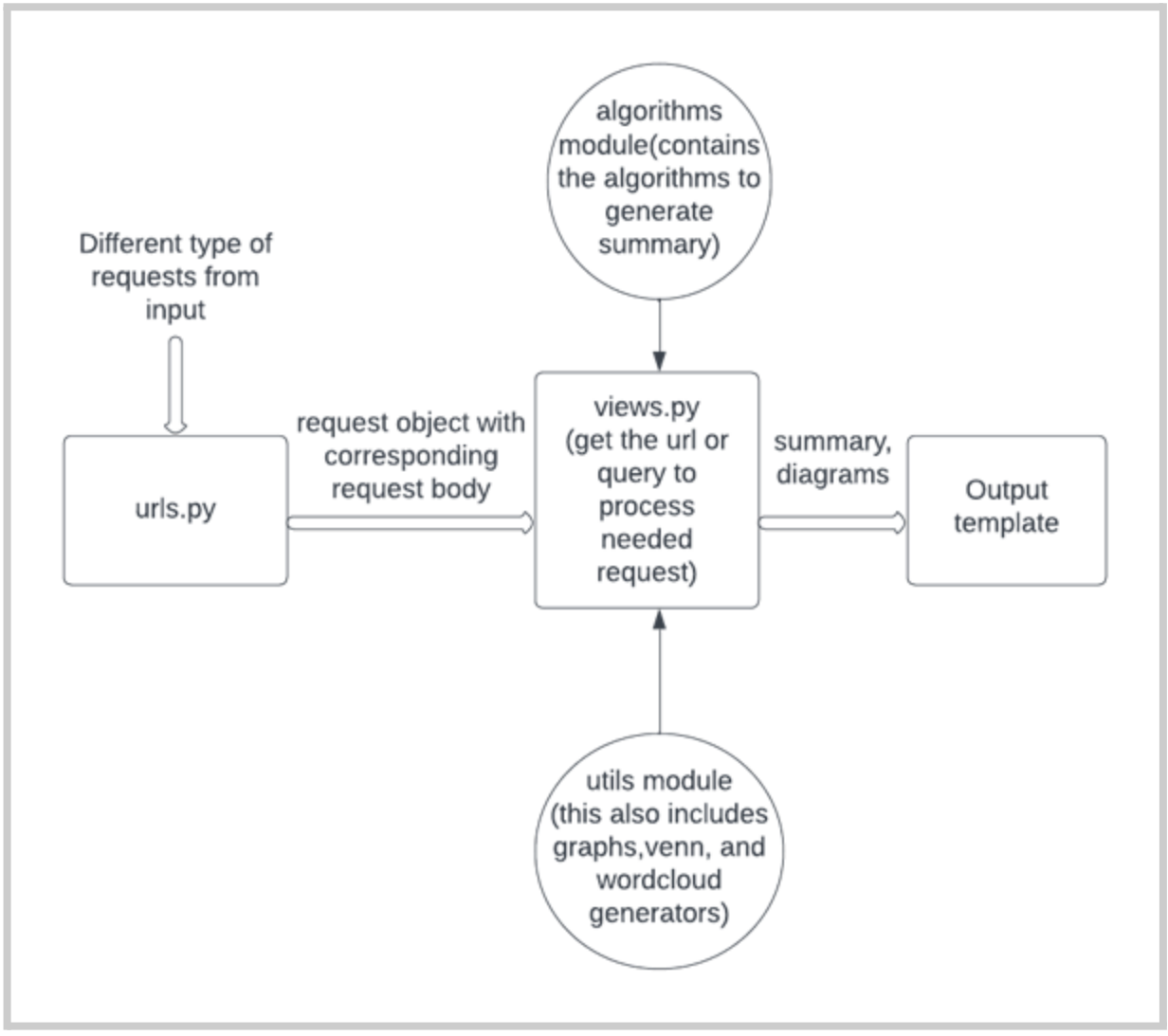


Fig.51: Work for proposed methodology of web application

CHAPTER 6

EXPERIMENT

This chapter describes the experiential outputs and observations using different summarization models discussed in the previous chapter. Detailed experimental output with GUI obtained from all the summarization models has been discussed here.

6.1Data Preparation

The information that is to be embedded plays a vital role. In order to get meaningful output, the input should be acceptable and understandable. The dataset used here is a text document of one Wikipedia article live URL. The proposed automatic text summarization based on a single document system is implemented in the PYTHON program and the text summarization process is experimented with the document is collected from WIKIPEDIA Article. There were a number of different Python packages and modules used for different areas of functionality across this works and shows the input text taken for experimental purpose.

6.2 Experiment & Evaluation

The experiment is conducted to figure out the best possible way to solve the extractive text summarization using Five models like Summary by Sentence using LSA model, Summary by Ratio using TextRank algorithm, Summary by Word count using TextRank algorithm for doing extractive summarization on Wikipedia Live Urls .like :

Wiki URL 1: "https://en.wikipedia.org/wiki/History_of_India"
Wiki URL 2: "https://en.wikipedia.org/wiki/Kolkata"
Wiki URL 3: "https://en.wikipedia.org/wiki/Rabindranath_Tagore".

The experiment is conducted over different sets of Wikipedia Live Urls. First experiment is conducted on WikiURL 1 of Wikipedia url "https://en.wikipedia.org/wiki/History_of_India" ,

6.2.1 Experiment 1

First experiment is done with TextRank, LSA, T5 Transformer, GPT2, BART model in a GUI application on WikiURL1 "https://en.wikipedia.org/wiki/History_of_India"

6.2.1.1 Experiemnt by TextRank:

We have used the TextRank algorithm using Gensym Python Library for generating two types of extractive summary as shown in the output in the Fig.53 and Fig.54 ,

a) Summary by Ratio(ratio=0.1 percent),

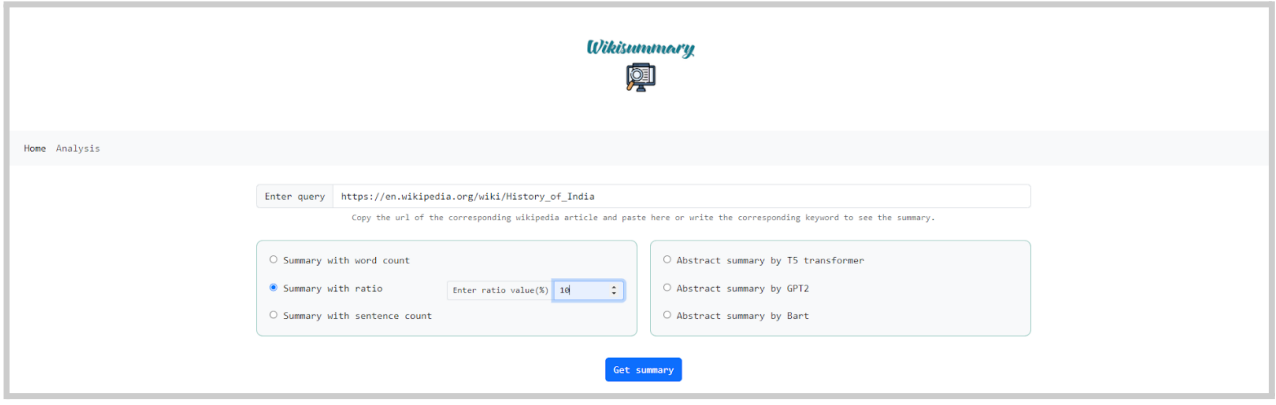


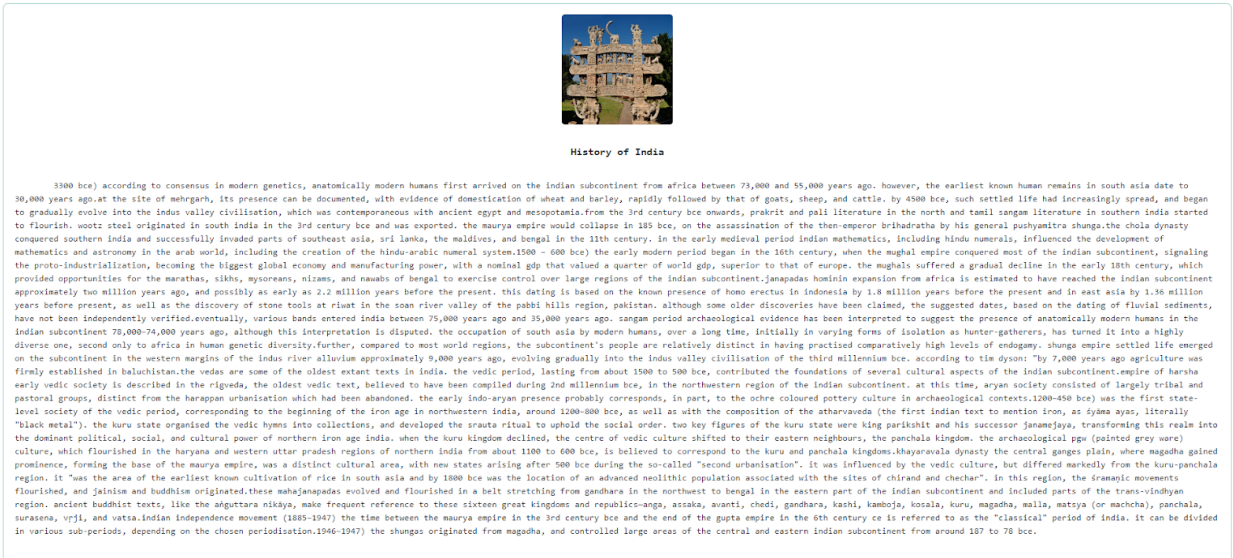
Fig.52: Home Page for Summary Generation

History of India

3300 bce) according to consensus in modern genetics, anatomically modern humans first arrived on the indian subcontinent from africa between 73,000 and 55,000 years ago. however, the earliest known human remains in south asia date to 30,000 years ago.at the site of mehrgarh, its presence can be documented, with evidence of domestication of wheat and barley, rapidly followed by that of goats, sheep, and cattle. by 4500 bce, such settled life had increasingly spread, and began to gradually evolve into the indus valley civilisation, which was contemporaneous with ancient egypt and mesopotamia.from the 3rd century bce onwards, prakrit and pali literature in the north and tamil sangam literature in southern india started to flourish. wootz steel originated in south india in the 3rd century bce and was exported. the maurya empire would collapse in 185 bce, on the assassination of the then-emperor brihadratha by his general pushyamitra shunga.indus valley civilisation this period, witnessing a hindu religious and intellectual resurgence, is known as the classical or golden age of india.indian cultural influence spread over many parts of southeast asia, which led to the establishment of indianised kingdoms in the region, forming the greater india.the chola dynasty conquered southern india and successfully invaded parts of southeast asia, sri lanka, the maldives, and bengal in the 11th century. in the early medieval period indian mathematics, including hindu numerals, influenced the development of mathematics and astronomy in the arab world, including the creation of the hindu-arabic numeral system.the delhi sultanate was founded in 1206 ce by central asian turks who ruled a major part of the northern indian subcontinent in the early 14th century, but declined in the late 14th century, and saw the advent of the deccan sultanates. the wealthy bengal sultanate also emerged as a major power, lasting over three centuries.1500 – 600 bce) the early modern period began in the 16th century, when the mughal empire conquered most of the indian subcontinent, signaling the proto-industrialization, becoming the biggest global economy and manufacturing power, with a nominal gdp that valued a quarter of world gdp, superior to that of europe. the mughals suffered a gradual decline in the early 18th century, which provided opportunities for the marathas, sikhs, mysoreans, nizams, and nawabs of bengal to exercise control over large regions of the indian subcontinent.janapadas hominin expansion from africa is estimated to have reached the indian subcontinent approximately two million years ago, and possibly as early as 2.2 million years before the present. this dating is based on the known presence of homo erectus in indonesia by 1.8 million years before the present and in east asia by 1.36 million years before present, as well as the discovery of stone tools at riwat in the soan river valley of the pabbi hills region, pakistan. although some older discoveries have been claimed, the suggested dates, based on the dating of fluvial sediments, have not been independently verified.eventually, various bands entered india between 75,000 years ago and 35,000 years ago. sangam period archaeological evidence has been interpreted to suggest the presence of anatomically modern humans in the indian subcontinent 78,000–74,000 years ago, although this interpretation is disputed. the occupation of south asia by modern humans, over a long time, initially in varying forms of isolation as hunter-gatherers, has turned it into a highly diverse one, second only to africa in human genetic diversity.further, compared to most world regions, the subcontinent's people are relatively distinct in having practised comparatively high levels of endogamy. shunga empire settled life emerged on the subcontinent in the western margins of the indus river alluvium approximately 9,000 years ago, evolving gradually into the indus valley civilisation of the third millennium bce. according to tim dyson: "by 7,000 years ago agriculture was firmly established in baluchistan.the vedas are some of the oldest extant texts in india. the vedic period, lasting from about 1500 to 500 bce, contributed the foundations of several cultural aspects of the indian subcontinent.empire of harsha early vedic society is described in the rigveda, the oldest vedic text, believed to have been compiled during 2nd millennium bce, in the northwestern region of the indian subcontinent. at this time, aryan society consisted of largely tribal and pastoral groups, distinct from the harappan urbanisation which had been abandoned. the early indo-aryan presence probably corresponds, in part, to the ochre coloured pottery culture in archaeological contexts.1200–450 bce) was the first state-level society of the vedic period, corresponding to the beginning of the iron age in northwestern india, around 1200–800 bce, as well as with the composition of the atharvaveda (the first indian text to mention iron, as śyāma ayas, literally "black metal"). the kuru state organised the vedic hymns into collections, and developed the srauta ritual to uphold the social order. two key figures of the kuru state were king parikshit and his successor janamejaya, transforming this realm into the dominant political, social, and cultural power of northern iron age india. when the kuru kingdom declined, the centre of vedic culture shifted to their eastern neighbours, the panchala kingdom. the archaeological pgw (painted grey ware) culture, which flourished in the haryana and western uttar pradesh regions of northern india from about 1100 to 600 bce, is believed to correspond to the kuru and panchala kingdoms.gurjara-pratihara empire during the late vedic period, the kingdom of videha emerged as a new centre of vedic culture, situated even farther to the east (in what is today nepal and bihar state in india); reaching its prominence under the king janaka, whose court provided patronage for brahmin sages and philosophers such as yajnavalkya, aruni, and gārgī vāchaknavi. the later part of this period corresponds with a consolidation of increasingly large states and kingdoms, called mahajanapadas, all across northern india.khayaravala dynasty the central ganges plain, where magadha gained prominence, forming the base of the maurya empire, was a distinct cultural area, with new states arising after 500 bce during the so-called "second urbanisation". it was influenced by the vedic culture, but differed markedly from the kuru-panchala region. it "was the area of the earliest known cultivation of rice in south asia and by 1800 bce was the location of an advanced neolithic population associated with the sites of chirand and chechar". in this region, the śramaṇic movements flourished, and jainism and buddhism originated.these mahajanapadas evolved and flourished in a belt stretching from gandhara in the northwest to bengal in the eastern part of the indian subcontinent and included parts of the trans-vindhyan region. ancient buddhist texts, like the aṅguttara nikāya, make frequent reference to these sixteen great kingdoms and republics—anga, assaka, avanti, chedi, gandhara, kashi, kamboja, kosala, kuru, magadha, malla, matsya (or machcha), panchala, surasena, vṛji, and vatsa.this period saw the second major rise of urbanism in india after the indus valley civilisation.especially focused in the central ganges plain but also spreading across vast areas of the northern and central indian subcontinent, this culture is characterized by the emergence of large cities with massive fortifications, significant population growth, increased social stratification, wide-ranging trade networks, construction of public architecture and water channels, specialized craft industries (e.g., ivory and carnelian carving), a system of weights, punch-marked coins, and the introduction of writing in the form of brahmi and kharosthi scripts. the language of the gentry at that time was sanskrit, while the languages of the general population of northern india are referred to as prakrits.indian independence movement (1885–1947) the time between the maurya empire in the 3rd century bce and the end of the gupta empire in the 6th century ce is referred to as the "classical" period of india. it can be divided in various sub-periods, depending on the chosen periodisation.1946–1947) the shungas originated from magadha, and controlled large areas of the central and eastern indian subcontinent from around 187 to 78 bce.however, after the death of agnimitra, the empire rapidly disintegrated; inscriptions and coins indicate that much of northern and central india consisted of small kingdoms and city-states that were independent of any shunga hegemony. the empire is noted for its numerous wars with both foreign and indigenous powers.

Fig.53: Summary by Ratio using TextRank

b) Summary by Word count(set word count=1000)



History of India

3300 bce) according to consensus in modern genetics, anatomically modern humans first arrived on the indian subcontinent from africa between 73,000 and 55,000 years ago. however, the earliest known human remains in south asia date to 30,000 years ago.at the site of mehrgarh, its presence can be documented, with evidence of domestication of wheat and barley, rapidly followed by that of goats, sheep, and cattle. by 4500 bce, such settled life had increasingly spread, and began to gradually evolve into the indus valley civilisation, which was contemporaneous with ancient egypt and mesopotamia.from the 3rd century bce onwards, prakrit and pali literature in the north and tamil sangam literature in southern india started to flourish. wootz steel originated in south india in the 3rd century bce and was exported. the maurya empire would collapse in 185 bce, on the assassination of the then-emperor brihadratha by his general pushyamitra shunga.the chola dynasty conquered southern india and successfully invaded parts of southeast asia, sri lanka, the maldives, and bengal in the 11th century. in the early medieval period indian mathematics, including hindu numerals, influenced the development of mathematics and astronomy in the arab world, including the creation of the hindu-arabic numeral system.1500 – 600 bce) the early modern period began in the 16th century, when the mughal empire conquered most of the indian subcontinent, signaling the proto-industrialization, becoming the biggest global economy and manufacturing power, with a nominal gdp that valued a quarter of world gdp, superior to that of europe. the mughals suffered a gradual decline in the early 18th century, which provided opportunities for the marathas, sikhs, mysoreans, nizams, and nawabs of bengal to exercise control over large regions of the indian subcontinent.janapadas hominin expansion from africa is estimated to have reached the indian subcontinent approximately two million years ago, and possibly as early as 2.2 million years before the present. this dating is based on the known presence of homo erectus in indonesia by 1.8 million years before the present and in east asia by 1.36 million years before present, as well as the discovery of stone tools at riwat in the soan river valley of the pabbi hills region, pakistan. although some older discoveries have been claimed, the suggested dates, based on the dating of fluvial sediments, have not been independently verified.eventually, various bands entered india between 75,000 years ago and 35,000 years ago. sangam period archaeological evidence has been interpreted to suggest the presence of anatomically modern humans in the indian subcontinent 78,000–74,000 years ago, although this interpretation is disputed. the occupation of south asia by modern humans, over a long time, initially in varying forms of isolation as hunter-gatherers, has turned it into a highly diverse one, second only to africa in human genetic diversity.further, compared to most world regions, the subcontinent's people are relatively distinct in having practised comparatively high levels of endogamy. shunga empire settled life emerged on the subcontinent in the western margins of the indus river alluvium approximately 9,000 years ago, evolving gradually into the indus valley civilisation of the third millennium bce. according to tim dyson: "by 7,000 years ago agriculture was firmly established in baluchistan.the vedas are some of the oldest extant texts in india. the vedic period, lasting from about 1500 to 500 bce, contributed the foundations of several cultural aspects of the indian subcontinent.empire of harsha early vedic society is described in the rigveda, the oldest vedic text, believed to have been compiled during 2nd millennium bce, in the northwestern region of the indian subcontinent. at this time, aryan society consisted of largely tribal and pastoral groups, distinct from the harappan urbanisation which had been abandoned. the early indo-aryan presence probably corresponds, in part, to the ochre coloured pottery culture in archaeological contexts.1200–450 bce) was the first state-level society of the vedic period, corresponding to the beginning of the iron age in northwestern india, around 1200–800 bce, as well as with the composition of the atharvaveda (the first indian text to mention iron, as śyāma ayas, literally "black metal"). the kuru state organised the vedic hymns into collections, and developed the srauta ritual to uphold the social order. two key figures of the kuru state were king parikshit and his successor janamejaya, transforming this realm into the dominant political, social, and cultural power of northern iron age india. when the kuru kingdom declined, the centre of vedic culture shifted to their eastern neighbours, the panchala kingdom. the archaeological pgw (painted grey ware) culture, which flourished in the haryana and western uttar pradesh regions of northern india from about 1100 to 600 bce, is believed to correspond to the kuru and panchala kingdoms.khayaravala dynasty the central ganges plain, where magadha gained prominence, forming the base of the maurya empire, was a distinct cultural area, with new states arising after 500 bce during the so-called "second urbanisation". it was influenced by the vedic culture, but differed markedly from the kuru-panchala region. it "was the area of the earliest known cultivation of rice in south asia and by 1800 bce was the location of an advanced neolithic population associated with the sites of chirand and chechar". in this region, the śramaṇic movements flourished, and jainism and buddhism originated.these mahajanapadas evolved and flourished in a belt stretching from gandhara in the northwest to bengal in the eastern part of the indian subcontinent and included parts of the trans-vindhyan region. ancient buddhist texts, like the aṅguttara nikāya, make frequent reference to these sixteen great kingdoms and republics—anga, assaka, avanti, chedi, gandhara, kashi, kamboja, kosala, kuru, magadha, malla, matsya (or machcha), panchala, surasena, vṛji, and vatsa.indian independence movement (1885–1947) the time between the maurya empire in the 3rd century bce and the end of the gupta empire in the 6th century ce is referred to as the "classical" period of india. it can be divided in various sub-periods, depending on the chosen periodisation.1946–1947) the shungas originated from magadha, and controlled large areas of the central and eastern indian subcontinent from around 187 to 78 bce.

Fig.54: Summary by Word count using TextRank

We are representing the wordcloud for the most frequent words present in the Summary by Ratio as shown in Fig.55 and Summary by Word count as shown in the Fig. 56.

Fig.55: Word Cloud Summary by Ratio using TextRank

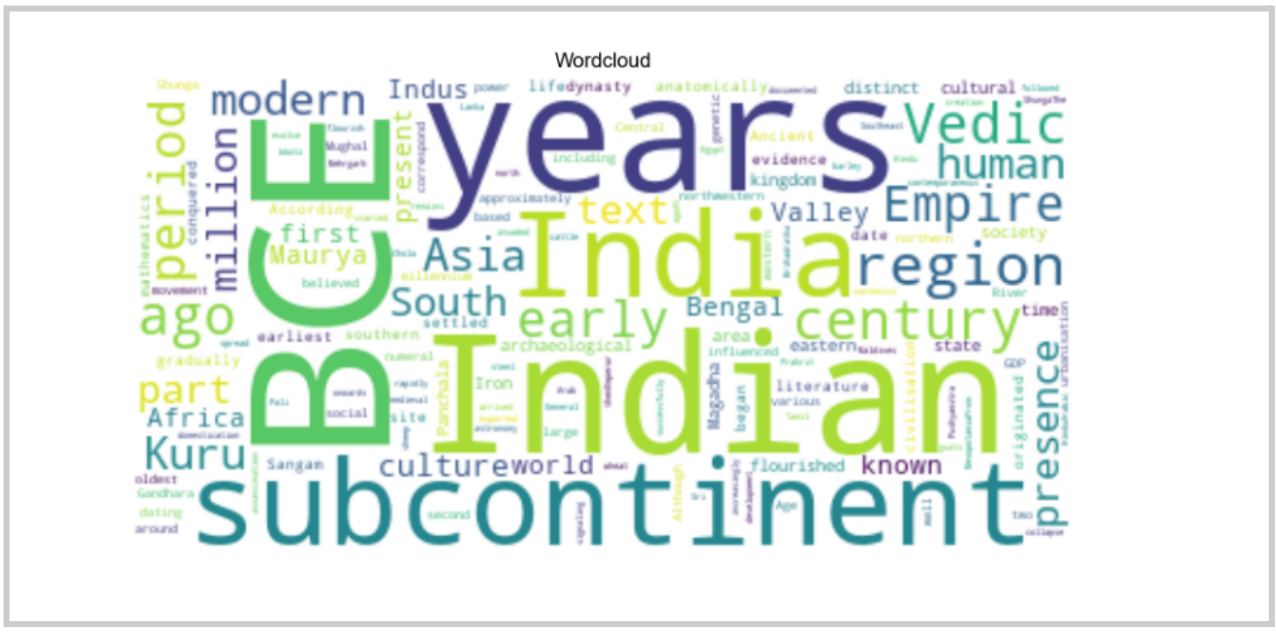


Fig.56: Word Cloud Summary by Word count using TextRank

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary by Ratio as shown in the Fig.57 and Summary by Word count as shown in the Fig.58.

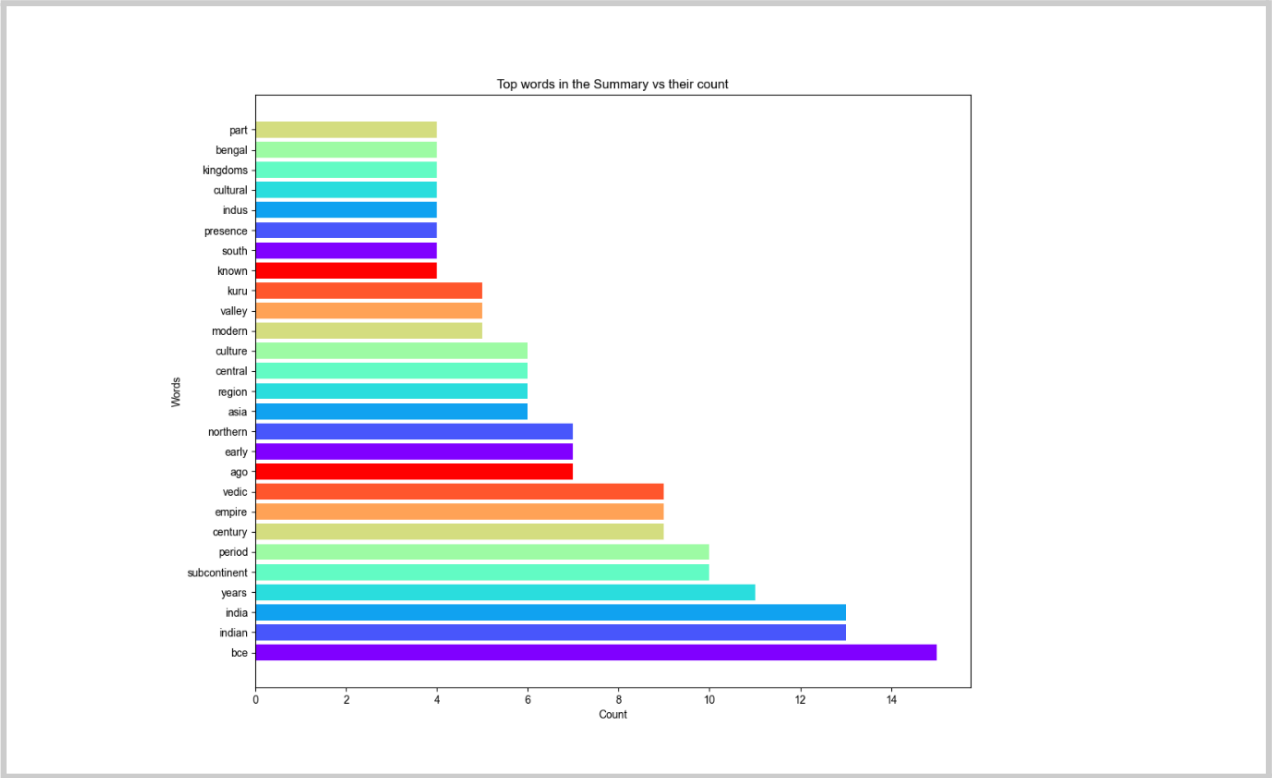


Fig.57: Most Common Word vs Count Summary by Ratio count using TextRank

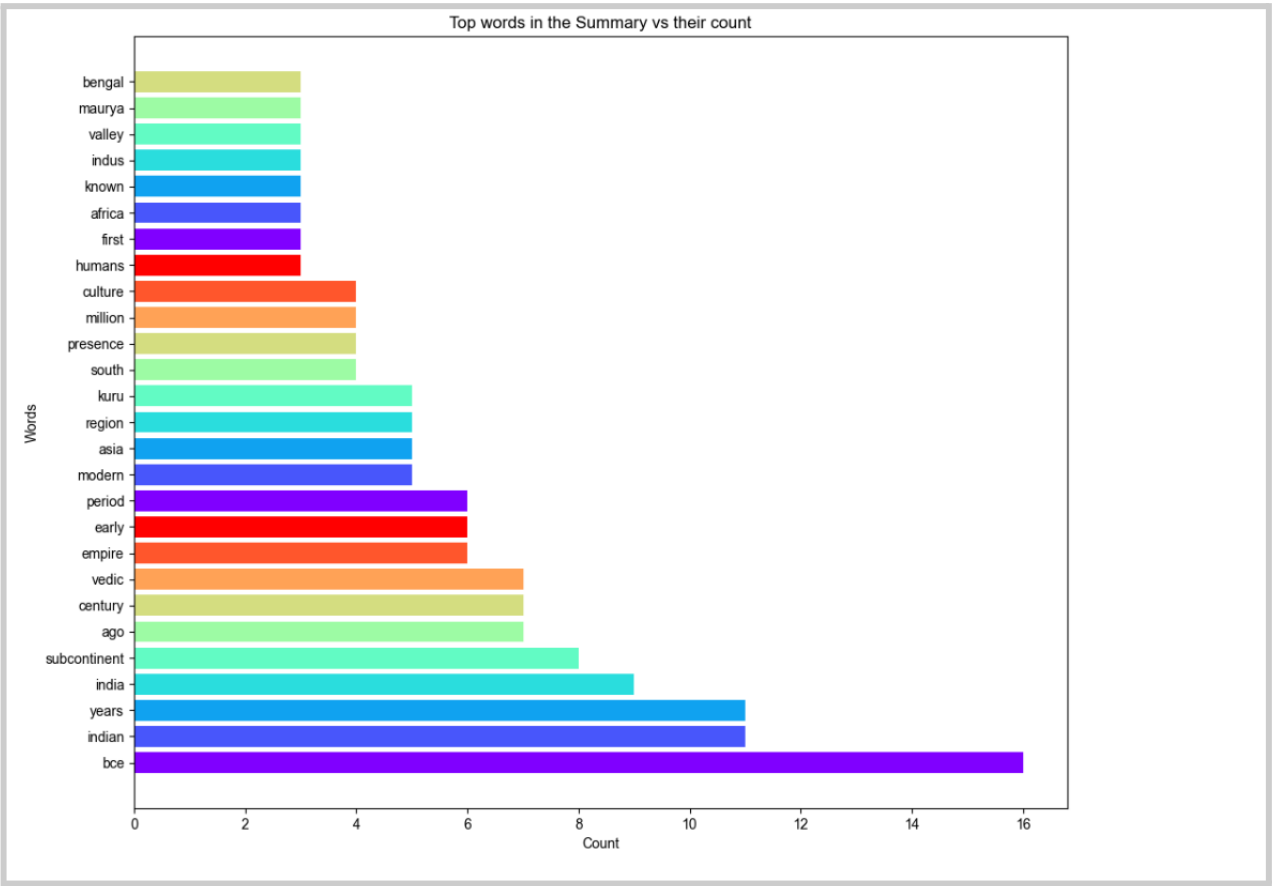


Fig.58: Most Common Word vs Count Summary by Word count using TextRank

6.2.1.2 Experiment by LSA:

We have used the LSA algorithm using Sumy Python Library for generating extractive summary by sentences (no of sentences=50)as shown in the output in the Fig. 59

