Abstractive and extractive based YouTube transcript summarization: a hybrid approach

Naidila Sadashiv[1,3], Aneesha Krishna Maiya[2,3], Geetha Shivareddy[2,3], Akash Reddy[2,3]
[1]Department of Information Science and Engineering, JSS Academy of Technical Education, Bengaluru, India
[2]Department of Computer Science and Engineering, JSS Academy of Technical Education, Bengaluru, India
[3]Visvesvaraya Technological University, Jnana Sangama, Belagavi, India

Article Info

Article history:

Received Nov 22, 2024
Revised Jul 6, 2025
Accepted Oct 15, 2025

Keywords:

Abstractive summarization
Extractive summarization
K-Means clustering algorithm
Natural language processing
Pre-trained models
ROUGE scores
Transformers

ABSTRACT

The rapid advancement in the field of communication and ubiquitous access to computing has led to the proliferation of large amounts of video content on YouTube and other social media platforms. However, getting precise information from the video in concise textual manner remains a challenge. Different extractive and abstractive text summarization methods are prevalent in the literature. In this paper, classical extractive text summarization methods Luhn's algorithm, TextRank algorithm and Keyword- based summarization are combined to develop a combined extractive (CE) method. To enhance its performance, bidirectional and auto-regressive transformers (BART) is investigated and integrated as a hybrid model. Further, we explore how K-means clustering algorithm can be used for text summarization in general and with the proposed hybrid approach for improvement in text summarization. Using CNN/DailyMail dataset, assessment of text summarization methods based on ROUGE scores and time taken for summary generation is carried out. Based on the ROUGE score, we observe that the proposed hybrid method - 0.2644 is better than traditional extractive summarization methods. The combination of hybrid method with K-means further improved the score to 0.3227. The time taken by them for summary generation are 138.09 and 142.16 seconds respectively. This work experimented with different classical and transformer-based text summarization techniques to explore the complementary aspects and the results obtained are comparable with that of existing models with less time for text summarization.

Corresponding Author:

Naidila Sadashiv
Department of Information Science and Engineering, JSS Academy of Technical Education
Bengaluru, India
Email: naidila@jssateb.ac.in

1. INTRODUCTION

YouTube is a social networking and online video sharing platform having millions of videos related to a wide range of categories such as education, entertainment, health domain and many more. The vast amount of data available in YouTube makes it difficult for its users to manually filter different videos to find useful information. This necessitates techniques for video summarization that can provide concise information in space and time efficient manner retaining the semantics of the information. However, the dynamic and diverse nature of video data makes video transcript summarization very challenging. YouTube provides video transcripts and are explored to improve the process of video summarization.

In the literature, the approaches for text summarization are mainly categorized as extractive summarization and abstractive summarization techniques. Extractive summarization involves selecting phrases and sentences directly from the original transcript. The sentences are scored based on considering parameters such as term frequency, relevance, position of text and rearrange in a meaningful way to create a summary. Abstractive summarization produces a summary containing new phrases and sentences that are capable of representing key information of the video. Various abstractive summarization techniques have been developed recently such as sequence-to-sequence models using recurrent neural network (RNN), long short-term memory (LSTM) [1], pointer-generator networks [2], and transformer-based models like bidirectional encoder representations from transformers (BERT) and GPT models. The literature survey provides methods and models that are available to summarize the transcript [3]-[6]. The primary emphasis of these models is regarding the readability, understandability and grammar of the generated summary.

Despite significant advancement in video transcript summarization, due to the proliferation of vast amounts of dynamic video content, efficient transcript summarization is still a challenge. Video transcript summarization is under explored in many of the domains and applications. There exists good scope of its applications in various domains that includes generating short notes of class lectures in education field to assistance for the review of medical procedure recordings at large more quickly.

To address these challenges the work in this paper explores the classical and transformer-based model and proposes a hybrid video summarization framework that combines the strengths of extractive and abstractive techniques. By integrating the classical and transformer-based model the current work aims to get the best of these models together to improve the efficiency and informativeness of generated summaries. The proposed work also includes an evaluation of these models, discussion and comparative analysis of the models with the existing models as this is essential for decision making during model selection in specific use cases.