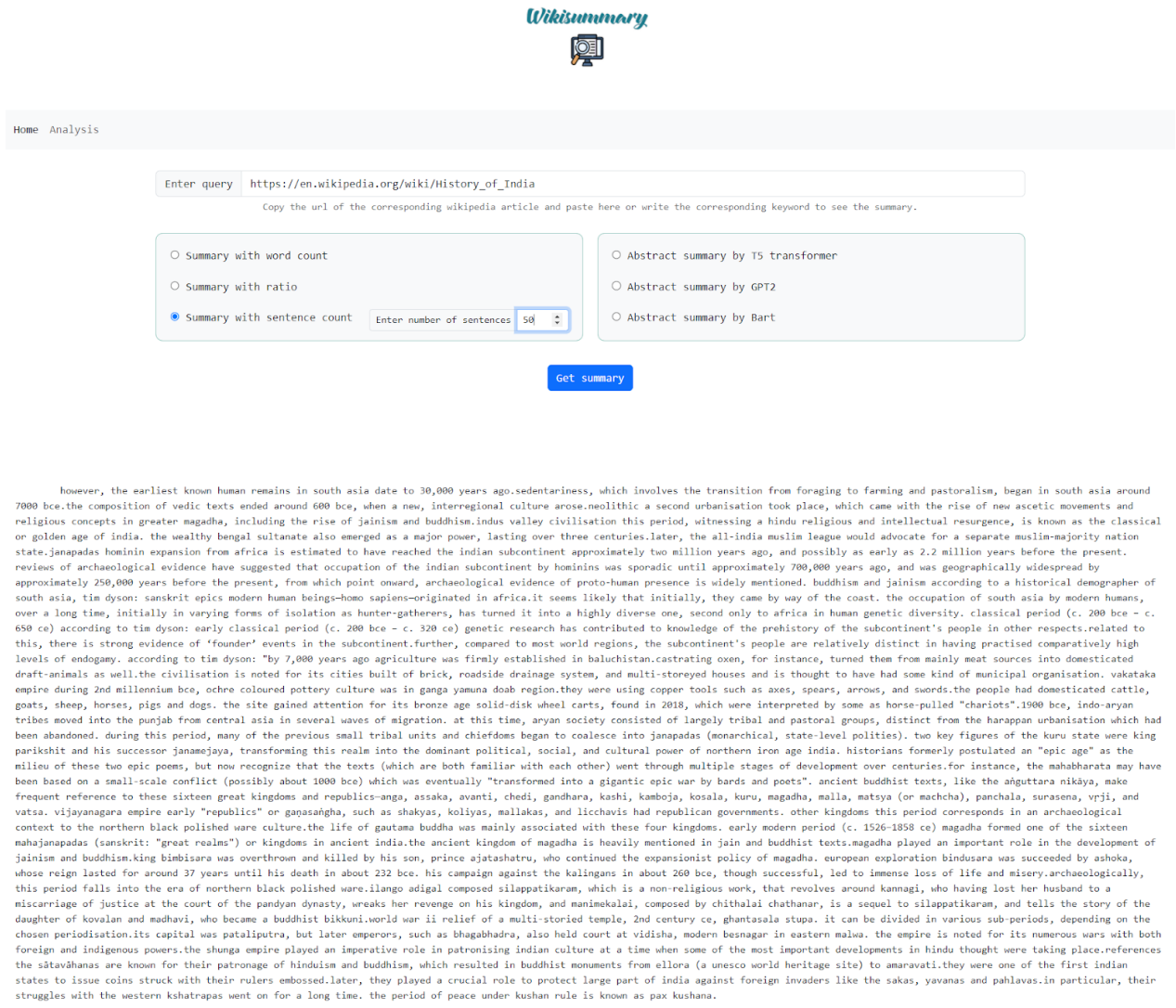


Fig.59: Summary by Sentences using LSA

We are representing the Word Cloud for the most frequent words present in the Summary by Sentences as shown in Fig. 60.

Fig.60: Word Cloud Summary by Sentences using LSA

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary by Sentences as shown in Fig. 61.

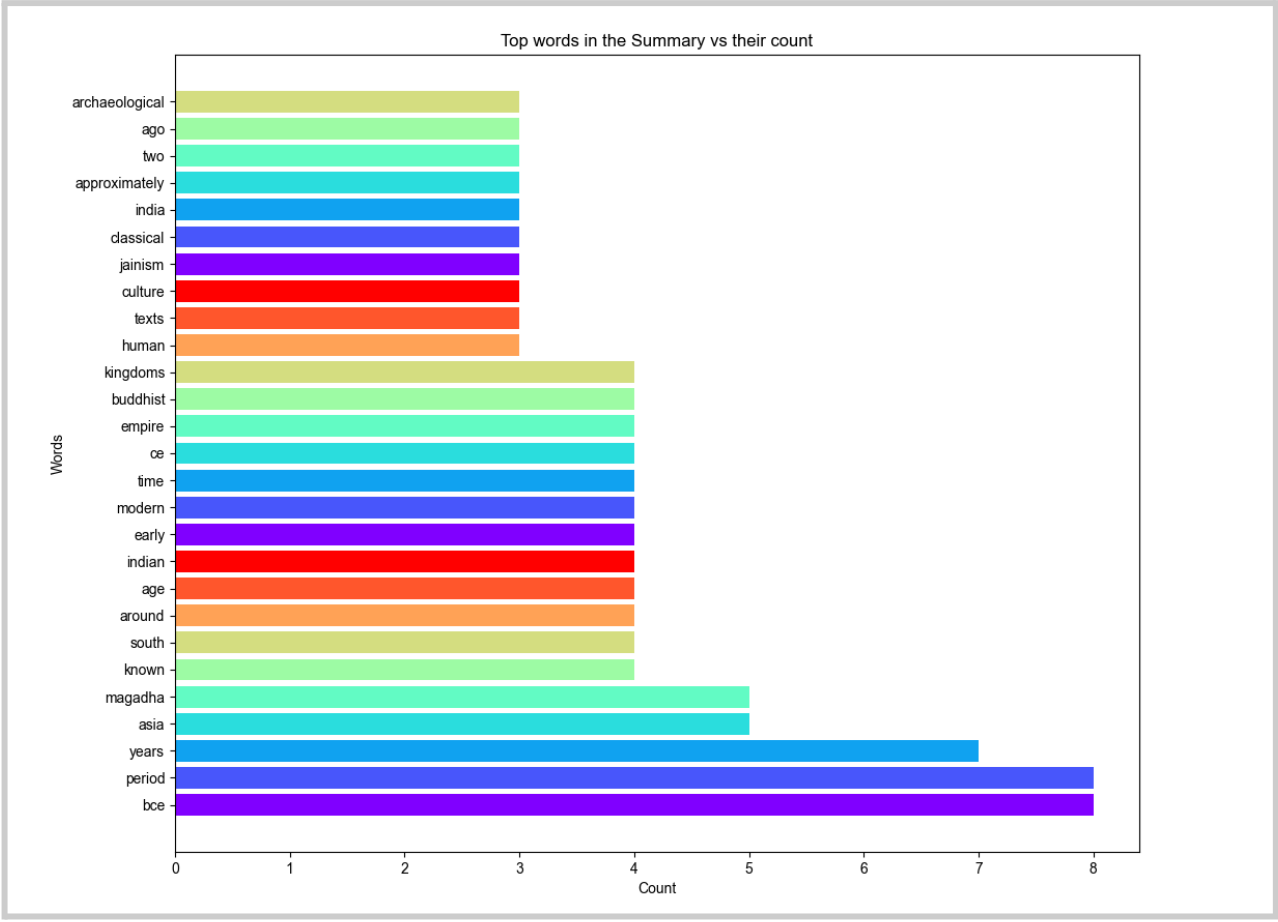


Fig.61: Most Common Word vs Count of Summary by LSA

6.2.1.3 Experiment by T5 Transformer:

We have used the T5 (Text-to-Text Transformer) model using torch, transformers, AutoTokenizer, AutoModelForSeq2SeqLM Python Libraries for generating abstractive summary as shown in the output in Fig. 62.

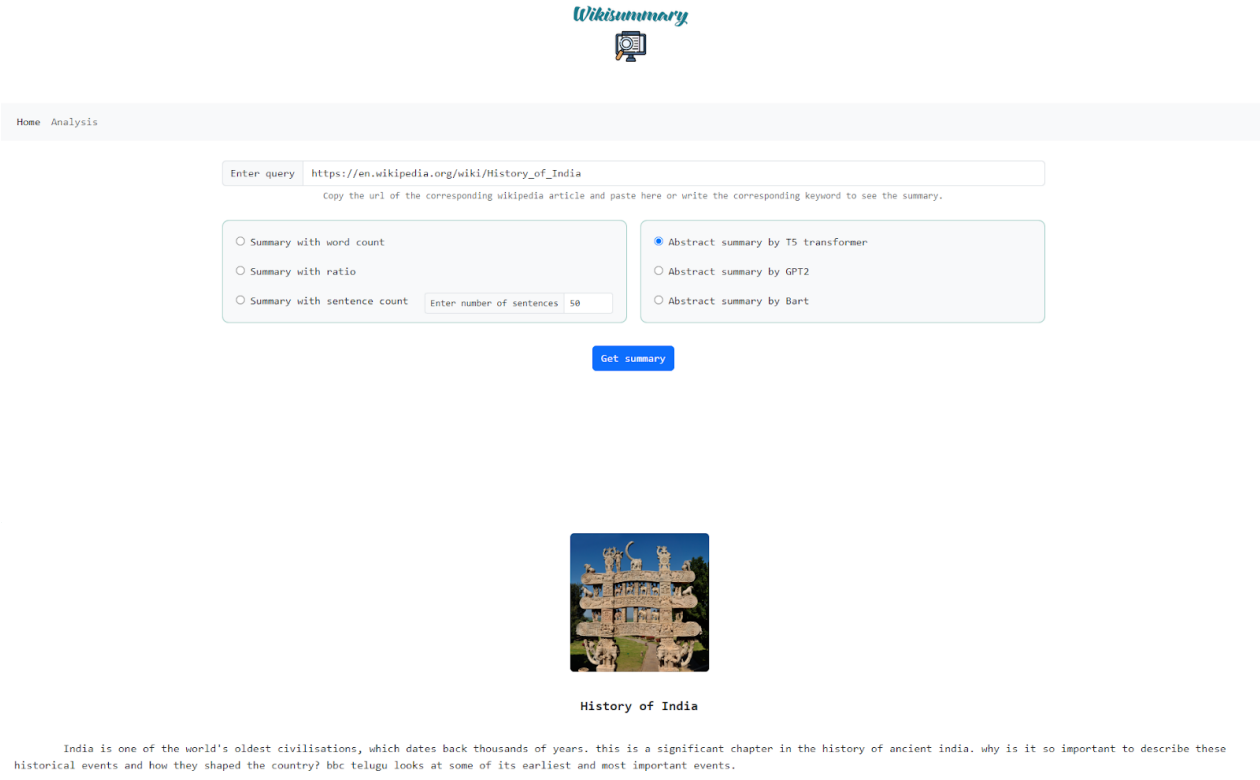


Fig.62: Summary by T5 Transformer

We are representing the Word Cloud for the most frequent words present in the Summary as shown in the Fig. 63

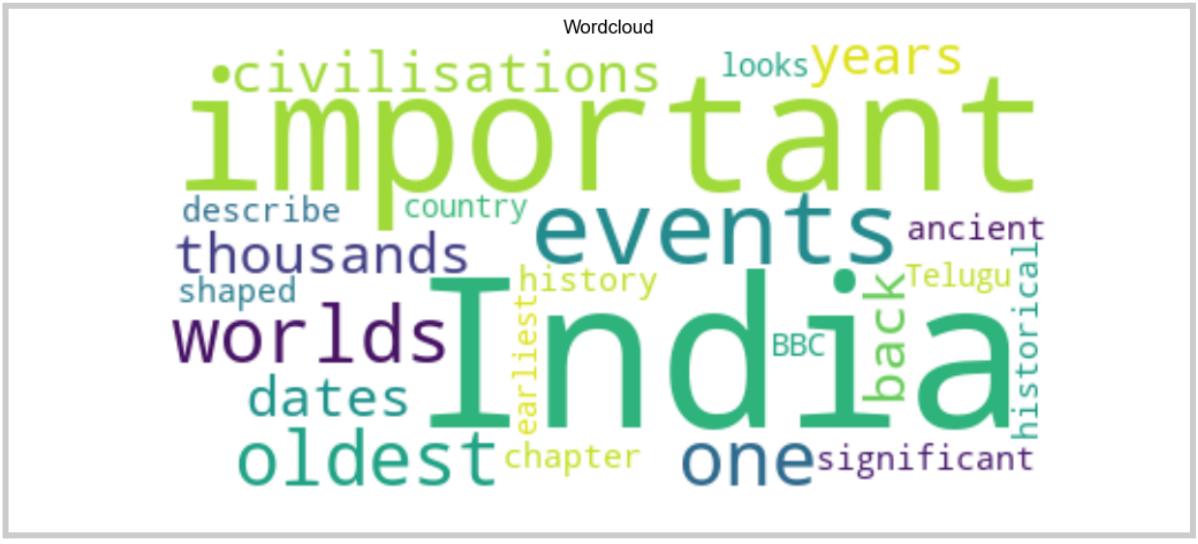


Fig.63: Word Cloud of Summary by T5 Transformer

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in the Fig. 64.

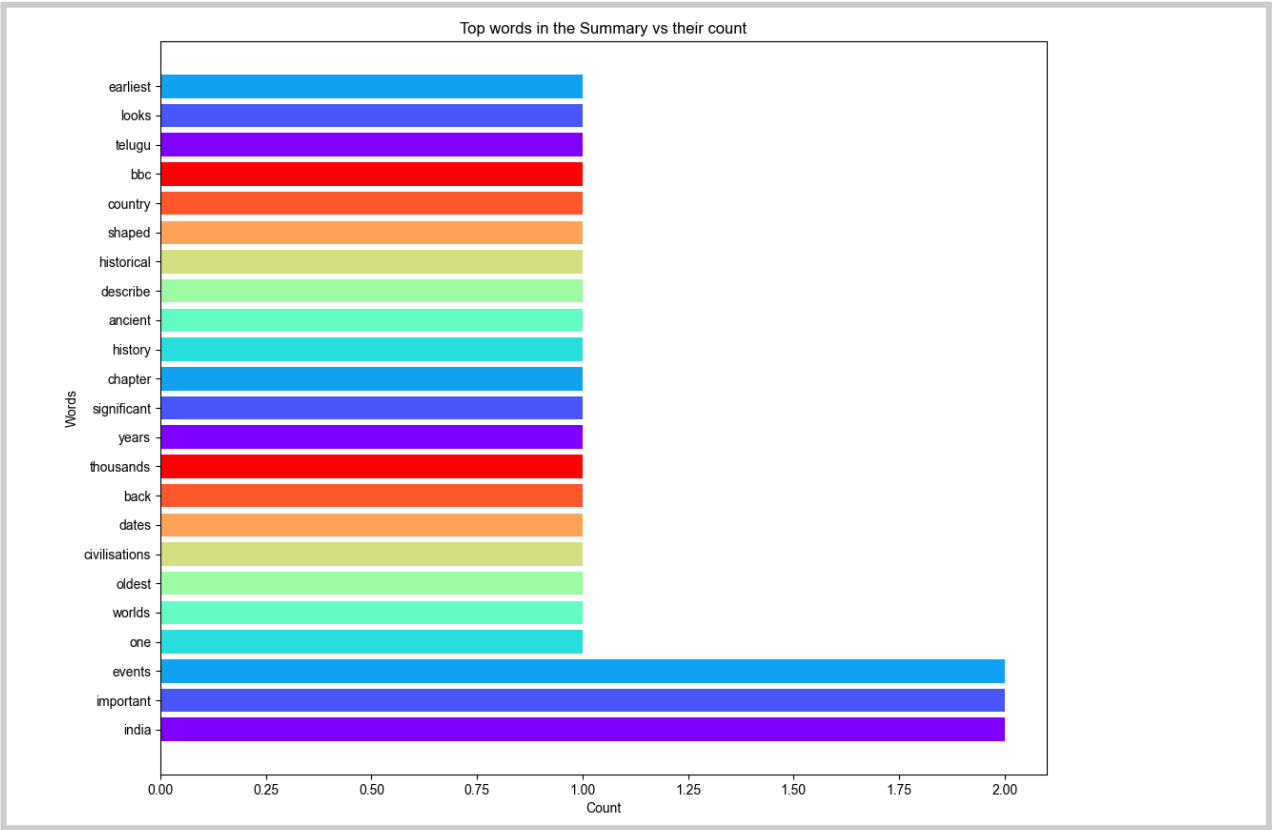


Fig.64: Most Common Word vs Count of Summary by T5 Transformer

6.2.1.4 Experiment by GPT2 model:

We have used the GPT2 model using torch, transformers, GPT2Tokenizer, GPT2LMHeadModel Python Libraries for generating abstractive summary as shown in the output in Fig. 65.

History of India

prehistoric era (until c. 3300 bce) according to consensus in modern genetics, anatomically modern humans first arrived on the indian subcontinent from africa between 73,000 and 55,000 years ago. however, the earliest known human remains in south asia date to 30,000 years ago. sedentariness, which involves the transition from foraging to farming and pastoralism, began in south asia around 7000 bce. at the site of mehrgarh, its presence can be documented, with evidence of domestication of wheat and barley, rapidly followed by that of goats, sheep, and cattle. by 4500 bce, such settled life had increasingly spread, and began to gradually evolve into the indus valley civilisation, which was contemporaneous with ancient egypt and mesopotamia. this civilisation flourished between 2500 bce and 1900 bce in present-day pakistan and north-western india, and was noted for its urban planning, baked brick houses, elaborate drainage, and water supply. paleolithic early on in the second millennium bce, persistent drought caused the population of the indus valley to scatter from large urban centres to villages. around the same time, indo-aryan tribes moved into the punjab from central asia in several waves of migration. the vedic period (1500–500 bce) was marked by the composition of their large collections of hymns called vedas. their varna system, which evolved into the caste system, consisted of a hierarchy of priests, warriors, free peasants, and servants. the pastoral and nomadic indo-aryans spread from the punjab into the gangetic plain, large swaths of which they deforested for agriculture. the composition of vedic texts ended around 600 bce, when a new, interregional culture arose. then, small chieftaincies (janapadas) were consolidated into larger states (mahajanapadas). neolithic a second urbanisation took place, which came with the rise of new ascetic movements and religious concepts in greater magadha, including the rise of jainism and buddhism. these opposed the growing influence of brahmanism and the primacy of rituals—often presided by brahmin priests—that had come to be associated with vedic religion. in response to the success of these movements, the latter was synthesised with the preexisting religious cultures of the subcontinent, giving rise to hinduism. bronze age (c. 3300 – c. 1800 bce) chandragupta maurya with the guidance of acharya chanakya succeeded in overthrowing the nanda empire and established the first great empire in ancient india, the maurya empire. most of the indian subcontinent was conquered by the maurya empire during the 4th and 3rd centuries bce. from the 3rd century bce onwards, prakrit and pali literature in the north and tamil sangam literature in southern india started to flourish. wootz steel originated in south india in the 3rd century bce and was exported. the maurya empire would collapse in 185 bce, on the assassination of the then-emperor brihadratha by his general pushyamitra shunga. shunga would go on to form the shunga empire in the north and northeast of the subcontinent, while the greco-bactrian kingdom would claim the northwest and found the indo-greek kingdom. various parts of india were ruled by numerous dynasties, including the gupta empire in the 4-6th centuries ce. indus valley civilisation this period, witnessing a hindu religious and intellectual resurgence, is known as the classical or golden age of india. during this time, aspects of indian civilisation, administration, culture, and religion spread to much of asia. kingdoms in southern india had maritime business links with the middle east and the mediterranean. indian cultural influence spread over many parts of southeast asia, which led to the establishment of indianised kingdoms in the region, forming the greater india. ochre coloured pottery culture the most significant event between the 7th and 11th century was the tripartite struggle centred on kannauj that lasted for more than two centuries between the pala empire, rashtrakuta empire, and gurjara-pratihara empire. southern india saw the rise of multiple imperial powers from the middle of the fifth century, most notably the chalukya, chola, pallava, chera, pandyan, and western chalukya empires. the chola dynasty conquered southern india and successfully invaded parts of southeast asia, sri lanka, the maldives, and bengal in the 11th century. in the early medieval period indian mathematics, including hindu numerals, influenced the development of mathematics and astronomy in the arab world, including the creation of the hindu and

Fig.65: Summary by GPT2

We are representing the wordcloud for the most frequent words present in the Summary as shown in Fig. 66.

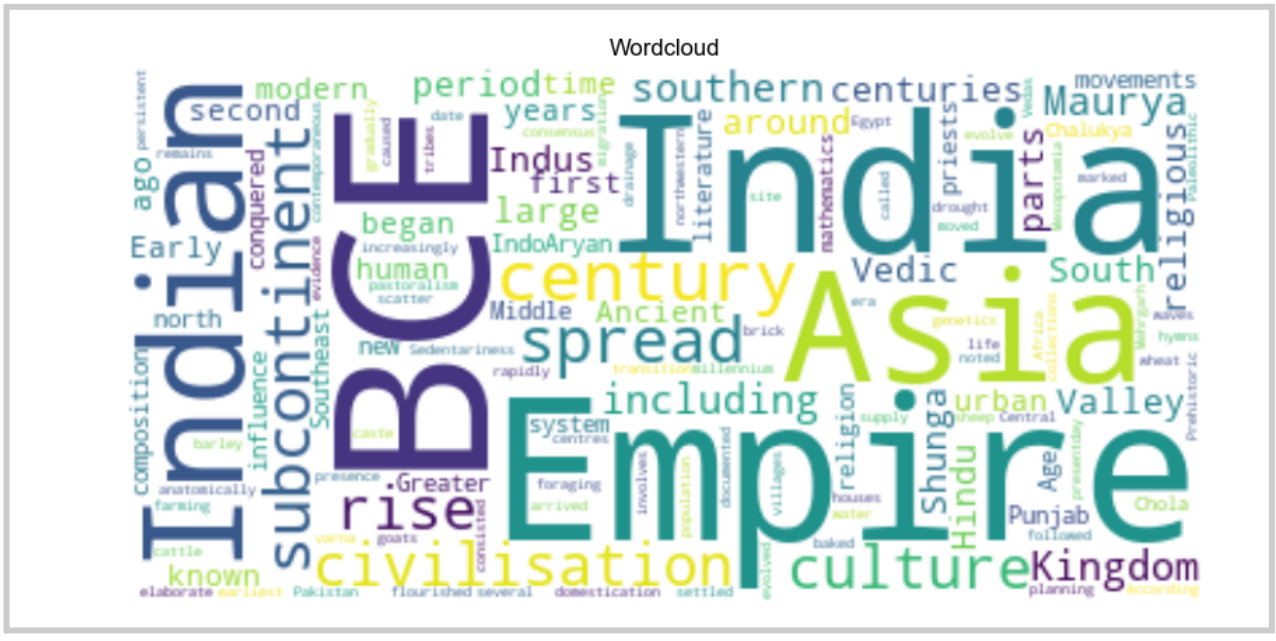


Fig.66: Word Cloud of Summary by GPT2

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in Fig. 67.

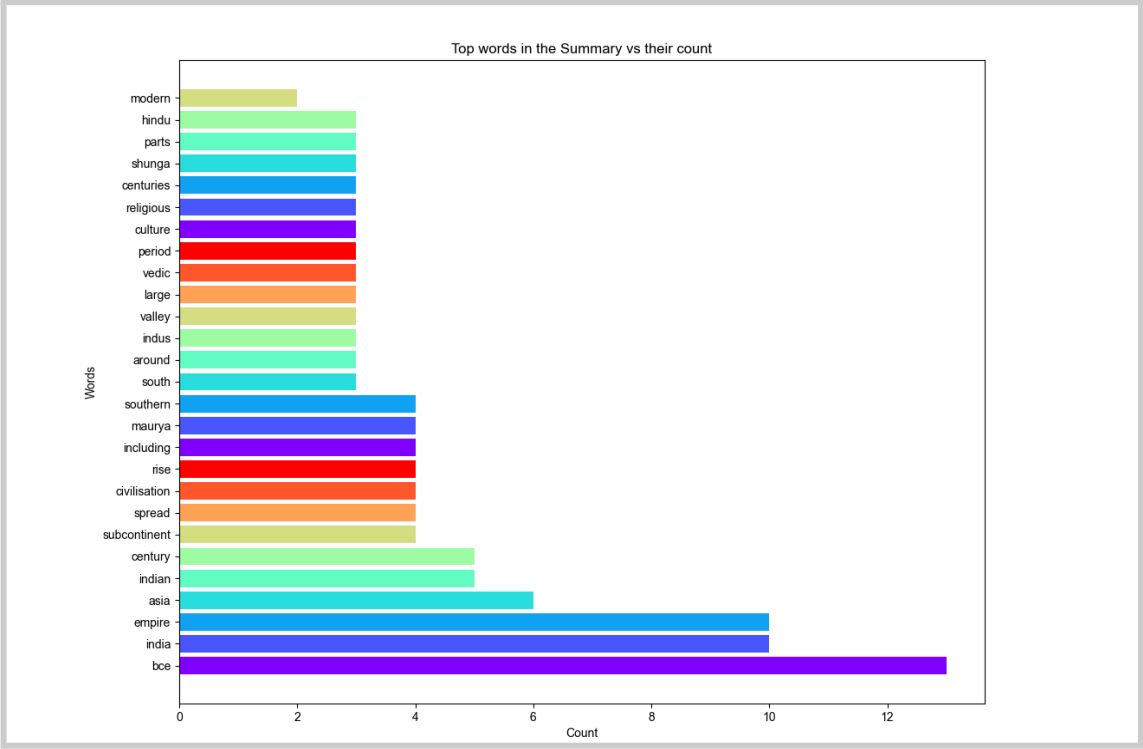


Fig.67: Most Common Word vs Count of Summary by GPT2

6.2.1.5 Experiment by BART model:

We have used the BART model using BartForConditionalGeneration, BartTokenizer, BartConfig Python Library for generating abstractive summary as shown in the output in Fig.68.

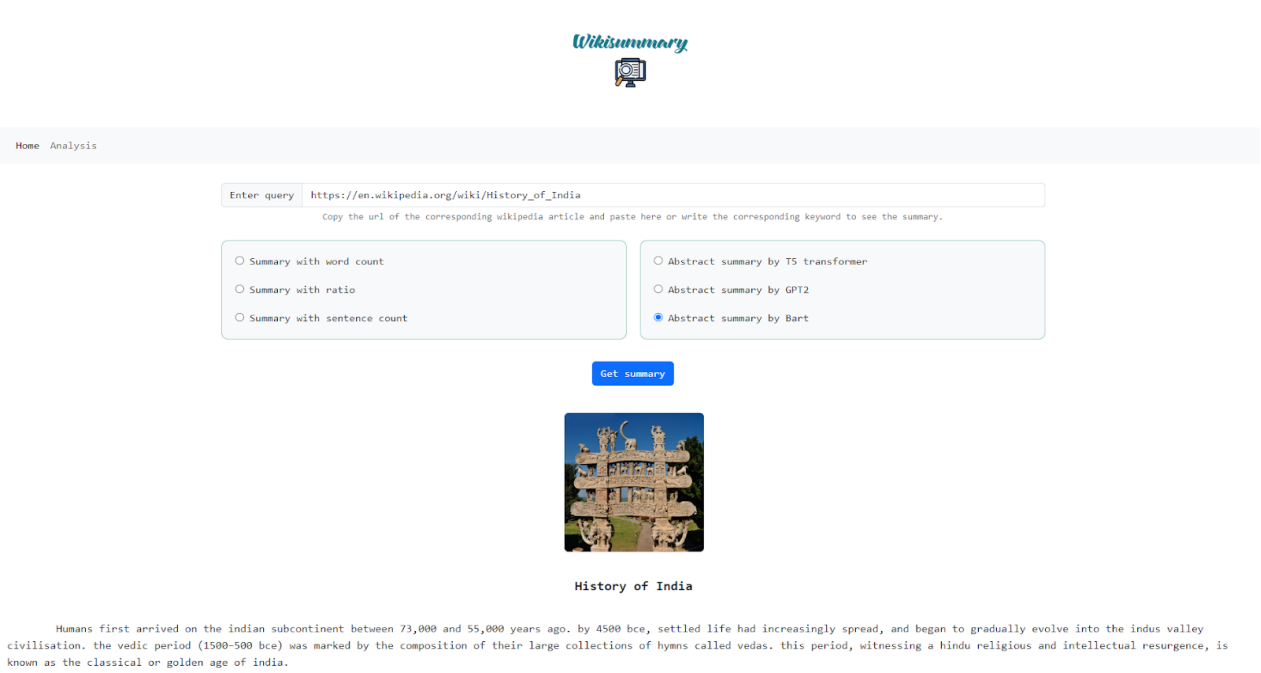


Fig.68: Summary by BART

We are representing the wordcloud for the most frequent words present in the Summary as shown in Fig.69.

Fig.69: Word Cloud of Summary by BART

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in the Fig.70.

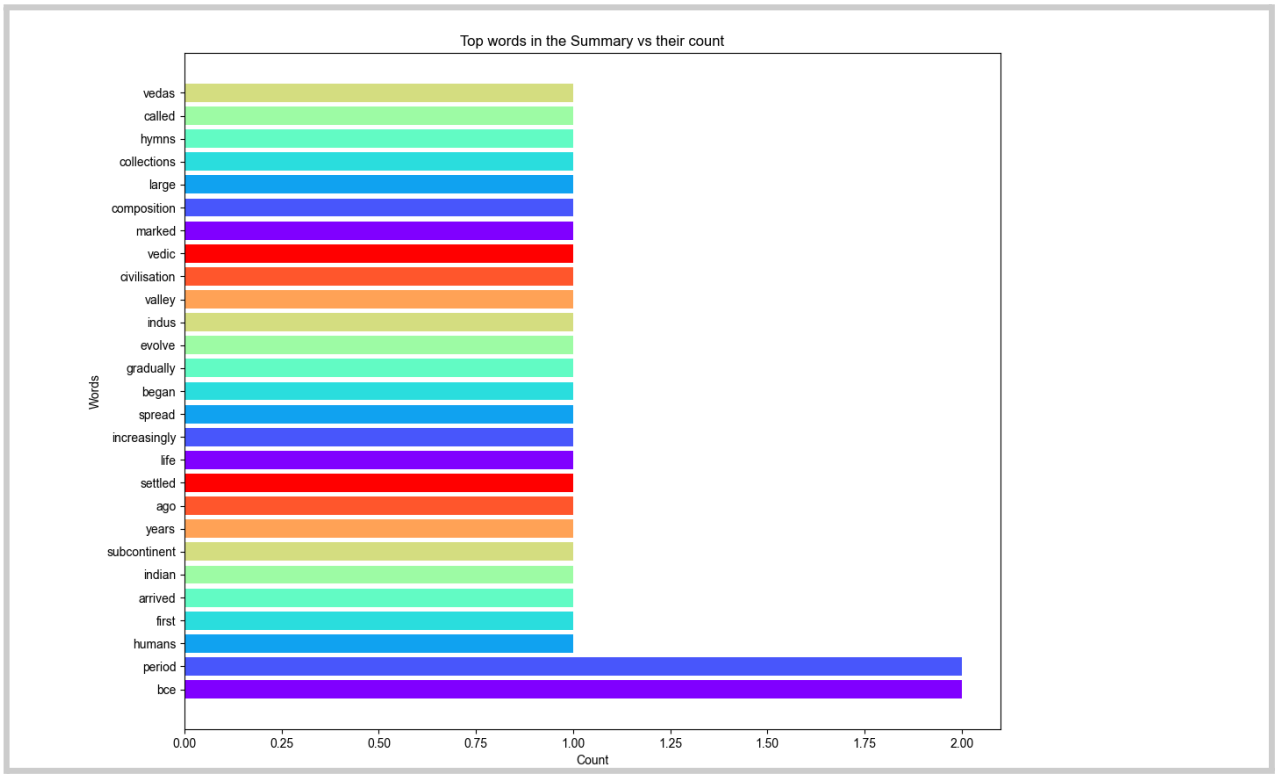


Fig.70: Most Common Word vs Count of Summary by BART

6.2.2 Experiment 2

Second experiment is done with TextRank, LSA, T5 Transformer, GPT2, BART model in a GUI application on WikiURL2 ”https://en.wikipedia.org/wiki/Kolkata” .

6.2.2.1 Experiemnt by TextRank

We have used the TextRank algorithm using Gensym Python Library for generating two types of extractive summary as shown in the output in the Fig.71 and Fig.72 ,

a) Summary by Ratio(ratio=0.1 percent),

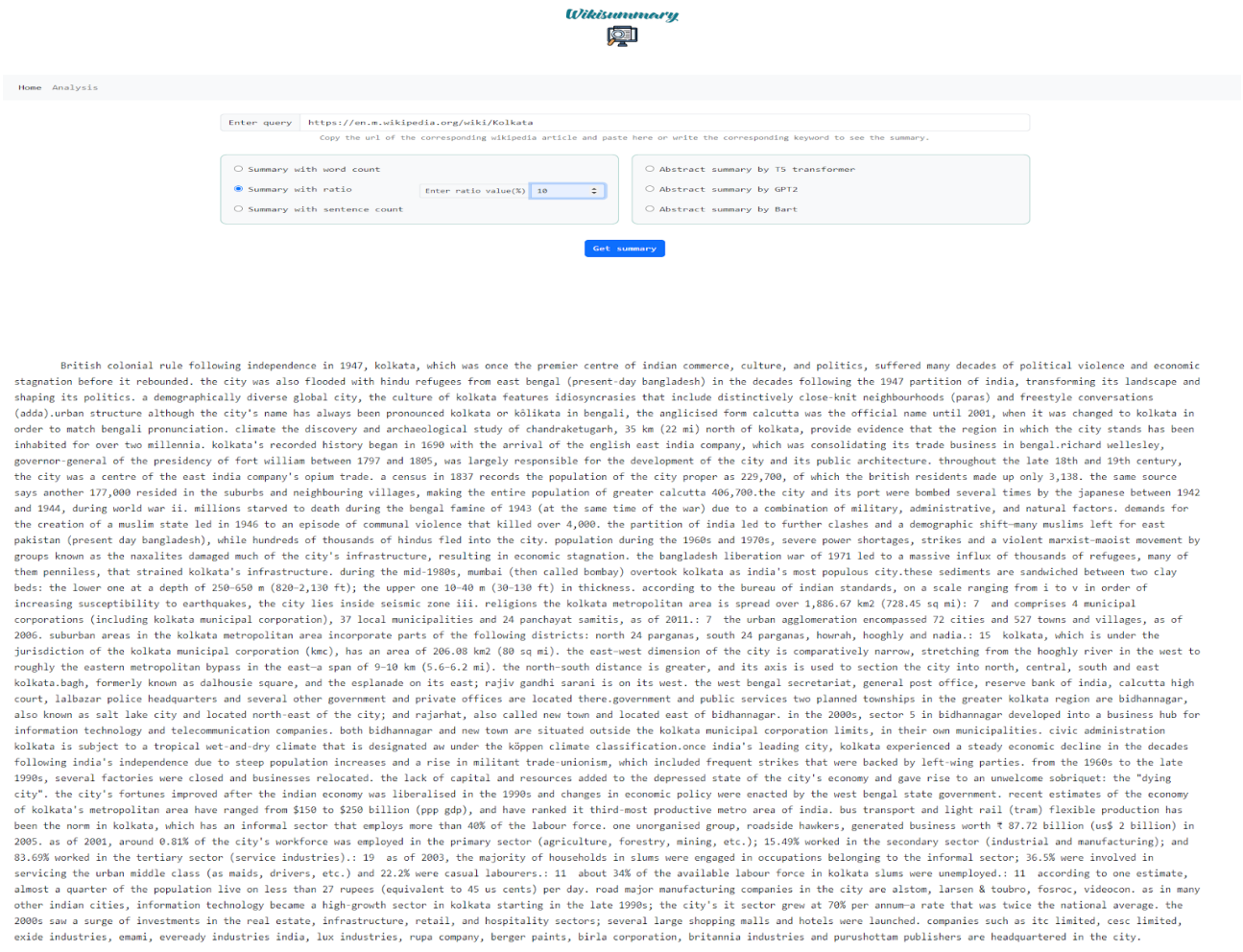


Fig.71: Summary by Ratio using TextRank

b) Summary by Word count(set word count=1000)

British colonial rule following independence in 1947, kolkata, which was once the premier centre of indian commerce, culture, and politics, suffered many decades of political violence and economic stagnation before it rebounded. the city was also flooded with hindu refugees from east bengal (present-day bangladesh) in the decades following the 1947 partition of india, transforming its landscape and shaping its politics. a demographically diverse global city, the culture of kolkata features idiosyncrasies that include distinctively close-knit neighbourhoods (paras) and freestyle conversations (adda).urban structure although the city's name has always been pronounced kolkata or kôlikata in bengali, the anglicised form calcutta was the official name until 2001, when it was changed to kolkata in order to match bengali pronunciation. climate the discovery and archaeological study of chandraketugarh, 35 km (22 mi) north of kolkata, provide evidence that the region in which the city stands has been inhabited for over two millennia. kolkata's recorded history began in 1690 with the arrival of the english east india company, which was consolidating its trade business in bengal.richard wellesley, governor-general of the presidency of fort william between 1797 and 1805, was largely responsible for the development of the city and its public architecture. throughout the late 18th and 19th century, the city was a centre of the east india company's opium trade. a census in 1837 records the population of the city proper as 229,700, of which the british residents made up only 3,138. the same source says another 177,000 resided in the suburbs and neighbouring villages, making the entire population of greater calcutta 406,700.these sediments are sandwiched between two clay beds: the lower one at a depth of 250–650 m (820–2,130 ft); the upper one 10–40 m (30–130 ft) in thickness. according to the bureau of indian standards, on a scale ranging from i to v in order of increasing susceptibility to earthquakes, the city lies inside seismic zone iii. religions the kolkata metropolitan area is spread over 1,886.67 km2 (728.45 sq mi): 7 and comprises 4 municipal corporations (including kolkata municipal corporation), 37 local municipalities and 24 panchayat samitis, as of 2011.: 7 the urban agglomeration encompassed 72 cities and 527 towns and villages, as of 2006. suburban areas in the kolkata metropolitan area incorporate parts of the following districts: north 24 parganas, south 24 parganas, howrah, hooghly and nadia.: 15 kolkata, which is under the jurisdiction of the kolkata municipal corporation (kmc), has an area of 206.08 km2 (80 sq mi). the east–west dimension of the city is comparatively narrow, stretching from the hooghly river in the west to roughly the eastern metropolitan bypass in the east–a span of 9–10 km (5.6–6.2 mi). the north–south distance is greater, and its axis is used to section the city into north, central, south and east kolkata.bagh, formerly known as dalhousie square, and the esplanade on its east; rajiv gandhi sarani is on its west. the west bengal secretariat, general post office, reserve bank of india, calcutta high court, lalbazar police headquarters and several other government and private offices are located there.government and public services two planned townships in the greater kolkata region are bidhannagar, also known as salt lake city and located north-east of the city; and rajarhat, also called new town and located east of bidhannagar. in the 2000s, sector 5 in bidhannagar developed into a business hub for information technology and telecommunication companies. both bidhannagar and new town are situated outside the kolkata municipal corporation limits, in their own municipalities. civic administration kolkata is subject to a tropical wet-and-dry climate that is designated aw under the köppen climate classification.once india's leading city, kolkata experienced a steady economic decline in the decades following india's independence due to steep population increases and a rise in militant trade-unionism, which included frequent strikes that were backed by left-wing parties. from the 1960s to the late 1990s, several factories were closed and businesses relocated. the lack of capital and resources added to the depressed state of the city's economy and gave rise to an unwelcome sobriquet: the "dying city". the city's fortunes improved after the indian economy was liberalised in the 1990s and changes in economic policy were enacted by the west bengal state government. recent estimates of the economy of kolkata's metropolitan area have ranged from $150 to $250 billion (ppp gdp), and have ranked it third-most productive metro area of india. bus transport and light rail (tram) flexible production has been the norm in kolkata, which has an informal sector that employs more than 40% of the labour force. one unorganised group, roadside hawkers, generated business worth ₹ 87.72 billion (us$ 2 billion) in 2005. as of 2001, around 0.81% of the city's workforce was employed in the primary sector (agriculture, forestry, mining, etc.); 15.49% worked in the secondary sector (industrial and manufacturing); and 83.69% worked in the tertiary sector (service industries).: 19 as of 2003, the majority of households in slums were engaged in occupations belonging to the informal sector; 36.5% were involved in servicing the urban middle class (as maids, drivers, etc.) and 22.2% were casual labourers.: 11 about 34% of the available labour force in kolkata slums were unemployed.: 11 according to one estimate, almost a quarter of the population live on less than 27 rupees (equivalent to 45 us cents) per day. road major manufacturing companies in the city are alstom, larsen & toubro, fosroc, videocon. as in many other indian cities, information technology became a high-growth sector in kolkata starting in the late 1990s; the city's it sector grew at 70% per annum–a rate that was twice the national average. the 2000s saw a surge of investments in the real estate, infrastructure, retail, and hospitality sectors; several large shopping malls and hotels were launched. companies such as itc limited, cesc limited, exide industries, emami, eveready industries india, lux industries, rupa company, berger paints, birla corporation, britannia industries and purushottam publishers are headquartered in the city.

Fig.72: Summary by Word count using TextRank

We are representing the wordcloud for the most frequent words present in the Summary by Ratio as shown in the Fig. 73 and Summary by Word count as shown in Fig. 74.

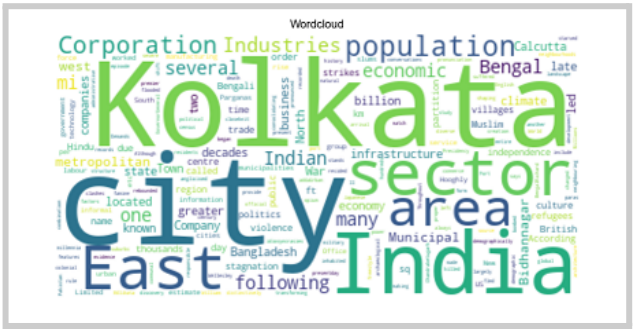


Fig.73: Word Cloud Summary by Ratio using TextRank

Fig.74: Word Cloud Summary by Word count using TextRank

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary by Ratio as shown in the Fig.75 and Summary by Word count as shown in the Fig.76.

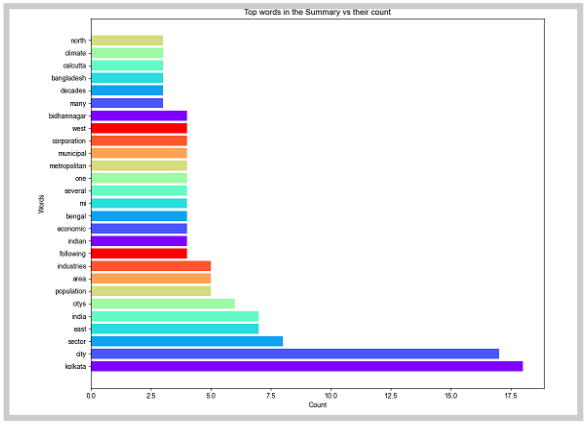


Fig.75: Most Common Word vs Count Summary by Ratio count using TextRank

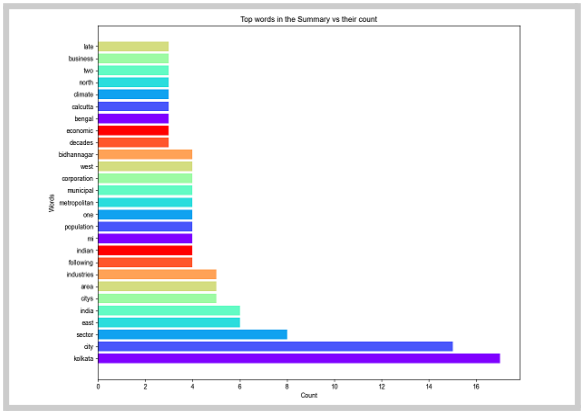


Fig.76: Most Common Word vs Count Summary by Word count using TextRank

6.2.2.2 Experiment by LSA

We have used the LSA algorithm using Sumy Python Library for generating extractive summary by sentences (no of sentences=50)as shown in the output in the Fig. 77

It is the primary business, commercial, and financial hub of eastern india and the main port of communication for north-east india.it is a part of kolkata metropolitan area (or known as greater kolkata) which has a population of over 1.41 crore (14.1 million) residents making it the third-most populous metropolitan area in india.in 2021, the kolkata metropolitan area crossed 1.5 crore (15 million) registered voters.it has the highest number of nobel laureates among all cities in india.british colonial rule following independence in 1947, kolkata, which was once the premier centre of indian commerce, culture, and politics, suffered many decades of political violence and economic stagnation before it rebounded. a demographically diverse global city, the culture of kolkata features idiosyncrasies that include distinctively close-knit neighbourhoods (paras) and freestyle conversations (adda). though home to major cricketing venues and franchises, kolkata stands out in india for being the country's centre of association football and also having strong culture in other sports less widespread elsewhere. kolkata's recorded history began in 1690 with the arrival of the english east india company, which was consolidating its trade business in bengal.these rights were transferred to the east india company in 1698. facing frequent skirmishes with french forces, the british began to upgrade their fortifications in 1756.the coalescence of british and indian culture resulted in the emergence of a new babu class of urbane indians, whose members were often bureaucrats, professionals, newspaper readers, and anglophiles; they usually belonged to upper-caste hindu communities. calcutta continued to be a centre for revolutionary organisations associated with the indian independence movement. during the mid-1980s, mumbai (then called bombay) overtook kolkata as india's most populous city.since 2000, the information technology (it) services sector has revitalised kolkata's stagnant economy.the city is also experiencing marked growth in its manufacturing base. bengal basin comprises three structural units: shelf or platform in the west; central hinge or shelf/slope break; and deep basinal part in the east and southeast. the shelf and hinge zones have many faults, among them some are active.these sediments are sandwiched between two clay beds: the lower one at a depth of 250–650 m (820–2,130 ft); the upper one 10–40 m (30–130 ft) in thickness. religions the kolkata metropolitan area is spread over 1,886.67 km2 (728.45 sq mi): 7 and comprises 4 municipal corporations (including kolkata municipal corporation), 37 local municipalities and 24 panchayat samitis, as of 2011.characterised by 19th-century architecture and narrow alleyways, it includes areas such as jorasanko, rajabazar, maniktala, ultadanga, shyambazar, shobhabazar, bagbazar, cossipore, sinthee etc.are also within the city of kolkata (as a metropolitan structure). the west bengal secretariat, general post office, reserve bank of india, calcutta high court, lalbazar police headquarters and several other government and private offices are located there. south kolkata developed after india gained independence in 1947; it includes upscale neighbourhoods such as bhawanipore, alipore, ballygunge, kasba, dhakuria, santoshpur, garia, golf green, tollygunge, new alipore, behala, barisha etc.are also within the city of kolkata (as a metropolitan structure). both bidhannagar and new town are situated outside the kolkata municipal corporation limits, in their own municipalities.according to a united nations development programme report, its wind and cyclone zone is "very high damage risk".summers (march–june) are hot and humid, with temperatures in the low 30s celsius; during dry spells, maximum temperatures sometime exceed 40 °c (104 °f) in may and june. winter lasts for roughly 2+1⁄2 months, with seasonal lows dipping to 9–11 °c (48–52 °f) in december and january.often, in april–june, the city is struck by heavy rains or dusty squalls that are followed by thunderstorms or hailstorms, bringing cooling relief from the prevailing humidity.the highest monthly rainfall total occurs in july and august.the city receives 2,107 hours of sunshine per year, with maximum sunlight exposure occurring in april. kolkata has been hit by several cyclones; these include systems occurring in 1737 and 1864 that killed thousands.as of 2008, sulphur dioxide and nitrogen dioxide annual concentration were within the national ambient air quality standards of india, but respirable suspended particulate matter levels were high, and on an increasing trend for five consecutive years, causing smog and haze.once india's leading city, kolkata experienced a steady economic decline in the decades following india's independence due to steep population increases and a rise in militant trade-unionism, which included frequent strikes that were backed by left-wing parties. bus transport and light rail (tram) flexible production has been the norm in kolkata, which has an informal sector that employs more than 40% of the labour force.: 11 about 34% of the available labour force in kolkata slums were unemployed. road major manufacturing companies in the city are alstom, larsen & toubro, fosroc, videocon. the 2000s saw a surge of investments in the real estate, infrastructure, retail, and hospitality sectors; several large shopping malls and hotels were launched.air the demonym for residents of kolkata are calcuttan and kolkatan.: 4 : 92 the authorised slums (with access to basic services like water, latrines, trash removal by the kolkata municipal corporation) can be broadly divided into two groups–bustees, in which slum dwellers have some long term tenancy agreement with the landowners; and udbastu colonies, settlements which had been leased to refugees from present-day bangladesh by the government.: 5 the unauthorised slums (devoid of basic services provided by the municipality) are occupied by squatters who started living on encroached lands–mainly along canals, railway lines and roads.: 23 mother teresa was awarded the nobel peace prize for founding and working with the missionaries of charity in kolkata–an organisation "whose primary task was to love and care for those persons nobody was prepared to look after". bengali hindus form the majority of kolkata's population; marwaris, biharis and urdu-speaking muslims compose large minorities. among kolkata's smaller communities are chinese, tamils, nepalis, pathans/afghans (locally known as kabuliwala) odias, telugus, gujaratis, anglo-indians, armenians, bengali muslims, greeks, tibetans, maharashtrians, konkanis, malayalees, punjabis and parsis. india's sole chinatown is in eastern kolkata; once home to 20,000 ethnic chinese, its population dropped to around 2,000 as of 2009 as a result of multiple factors including repatriation and denial of indian citizenship following the 1962 sino-indian war, and immigration to foreign countries for better economic opportunities. kolkata reported 67.6% of special and local laws crimes registered in 35 large indian cities during 2004. media kolkata's administrative agencies have areas of jurisdiction that do not coincide. further reading state-owned bharat sanchar nigam limited, or bsnl, as well as private enterprises, among them vodafone idea, bharti airtel, reliance jio are the leading telephone and cell phone service providers in the city.: 25–26 : 179 with kolkata being the first city in india to have cell phone and 4g connectivity, the gsm and cdma cellular coverage is extensive. as of 2010, kolkata has 7 percent of the total broadband internet consumers in india; bsnl, vsnl, tata indicom, sify, hathway, airtel, and jio are among the main vendors.

Fig.77: Summary by Sentences using LSA

We are representing the Word Cloud for the most frequent words present in the Summary by Sentences as shown in Fig. 78.

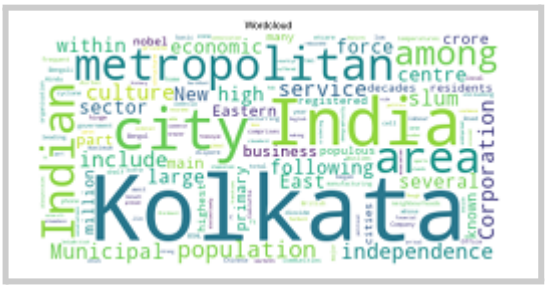


Fig.78: Word Cloud Summary by Sentences using LSA

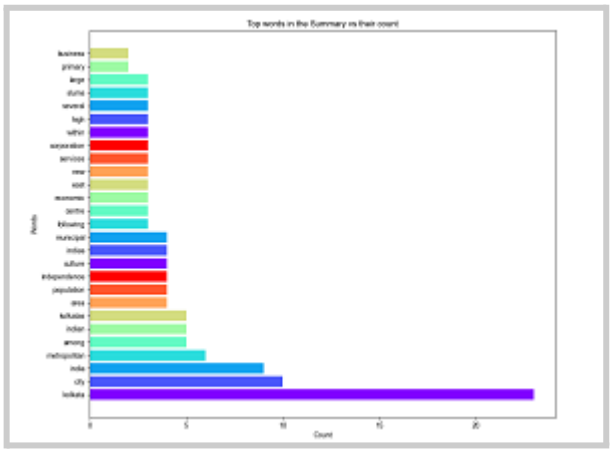


Fig.79: Most Common Word vs Count of Summary by LSA

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary by Sentences as shown in Fig. 79.

6.2.2.3 Experiment by T5 Transformer

We have used the T5 (Text-to-Text Transformer) model using torch, transformers, AutoTokenizer, AutoModelForSeq2SeqLM Python Libraries for generating abstractive summary as shown in the output in Fig. 80.

Kolkata

Kolkata is one of india's most populous cities, with a population of more than 1.5 million, according to the latest official figures. the bbc looks at the history of the city and how it is based on its historical origins and why it doesn't always be associated with its cultural and cultural heritage. these are some of

Fig.80: Summary by T5 Transformer

We are representing the Word Cloud for the most frequent words present in the Summary as shown in Fig. 81.

Fig.81: Word Cloud of Summary by T5 Transformer

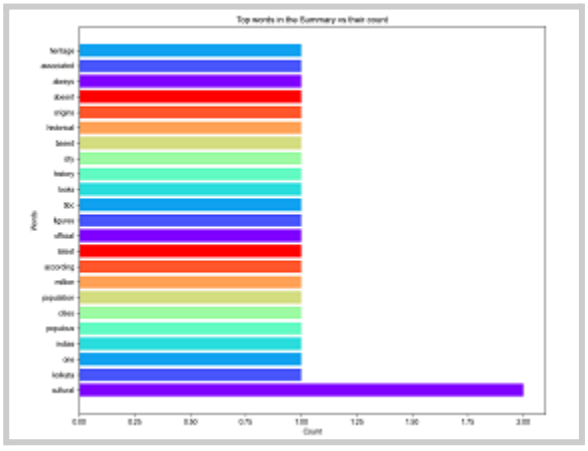


Fig.82: Most Common Word vs Count of Summary by T5 Transformer

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in the Fig. 82.

6.2.2.4 Experiment by GPT2 model

We have used the GPT2 model using torch, transformers, GPT2Tokenizer, GPT2LMHeadModel Python Libraries for generating abstractive summary as shown in the output in Fig. 83.