There exists a good amount of literature work on text summarization using classical approaches using extractive and abstractive methods. Luhn [7] proposed sentence extraction approach based on the word frequency and this work has made its way to many text summarization methods. In 2004 a graph-based ranking named “TextRank was proposed by Mihalcea and Tarau [8]. It is similar to the PageRank algorithm developed by Google to rank web pages. A graph is created, where nodes represent sentences and edges between nodes are undirected and weighted using metrics like cosine similarity or Jaccard similarity. Sentences were selected based on the weights between two nodes. Babar and Patil [9], the authors employ two distinct methodologies to create two types of summaries. In the first approach, eight different sentence features are used to score the sentence and later the sentences are ranked using fuzzy logic. The 2nd approach generates a summary by latent semantic analysis. The highest scored sentences from individual summaries and common sentences in both summaries are integrated to produce a final summary. Gabriela *et al.* [10] have used adjective-noun pairing and selecting sentences based on TF-IDF scores to summarize hotel reviews. The sentiment polarity is calculated to generate the text summary of the review.

Pawar *et al.* [11], have created a system which groups documents as clusters and chooses key sentences using TF/IDF, Jaccard Similarity and Euclidean distances. A compression method based on determining the shortest path in the graph generated is used to generate the final summary. A system to summarize news content based on fuzzy logic has been proposed [12]. First special keywords such as time, person’s name and location are extracted and are scored based on word features. Sentence features are used to determine significance of each sentence and suitable weights for a news article were assigned using genetic algorithm to create the final news article summary.

Tomer and Kumar [13] have proposed text summarization from multi document based on firefly algorithm in which fitness function is based on topic relation, cohesion and readability factor. The best sentence is determined by fitness function and subsequent sentences will move towards it similar to fireflies moving towards brightest firefly after a set number of repetitions of the algorithm. Veningston *et al.* [14], the authors have created a system for summarizing multiple documents based on user preference. The summary is generated by employing recursive neural network and convolutional neural network. User’s historical data is encoded using long-short term memory and is compared with the candidate summary produced by Siamese network. The combined output is fed via a multi-layered perceptron to obtain a final summary. Backpropagation is used to update the weights in each layer if the summary produced is not adequate.

Muniraj *et al.* [15] have presented the HNTSumm algorithm for news summarization based on a hybrid approach consisting of extractive and abstractive summarization methods with transliterated words. A neural embedding model trained on a dataset with transliterated words is used to generate word embeddings. In the hybrid approach, the textrank algorithm is used for extractive summarization and abstractive summarization involves a hybrid seq2seq model using bidirectional LSTM for both encoder and decoder. Liu and Lapata [16] for the first-time applied BERT for text summarization and proposed a general framework for both abstractive summarization and extractive summarization using this model. They developed a novel model named BERTSumExt for extractive summarization.

Kumar *et al.* [17] have developed a web-based tool to generate YouTube video summaries using natural language processing (NLP) and the flask framework. The tools assist in retrieving essential information and fosters YouTube video access. Shudan *et al.* [18] have presented a transformer-based system for summarizing the YouTube videos and explored its benefits while highlighting the limitations of traditional approaches. ROUGE scores were the metrics considered for evaluation of the system. Jadhav *et al.* [19] have proposed a tool for summarizing YouTube videos with the focus of regional language learning. It addresses the limitation of less summarizing tools for local language learning by leveraging NLP approaches and deep learning with sentimental analysis. Awais and Nawab [20] have focused on the Urdu language using an abstractive summarization approach using deep learning and transformer models.

Ulker and Ozer [21] have addressed the challenge of summarizing scientific articles amidst a large number of works in the research field. The authors have used scientific information extractor, encoded scientific content and have utilized graph based abstractive summarization in their proposed model. The model performance was evaluated using ROUGE scores and compared with the baseline model. On similar lines the authors in [22] have proposed T5LSTM-RNN model where the transformer model is used for preprocessing and generation of text. Ali *et al.* [23] have explored multi-document summarization (MDS) using a meta-heuristic algorithm named harmony search algorithm (HAS).