Kolkata

etymology kolkata (uk: /kɒlˈkɑːtə/ or /kɒlˈkʌtə/, us: /koʊlˈkɑːtɑː/, bengali: (listen); also known as calcutta /kælˈkʌtə/ which was the official name until 2001) is the capital of the indian state of west bengal. it is on the eastern bank of the hooghly river 80 km (50 mi) west of the border with bangladesh. it is the primary business, commercial, and financial hub of eastern india and the main port of communication for north-east india. according to the 2011 indian census, kolkata is the seventh-most populous city in india, with a population of 45 lakh (4.5 million) residents within the city. it is a part of kolkata metropolitan area (or known as greater kolkata) which has a population of over 1.41 crore (14.1 million) residents making it the third-most populous metropolitan area in india. in 2021, the kolkata metropolitan area crossed 1.5 crore (15 million) registered voters. the port of kolkata is india's oldest operating port and its sole major riverine port. kolkata is regarded as the cultural capital of india. kolkata is the second largest bengali-speaking city after dhaka. it has the highest number of nobel laureates among all cities in india. history in the late 17th century, the three villages that predated calcutta were ruled by the nawab of bengal under mughal suzerainty. after the nawab granted the east india company a trading licence in 1690, the area was developed by the company into an increasingly fortified trading post known as fort william. nawab siraj ud-daulah occupied calcutta in 1756, and the east india company retook it the following year. in 1793 the east india company was strong enough to abolish native rule, and assumed full sovereignty of the region. under company rule and later under the british raj, calcutta served as the capital of british-held territories in india until 1911. in that year, after assessing its geographical location, combined with growing nationalism in bengal (calcutta became the centre for the indian independence movement), the british moved the capital to the relatively more centrally located new delhi. british colonial rule following independence in 1947, kolkata, which was once the premier centre of indian commerce, culture, and politics, suffered many decades of political violence and economic stagnation before it rebounded. the city was also flooded with hindu refugees from east bengal (present-day bangladesh) in the decades following the 1947 partition of india, transforming its landscape and shaping its politics. a demographically diverse global city, the culture of kolkata features idiosyncrasies that include distinctively close-knit neighbourhoods (paras) and freestyle conversations (adda). kolkata is home to eastern india's film industry, known as tollywood, and cultural institutions, such as the academy of fine arts, the victoria memorial, the asiatic society, the indian museum, and the national library of india. among scientific institutions, kolkata hosts the agri horticultural society of india, the geological survey of india, the botanical survey of india, the calcutta mathematical society, the indian science congress association, the zoological survey of india, the institution of engineers, the anthropological survey of india and the indian public health association. four nobel laureates and two nobel memorial prize winners are associated with the city. though home to major cricketing venues and franchises, kolkata stands out in india for being the country's centre of association football and also having strong culture in other sports less widespread elsewhere. hence, kolkata is also known as 'the city of joy'. contemporary the word kolkata (bengali: কলকাতা) derives from kōlikata (bengali: কলিকাতা), the bengali language name of one of three villages that predated the arrival of the british, the other two villages were sutanuti and govindapur. geography there are several explanations for the etymology of this name: urban structure although the city's name has always been pronounced kolkata or kōlikata in bengali, the anglicised form calcutta was the official name until 2001

Fig.83: Summary by GPT2

We are representing the wordcloud for the most frequent words present in the Summary as shown in the Fig. 84

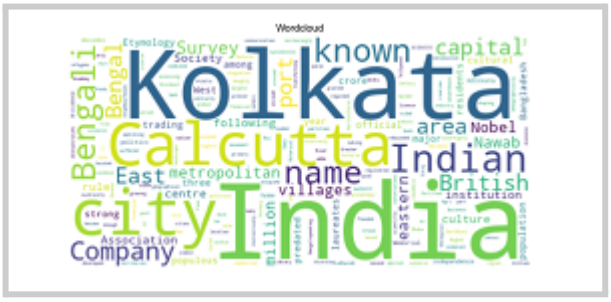


Fig.84: Word Cloud of Summary by GPT2

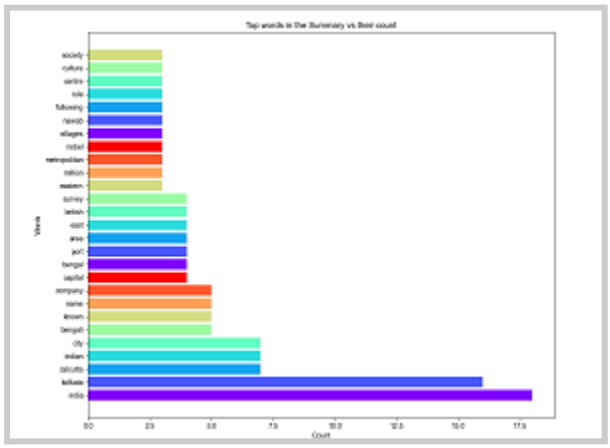
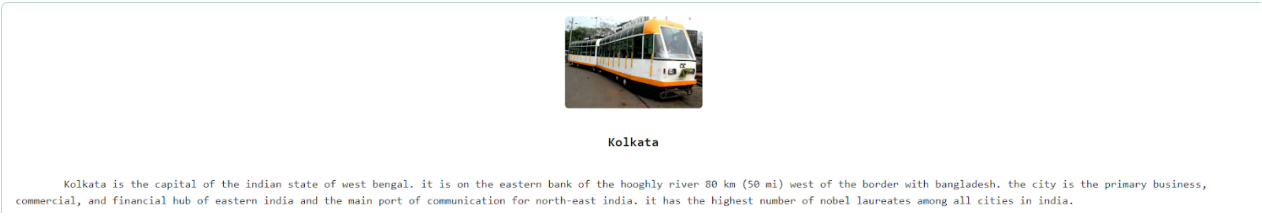


Fig.85: Most Common Word vs Count of Summary by GPT2

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in Fig. 85.

6.2.2.5 Experiment by BART model

We have used the BART model using BartForConditionalGeneration, BartTokenizer, BartConfig Python Library for generating abstractive summary as shown in the output in the Fig.86.



Kolkata

Kolkata is the capital of the indian state of west bengal. it is on the eastern bank of the hooghly river 80 km (50 mi) west of the border with bangladesh. the city is the primary business, commercial, and financial hub of eastern india and the main port of communication for north-east india. it has the highest number of nobel laureates among all cities in india.

Fig.86: Summary by BART

We are representing the wordcloud for the most frequent words present in the Summary as shown in Fig.87.

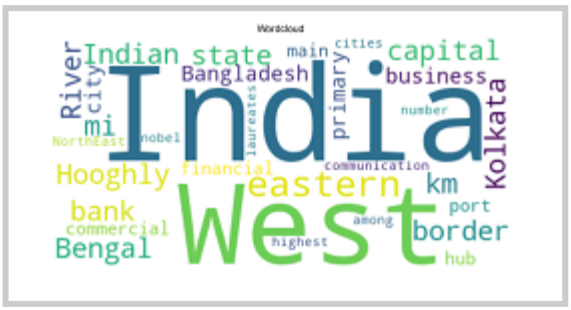


Fig.87 : Word Cloud of Summary by BART

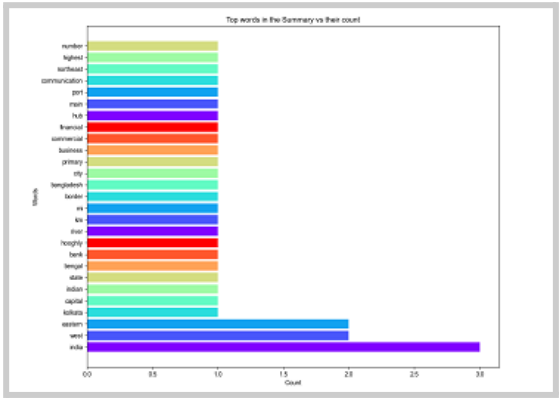


Fig.88: Most Common Word vs Count of Summary by BART

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in the Fig.88.

6.2.3 Experiment 3

Second experiment is done with TextRank, LSA, T5 Transformer, GPT2, BART model in a GUI application on WikiURL3 ”https://en.wikipedia.org/wiki/Rabindranath_Tagore”

6.2.3.1 Experiemnt by TextRank

We have used the TextRank algorithm using Gensym Python Library for generating two types of extractive summary as shown in the output in the Fig.89 and Fig.90 ,

a) Summary by Ratio(ratio=0.1 percent),

Rabindranath Tagore

Referred to as "the bard of bengal", tagore was known by sobriquets: gurudeb, kobiguru, biswokobi. life and events a bengali brahmin from calcutta with ancestral gentry roots in burdwan district and jessore, tagore wrote poetry as an eight-year-old. at the age of sixteen, he released his first substantial poems under the pseudonym bhānusiṁha ("sun lion"), which were seized upon by literary authorities as long-lost classics. by 1877 he graduated to his first short stories and dramas, published under his real name.as an exponent of the bengal renaissance, he advanced a vast canon that comprised paintings, sketches and doodles, hundreds of texts, and some two thousand songs; his legacy also endures in his founding of visva-bharati university. early life: 1861–1878 tagore modernised bengali art by spurning rigid classical forms and resisting linguistic strictures.works the youngest of 13 surviving children, tagore (nicknamed "rabi") was born on 7 may 1861 in the jorasanko mansion in calcutta, the son of debendranath tagore (1817–1905) and sarada devi (1830–1875). drama tagore was raised mostly by servants; his mother had died in his early childhood and his father travelled widely. the tagore family was at the forefront of the bengal renaissance.her abrupt suicide in 1884, soon after he married, left him profoundly distraught for years. short stories tagore largely avoided classroom schooling and preferred to roam the manor or nearby bolpur and panihati, which the family visited. his brother hemendranath tutored and physically conditioned him–by having him swim the ganges or trek through hills, by gymnastics, and by practising judo and wrestling.my father, seated amidst the throng of worshippers, would sometimes add his voice to the hymn of praise, and finding a stranger joining in their devotions they would wax enthusiastically cordial, and we would return loaded with the sanctified offerings of sugar crystals and other sweets. songs (rabindra sangeet) he wrote 6 poems relating to sikhism and a number of articles in bengali children's magazine about sikhism. tagore returned to jorosanko and completed a set of major works by 1877, one of them a long poem in the maithili style of vidyapati.art works because debendranath wanted his son to become a barrister, tagore enrolled at a public school in brighton, east sussex, england in 1878. he stayed for several months at a house that the tagore family owned near brighton and hove, in medina villas; in 1877 his nephew and niece–suren and indira devi, the children of tagore's brother satyendranath–were sent together with their mother, tagore's sister-in-law, to live with him. he briefly read law at university college london, but again left school, opting instead for independent study of shakespeare's plays coriolanus, and antony and cleopatra and the religio medici of thomas browne.london's india society published the work in a limited edition, and the american magazine poetry published a selection from gitanjali. in november 1913, tagore learned he had won that year's nobel prize in literature: the swedish academy appreciated the idealistic–and for westerners–accessible nature of a small body of his translated material focused on the 1912 gitanjali: song offerings. he was awarded a knighthood by king george v in the 1915 birthday honours, but tagore renounced it after the 1919 jallianwala bagh massacre. renouncing the knighthood, tagore wrote in a letter addressed to lord chelmsford, the then british viceroy of india, "the disproportionate severity of the punishments inflicted upon the unfortunate people and the methods of carrying them out, we are convinced, are without parallel in the history of civilised governments...the time has come when badges of honour make our shame glaring in their incongruous context of humiliation, and i for my part wish to stand, shorn of all special distinctions, by the side of my country men." theft of nobel prize in 1919, he was invited by the president and chairman of anjuman-e-islamia, syed abdul majid to visit sylhet for the first time.in november 1912 tagore began touring the united states and the united kingdom, staying in butterton, staffordshire with andrews's clergymen friends. from may 1916 until april 1917, he lectured in japan and the united states. he denounced nationalism. his essay "nationalism in india" was scorned and praised; it was admired by romain rolland and other pacifists. primary shortly after returning home the 63-year-old tagore accepted an invitation from the peruvian government.he wrote in 1932, while on a visit to iran, that "each country of asia will solve its own historical problems according to its strength, nature and needs, but the lamp they will each carry on their path to progress will converge to illuminate the common ray of knowledge." further reading known mostly for his poetry, tagore wrote novels, essays, short stories, travelogues, dramas, and thousands of songs.

Fig.89: Summary by Ratio using TextRank

b) Summary by Word count(set word count=1000)

Rabindranath Tagore

Referred to as "the bard of bengal", tagore was known by sobriquets: gurudeb, kobiguru, biswokobi. life and events a bengali brahmin from calcutta with ancestral gentry roots in burdwan district and jessore, tagore wrote poetry as an eight-year-old. at the age of sixteen, he released his first substantial poems under the pseudonym bhānusiṁha ("sun lion"), which were seized upon by literary authorities as long-lost classics. by 1877 he graduated to his first short stories and dramas, published under his real name.as an exponent of the bengal renaissance, he advanced a vast canon that comprised paintings, sketches and doodles, hundreds of texts, and some two thousand songs; his legacy also endures in his founding of visva-bharati university. early life: 1861–1878 tagore modernised bengali art by spurning rigid classical forms and resisting linguistic strictures.gitanjali (song offerings), gora (fair-faced) and ghare-baire (the home and the world) are his best-known works, and his verse, short stories, and novels were acclaimed–or panned–for their lyricism, colloquialism, naturalism, and unnatural contemplation.works the youngest of 13 surviving children, tagore (nicknamed "rabi") was born on 7 may 1861 in the jorasanko mansion in calcutta, the son of debendranath tagore (1817–1905) and sarada devi (1830–1875). drama tagore was raised mostly by servants; his mother had died in his early childhood and his father travelled widely. the tagore family was at the forefront of the bengal renaissance.her abrupt suicide in 1884, soon after he married, left him profoundly distraught for years. short stories tagore largely avoided classroom schooling and preferred to roam the manor or nearby bolpur and panihati, which the family visited. his brother hemendranath tutored and physically conditioned him–by having him swim the ganges or trek through hills, by gymnastics, and by practising judo and wrestling.there tagore read biographies, studied history, astronomy, modern science, and sanskrit, and examined the classical poetry of kālidāsa. during his 1-month stay at amritsar in 1873 he was greatly influenced by melodious gurbani and nanak bani being sung at golden temple for which both father and son were regular visitors.my father, seated amidst the throng of worshippers, would sometimes add his voice to the hymn of praise, and finding a stranger joining in their devotions they would wax enthusiastically cordial, and we would return loaded with the sanctified offerings of sugar crystals and other sweets. songs (rabindra sangeet) he wrote 6 poems relating to sikhism and a number of articles in bengali children's magazine about sikhism. tagore returned to jorosanko and completed a set of major works by 1877, one of them a long poem in the maithili style of vidyapati.art works because debendranath wanted his son to become a barrister, tagore enrolled at a public school in brighton, east sussex, england in 1878. he stayed for several months at a house that the tagore family owned near brighton and hove, in medina villas; in 1877 his nephew and niece–suren and indira devi, the children of tagore's brother satyendranath–were sent together with their mother, tagore's sister-in-law, to live with him. he briefly read law at university college london, but again left school, opting instead for independent study of shakespeare's plays coriolanus, and antony and cleopatra and the religio medici of thomas browne.lively english, irish, and scottish folk tunes impressed tagore, whose own tradition of nidhubabu-authored kirtans and tappas and brahmo hymnody was subdued. in 1880 he returned to bengal degree-less, resolving to reconcile european novelty with brahmo traditions, taking the best from each. after returning to bengal, tagore regularly published poems, stories, and novels.the period 1891–1895, tagore's sadhana period, named after one of his magazines, was his most productive; in these years he wrote more than half the stories of the three-volume, 84-story galpaguchchha. its ironic and grave tales examined the voluptuous poverty of an idealised rural bengal. repudiation of knighthood in 1901 tagore moved to santiniketan to found an ashram with a marble-floored prayer hall–the mandir–an experimental school, groves of trees, gardens, a library. there his wife and two of his children died.london's india society published the work in a limited edition, and the american magazine poetry published a selection from gitanjali. in november 1913, tagore learned he had won that year's nobel prize in literature: the swedish academy appreciated the idealistic–and for westerners–accessible nature of a small body of his translated material focused on the 1912 gitanjali: song offerings. he was awarded a knighthood by king george v in the 1915 birthday honours, but tagore renounced it after the 1919 jallianwala bagh massacre. renouncing the knighthood, tagore wrote in a letter addressed to lord chelmsford, the then british viceroy of india, "the disproportionate severity of the punishments inflicted upon the unfortunate people and the methods of carrying them out, we are convinced, are without parallel in the history of civilised governments...the time has come when badges of honour make our shame glaring in their incongruous context of humiliation, and i for my part wish to stand, shorn of all special distinctions, by the side of my country men." theft of nobel prize in 1919, he was invited by the president and chairman of anjuman-e-islamia, syed abdul majid to visit sylhet for the first time.in november 1912 tagore began touring the united states and the united kingdom, staying in butterton, staffordshire with andrews's clergymen friends. from may 1916 until april 1917, he lectured in japan and the united states. he denounced nationalism. his essay "nationalism in india" was scorned and praised; it was admired by romain rolland and other pacifists. primary shortly after returning home the 63-year-old tagore accepted an invitation from the peruvian government.he wrote in 1932, while on a visit to iran, that "each country of asia will solve its own historical problems according to its strength, nature and needs, but the lamp they will each carry on their path to progress will converge to illuminate the common ray of knowledge." further reading known mostly for his poetry, tagore wrote novels, essays, short stories, travelogues, dramas, and thousands of songs.

Fig.90: Summary by Word count using TextRank

We are representing the wordcloud for the most frequent words present in the Summary by Ratio as shown in Fig. 91 and Summary by Word count as shown in the Fig. 92.

Fig.91: Word Cloud Summary by Ratio using TextRank

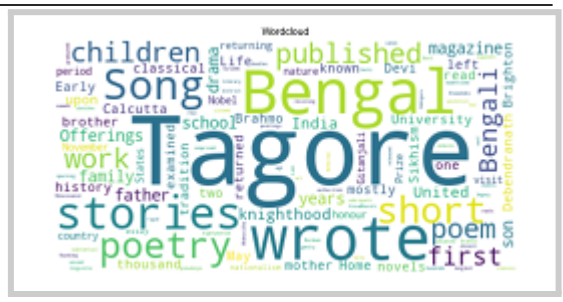


Fig.92: Word Cloud Summary by Word count using TextRank

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary by Ratio as shown in the Fig.93 and Summary by Word count as shown in the Fig.94.

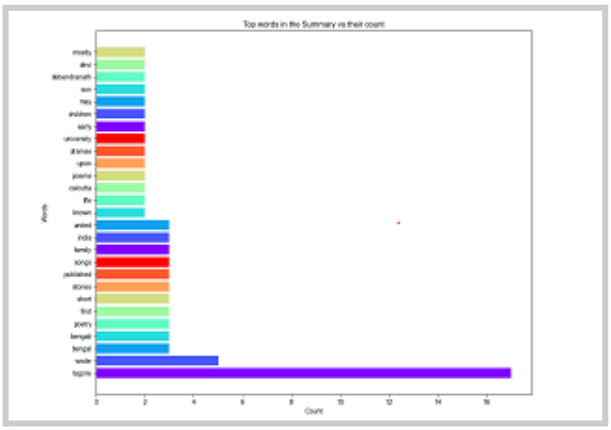


Fig.93: Most Common Word vs Count Summary by Ratio using TextRank

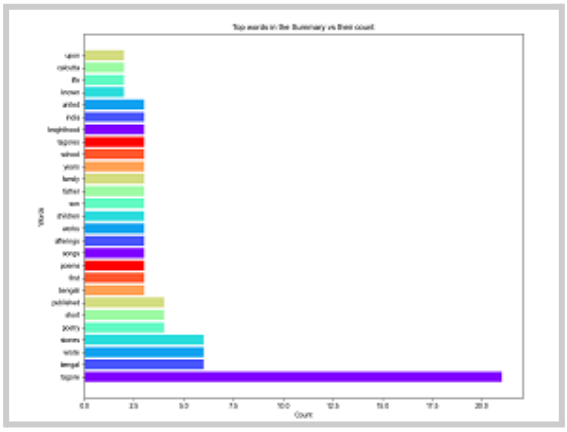


Fig.94: Most Common Word vs Count Summary by Word count using TextRank

6.2.3.2 Experiment by LSA

We have used the LSA algorithm using Sumy Python Library for generating extractive summary by sentences (no of sentences=50)as shown in the output in the Fig. 95

Rabindranath Tagore

Family history rabindranath tagore fras (/rəˈbɪndrəno:t təˈgɔ:r/ (listen); pronounced ; 7 may 1861 – 7 august 1941) was a bengali poet, writer, playwright, composer, philosopher, social reformer and painter.referred to as "the bard of bengal", tagore was known by sobriquets: gurudeb, kobiguru, biswokobi. at the age of sixteen, he released his first substantial poems under the pseudonym bhānusiṃha ("sun lion"), which were seized upon by literary authorities as long-lost classics. early life: 1861–1878 tagore modernised bengali art by spurning rigid classical forms and resisting linguistic strictures.his compositions were chosen by two nations as national anthems: india's "jana gana mana" and bangladesh's "amar shonar bangla".they were pirali brahmin ('pirali' historically carried a stigmatized and pejorative connotation) who originally belonged to a village named kush in the district named burdwan in west bengal.yet another brother, jyotirindranath, was a musician, composer, and playwright. jyotirindranath's wife kadambari devi, slightly older than tagore, was a dear friend and powerful influence.her abrupt suicide in 1884, soon after he married, left him profoundly distraught for years. tagore loathed formal education–his scholarly travails at the local presidency college spanned a single day.years later he held that proper teaching does not explain things; proper teaching stokes curiosity: novels after his upanayan (coming-of-age rite) at age eleven, tagore and his father left calcutta in february 1873 to tour india for several months, visiting his father's santiniketan estate and amritsar before reaching the himalayan hill station of dalhousie. during his 1-month stay at amritsar in 1873 he was greatly influenced by melodious gurbani and nanak bani being sung at golden temple for which both father and son were regular visitors.he mentions about this in his my reminiscences (1912) poetry the golden temple of amritsar comes back to me like a dream.my father, seated amidst the throng of worshippers, would sometimes add his voice to the hymn of praise, and finding a stranger joining in their devotions they would wax enthusiastically cordial, and we would return loaded with the sanctified offerings of sugar crystals and other sweets.as a joke, he claimed that these were the lost works of newly discovered 17th-century vaiṣṇava poet bhānusiṃha. he stayed for several months at a house that the tagore family owned near brighton and hove, in medina villas; in 1877 his nephew and niece–suren and indira devi, the children of tagore's brother satyendranath–were sent together with their mother, tagore's sister-in-law, to live with him. after returning to bengal, tagore regularly published poems, stories, and novels.these had a profound impact within bengal itself but received little national attention. in 1883 he married 10-year-old mrinalini devi, born bhabatarini, 1873–1902 (this was a common practice at the time). its ironic and grave tales examined the voluptuous poverty of an idealised rural bengal. repudiation of knighthood in 1901 tagore moved to santiniketan to found an ashram with a marble-floored prayer hall–the mandir–an experimental school, groves of trees, gardens, a library.santiniketan and visva-bharati in 1912, tagore translated his 1910 work gitanjali into english.while on a trip to london, he shared these poems with admirers including william butler yeats and ezra pound." theft of nobel prize in 1919, he was invited by the president and chairman of anjuman-e-islamia, syed abdul majid to visit sylhet for the first time. impact and legacy in 1921, tagore and agricultural economist leonard elmhirst set up the "institute for rural reconstruction", later renamed shriniketan or "abode of welfare", in surul, a village near the ashram.with it, tagore sought to moderate gandhi's swaraj protests, which he occasionally blamed for british india's perceived mental – and thus ultimately colonial – decline. in the early 1930s he targeted ambient "abnormal caste consciousness" and untouchability.during a may 1932 visit to a bedouin encampment in the iraqi desert, the tribal chief told him that "our prophet has said that a true muslim is he by whose words and deeds not the least of his brother-men may ever come to any harm ..." tagore confided in his diary: "i was startled into recognizing in his words the voice of essential humanity. fifteen new volumes appeared, among them prose-poem works punashcha (1932), shes saptak (1935), and patraput (1936).his last five years were marked by chronic pain and two long periods of illness.these began when tagore lost consciousness in late 1937; he remained comatose and near death for a time.this was followed in late 1940 by a similar spell, from which he never recovered. a period of prolonged agony ended with tagore's death on 7 august 1941, aged 80.i want my friends, their touch, with the earth's last love. bibliography between 1878 and 1932, tagore set foot in more than thirty countries on five continents. in 1912, he took a sheaf of his translated works to england, where they gained attention from missionary and gandhi protégé charles f. andrews, irish poet william butler yeats, ezra pound, robert bridges, ernest rhys, thomas sturge moore, and others. from may 1916 until april 1917, he lectured in japan and the united states. their warm rapport ended when tagore pronounced upon il duce's fascist finesse. secondary on 1 november 1926 tagore arrived at hungary and spent some time on the shore of lake balaton in the city of balatonfüred, recovering from heart problems at a sanitarium. texts on 14 july 1927 tagore and two companions began a four-month tour of southeast asia.they visited bali, java, kuala lumpur, malacca, penang, siam, and singapore. in early 1930 he left bengal for a nearly year-long tour of europe and the united states.upon returning to britain–and as his paintings were exhibited in paris and london–he lodged at a birmingham quaker settlement.he wrote his oxford hibbert lectures and spoke at the annual london quaker meet. he visited aga khan iii, stayed at dartington hall, toured denmark, switzerland, and germany from june to mid-september 1930, then went on into the soviet union. in his other travels, tagore interacted with henri bergson, albert einstein, robert frost, thomas mann, george bernard shaw, h.g. vice-president of india m. hamid ansari has said that rabindranath tagore heralded the cultural rapprochement between communities, societies and nations much before it became the liberal norm of conduct." further reading known mostly for his poetry, tagore wrote novels, essays, short stories, travelogues, dramas, and thousands of songs.his works are frequently noted for their rhythmic, optimistic, and lyrical nature.this includes all versions of each work and fills about eighty volumes.

Fig.95: Summary by Sentences using LSA

We are representing the Word Cloud for the most frequent words present in the Summary by Sentences as shown in Fig. 96.

Fig.96: Word Cloud Summary by Sentences using LSA

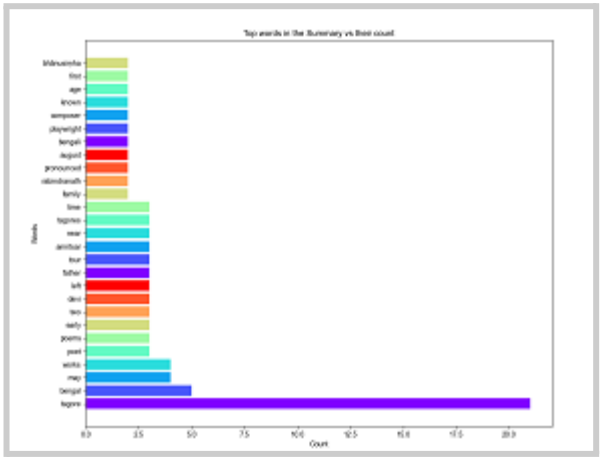
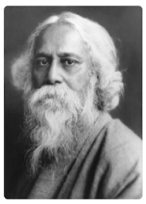


Fig.97: Most Common Word vs Count of Summary by LSA

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary by Sentences as shown in Fig. 97.

6.2.3.3 Experiment by T5 Transformer

We have used the T5 (Text-to-Text Transformer) model using torch, transformers, AutoTokenizer, AutoModelForSeq2SeqLM Python Libraries for generating abstractive summary as shown in the output in Fig. 98.



Rabindranath Tagore

Tributes have been paid to the legendary indian poet and poet rabi□□ta tagore, who has died aged 99. the bbc's geeta pandey looks at his legacy and his life. following is a timeline of his personal stories and memories of the bard of bengal, which has been published in the uk and india.

Fig.98: Summary by T5 Transformer

We are representing the Word Cloud for the most frequent words present in the Summary as shown in Fig. 99.

Fig.99: Word Cloud of Summary by T5 Transformer

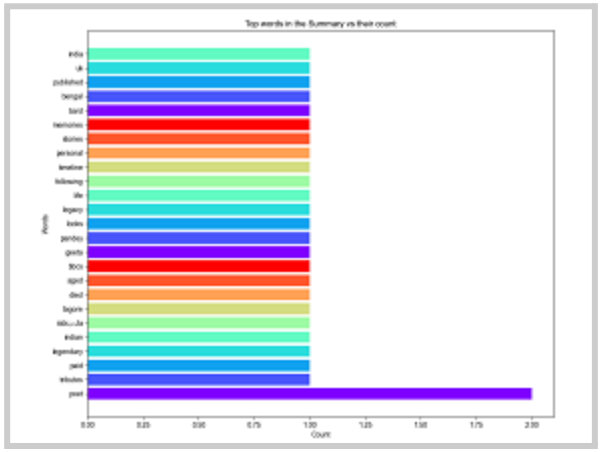
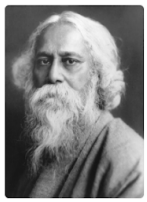


Fig.100: Most Common Word vs Count of Summary by T5 Transformer

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in the Fig. 100.

6.2.3.4 Experiment by GPT2 model

We have used the GPT2 model using torch, transformers, GPT2Tokenizer, GPT2LMHeadModel Python Libraries for generating abstractive summary as shown in the output in Fig. 101.



Rabindranath Tagore

family history rabindranath tagore fras (/rəˈbɪndrənɑːt təˈgɔːr/ (listen); pronounced ; 7 may 1861 – 7 august 1941) was a bengali poet, writer, playwright, composer, philosopher, social reformer and painter. he reshaped bengali literature and music as well as indian art with contextual modernism in the late 19th and early 20th centuries. author of the "profoundly sensitive, fresh and beautiful" poetry of gitanjali, he became in 1913 the first non-european and the first lyricist to win the nobel prize in literature. tagore's poetic songs were viewed as spiritual and mercurial; however, his "elegant prose and magical poetry" remain largely unknown outside bengal. he was a fellow of the royal asiatic society. referred to as "the bard of bengal", tagore was known by sobriquets: gurudeb, kobiguru, biswokobi. life and events a bengali brahmin from calcutta with ancestral gentry roots in burdwan district and jessore, tagore wrote poetry as an eight-year-old. at the age of sixteen, he released his first substantial poems under the pseudonym bhānusiṃha ("sun lion"), which were seized upon by literary authorities as long-lost classics. by 1877 he graduated to his first short stories and dramas, published under his real name. as a humanist, universalist, internationalist, and ardent critic of nationalism, he denounced the british raj and advocated independence from britain. as an exponent of the bengal renaissance, he advanced a vast canon that comprised paintings, sketches and doodles, hundreds of texts, and some two thousand songs; his legacy also endures in his founding of visva-bharati university. early life: 1861–1878 tagore modernised bengali art by spurning rigid classical forms and resisting linguistic strictures. his novels, stories, songs, dance-dramas, and essays spoke to topics political and personal. gitanjali (song offerings), gora (fair-faced) and ghare-baire (the home and the world) are his best-known works, and his verse, short stories, and novels were acclaimed–or panned–for their lyricism, colloquialism, naturalism, and unnatural contemplation. his compositions were chosen by two nations as national anthems: india's "jana gana mana" and bangladesh's "amar shonar bangla". the sri lankan national anthem was inspired by his work. shelaidaha: 1878–1901 the name tagore is the anglicised transliteration of thakur. the original surname of the tagores was kushari. they were pirali brahmin ('pirali' historically carried a stigmatized and pejorative connotation) who originally belonged to a village named kush in the district named burdwan in west bengal. the biographer of rabindranath tagore, prabhat kumar mukhopadhyaya wrote in the first volume of his book rabindrajibani o rabindra sahitya prabeshak that santiniketan: 1901–1932 the kusharis were the descendants of deen kushari, the son of bhatta narayana; deen was granted a village named kush (in burdwan zilla) by maharaja kshitisura, he became its chief and came to be known as kushari. twilight years: 1932–1941 the last two days a storm has been raging, similar to the description in my song–jhauro jhauro borishe baridhara a hapless, homeless man drenched from top to toe standing on the roof of his steamer the last two days i have been singing this song over and over as a result the pelting sound of the intense rain, the wail of the wind, the sound of the heaving gorai river, have assumed a fresh life and found a new language and i have felt like a major actor in this new musical drama unfolding before me. travels – letter to indira devi. works the youngest of 13 surviving children, tagore (nicknamed "rabi") was born on 7 may 1861 in the jorasanko mansion in calcutta, the son of

Fig.101: Summary by GPT2

We are representing the wordcloud for the most frequent words present in the Summary as shown in Fig. 102.

Fig.102: Word Cloud of Summary by GPT2

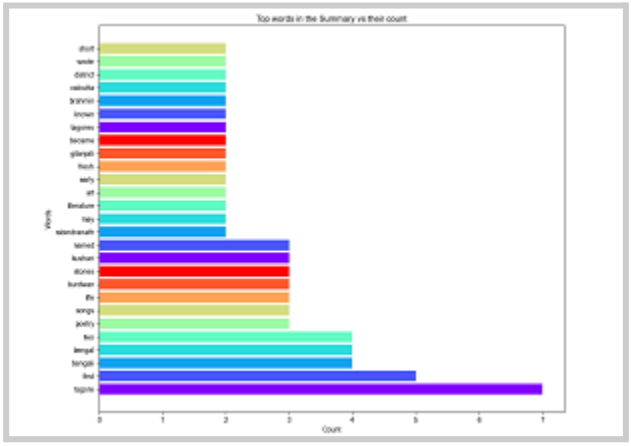


Fig.103: Most Common Word vs Count of Summary by GPT2

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in Fig. 103.

6.2.3.5 Experiment by BART model

We have used the BART model using BartForConditionalGeneration, BartTokenizer, BartConfig Python Library for generating abstractive summary as shown in the output in Fig.104.

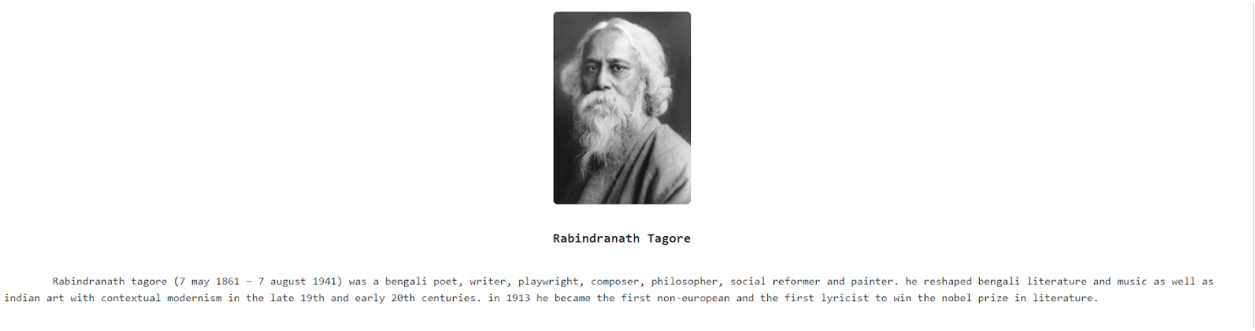


Fig.104: Summary by BART

We are representing the wordcloud for the most frequent words present in the Summary as shown in Fig.105.

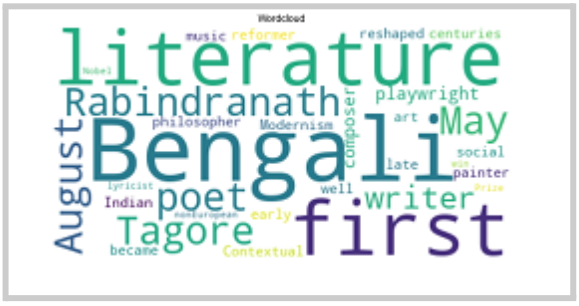


Fig. 105 : Word Cloud of Summary by BART

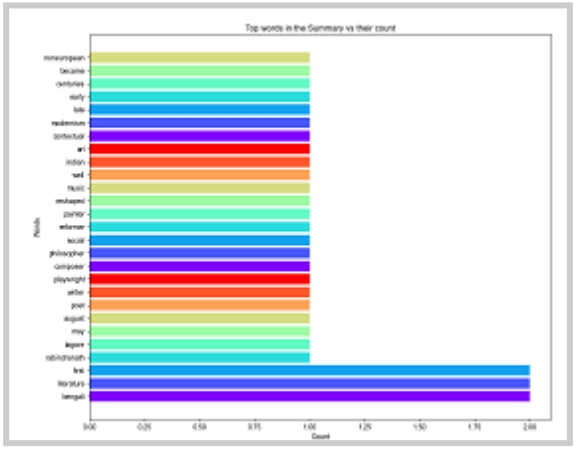


Fig.106: Most Common Word vs Count of Summary by BART

We have described the graphical chart representations of 27 nos of most common keywords extracted from the Summary as shown in the Fig.106

CHAPTER 7

RESULT

This chapter describes the key findings and results of the different models and different datasets performed on the previous chapter. Detailed results obtained from all the models (extractive and abstractive) and the different evaluation metrics used for evaluating the model predicted output summary is also presented here

Let us consider the result generated by each of the datasets discussed in the section above. We used the human generated summaries to perform a comprehensive analysis of the summaries generated by different models.Not only the prediction of output summary by our model is important, but also, evaluating that result is also a significant task without which we cannot confirm whether the output produced by our model is efficient or not. Below are the quantitative and qualitative methods for evaluation of the generated summary.

7.1. Evaluation Metrics

In general, there are three types of evaluations: coselection-based assessment (with a reference summary), document-based assessment (with the original document), and content-based assessment (without reference summary) [32]. We briefly discuss them as follows.

(a) Coselection-Based Evaluation Metrics.

This evaluation technique is based on keywords in the system summary, and it necessitates a comparison of reference summaries of the documents. The reference summary and system summary's common words are chosen and assessed separately. Recall, F-score, and Precision are the measurements .

(b) Content-Based Evaluation Metrics.

This technique assesses the summarizing system in terms that are widely understood.The outline can not get a network of thoughts,a stream of sentences,the relatedness of sentences to previous phrases, or content curiosity. Every one of these difficulties may be addressed using a content-based approach. We show some content-based assessment methodologies that take into account a text's varied features. It just necessitates a system overview, which contains metrics like cohesiveness, nonredundancy, and readability .

(c) Document-Based Evaluation Metrics.

When two phrases in a document have the same relevance, but neither is included in the reference summary, these evaluation metrics fail to assess the system summary properly.

Regarding this paper, we have used coselection-based metrics for evaluation, especially the ROUGE framework, which is explained in more detail hereinafter.

7.2 ROUGE

Since the mid-2000, the ROUGE metric has been broadly utilized for programmed assessment of outlines [16]. Lin called it Recall-Oriented Understudy for Gisting Evaluation (ROUGE), and he presented various measurements that help in naturally deciding the nature of anoutlinebycomparingitwithhuman(reference)synopses considered mostly as the ground truth. Different types of ROUGEs are used in comparing different sentences. The granularity of texts compared between the system summaries and reference summaries can be thought of as ROUGE-L, ROUGE-N, and ROUGE-S.

(a) **ROUGE-N** identifies overlap between unigrams, bigrams, trigrams, and higher-order n-grams

(b) **ROUGE-L** uses the longest common sentence (LCS) to determine the most extended corresponding sequence of terms. LCS has the benefit of demanding in-sequence matches that capture sentence-level word order rather than sequential matches. You do not need to specify an n-gram length since it contains the longest in-sequence typical n-grams by default

(c) **ROUGE-S** is any pair of words in the proper order of a phrase, accounting for gaps. This is referred to as skip-gram concurrence. Skip-bigram, for example, tests the overlap between word pairs with a limit of two spaces between them. For example, the skip bigrams for the term“dog in the basket”will be“dog in,dog the dog basket,in the,in a basket,the basket”

7.3 Evaluation Measures

The evaluation measures used are for context evaluation which comes under co-selection. In co-selection there are three types of measures, they are precision, recall, f-measure.

Precision (P) can be defined as the fraction of the number of sentences common in manual and automated summary to the number of sentences in the automated summary. Precision (P) can be procured using equation .

$$\text{Precision } (P) = \frac{S(manual) \cap S(auto)}{S(auto)}$$

Recall (R) can be defined as the ratio of the number of sentences common in manual and automated summary to the manual summary. The recall can be calculated using equation.

$$\text{Recall } (R) = \frac{S(manual) \cap S(auto)}{S(manual)}$$

F-measure (F) can be defined as the harmonic mean of precision and recall. F-measure (F) can be procured using equation .

$$\text{F-measure } (F) = \frac{2 * Recall\ (R) * Precision\ (P)}{Recall\ (R) + Precision\ (P)}$$

7.3.1 Tabular Representation of The Result

As for the purpose of evaluation we have selected the Three wikipedia URLs as datasets for text summarization using five different models like, TextRank, LSA, T5 Transformer, GPT2, BART . Three wikipedia URLs are as follows:

Wiki URL 1 : “https://en.wikipedia.org/wiki/History_of_India”

Wiki URL 2 : “https://en.wikipedia.org/wiki/Kolkata”

Wiki URL 3 : “https://en.wikipedia.org/wiki/Rabindranath_Tagore”

Summary Evaluation Using Text Rank Model:

Table 1. is showing the comparison of Rouge values for the Text Rank algorithm used for extractive summary with respect to the human created summary where we used the input data set Wiki URL1, Wiki URL2, Wiki URL3 .

Wikipedia Live URL	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Wiki URL 1	0.61691542	0.465290807	0.530481279	0.457295374	0.29271071	0.35694444	0.5	0.37711069	0.429946519
Wiki URL 2	0.68609865	0.25542571	0.37226277	0.507936508	0.18348624	0.269587191	0.533632287	0.19866444	0.289537709
Wiki URL 3	0.72222222	0.193740686	0.305522911	0.59	0.12812161	0.210526313	0.633333333	0.16989568	0.267920091

Table 1: Rouge values of the proposed method Text Rank

Summary Evaluation Using LSA Model:

Table 2. is showing the comparison of Rouge values for the LSA algorithm used for extractive summary with respect to the human created summary where we used the input data set Wiki URL1, Wiki URL2, Wiki URL3.

Wikipedia Live URL	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Wiki URL 1	0.465174129	0.28462709	0.35316336	0.20462633	0.123126338	0.153743311	0.231343284	0.14155251	0.175637389
Wiki URL 2	0.582959641	0.19033675	0.28697571	0.35555556	0.12626832	0.186356069	0.394618834	0.12884334	0.194260482
Wiki URL 3	0.733333333	0.17553191	0.2832618	0.595	0.129207383	0.212310434	0.655555556	0.15691489	0.253218881

Table 2: Rouge values of the proposed method LSA

Summary Evaluation Using T5 Transformer Model:

Table 3. is showing the comparison of Rouge values for the T5 Transformer Model used for abstractive summary with respect to the human created summary where we used the input data set Wiki URL1, Wiki URL2, Wiki URL3.
..

Wikipedia Live URL	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Wiki URL 1	0.024875622	0.41666667	0.04694836	0.003558719	0.07692308	0.00680272	0.012437811	0.208333333	0.02347418
Wiki URL 2	0.044843049	0.4	0.08064516	0.00952381	0.125	0.01769911	0.035874439	0.32	0.06451613
Wiki URL 3	0.044444444	0.32	0.07804878	0.005	0.03846154	0.00884956	0.027777778	0.2	0.04878049

Table 3: Rouge values of the proposed method T5 Transformer

Summary Evaluation Using GPT2 Model:

Table 4. is showing the comparison of Rouge values for the GPT2 model used for extractive summary with respect to the human created summary where we used the input data sets as Wiki URL1, Wiki URL2, Wiki URL3.

Wikipedia Live URL	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Wiki URL 1	0.98756219	0.93632075	0.96125907	0.97330961	0.90413223	0.937446439	0.98756219	0.93632075	0.96125907
Wiki URL 2	0.9955157	0.61325967	0.75897435	0.98412698	0.57943925	0.72941176	0.9955157	0.61325967	0.75897435
Wiki URL 3	0.98888889	0.41203704	0.58169934	0.97	0.36672968	0.532235936	0.98888889	0.41203704	0.58169934

Table 4: Rouge values of the proposed method GPT2

Summary Evaluation Using BART Model:

Table 5. is showing the comparison of Rouge values for the BART model used for extractive summary with respect to the human created summary where we used the input data sets as Wiki URL1, Wiki URL2, Wiki URL3.

Wikipedia Live URL	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Wiki URL 1	0.07960199	0.82051282	0.14512471	0.03558719	0.42553191	0.065681444	0.064676617	0.666666667	0.11791383
Wiki URL 2	0.11659193	0.66666667	0.19847328	0.05396825	0.4047619	0.095238093	0.089686099	0.512820513	0.15267175
Wiki URL 3	0.18333333	0.84615385	0.30136986	0.155	0.775	0.258333331	0.183333333	0.846153846	0.30136986

Table 5: Rouge values of the proposed method BART

Now ,we have evaluated the Rouge values using five proposed methods on WikiURL 1, WikiURL 2 and WikiURL 3 respectively.

Summary Evaluation Using Wikipedia Url 1:

Name of the Models	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
TextRank	0.616915423	0.46529081	0.53048128	0.45729537	0.29271071	0.35694444	0.5	0.37711069	0.42994652
LSA	0.465174129	0.28462709	0.35316336	0.20462633	0.12312634	0.153743311	0.231343284	0.14155251	0.17563739
T5 Transformer	0.024875622	0.41666667	0.04694836	0.00355872	0.07692308	0.0068072	0.012437811	0.20833333	0.02347418
GPT2	0.987562189	0.93632075	0.96125907	0.97330961	0.90413223	0.937446439	0.987562189	0.93632075	0.96125907
BART	0.07960199	0.82051282	0.14512471	0.03558719	0.42553191	0.065681444	0.064676617	0.66666667	0.11791383

Table 6: Comparison of Rouge values using five proposed methods on WikiURL 1

Summary Evaluation Using Wikipedia Url 2:

Name of the Models	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
TextRank	0.68609865	0.25542571	0.37226277	0.50793651	0.183486239	0.26958719	0.533632287	0.19866444	0.28953709
LSA	0.58295964	0.19033675	0.286975714	0.35555556	0.12626832	0.18635607	0.394618834	0.12884334	0.194260482
T5 Transformer	0.04484305	0.4	0.080645159	0.00952381	0.125	0.01769911	0.035874439	0.32	0.064516127
GPT2	0.9955157	0.61325967	0.758974354	0.98412698	0.579439252	0.72941176	0.995515695	0.61325967	0.758974354
BART	0.11659193	0.66666667	0.19847328	0.05396825	0.404761905	0.09523809	0.089686099	0.51282051	0.152671753

Table 7: Comparison of Rouge values using five proposed methods on WikiURL 2

Summary Evaluation Using Wikipedia Url 3:

Name of the Models	ROUGE - 1			ROUGE - 2			ROUGE - L		
	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
TextRank	0.72222222	0.193740686	0.30552291	0.59	0.128121607	0.210526313	0.63333333	0.16989568	0.26792009
LSA	0.73333333	0.175531915	0.2832618	0.595	0.129207383	0.212310434	0.65555556	0.15691489	0.25321888
T5 Transformer	0.04444444	0.32	0.07804878	0.005	0.038461538	0.008849555	0.0277778	0.2	0.04878049
GPT2	0.98888889	0.412037037	0.58169934	0.97	0.366729679	0.532235936	0.98888889	0.41203704	0.58169934
BART	0.18333333	0.846153846	0.30136986	0.155	0.775	0.258333331	0.18333333	0.84615385	0.30136986

Table 8: Comparison of Rouge values using five proposed methods on WikiURL 3

7.3.2 Graphical Representation

Summary Evaluation Using TextRank Model:

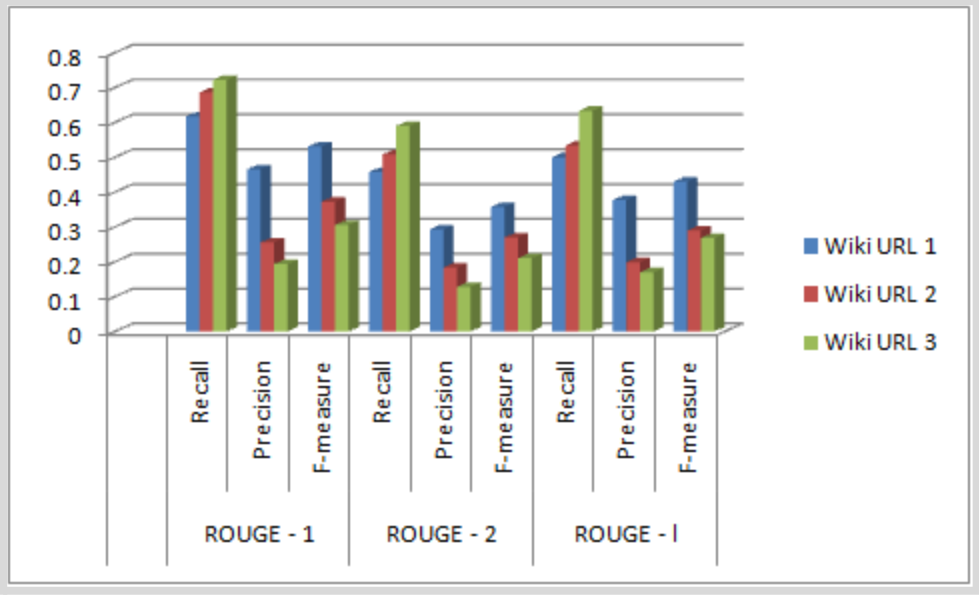


Fig.107: Similarity Chart for the Precision, Recall and F-Score (TextRank)

Summary Evaluation Using LSA Model:

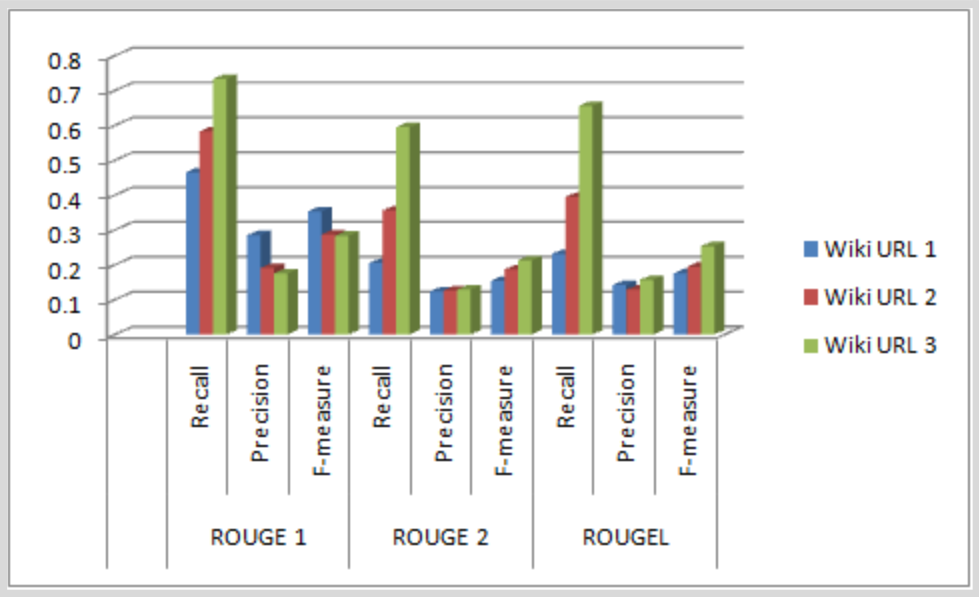


Fig.108: Similarity Chart for the Precision, Recall and F-Score (LSA)

Summary Evaluation Using T5 Transformer Model:

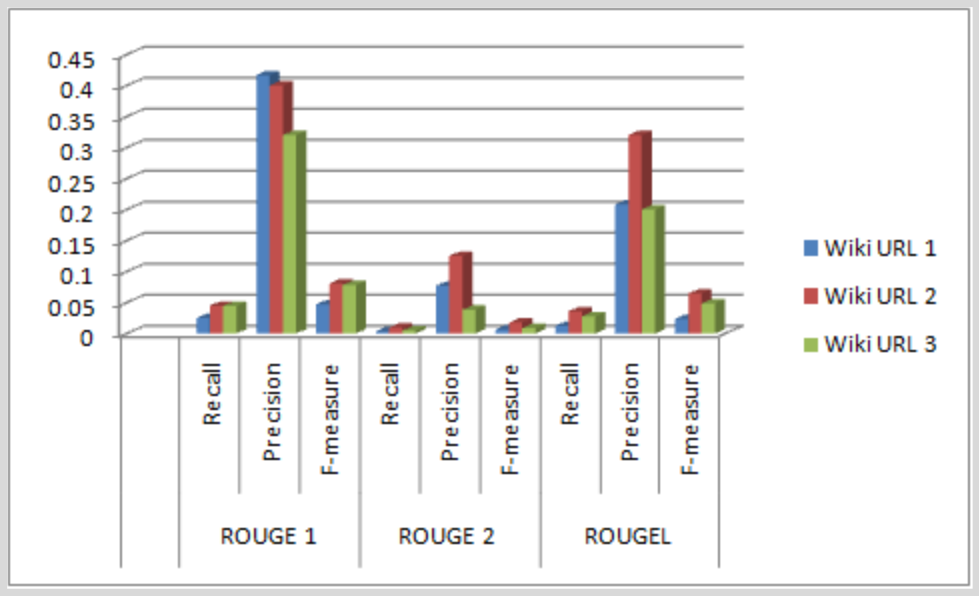


Fig.109: Similarity Chart for the Precision, Recall and F-Score (T5 Transformer)

Summary Evaluation Using GPT2 Model:

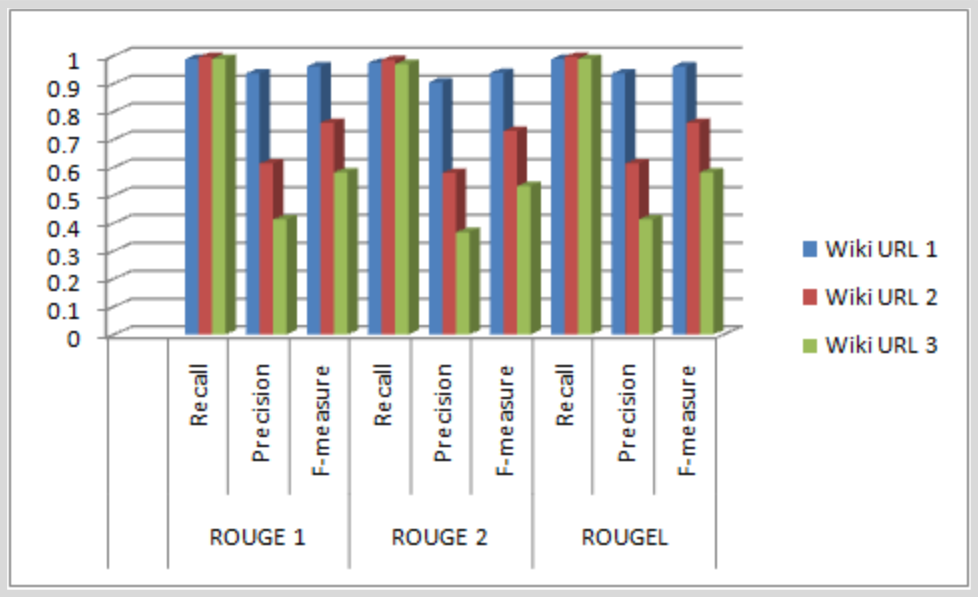


Fig.110: Similarity Chart for the Precision, Recall and F-Score (GPT2)

Summary Evaluation Using BART Model:

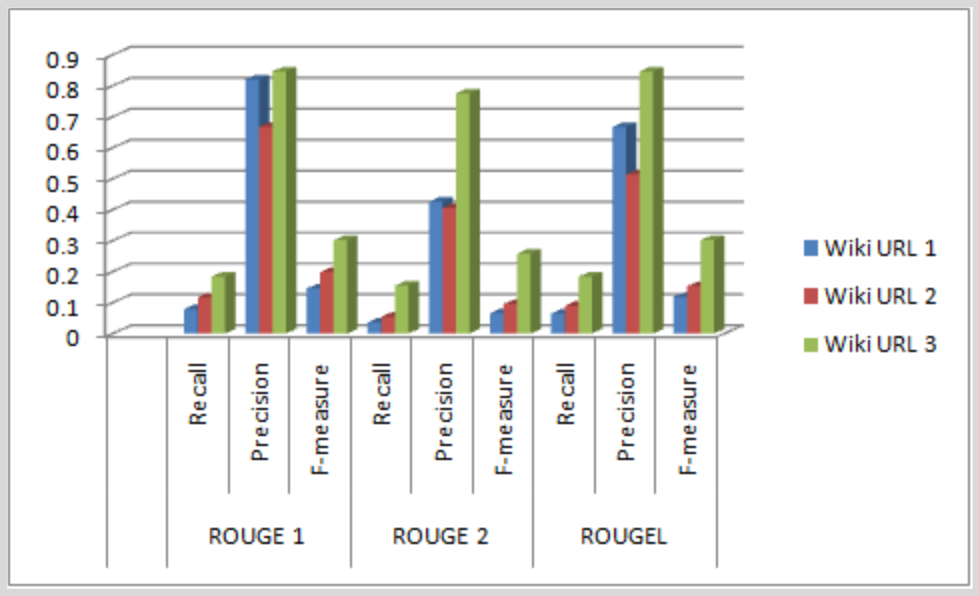


Fig.111: Similarity Chart for the Precision, Recall and F-Score (BART)

Summary Evaluation Using Wikipedia Url 1:

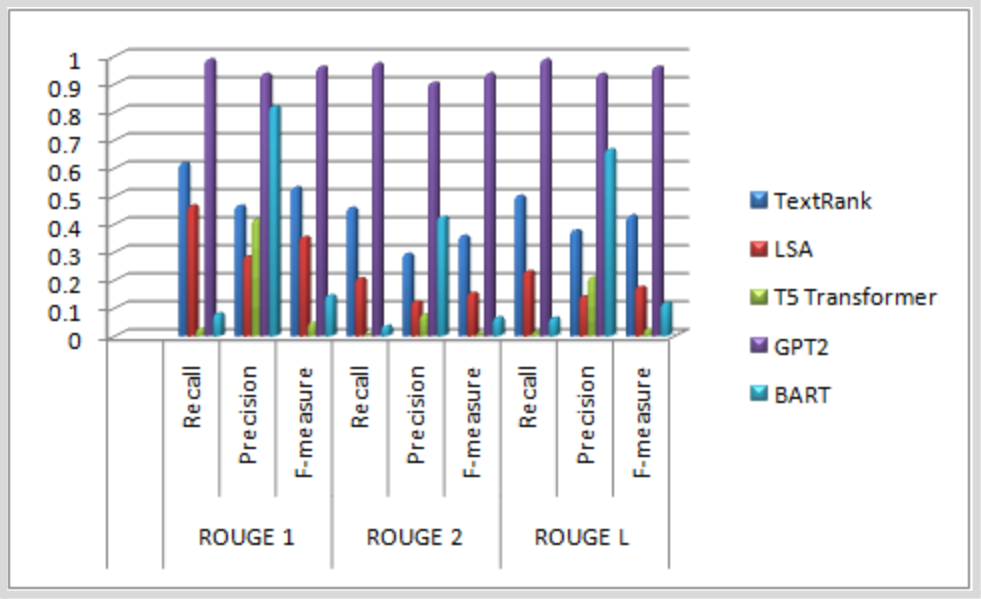


Fig.112: Similarity Chart for the Precision, Recall and F-Score on Wiki URL1

Summary Evaluation Using Wikipedia Url 2:

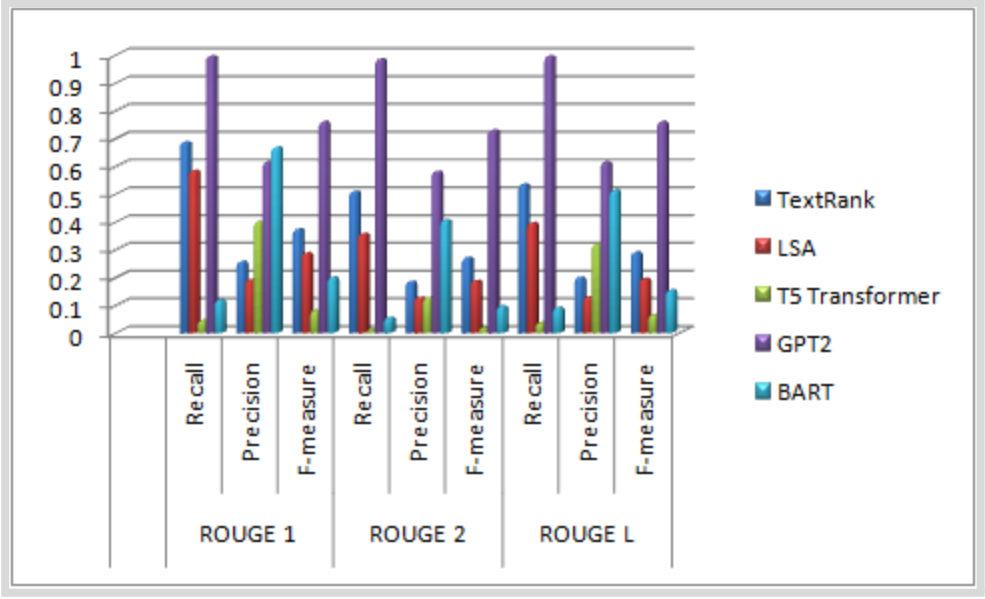


Fig.113: Similarity Chart for the Precision, Recall and F-Score on Wiki URL2

Summary Evaluation Using Wikipedia Url 3:

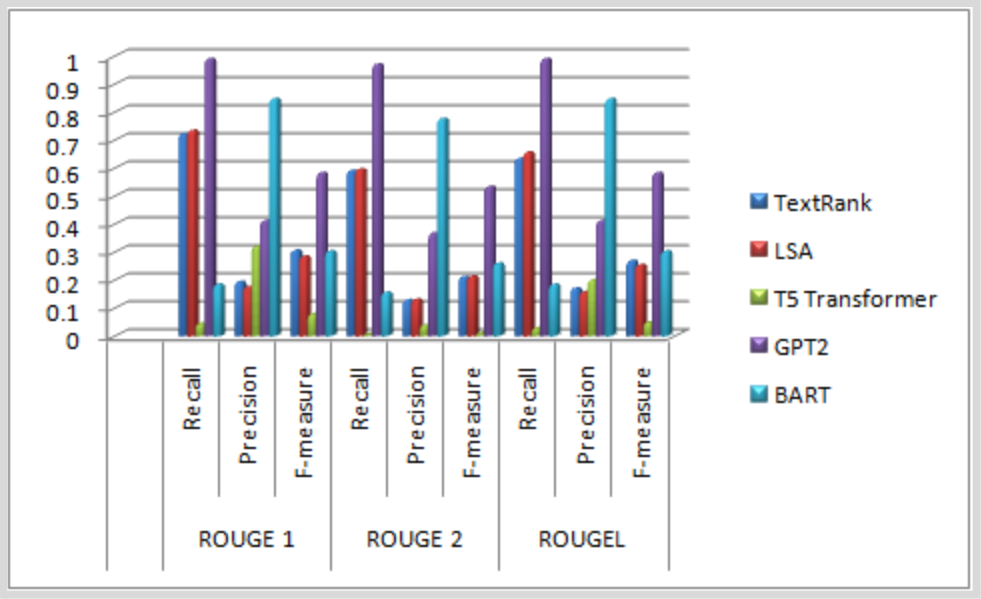


Fig.114: Similarity Chart for the Precision, Recall and F-Score on Wiki URL3

CHAPTER 8

ANALYTICS

This chapter describes how the Venn diagram can be used to effectively explore and visualize results using keywords extracted from summary generated by different algorithms. By using a familiar and intuitive diagram for set relations, the results could provide not only the retrieval of keywords, but also its relevance to the summary.