The existing work based on text summarization surveyed in the literature has contributed significantly for NLP. However, from our viewpoint they have not much experimented with the combination of extractive and abstractive approaches that experiments with classical and transformer-based models. Our experimental study aims to advance the text summarization by integrating the best of existing approaches that shall contribute to address the time and resource limitation with improved summary generation. Text summarization based on classical methods and combination of these with transformer-based models will help to explore different approaches that may lead to optimal summary generation with less time and resource overhead.

With advancements of the transformer model applied for NLP is promising, however, the limitation is it is time and resource consuming due to the complex processing. This motivates the current work to propose a hybrid model based on the combination extractive and abstractive model. In this direction in the proposed work, the research objectives of this paper are as follows: experiment the complementary aspects of extractive and abstractive approaches with transformer-based models for NLP.

a) Develop a combined extractive (CE) text summarization model and later develop an abstractive based CE hybrid model for YouTube transcript summarization.
b) Evaluate the performance of the proposed hybrid approach and compare with baseline approaches for text summarization.
c) Evaluate the performance of different variants of K-means clustering algorithms with the proposed hybrid approach for text summarization based on ROUGE scores and time metrics.

The rest of the paper is structured as follows. In section 2 discusses method and implementation of different approaches to text summarization. In section 3 discusses the results obtained for different approaches. Finally, section 4 discusses conclusion and future works.

2. METHOD

The methodology section starts with a brief introduction on preprocessing steps employed to improve the overall data quality. It is followed by discussion on the punctuation restoration model. Subsequently we present a combined-extractive algorithm combining the main features of the three extractive algorithms namely Luhn algorithm, TextRank algorithm and keyword-based method. Next, we discuss the working of hybrid approach for text summarization Finally, we end this section by explaining the working of different versions of K-means clustering algorithm.

2.1. Preprocessing

Preprocessing refers to the series of methods performed on raw text to improve the quality of the summary/text generated. The list of preprocessing steps which are applied to improve the text quality are mentioned below:

a) Cleaning text – Symbols, punctuations, special characters or any HTML tags are removed from input text.
b) Tokenization – The process of breaking the input text data as words or sentences.
c) Stopwords removal – Stopwords such as ‘and’, ‘the’, ‘is’ so on are words having no significant meaning in a sentence. These words are removed from the text data.
d) Expanding contractions –It involves expanding frequently used contraction words such as “hasn’t” to has not, “isn’t” to is not and so on.
e) Case conversion – Usually the input text data is changed to lowercase format for ensuring text homogeneity.

2.2. Punctuation restoration model

Usually, the punctuation symbols present in the transcript are removed during preprocessing, therefore it is necessary to restore the punctuation of the document before applying any extractive algorithms on it. A bert-base-uncased model (felflare/bert-restore-punctuation) provided by hugging face library is used. A single document having the entire video’s transcript is given to the model that generates the document with necessary punctuation symbols.

2.3. Extractive summarization

2.3.1. Extractive summarization using BERT

BERT is a transformer-based model developed by researchers at Google [24]. The BERT model reads the text in both the directions. Hence the model can capture context on either side of a token and hence it has a deeper and better understanding about meanings of words in different contexts. The BERT model is pre-trained on a large corpus of text using masked language modeling where the model has to predict randomly masked tokens and next sentence prediction where the model has to predict if a sentence follows another sentence among a given pair of sentences. In this paper we have explored the use of bert-extractive-summarizer to perform extractive summarization. It uses a BERT model (bert-base-uncased as default model) to generate embeddings for each sentence and then each sentence is scored based on its importance and relevance with respect to the overall context of the document. The highest ranked sentences are selected and combined together to form the summary. The parameters which can be adjusted for this model are given below.

a) body: (str) - The string which needs to be summarized.
b) ratio: (float) - The ratio of sentences needed in the final summary. We have used 0.25 as the ratio value.
c) min_length: (int) - Parameter to specify to remove sentences having less than 40 characters.
d) max_length: (int) - Parameter to specify to remove sentences greater than max length.
e) num_sentences (int) - Number of sentences to use. Overrides ratio if supplied.