Text summarization is an important task in natural language processing, and there are various algorithms and models that can be used to generate summaries of text. However, the accuracy and quality of these summaries can vary depending on factors such as the model architecture, training data, and hyperparameters.The Venn diagram provides a visual representation of the overlap and differences between the keywords generated by each model, and can help us to identify patterns and insights into the strengths and weaknesses of each model for summarizing text. By leveraging these insights, we can select the most appropriate model for your specific use case and optimize the effectiveness of your text summarization efforts.

8.1. Data Preparation:

By analyzing the keywords generated by different ML models for the same text, a Venn diagram can help identify patterns and relationships between the keywords, and reveal insights into the performance of the different models.
We have used five different models that generate summaries of three different wikipedia articles . Each summary consists of a set of keywords that capture the main ideas and concepts in the article. To compare the performance of the models, we create a Venn diagram that shows the overlap between the keywords generated by each model on different wikipedia articles.

8.2. Visualization by Venn Diagram

After generating the summaries, we have analyzed the keywords from each summary using a Venn diagram to visualize the overlap between the keywords generated by each model for Extractive Summary, Abstractive summary and both of two types of summarizations.

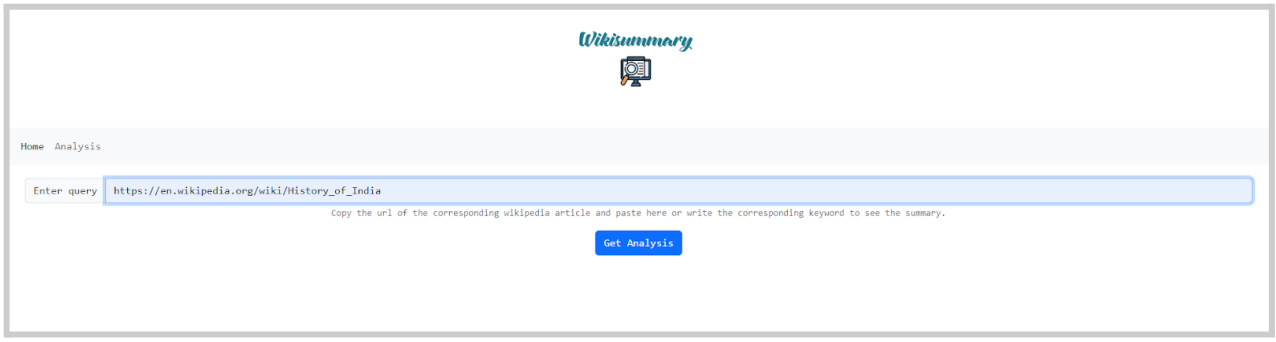


Fig.115: Analysis of Summarization

8.2.1. Analysis For Extractive Summary

We have used three models like Summary by Sentence using LSA model, Summary by Ratio using TextRank algorithm, Summary by Word count using TextRank algorithm for doing extractive summarization on Wikipedia Live Urls like :

Wiki URL 1: ”https://en.wikipedia.org/wiki/History_of_India”
Wiki URL 2: ”https://en.wikipedia.org/wiki/Kolkata”
Wiki URL 3: ”https://en.wikipedia.org/wiki/Rabindranath_Tagore”

Venn diagram presentation on Wiki URL 1:

First Venn Diagram Fig. 116 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated using Summary by Sentence, Summary by Ratio(0.1), Summary by Word count(1000) where we have used Wiki URL1 as input data set.

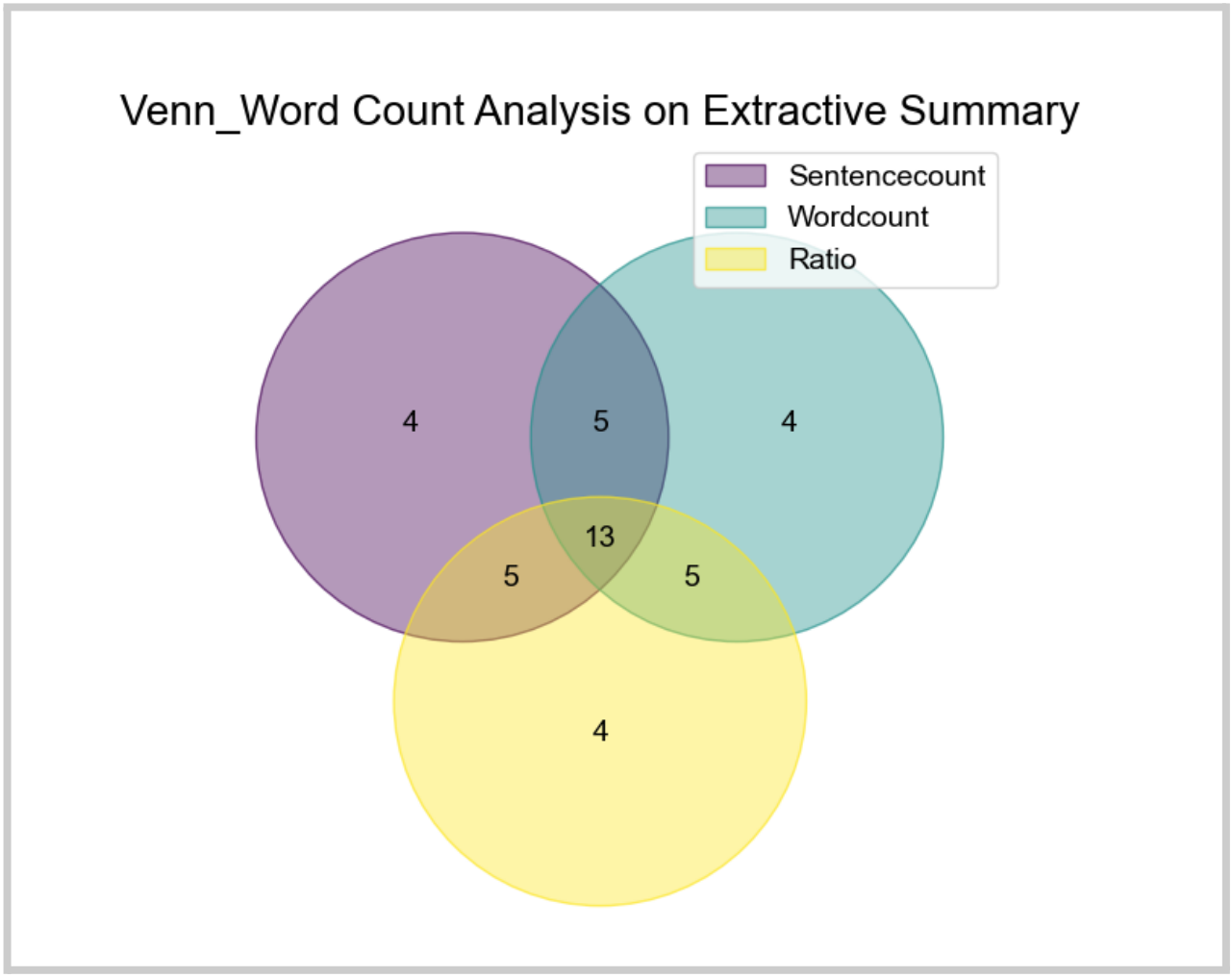


Fig.116: Venn diagram for Word Count analysis of Extractive Summarization

Second Venn Diagram Fig. 117 is showing the word cloud of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated using Summary by Sentence (50 nos of sentences) , Summary by Ratio(0.1), Summary by Word Count(1000 nos of word) where we have used Wiki URL1 as input data set.

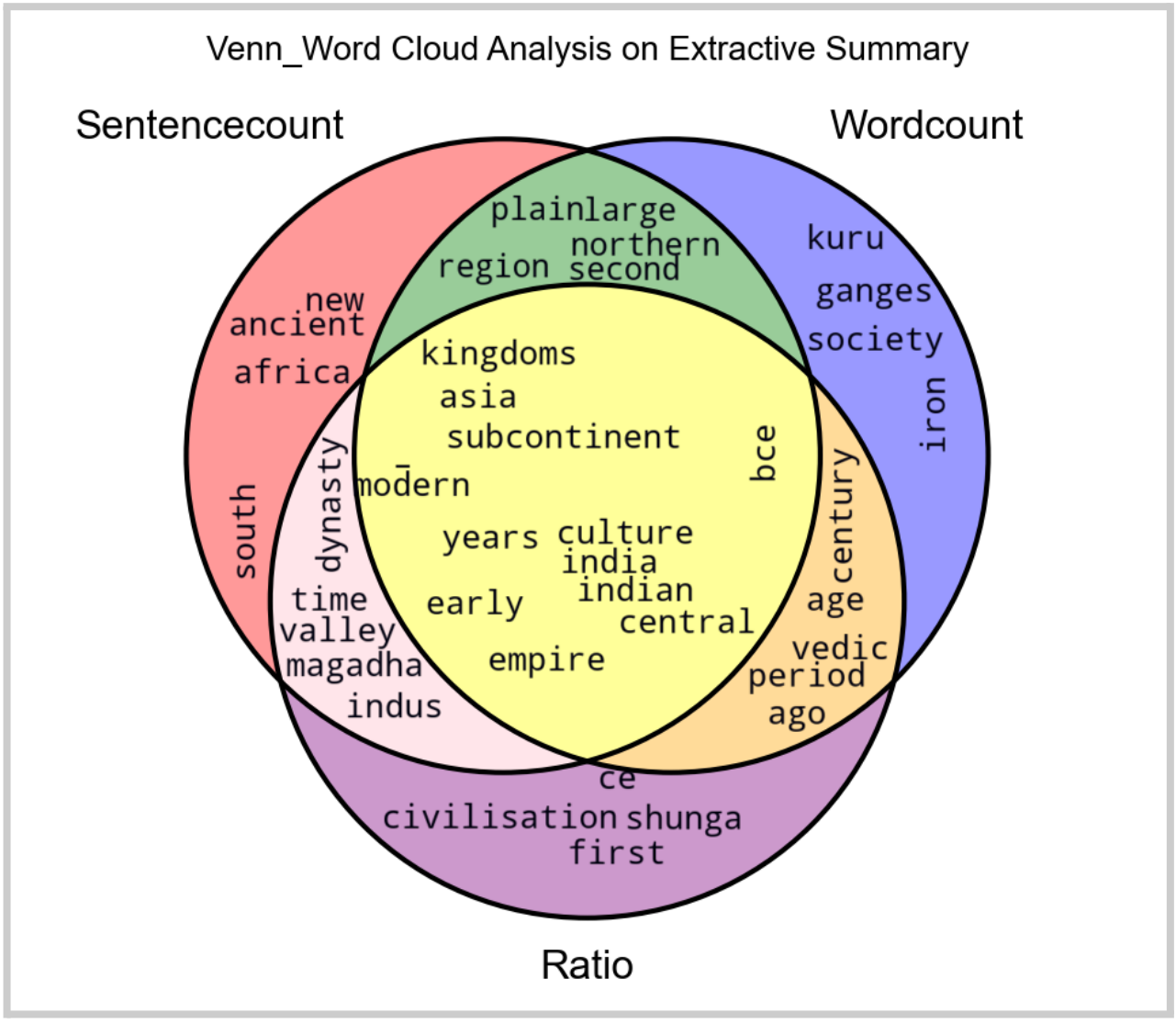


Fig.117: Venn diagram with Word Cloud for analysis of Extractive Summarization

We notice that three models have identified thirteen nos of keywords "bce","indian","kingdoms","asia","subcontinent","modern","years","culture","india","early","central","empire", in an article about Indian history, those have appeared in the intersection area of three models and get the highest weightage as most common keywords extracted from three different kinds of summary.

Venn diagram presentation on Wiki URL 2:

First Venn Diagram Fig. 118 is showing the word cloud of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated using Summary by Sentence (50 nos of sentences) , Summary by Ratio(0.1), Summary by Word Count(1000 nos of word) where we have used Wiki URL2 as input data set.

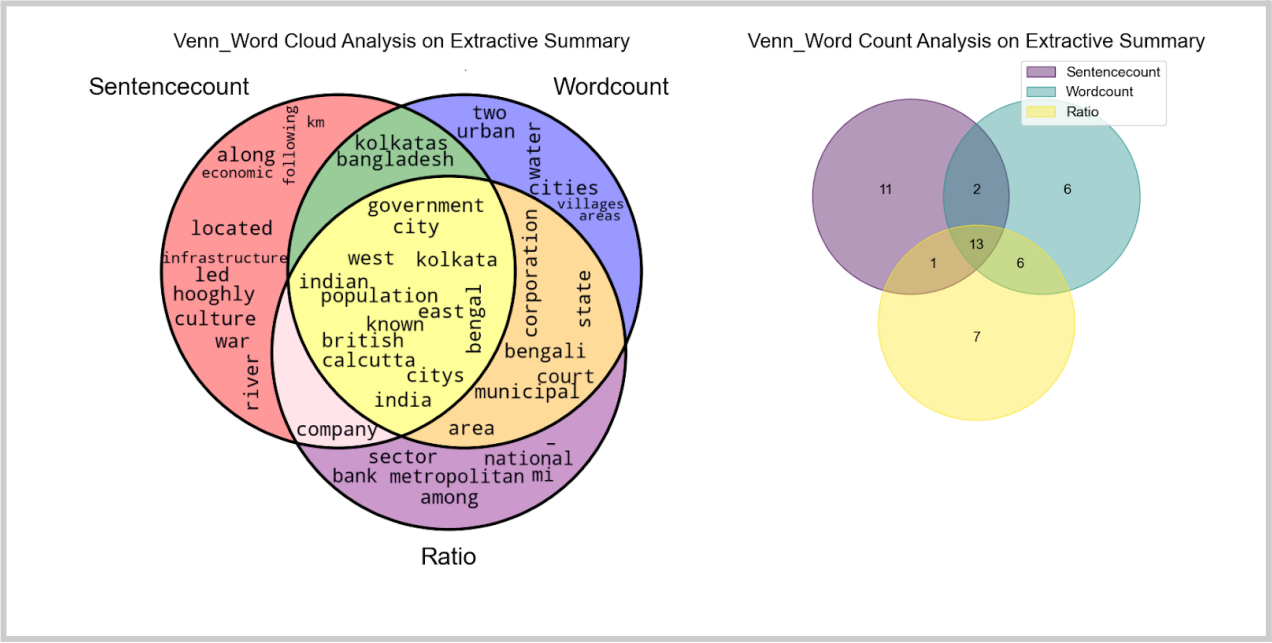


Fig.118: First Venn diagram with Word Cloud for analysis of Extractive Summarization
Second Venn diagram with Word Count for analysis of Extractive Summarization

Second Venn Diagram Fig. 118 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated using Summary by Sentence, Summary by Ratio(0.1), Summary by Word count(1000) where we have used Wiki URL2 as input data set.

We notice that three models have identified thirteen nos of keywords "government", "city", "west", "kolkata", "indian", "population", "east", "bengal", "known", "british","calcutta","citys", "india" in an article about Kolkata those have appeared in the intersection area of three models and get the highest weightage as most common keywords extracted from three different kinds of summary.

Venn diagram presentation on Wiki URL 3:

First Venn Diagram Fig.119 is showing the word cloud of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated using Summary by Sentence (50 nos of sentences) , Summary by Ratio(0.1), Summary by Word Count(1000 nos of word) where we have used Wiki URL3 as input data set.

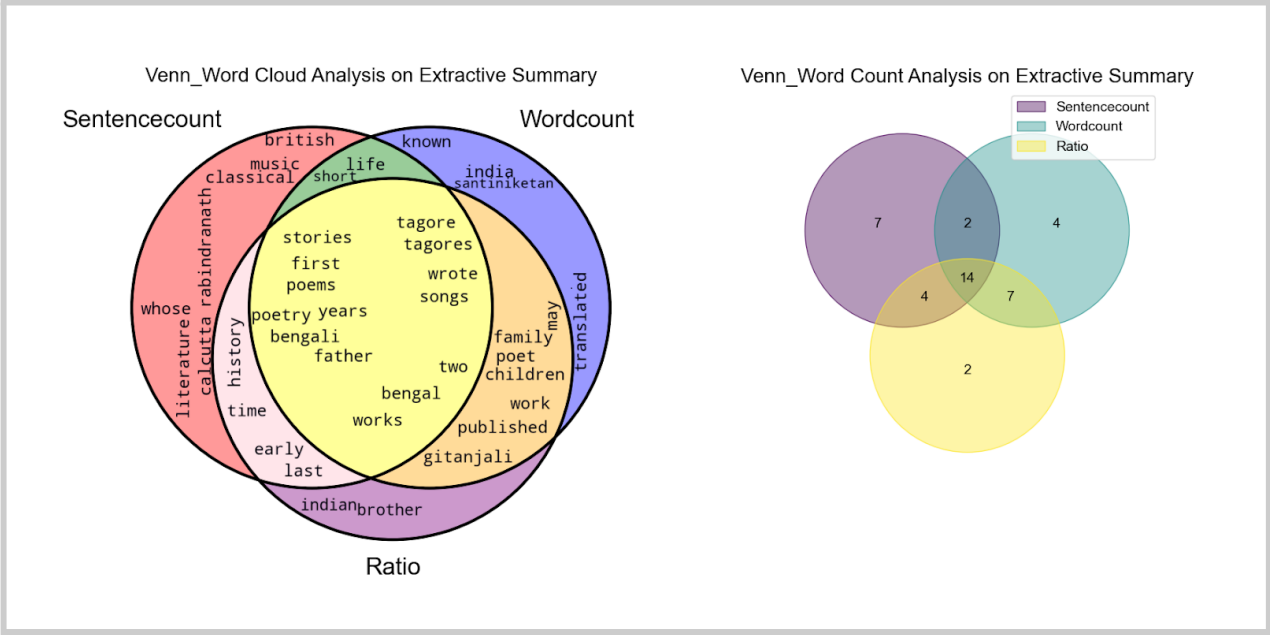


Fig.119: First Venn diagram with Word Cloud for analysis of Extractive Summarization
Second Venn diagram with Word Count for analysis of Extractive Summarization

Second Venn Diagram Fig. 119 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated using Summary by Sentence, Summary by Ratio(0.1), Summary by Word count(1000) where we have used Wiki URL3 as input data set.

We notice that three models have identified fourteen nos of keywords "tagore", "stories", "tagores", "first", "poems", “wrote", "years", "songs", "poetry", "bengali", "father", "two", "bengal", "works" in an article about Rabindranath Tagore, those have appeared in the intersection area of three models and get the highest weightage as most common keywords extracted from three different kinds of summary.

8.2.2. Analysis for Abstractive Summary

We have used three ML models like Summary using T5 Transformer model, Summary using GPT2, Summary using BART model for doing abstractive summarization on Wikipedia Live Urls like : :

Wiki URL 1:"https://en.wikipedia.org/wiki/History_of_India"
Wiki URL 2:"https://en.wikipedia.org/wiki/Kolkata"
Wiki URL 3:"https://en.wikipedia.org/wiki/Rabindranath_Tagore"

Venn diagram presentation on Wiki URL 1:

First Venn Diagram Fig. 120 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated Summary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL1 as input data set.

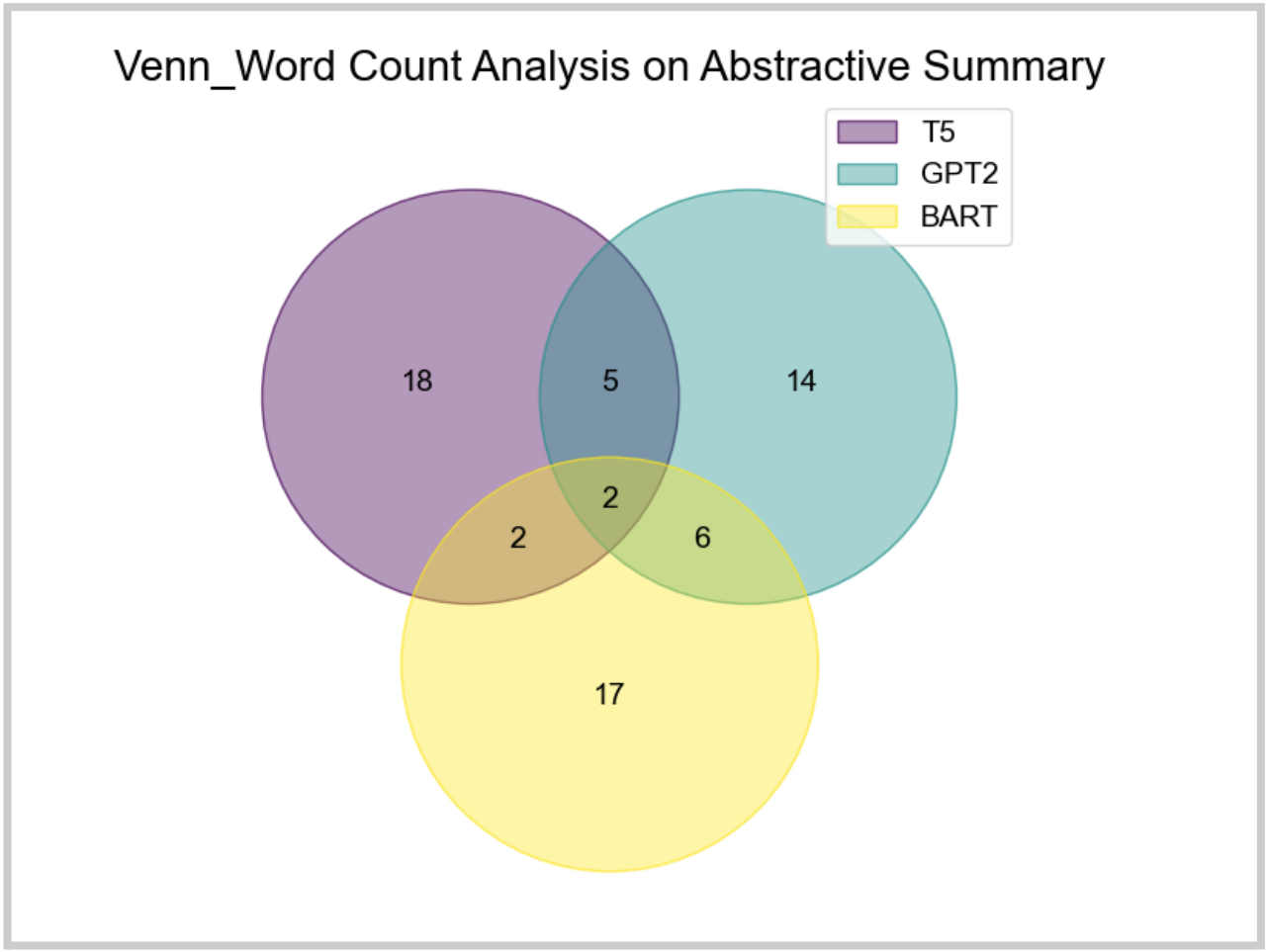


Fig.120: Venn diagram for Word Count Analysis of Abstractive Summarization

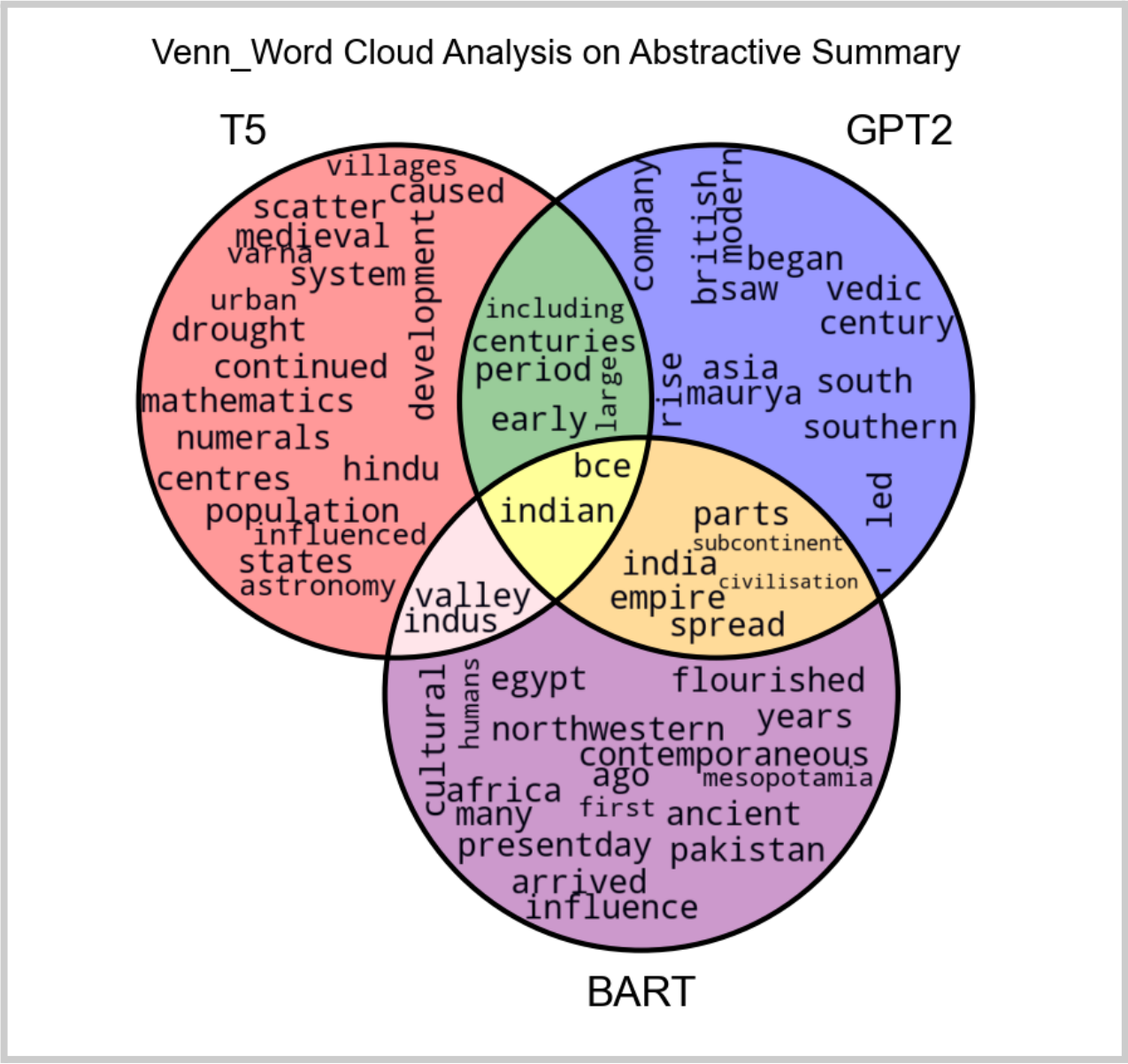


Fig.121: Venn diagram with Word Cloud for Analysis of Abstractive Summarization

Second Venn Diagram Fig. 121 is showing the word cloud of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated usingSummary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL1 as input data set.

We notice that three models have identified two keywords "bce", "indian" in an article about Indian history, those have appeared in the intersection area of three models and get the highest weightage as most common keywords extracted from three different kinds of summary.

Venn diagram presentation on Wiki URL 2:

First Venn Diagram Fig.122 is showing the word cloud of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated usingSummary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL2 as input data set.

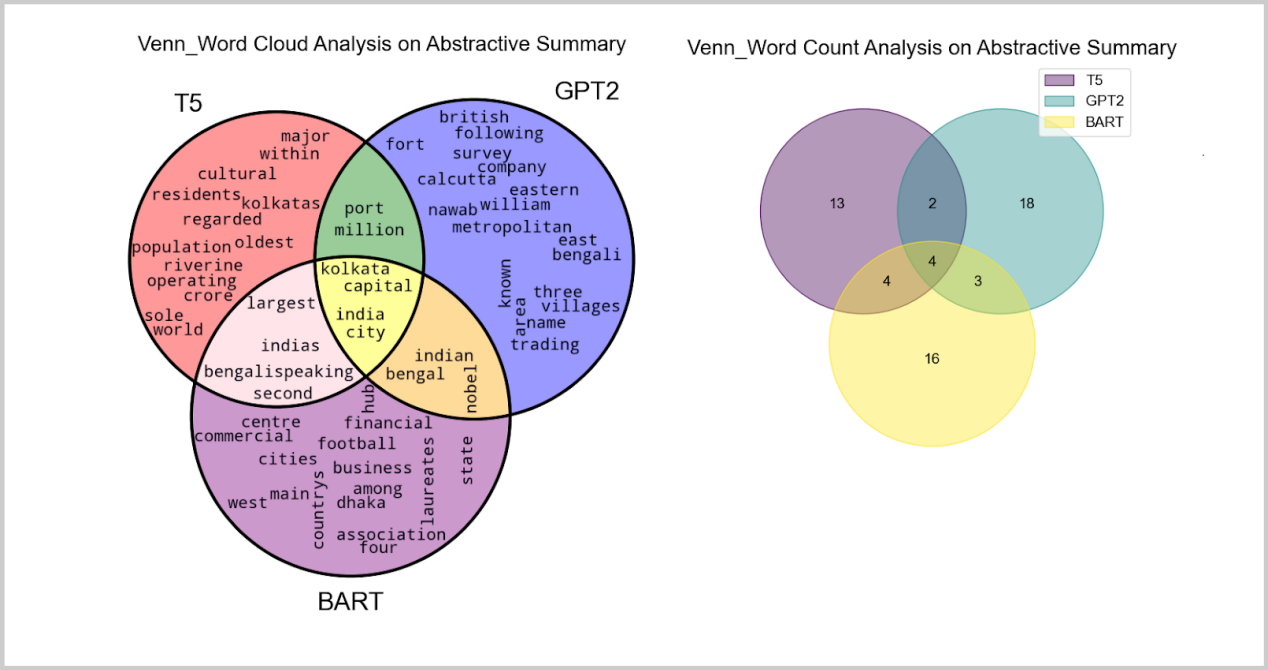


Fig.122: First Venn diagram with Word Cloud for analysis of Abstractive Summarization Second Venn diagram for Word Count Analysis of Abstractive Summarization

Second Venn Diagram Fig.122 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated Summary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL2 as input data set.

We notice that three models have identified four keywords "kolkata", "capital", "india", "city" in an article about Kolkata, those have appeared in the intersection area of three models and get the highest weightage as most common keywords extracted from three different kinds of summary.

Venn diagram presentation on Wiki URL 3:

First Venn Diagram Fig.123 is showing the word cloud of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated usingSummary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL3 as input data set.

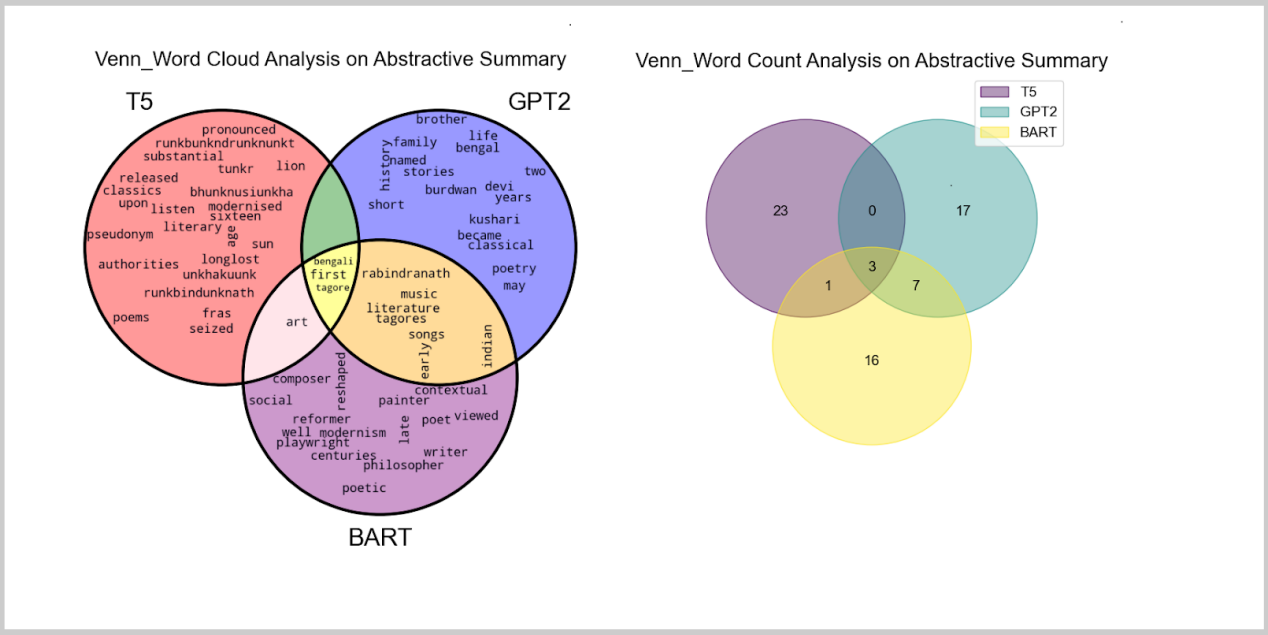


Fig.123: First Venn diagram with Word Cloud for analysis of Abstractive Summarization Second Venn diagram for Word Count Analysis of Abstractive Summarization

First Venn Diagram Fig.123 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated Summary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL3 as input data set.

We notice that three models have identified three keywords "bengali", "first", "tagore" in an article about Rabindranath, those have appeared in the intersection area of three models and get the highest weightage as most common keywords extracted from three different kinds of summary.

8.2.3. Analysis for Extractive and Abstractive Summary

We have used five models LSA, TextRank, T5 Transformer, GPT2, BART models for doing Extractive and Abstractive Summary both on Wikipedia Live Urls like:
Wiki URL 1:"https://en.wikipedia.org/wiki/History_of_India"
Wiki URL 2:"https://en.wikipedia.org/wiki/Kolkata"
Wiki URL 3:"https://en.wikipedia.org/wiki/Rabindranath_Tagore"

Venn diagram presentation on Wiki URL 1:

First Venn Diagram Fig.124 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated Summary by LSA , Summary by TextRank,Summary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL1 as input data set.

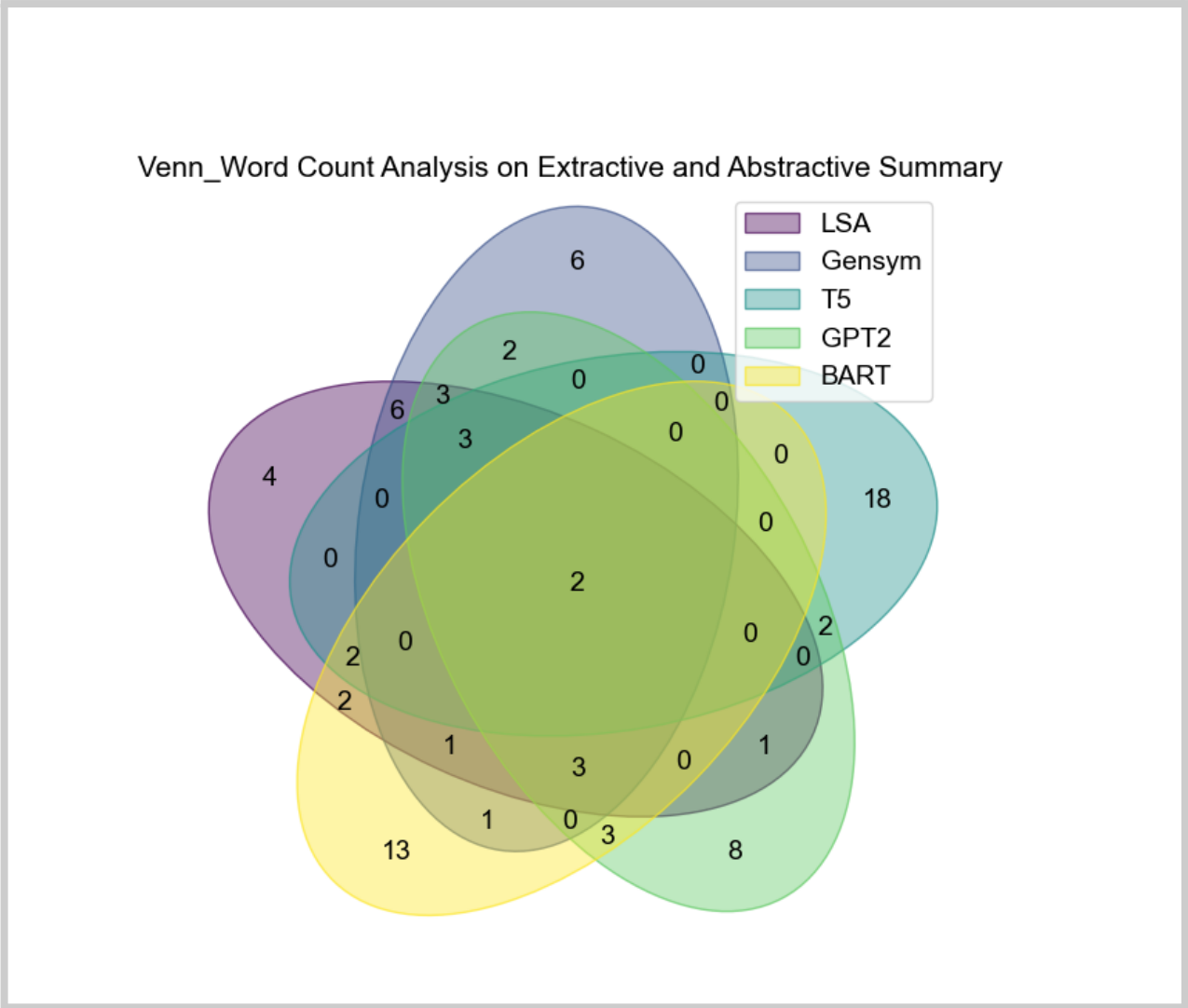


Fig.124: Venn diagram for analysis of Extractive and Abstractive Summarization

Venn diagram presentation on Wiki URL 2:

Second Venn Diagram Fig. 125 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated Summary by LSA , Summary by TextRank,Summary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL2 as input data set.

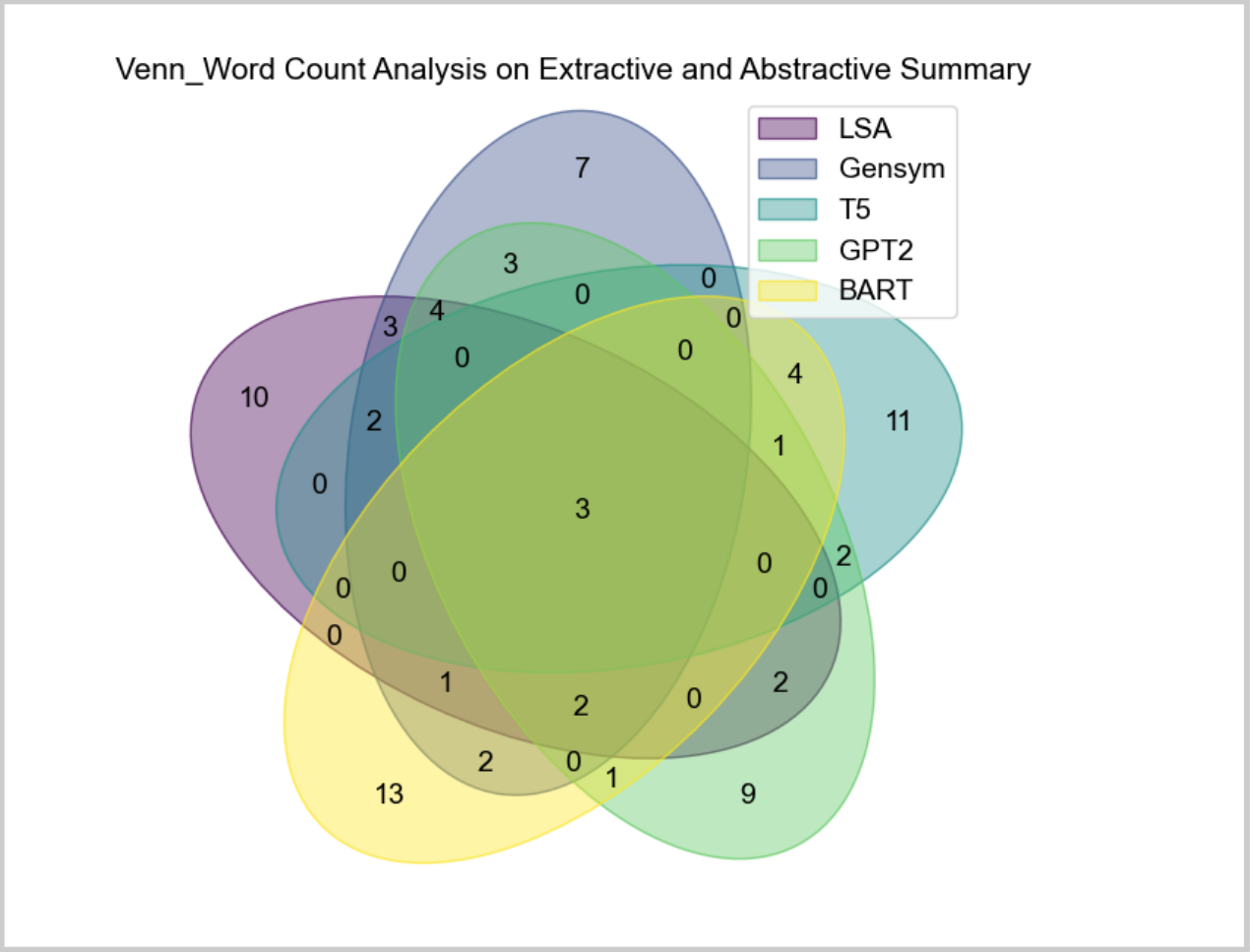


Fig.125: Venn diagram for analysis of Extractive and Abstractive Summarization

Venn diagram presentation on Wiki URL 3:

Third Venn Diagram Fig.126 is showing the word count of each overlapping area and non overlapping area of the most common keywords extracted from three different kind of WikiSummary generated Summary by LSA , Summary by TextRank,Summary using T5 Transformer model, Summary using GPT2, Summary using BART model where we have used Wiki URL3 as input data set.

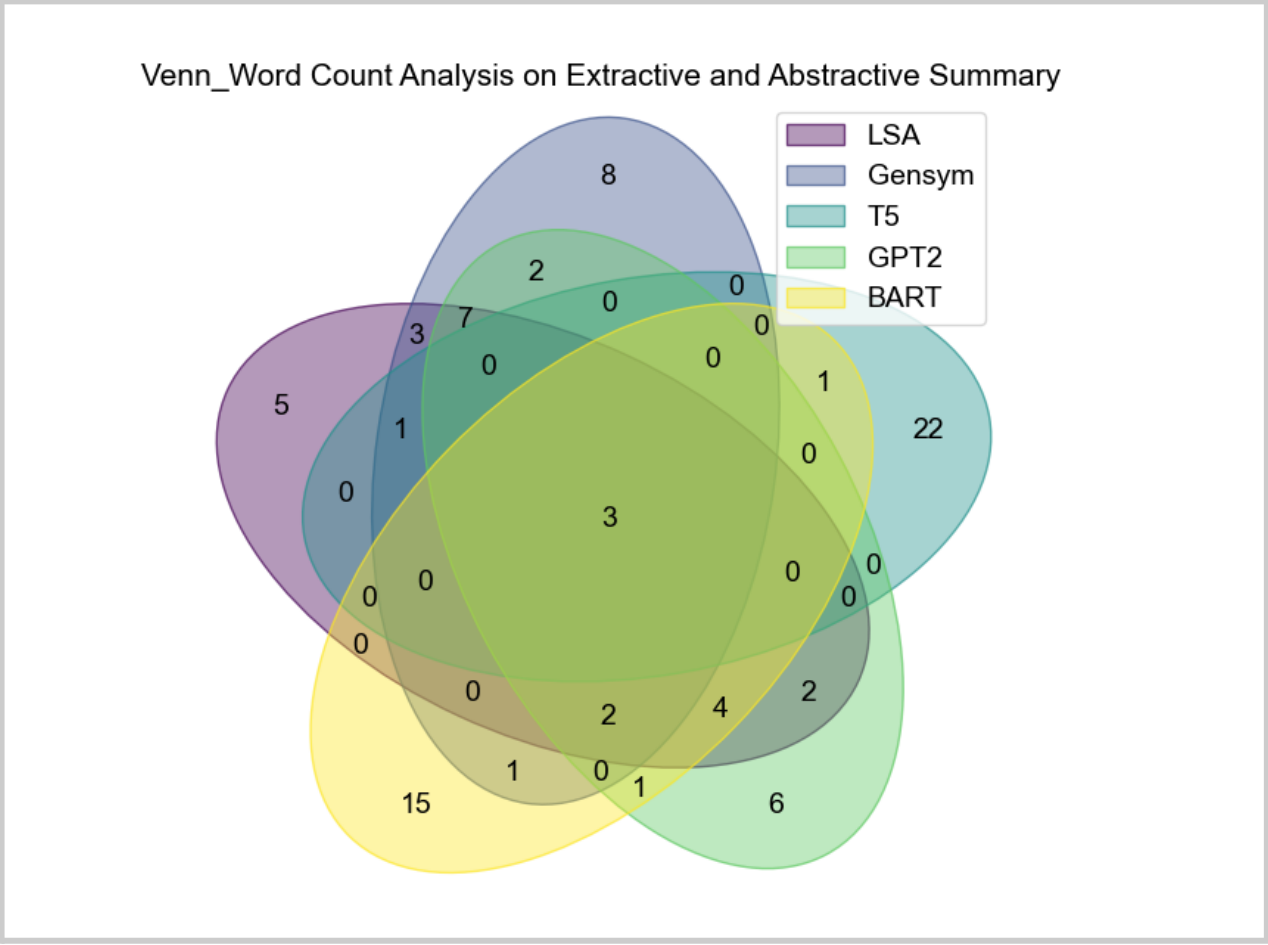


Fig.126: Venn diagram for analysis of Extractive and Abstractive Summarization

8.2.4. Consensus Analysis for Extractive and Abstractive Summary

Analyze the Venn diagram to identify patterns and relationships between the keywords. Look for areas of overlap between the circles, which represent keywords that are shared between two or more categories.

For example, if the Venn diagram shows a significant overlap between the keywords generated by two or more models, this suggests that these models may have a higher degree of accuracy and relevance in generating important information related to the text. This insight can guide the selection of the most appropriate model for future text summarization tasks.

On the other hand, if there is little or no overlap between the keywords generated by different models, this could indicate a lack of consensus or agreement on the important information in the text. This insight can guide further investigation into the reasons for the discrepancies and potentially inform decisions about the most appropriate way to summarize the text.

However, upon closer inspection, in Fig.124 We notice that three models have identified some keywords that are not present in the summaries generated by the other two models. For example, we also notice that five models have identified two keywords "Indian " and “bce” in an article about Indian history, whereas the other keywords are not included by those models. This suggests that those 2 keywords extracted may be better at identifying more nuanced or specific information in the article.Similarly, in Fig.125, three keywords "City", “Kolkata” and “India” in an article about Kolkata and in Fig.126, three keywords "Bengali", “First” and “Tagore” in an article about Rabindranath Tagore have been identified by five models.

This insight suggests that which model may have a higher degree of accuracy in generating relevant and important information related to the wikipedia articles. This can guide the selection of the most appropriate model for future text summarization tasks related to this topic.

Overall, the Venn diagram provided valuable insights into the performance and accuracy of the different ML models for text summarization. By identifying patterns and relationships between the keywords generated by each model, the Venn diagram helped the researchers to make informed decisions about which model to use for future summarization tasks, depending on the specific goals and requirements of the task.By leveraging these insights, we can select the most appropriate model for our specific use case and optimize the effectiveness of your text summarization efforts.

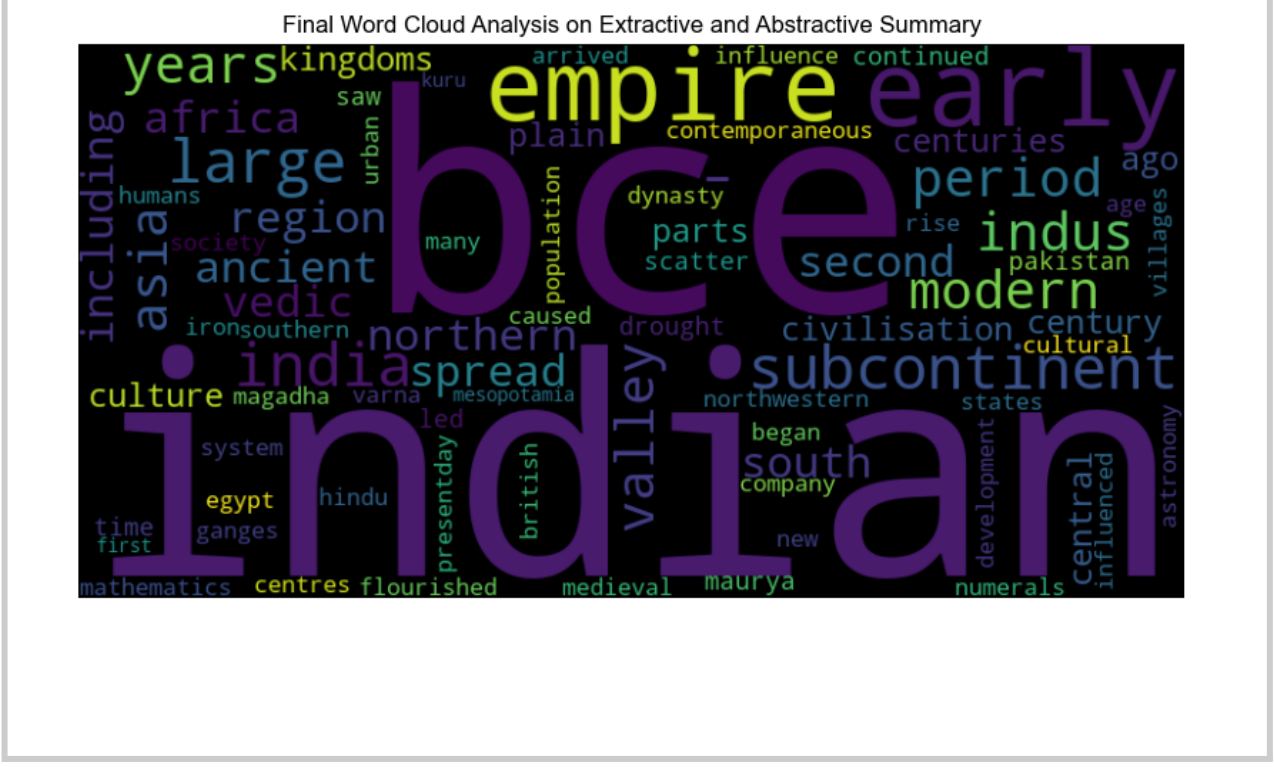


Fig.127: Final Word Cloud for analysis of Keywords extracted from Extractive and Abstractive Summarization URL1

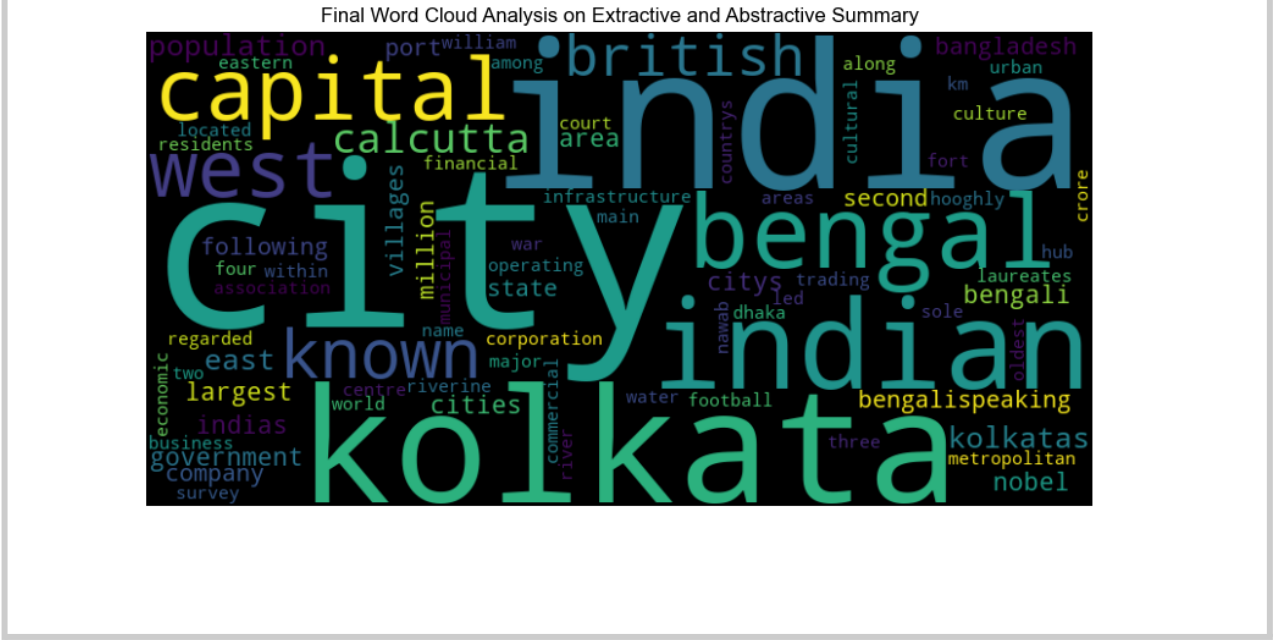


Fig.128: Final Word Cloud for analysis of Keywords extracted from Extractive and Abstractive Summarization URL2

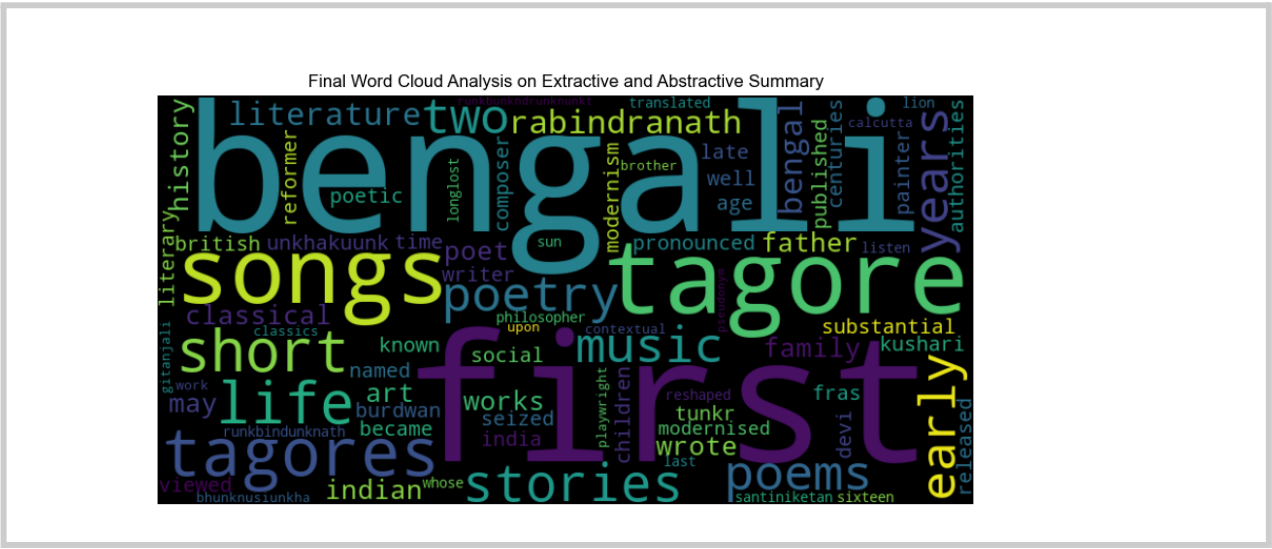


Fig.129: Final Word Cloud for analysis of Keywords extracted from Extractive and Abstractive Summarization URL3

Finally with the new sets of keywords extracted from different summaries we have generated a new word cloud as shown in Fig.127, Fig.128, Fig.129 where the we have given the most weightage to those keywords which are present in the overlapping area of all five models and the least weightage which are only appeared in one keyword sets generated by only one model.We have assigned all the weightage in accordance with the appearance of every keyword with respect to the number of overlapping area of models.

CHAPTER 9

CONCLUSION AND FUTURE WORK

The consensus-based approach has been shown to be a promising method for multi-document summarization of Wikipedia pages. By leveraging multiple summarization algorithms and combining their outputs using a voting mechanism, the approach is able to generate high-quality summaries that outperform several state-of-the-art methods. The approach is particularly well-suited for summarization of large and complex information sources such as Wikipedia, where the use of a single summarization algorithm may not be sufficient to capture all the important information.