2.3.2. Extractive summarization using CE method

Three different extractive summarization algorithms Luhn algorithm, TextRank algorithm and keyword-based method are combined as a single method [25] is presented in Algorithm 1 and are depicted in Figure 1. A flowchart representation of the CE method is shown in Figure 2. Here, frequency of common words found in a sentence and distance between them is used to assign scores in Luhn’s algorithm. TextRank algorithm considers the similarity between sentences while scoring sentences and keyword-based method scores sentences based on number of keywords present in a sentence. First the document containing the entire video transcript is passed to the punctuation restoration model to restore its punctuation. Different preprocessing steps like tokenization, and stopword removal are performed on punctuation restored documents. Scores are calculated using these three different extractive algorithms and we are calculating the final score of the sentence using (1).

$$S(i) = \frac{3\times S_t(i)+2\times S_k(i)+ 1\times S_l(i)}{6} \tag{1}$$

Where,
S(i) = final score of i’th sentence.
$S_t(i)$ = score of i’th sentence obtained by performing TextRank algorithm.
$S_k(i)$ = score of i’th sentence obtained by performing keyword-based method.
$S_l(i)$ = score of i’th sentence obtained through Luhn’s algorithm.
Priority for individual methods is assigned as per (1). We found that priority for TextRank, keyword-based and Luhn algorithms should be in the ratio of 2:1:3 to obtain maximum ROUGE scores.

Algorithm 1. Combined – extractive algorithm

```
Input: Punctuation restored transcript of the video
Output: Summary of the video
1)  Calculate sentence scores using Luhn algorithm => sentence_scores_luhn
2)  Calculate sentence scores using Keyword-Based method => sentence_score_keyword
3)  Calculate sentence scores using TextRank method => sentence_score_keyword
4)  Use sentence_scores_luhn, sentence_score_keyword, sentence_score_keyword to calculate
    final scores of sentences based on eq (1) => combined_sentence_score
5)  Create summary by considering n/2 sentences having highest combined_sentence_score
```

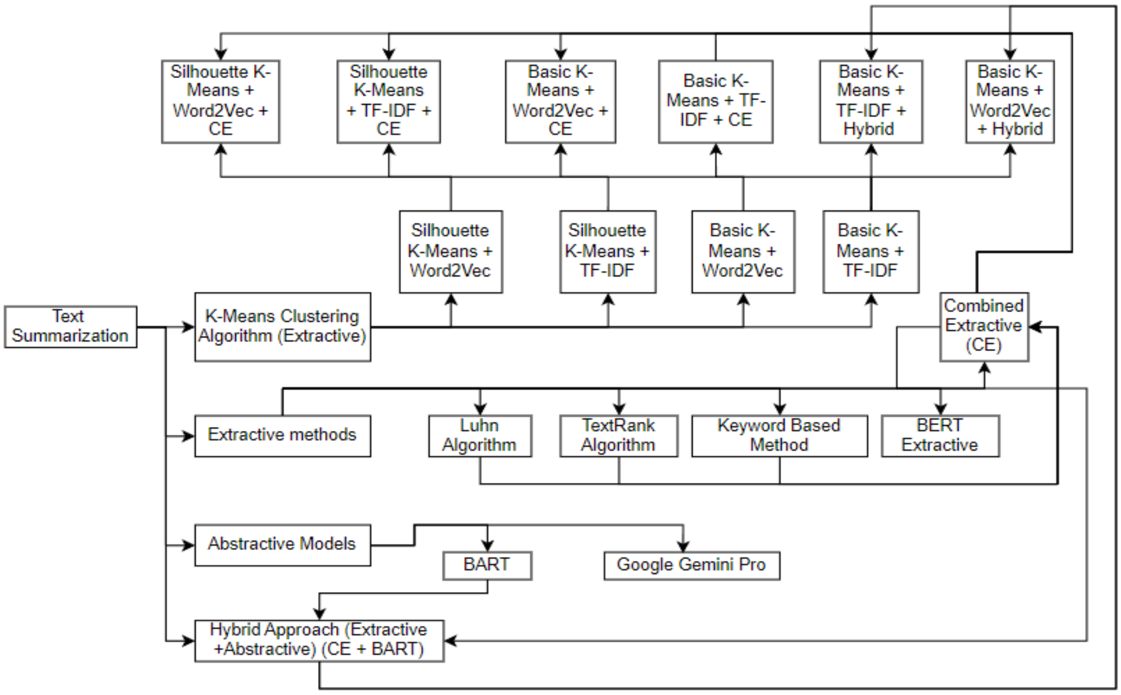

Text Summarization
K-Means Clustering Algorithm (Extractive)
Extractive methods
Abstractive Models
Hybrid Approach (Extractive +Abstractive) (CE + BART)
Silhouette K-Means + Word2Vec + CE
Silhouette K-Means + TF-IDF + CE
Basic K-Means + Word2Vec + CE
Basic K-Means + TF-IDF + CE
Basic K-Means + TF-IDF + Hybrid
Basic K-Means + Word2Vec + Hybrid
Silhouette K-Means + Word2Vec
Silhouette K-Means + TF-IDF
Basic K-Means + Word2Vec
Basic K-Means + TF-IDF
Combined Extractive (CE)
Luhn Algorithm
TextRank Algorithm
Keyword Based Method
BERT Extractive
BART
Google Gemini Pro

Figure 1. Master view of text summarization techniques used in CE and hybrid approach

2.4. Abstractive summarization

2.4.1. Abstractive summarization using bidirectional and auto-regressive transformers (BART)

BART introduced by Lewis *et al.* [26] in 2019 is a sequence-to-sequence transformer-based architecture. It is pre-trained using both bidirectional and auto-regressive objectives. During the bidirectional phase, the BART model tries to predict randomly masked tokens based on the surrounding context. In the auto-regressive phase the model tries to generate the target sequence one token at a time using the previously generated tokens. We are using the "facebook/bart-large" model and a tokenizer for this model from hugging face library. The max_length as input for this model is 1024 tokens hence the input is divided into smaller chunks and summary is produced for each chunk and later combined together to generate a complete summary.

The max_length parameter controls the maximum length of the summary generated, here we have set it as 150. The min_length parameter controls the minimum length of the summary generated and we are assigning it dynamically to 50% of the number of words in the chunk. By keeping the min_length as 50% we can ensure that the summary generated for each chunk won't be short, thus reducing the chance of missing out key points. The length_penalty parameter is used to control the length of the summary generated. Higher value for this parameter forces the model to generate shorter summary. We have set length_penalty as 2.0 to encourage model to generate a slightly longer summary for each chunk. The num_beams parameter indicates the number of beams used for beam search. A higher value for this parameter indicates the model to generate diverse summary but model may take more time to generate summaries. Here we are assigning it a value of 4 so that the model can generate a slightly diverse summary without taking too much time.

2.4.2. Abstractive summarization using Google Gemini Pro model

Google's Gemini model is an advanced artificial intelligence (AI) model developed by Google DeepMind. This model combines the strength of large language models and reinforcement learning through human feedback. It is capable of handling different tasks such as text generation, translation, summarization and question answering. The model has three different versions: Gemini Ultra, Gemini Pro and Gemini Nano. The Gemini Pro model takes the punctuation restored transcript as the input along with a text prompt indicating what the model should perform. Based on the prompt provided, the model generates the summary in a customized manner.

2.5. Text summarization using hybrid approach - combination of CE methods and abstractive

Our proposed hybrid approach involves a combination of both extractive method and abstractive model to generate the summary as shown in Figure 2. First, we perform extractive summarization using the combined-extractive algorithm described in section 2.3.2. The intermediate summary is divided into chunks and is passed as input to the BART model discussed in section 2.4.1 to perform abstractive summarization. The model returns a summary for each chunk which is combined together to generate the final summary of the video.