Keyword extraction is an essential task in text summarization, and the consensus-based approach has emerged as a promising method for improving the accuracy and robustness of this process. By combining the outputs of multiple summarization algorithms, the consensus-based approach can overcome the limitations of individual algorithms and produce a more comprehensive and accurate summary of the input text. Our review of the literature suggests that the consensus-based approach has been successfully applied to a wide range of texts, including news articles, scientific papers, and

Wikipedia pages. Several studies have shown that the consensus-based approach outperforms individual algorithms and other state-of-the-art approaches in terms of content coverage, coherence, and readability.
Despite its success, there are still several challenges that need to be addressed to further enhance the effectiveness of the consensus-based approach. These challenges include determining the optimal weights assigned to each algorithm, accounting for changes in the quality of outputs over time, and incorporating additional sources of information to refine the keyword extraction process. Future research in these areas has the potential to enhance the accuracy and effectiveness of the consensus-based approach and further advance the field of text summarization.

In conclusion, the consensus-based approach is a promising method for keyword extraction from multiple summarization algorithms, and its success has been demonstrated across a range of text types and domains. While there are still challenges to be addressed, the consensus-based approach offers a viable solution for producing accurate and comprehensive summaries of large volumes of text.

Future research can explore the use of additional techniques such as machine learning and external knowledge integration to further improve the accuracy and effectiveness of the consensus-based approach for summarization of Wikipedia pages.
This study frames future work on the consensus approach for keyword extraction from multiple summarization algorithms:

Weight optimization: One of the main challenges of the consensus approach is determining the optimal weights assigned to each algorithm. Future research can investigate the use of machine learning techniques, such as regression or neural networks, to estimate the weights based on the characteristics of the input text and the performance of the individual algorithms.

Noise handling: The current consensus approach assumes that all the outputs from the individual algorithms are of equal quality. However, in practice, some outputs may be noisy or of low quality, which can negatively affect the accuracy of the consensus-based keyword extraction. Future research can explore the use of techniques such as outlier detection, quality estimation, and clustering to identify and filter out the noisy or low-quality outputs.

Transparency and interpretability: The consensus approach is often criticized for its lack of transparency and interpretability, as it is not always clear why certain keywords are selected and how the consensus is reached.

Future research can develop more transparent and interpretable approaches that can provide insights into the decision-making process of the consensus-based keyword extraction.

External knowledge integration: The current consensus approach relies solely on the outputs of the individual algorithms, without taking into account any external knowledge sources. Future research can investigate the use of additional sources of information, such as document metadata, domain-specific ontologies, or external knowledge graphs, to further refine the keyword extraction process and improve the accuracy of the consensus-based approach.

Evaluation metrics: Finally, future research can develop more comprehensive and domain-specific evaluation metrics to assess the effectiveness and robustness of the consensus-based approach for keyword extraction. These metrics should take into account the different aspects of keyword extraction, such as coverage, coherence, and relevance, and should be validated on large and diverse datasets from different domains and text types.

Overall, our results demonstrate the potential of the consensus-based approach as a practical and scalable solution for multi-document summarization of large and diverse information sources.

BIBLIOGRAPHY

[1] Gupta, A., Chugh, D., Anjum, Katarya, R. (2022, March 31st). "Automated News Summarization Using Transformers". In: Aurelia, S., Hiremath, S.S., Subramanian, K., Biswas, S.K. (eds) Sustainable Advanced Computing. Lecture Notes in Electrical Engineering, *Conference paper First Online:Part of the Lecture Notes in Electrical Engineering book series*, vol 840. Springer, Singapore. https://doi.org/10.1007/978-981-16-901

[2] AYESHA AYUB SYED1, FORD LUMBAN GAOL 1, (Senior Member, IEEE), AND TOKURO MATSUO2,3, (Member, IEEE),1Department of Doctor of Computer Science, Bina Nusantara University, Jakarta 11480, Indonesia ,2 Graduate School of Industrial Technology, Advanced Institute of Industrial Technology, Tokyo 140-0011, Japan, 3 Department of M-Commerce and Multimedia Applications, Asia University, Taichung 41354, Taiwan, "A Survey of the State-of-the-Art Models in Neural Abstractive Text Summarization," IEEE Received December 17, 2020, accepted January 3, 2021, date of publication January 19, 2021, date of current version January 25, 2021.

[3] Jean Louis Ebongue Kedieng Fendji 1,*, Désiré Manuel Taira 2, Marcellin Atemkeng 3, * and Adam Musa Ali 2 ,1 Department of Computer Engineering, University Institute of Technology, University of Ngaoundere, Ngaoundere P.O. Box 454, Cameroon ,2 Department of Mathematics and Computer Science, Faculty of Science, University of Ngaoundere, Ngaoundere P.O. Box 454, Cameroon,3 Department of Mathematics, Rhodes University, Grahamstown 6140, South Africa, "WATS-SMS: A T5-Based French Wikipedia Abstractive Text Summarizer for SMS", *Future Internet 2021, 13, 238. https://doi.org/10.3390/fi13090238 www.mdpi.com/journal/futureinternet*

[4] SAI TEJA P, SHAIK AMEEN, G RAVI VARMA, M MOUNISHA Under esteemed guidance of Dr. K. Selvani Deepthi Associate Professor, DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY & SCIENCES , "TEXT SUMMARIZATION USING NEURAL NETWORK ",*SANGIVALASA, Bheemili mandal VISAKHAPATNAM – 531162 MARCH 2020*

[5] Divakar Yadav,1 Naman Lalit,1Riya Kaushik,1Yogendra Singh,1 Mohit,1 Dinesh,1 Arun Kr. Yadav,1 Kishor V. Bhadane,2Adarsh Kumar,3and Baseem Khan4,1Department of Computer Science and Engineering, NIT Hamirpur (HP), Hamirpur, India, 2 Amrutvahini College of Engineering Sangamner, Ghulewadi, Maharashtra, India, 3Department of Systemics, School of Computer Sciences, UPES, Dehradun, India, 4 Department of Electrical and Computer Engineering, Institute of Technology, Hawassa University, Hawassa, Ethiopia,"Qualitative Analysis of Text Summarization Techniques and Its Applications in Health Domain", *Hindawi Computational Intelligence and Neuroscience, Volume 2022| Article ID 3411881|14 pages https://doi.org/10.1155/2022/3411881*

[6] Ankit Kumar ,Zixin Luo, Ming Xu, "Text Summarization using Natural Language Processing," WPI, Juniper Networks ,March 2rd, 2018.

[7] Federico Barrios1, Federico L´opez1, Luis Argerich1, Rosita Wachenchauzer12, 1 Facultad de Ingenier´ıa, Universidad de Buenos Aires, Ciudad Aut´onoma de Buenos Aires, Argentina., 2 Universidad Nacional de Tres de Febrero, Caseros, Argentina, "Variations of the Similarity Function of TextRank for Automated Summarization," *arXiv:1602.03606v1 [cs.CL], 11 Feb 2016.*

[8] Muthiah, Kaarthic (2020), "Automatic Coherent and Concise Text Summarization using Natural Language Processing." *Masters thesis, Dublin, National College of Ireland*

[9] Chirantana Mallick ,Ajit Kumar Das ,Madhurima Dutta, Asit Kumar, Das and Apurba Sarkar, "Graph-Based Text Summarization Using Modified TextRank" (eds) Soft Computing in Data Analytics . *Advances in Intelligent Systems and Computing, vol 758. Springer, Singapore. https://doi.org/10.1007/978-981-13-0514-6_14 , (2019).*

[10] Makbule Gulcin Ozsoy and Ferda Nur Alpaslan Department of Computer Engineering, Middle East Technical University, Turkey Ilyas Cicekli Department of Computer Engineering, Hacettepe University, Turkey,"Text summarization using Latent Semantic Analysis",*Journal of Information Science 37(4) 405–417 ,The Author(s) 2011 Reprints and permission: sagepub. co.uk/journalsPermissions.nav DOI: 10.1177/0165551511408848 jis.sagepub.com*

[11] Aarti Patil Sinhgad Institute of Technology & Sciences, Narhe, Pune ,Komal Pharande Sinhgad Institute of Technology & Sciences, Narhe, Pune ,Dipali Nale Sinhgad Institute of Technology & Sciences, Narhe, Pune ,Roshani Agrawal Sinhgad Institute of Technology & Sciences, Narhe, Pune, "Automatic Text Summarization," *International Journal of Computer Applications (0975 – 8887) Volume 109 – No. 17 , January 2015.*

[12]Mr.S.A.Babar,M.Tech-CSE,RIT, "Text Summarization:An Overview," published in www.researchgate.net/publication, on 21 May 2014

[13] TalBaumel Dept. of Computer Science Ben-Gurion University Beer-Sheva, Israel,MatanEyal Dept. of Computer Science Ben-Gurion University Beer-Sheva, Israel,MichaelElhadad Dept. of Computer Science Ben-Gurion University Beer-Sheva, Israel ,"Query Focused Abstractive Summarization: Incorporating Query Relevance, Multi-Document Coverage, and Summary Length Constraints into seq2seqModels", *arXiv:1801.07704v2 [cs.CL],25 Jan 2018*

[14] Hritvik Gupta,5th Sem B. Tech, Computer science and Engineering, Geetanjali institute of technical studies Udaipur, Rajasthan, Mayank Patel Associate Professor, Computer science and Engineering,Geetanjali institute of technical studies Udaipur, Rajasthan,"METHOD OF TEXT SUMMARIZATION USING LSA AND SENTENCE BASED TOPIC MODELLING WITH BERT ",*2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS) | 978-1-7281-9537-7/20/$31.00 ©2021 IEEE| DOI: 10.1109/ICAIS50930.2021.9395976*

[15] Jaishree Ranganathan, Gloria Abuka Department of Computer Science Middle Tennessee State University Murfreesboro, TN, USA ,"Text Summarization using Transformer Model",*2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS) | 979-8-3503-2048-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/SNAMS58071.2022.10062698*

[16] Spandan Patil1, Lokshana Chavan2, Janhvi Mukane3, Dr. Deepali Vora4, Prof. Vidya Chitre5 Department of Information Technology, Vidyalankar Institute of Technology, Mumbai, India1, 2, 3, 5 Department of Information Technology, Symbiosis Institute of Technology, Pune, India4 ,"State-of-the-Art Approach to e-Learning with Cutting Edge NLP Transformers: Implementing Text Summarization, Question and Distractor Generation, Question Answering",*(IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 13, No. 1, 2022*

[17] Aman Burman1 and Eric Bradford2,3 , 1Dubai College 2 Technical Product Manager at Apple , 3 Masters of Engineering from Massachusetts Institute of Technology, "Building an Optimized algorithm that provides summaries of legal documents",

"https://www.researchgate.net/publication/365294672_Building_an_Optimized_algorithm_that_provides_summaries_of_legal_documents". November 2022

[18] Soma Shrenikaa, GouriPriya Raminia, Lunsavath Bhagavathi Devia, B Geetavanib a Under Graduate Student, Computer Science and Engineering, Institute of Aeronautical Engineering, Hyderabad, India, bAssistant Professor, Computer Science and Engineering, Institute of Aeronautical Engineering, Hyderabad, India."Abstractive Text Summarization By Using Deep Learning Model",*Turkish Journal of Computer and Mathematics Education Vol.12 No.14 (2021), 2404– 2411*

[19] C.R.Dhivyaa Assistant Professor Computer Science and Engineering Kongu Engineering College Perundurai, India ,K.Nithya Assistant Professor Computer Science and Engineering Kongu Engineering College Perundurai, India,T. Janani Assistant professor Information Technology Bannari Amman Institute of Technology,Sathyamangalam,India, K. Sathis Kumar Assistant professor grade II Computer Science and Engineering Bannari Amman Institute of Technology Sathyamangalam,India,N.Prashanth PG Scholar, Computer Technology-PG Kongu Engineering College Perundurai, India,"Transliteration based Generative Pre-trained Transformer 2 Model for Tamil Text Summarization", *2022 International Conference on Computer Communication and Informatics (ICCCI), Jan. 25 – 27, 2022, Coimbatore, INDIA | 978-1-6654-8035-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCCI54379.2022.9740991*

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova,"BERT:Pre-training of Deep Bidirectional Transformers for Language Understanding" , arXiv:1810.04805v2 [cs.CL] 24 May 2019

[21] Varun Deokar1, Kanishk Shah2 ,1Student, Dept. of Information Technology, Vidyalankar Institute of Technology, Maharashtra, India ,2Student, Dept. of Computer Science, D.J. Sanghvi College of Engineering, Maharashtra, India ,"Automated Text Summarization of News Articles ",*International Research Journal of Engineering and Technology (IRJET),ISO 9001:2008 Certified Journal | Volume: 08 Issue: 09 | Sep 2021*

[22] Farrel Octavianus,Computer Science Department, School of Computer Science, Bina Nusantara University Jakarta, Indonesia, Derwin Suhartono Computer Science Department, School of Computer Science, Bina Nusantara University Jakarta, Indonesia,Albert Wihardi Computer Science Department, School of Computer Science, Bina Nusantara University Jakarta, Indonesia albert,Muhamad Keenan Ario Computer Science Department, School of Computer Science, Bina Nusantara University Jakarta, Indonesia ,"Automated Text Summarization and Topic Detection on News Aggregation System Using BART and SVM ",*2022 International Symposium on Information Technology and Digital Innovation (ISITDI) | 978-1-6654-6116-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISITDI55734.2022.9944521*

[23] Babar S, Tech-Cse M, Rit (2013) Text Summarization:An Overview

[24] Christian H, Agus M, Suhartono D (2016) Single Document Automatic Text Summarization using Term Frequency-Inverse Document Frequency (TFIDF). ComTech: Computer, Mathematics and Engineering Applications 7:285 . https://doi.org/10.21512/comtech.v7i4.3746

[25] Nomoto T (2005) Bayesian Learning in Text Summarization

[26] Graves A (2013) Generating Sequences With Recurrent Neural Networks. CoRR abs/1308.0850:

[27] Nallapati R, Xiang B, Zhou B (2016) Sequence-to-Sequence RNNs for Text Summarization. CoRR abs/1602.06023:

[28] Hochreiter S, Schmidhuber J (1997) Long Short-Term Memory. Neural Comput 9:1735–1780 https://doi.org/10.1162/neco.1997.9.8.1735

[29] Shi T, Keneshloo Y, Ramakrishnan N, Reddy CK (2018) Neural Abstractive Text Summarization with Sequence-to-Sequence Models. CoRR abs/1812.02303:

[30] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is All you Need. ArXiv abs/1706.03762:

[31] Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, Cistac P, Rault T, Louf R, Funtowicz M, Brew J (2019) HuggingFace's Transformers: State-ofthe-art Natural Language Processing. CoRR abs/1910.03771:

[32] Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu PJ (2019) Exploring the Limits of Transfer Learning with a Unified Textto-Text Transformer. CoRR abs/1910.10683:

[33] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H, He Q (2021) A Comprehensive Survey on Transfer Learning. Proceedings of the IEEE 109:43–76 . https://doi.org/10.1109/JPROC.2020.3004555

[34] Devlin J, Chang M-W, Lee K, Toutanova K (2018) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. CoRR abs/1810.04805:

[35] Radford A (2018) Improving Language Understanding by Generative PreTraining

[36] Lewis M, Liu Y, Goyal N, Ghazvininejad M, Mohamed A, Levy O, Stoyanov V, Zettlemoyer L (2019) BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. CoRR abs/1910.13461:

[37] J. Devlin, M. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., 2019, pp. 4171–4186.

[38] A. Radford, ''Improving language understanding by generative pretraining,'' OpenAI J., to be published.

[39] K. Jing and J. Xu, ''A survey on neural network language models,'' 2019, arXiv:1906.03591. [Online]. Available: http://arxiv.org/abs/1906. 03591

[40] Y.Bengio,R.Ducharme,P.Vincent,andC.Jauvin,''A Neural Probabilistic language model,'' J. Mach. Learn. Res., vol. 3, pp. 1137–1155, Feb. 2003, doi: 10.1162/153244303322533223.

[41] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ''Efficient estimation of word representations in vector space,''in Proc. 1st Int. Conf. Learn. Represent. (ICLR), 2013, pp. 1–12.

[42] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, ''Pre-trained models for natural language processing: A survey,'' Sci. China Technol. Sci., vol. 63, no. 10, pp. 1872–1897, Oct. 2020, doi: 10.1007/s11431-0201647-3.

[43] S. Song, H. Huang, and T. Ruan, ''Abstractive text summarization using LSTM-CNNbaseddeeplearning,''*Multimedia Tools Appl.,vol.78,no.1, pp. 857–875, Jan. 2019, doi: 10.1007/s11042-018-5749-3.*

[44] J. L. Elman, ''Finding structure in time,'' *Cognit. Sci., vol. 14, no. 2, pp. 179–211, Mar. 1990, doi: 10.1016/0364-0213(90)90002-E.*

[45] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk,andY.Bengio,''Learning phrase representations using RNN encoder–decoder for statistical machine translation,'' in *Proc.Conf.Empirical Methods Natural Lang. Process.(EMNLP),2014,pp. 1724–1734,doi: 10.3115/v1/d14-1179.*

[46] D. Bahdanau, K. Cho, and Y. Bengio, ''Neural machine translation by jointly learning to align and translate,'' *in Proc. 3rd Int. Conf. Learn. Represent., ICLR, 2015, pp. 1–15.*

[47] S. Hochreiter and J. Schmidhuber, ''Long short-term memory,'' *Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/ neco.1997.9.8.1735.*

[48] I. Sutskever, O. Vinyals, and Q. V. Le, ‘‘Sequence to sequence learning with neural networks,’’ in *Proc. Adv. Neural Inf. Process. Syst., vol. 4, Sep. 2014, pp. 3104–3112. [Online]. Available: https://proceedings. neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2Paper.pdfv*

[49] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, ‘‘Empirical evaluation of gated recurrent neural networks on sequence modeling,’’ *in Proc. NIPS Workshop Deep Learn., 2014.*

[50] M.SchusterandK.K.Paliwal,‘‘Bidirectional recurrent neural networks,’’ *IEEE Trans. Signal Process., vol. 45, no. 11, pp. 2673–2681, 1997.*

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, ‘‘Attention is all you need,’’ in *Proc. Adv. Neural Inf. Process. Syst., NIPS, 2017, pp. 5998–6008. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/ 3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf*

[52] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, ‘‘BART: Denoising sequence-tosequence pre-training for natural language generation, translation, and comprehension,’’*inProc.58thAnnu.MeetingAssoc.Comput.Linguistics, 2020, doi: 10.18653/v1/2020.acl-main.703.*

[53] J. Gu, Z. Lu, H. Li, and V. O. K. Li, ‘‘Incorporating copying mechanism in sequence-to-sequence learning,’’ in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics (Long Papers), vol. 1, 2016, pp. 1631–1640, doi: 10.18653/v1/p16-1154.*

[54] A. See, P. J. Liu, and C. D. Manning, ‘‘Get to the point: Summarization with pointer-generator networks,’’ in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics (Long Papers), vol. 1, 2017, pp. 1073–1083, doi: 10.18653/v1/P17-1099.*

[55] Q.Chen,X.Zhu,Z.Ling,S.Wei,andH.Jiang,‘‘Distraction-based neural networks for document summarization,’’in*Proc.25thInt.JointConf.Artif. Intell., 2016. [Online]. Available: https://arxiv.org/abs/1610.08462*

[56] Luhn, H.P. The automatic creation of literature abstracts. *IBM J. Res. Dev. 1958, 2, 159–165.*

[57] Edmundson, H.P. New methods in automatic extracting. *J. ACM JACM 1969, 16, 264–285.*

[58] Christian, H.; Agus, M.P.; Suhartono, D. Single document automatic text summarization using term frequency-inverse document frequency (TF-IDF). *ComTech Comput. Math. Eng. Appl. 2016, 7, 285–294.*

[59] Sarkar, K. Automatic single document text summarization using key concepts in documents. J. *Inf. Process. Syst. 2013, 9, 602–620.*

[60] Erkan, G.; Radev, D. Lexpagerank: Prestige in multi-document text summarization. *In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain, 25–26 July 2004; pp. 365–371.*

[61] Uçkan, T.; Karcı, A. Extractive multi-document text summarization based on graph independent sets. *Egypt. Inform. J. 2020, 21, 145–157.*

[62] Patel, A.; Siddiqui, T.; Tiwary, U.S. A language independent approach to multilingual text summarization. *Large Scale Semant. Access Content 2007, 123–132.*

[63] Radev, D.R.; Allison, T.; Blair-Goldensohn, S.; Blitzer, J.; Celebi, A.; Dimitrov, S.; Drabek, E.; Hakim, A.; Lam, W.; Liu, D.; et al. MEAD-A Platform for Multidocument Multilingual Text Summarization; *European Language Resources Association (ELRA): Lisbon, Portugal, 2004.*

[64] Pontes, E.L.; González-Gallardo, C.-E.; Torres-Moreno, J.-M.; Huet, S. Cross-lingual speech-to-text summarization. *In Proceedings of the International Conference on Multimedia and Network Information System; Springer: Berlin/Heidelberg, Germany, 2018; pp. 385–395.*

[65] Zhu, J.; Wang, Q.; Wang, Y.; Zhou, Y.; Zhang, J.; Wang, S.; Zong, C. NCLS: Neural cross-lingual summarization. *arXiv 2019, arXiv:190900156.*

[66] Moratanch, N.; Chitrakala, S. A survey on extractive text summarization. *In Proceedings of the 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), Chennai, India, 10–11 January 2017; pp. 1–6.*

[67] Moratanch, N.; Chitrakala, S. A survey on abstractive text summarization. *In Proceedings of the 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), Nagercoil, India, 18–19 March 2016; pp. 1–7.*

[68] Hou, L.; Hu, P.; Bei, C. Abstractive document summarization via neural model with joint attention. *In Proceedings of theNational CCF Conference on Natural Language Processing and Chinese Computing; Springer: Berlin/Heidelberg, Germany, 2017; pp. 329–338.*

[69] Wang, S.; Zhao, X.; Li, B.; Ge, B.; Tang, D. Integrating extractive and abstractive models for long text summarization. *In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 305–312*

[70] Steinberger, J.; Poesio, M.; Kabadjov, M.A.; Ježek, K. Two uses of anaphora resolution in summarization. *Inf. Process. Manag. 2007, 43, 1663–1680.*

[71] Bhargava, R.; Sharma, Y. MSATS: Multilingual sentiment analysis via text summarization. *In Proceedings of the 2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence, Noida, India, 12–13 January 2017; pp. 71–76*

[72] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In AdvancesinNeuralInformationProcessingSystems; *2017; pp. 5998–6008. Available online: https://arxiv.org/pdf/1706.03762.pdf (accessed on 15 September 2021).*

[73] Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv 2018, arXiv:181004805.*

[74] Martin, L.; Muller, B.; Suárez, P.J.O.; Dupont, Y.; Romary, L.; de La Clergerie, É.V.; Seddah, D.; Sagot, B. Camembert: A tasty french language model. *arXiv 2019, arXiv:191103894.*

[75] Le, H.; Vial, L.; Frej, J.; Segonne, V.; Coavoux, M.; Lecouteux, B.; Allauzen, A.; Crabbé, B.; Besacier, L.; Schwab, D. Flaubert: Unsupervised language model pre-training for french. *arXiv 2019, arXiv:191205372.*

[76] Liu, Y.; Lapata, M. Text summarization with pretrained encoders. *arXiv 2019, arXiv:190808345*.

[77] Liu, Y. Fine-tune BERT for extractive summarization. *arXiv 2019, arXiv:190310318.*

[78] Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv 2019, arXiv:191010683.*

[79] K.Selvani Deepthi, Dr.Radhika Y(2015),"Extractive Text Summarization using Modified Weighing and Sentence Symmetric Feature Methods", in an International Journal of Modern Education and Computer Science, ISSN: 2075-0161 Volume 7 No-10 pp: 33-39, October 2015.

[80] N.RaneandS.Govilkar,''Recenttrendsindeeplearningbasedabstractive text summarization,'' Int. J. Recent Technol. Eng., vol. 8, no. 3, Sep. 2019.

[81] Valverde Tohalino, Jorge & Amancio, Diego. (2017). Extractive Multidocument Summarization Using Multilayer Networks. Physica A: Statistical Mechanics and its Applications. 503. 10.1016/j.physa.2018.03.013.

[82] J. N. Madhuri and R. Ganesh Kumar, "Extractive Text Summarization Using Sentence Ranking," 2019 International Conference on Data Science and Communication (IconDSC), Bangalore, India, 2019, pp.13.doi:

[83] 1109/IconDSC.2019.8817040 8. N. S. Shirwandkar and S. Kulkarni, "Extractive Text Summarization Using Deep Learning," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 2018, pp. 1-5. doi: 10.1109/ICCUBEA.2018.8697465

[84] Azar, Mahmood & Hamey, Len. (2016). Text Summarization Using Unsupervised Deep Learning. Expert Systems with Applications. 68. 10.1016/j.eswa.2016.10.017.

[85] N.M.Abdelaleem,H.A.Kader,andR.Salem,"Abriefsurvey on text summarization techniques," IJ of Electronics and Information Engineering, vol. 10, no. 2, pp. 103–116, 2019.

[86] A. Elrefaiy, A. R. Abas, and I. Elhenawy, "Review of recent techniques for extractive text summarization," Journal of Theoretical and Applied Information Technology, vol. 96, no. 23, pp. 7739–7759, 2018.

[87] A. Sinha, A. Yadav, and A. Gahlot, "Extractive text summarization using neural networks," 2018, https://arxiv.org/ abs/1802.10137.

[88] A. Nenkova and K. McKeown, "A survey of text summarizationtechniques,"inMiningTextData,pp.43–76,Springer, Boston, MA, USA, 2012.

[89] N. Vanetik, M. Litvak, E. Churkin, and M. Last, "An unsupervised constrained optimization approach to compressive summarization," Information Sciences, vol. 509, pp. 22–35, 2020.

[90] R. A. Garc´ıa-Herna´ndez and Y. Ledeneva, "Single extractive textsummarizationbasedonageneticalgorithm,"inMexican Conference on Pattern Recognition, pp. 374–383, Springer, Berlin, Germany, 2013.

[91] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: a comprehensivesurvey,"ExpertSystemswithApplications,vol.165, Article ID 113679, 2021.

[92] C.Khatri,G.Singh,andN.Parikh,"Abstractiveandextractive text summarization using document context vector and recurrent neural networks," 2018, https://arxiv.org/abs/1807. 08000. [6] S. Singla, N. Duhan, and U. Kalkal, "A novel approach for document ranking in digital libraries using extractive summarization," International Journal of Computer Applications, vol. 74, no. 18, pp. 25–31, 2013.

[93] R.Nallapati,B.Zhou,C.Gulcehre,andB.Xiang,"Abstractive text summarization using sequence-to-sequence rnns and beyond," 2016, https://arxiv.org/abs/1602.06023.

[94] Mihalcea, R., Tarau, P.: Textrank: Bringing order into texts. In: Lin, D., Wu, D. (eds.) Proceedings of EMNLP 2004. pp. 404–411. Association for Computational Linguistics, Barcelona, Spain (July 2004)

[95] Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. In: Proceedings of the 7th International World Wide Web Conference. pp. 161–172. Brisbane, Australia (1998), citeseer.nj.nec.com/ page98pagerank.html

[96] Beautifulsoup documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (2017). Accessed 30 Nov 2017

[97] Python 2.7.14 documentation. https://docs.python.org/2/index.html (2017). Accessed 30 Nov 2017

[98] Mihalcea, R.: Graph-based ranking algorithms for sentence extraction, applied to text summarization. In: Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, pp. 20. Association for Computational Linguistics (2004)

[99] Ozsoy MG, Cicekli I, Alpaslan FN. Text summarization of Turkish texts using Latent Semantic Analysis. In: Proceedings of the 23rd international conference on computational linguistics (Coling 2010) 2010: 869–876.

[100] Gong Y, Liu X. Generic text summarization using relevance measure and latent semantic analysis. In: Proceedings of SIGIR '01 2001.

[101] Steinberger J, Jezek K. Using Latent Semantic Analysis in text summarization and summary evaluation. In: Proceedings of ISIM '04 2004: 93–100.

[102] Colin Raffel, Noam Shazeer, et, al, "Exploring the Limits of Transfer Learning with Unified Text-to-Text Transformer", Journal of Machine Learning Research 21(2020), June 2020, arXiv:1910.10683.

[103] Elisa Terumi Rubel Schneider, Joao Vitor Andrioli de Souza, Yohan Bonescki Gumiel, Claudia Moro and Emerson Cabrera Paraiso, "A GPT-2 Language Model for Biomedical Texts in Portuguese", International Symposium on Computer-Based Medical Systems, 2021

[104] M. Lewis et al., "BART: Denoising Sequence-to-Sequence Pretraining for Natural Language Generation, Translation, and Comprehension," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 7871–7880, doi: 10.18653/v1/2020.acl-main.703

[105] S. Singla, N. Duhan, and U. Kalkal, "A novel approach for document ranking in digital libraries using extractive summarization," International Journal of Computer Applications, vol. 74, no. 18, pp. 25–31, 2013.

[106] Christopher D. Manning, Prabhakar Raghavan, H.S.: Introduction to Information Retrieval. Cambridge University Press (2008)

[107] Murray G, Renals S, Carletta J. Extractive summarization of meeting recordings. In: Proceedings of the 9th European conference on speech communication and technology 2005.

[108] Jacob Devlin, Ming-Wei Chang, et, al, "BERT: Pre-Training of Deep Bidirectional Transformer for Language Understanding", Google AI Language, May 2019, arXiv:1810.04805.