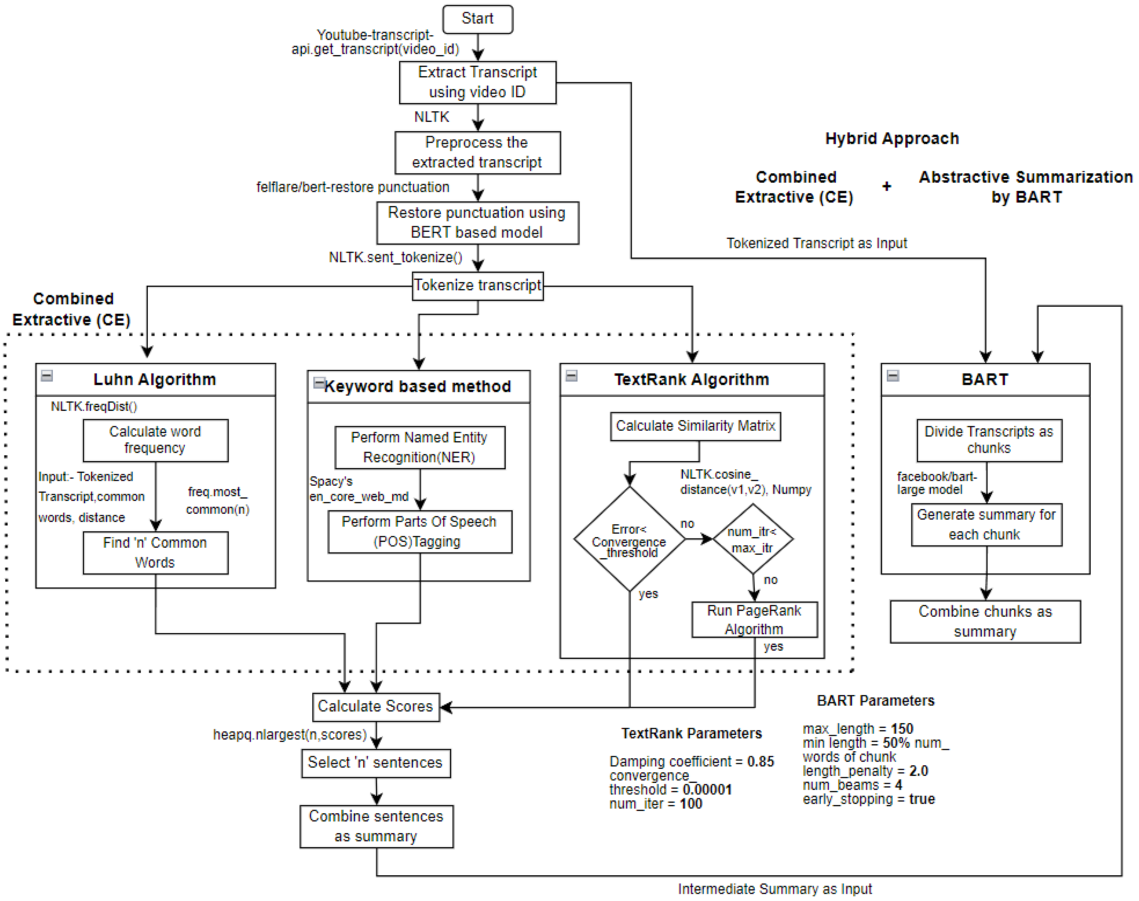

Start
Youtube-transcript-api.get_transcript(video_id)
Extract Transcript using video ID
NLTK
Preprocess the extracted transcript
felflare/bert-restore punctuation
Restore punctuation using BERT based model
NLTK.sent_tokenize()
Tokenize transcript
Hybrid Approach
Combined Extractive (CE)
+
Abstractive Summarization by BART
Tokenized Transcript as Input
Combined Extractive (CE)
Luhn Algorithm
NLTK.freqDist()
Calculate word frequency
Input:- Tokenized Transcript,common words, distance
freq.most_common(n)
Find 'n' Common Words
Keyword based method
Perform Named Entity Recognition(NER)
Spacy's en_core_web_md
Perform Parts Of Speech (POS)Tagging
TextRank Algorithm
Calculate Similarity Matrix
NLTK.cosine_distance(v1,v2), Numpy
Error< Convergence_threshold
no
num_itr< max_itr
no
yes
Run PageRank Algorithm
yes
BART
Divide Transcripts as chunks
facebook/bart-large model
Generate summary for each chunk
Combine chunks as summary
Calculate Scores
heapq.nlargest(n,scores)
Select 'n' sentences
Combine sentences as summary
TextRank Parameters
Damping coefficient = 0.85
convergence_threshold = 0.00001
num_iter = 100
BART Parameters
max_length = 150
min length = 50% num_words of chunk
length_penalty = 2.0
num_beams = 4
early_stopping = true
Intermediate Summary as Input

Figure 2. Flowchart representation of the CE method and hybrid approach

2.6. Text summarization using K-means clustering algorithm

The K-means clustering algorithm is one of the most popular algorithms used in unsupervised machine learning to group unlabeled datasets into k distinct clusters. In this paper we will be discussing 10 different variants of the K-means algorithm.

a) Basic K-means with TF-IDF vectorizer.
b) Basic K-means with Word2Vec.
c) Silhouette K-means with TF-IDF vectorizer.
d) Silhouette K-means with Word2Vec.
e) Basic K-means with TF-IDF vectorizer + CE method.
f) Basic K-means with Word2Vec + CE method.
g) Silhouette K-means with TF-IDF vectorizer + CE method.
h) Silhouette K-means with Word2Vec + CE method.
i) Basic K-means with Word2Vec model combined with hybrid approach.
j) Basic K-means with TF-IDF vectorizer combined with hybrid approach.

2.6.1 TF-IDF vectorizer

The TF-IDF is a scikit-learn tool used to convert textual data into numerical vectors which is used later to perform classification and clustering of text. Here, TF indicates the number of times a term is appearing in a document and IDF indicates how common or rare is the term in the document. The vectorizer generates a matrix of TF-IDF value for each term in the document. This data is later used during clustering.

2.6.2. Word2Vec

Word2Vec is a technique used to generate word embedding representing semantic relationships between words based on their context in a dataset. Here we are using the Word2Vec model of Gensim library to generate word vectors with K-means clustering algorithm to perform text summarization [27].

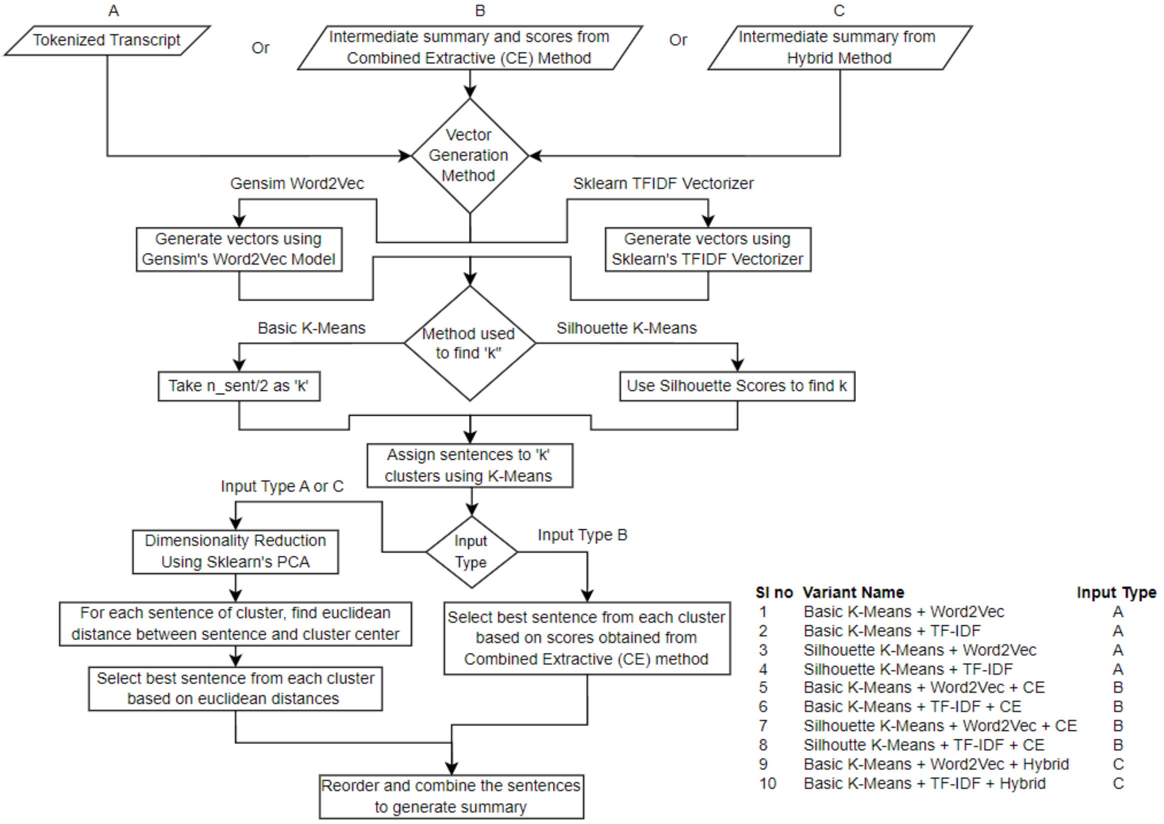
2.6.3. Silhouette score

The Silhouette score is one of the metrics which can be used to determine if the number of clusters chosen is appropriate while performing the K-means algorithm. The value of the Silhouette score lies between -1 to 1, where +1 indicates, the sample is assigned to the right cluster, 0 indicates at the boundary of the cluster and -1 is in the wrong cluster. The Silhouette score for a sample can be calculated as below:

$$s(i) = \frac{b(i) - a(i)}{(a(i)\,,b(i))} \tag{2}$$

where a(i) and b(i) denote the mean intra cluster distance and mean nearest distance of the sample ‘i’ and all other points in the cluster.

As shown in Figure. 3, input for the K-means algorithm can be a tokenized transcript or an intermediate summary generated either by CE method or by hybrid approach. Based on the approach used to convert textual data as vectors the summarization methods can be classified into two categories as Word2Vec based methods and TF-IDF vectorizer based method. These two categories are further classified into basic K-means and Silhouette K-means based on the method used to determine the number of clusters ‘k’.


A
Tokenized Transcript
Or
B
Intermediate summary and scores from Combined Extractive (CE) Method
Or
C
Intermediate summary from Hybrid Method
Vector Generation Method
Gensim Word2Vec
Sklearn TFIDF Vectorizer
Generate vectors using Gensim's Word2Vec Model
Generate vectors using Sklearn's TFIDF Vectorizer
Method used to find 'k"
Basic K-Means
Silhouette K-Means
Take n_sent/2 as 'k'
Use Silhouette Scores to find k
Assign sentences to 'k' clusters using K-Means
Input Type A or C
Input Type
Input Type B
Dimensionality Reduction Using Sklearn's PCA
For each sentence of cluster, find euclidean distance between sentence and cluster center
Select best sentence from each cluster based on euclidean distances
Select best sentence from each cluster based on scores obtained from Combined Extractive (CE) method
Reorder and combine the sentences to generate summary

Sl no	Variant Name	Input Type
1	Basic K-Means + Word2Vec	A
2	Basic K-Means + TF-IDF	A
3	Silhouette K-Means + Word2Vec	A
4	Silhouette K-Means + TF-IDF	A
5	Basic K-Means + Word2Vec + CE	B
6	Basic K-Means + TF-IDF + CE	B
7	Silhouette K-Means + Word2Vec + CE	B
8	Silhoutte K-Means + TF-IDF + CE	B
9	Basic K-Means + Word2Vec + Hybrid	C
10	Basic K-Means + TF-IDF + Hybrid	C

Figure 3. Different variants of K-means with extractive and hybrid methods for text summarization

The basic K-means variant methods simply take half the number of sentences as ‘k’ whereas the Silhouette K-means variant methods use silhouette scores to determine the optimal value of ‘k’. In Silhouette K-means it is necessary to fit the text data using K-means to calculate silhouette score. Starting from two till the number of sentences in the transcript, the data is fitted to K-means considering this number and silhouette score is calculated and stored in an array. The optimal value of ‘k’ is the index of value having maximum silhouette score + two among this entire array. As we have to fit the data every time to calculate the score, Silhouette K-means based methods generally take much more time than basic K-means based methods.

After determining the ‘k’ value the sentences are assigned to different clusters using K-means. Based on the input type, the clustering data is processed in different ways. If the input is obtained either as tokenized transcript or as intermediate summary by hybrid approach then we first perform dimensionality reduction using PCA. For each cluster, we calculate Euclidean distance between each sentence belonging to that cluster and the cluster center. One or more sentence having lower distance are selected from each cluster which are combined together to generate summary. If the input is obtained as an intermediate summary by CE method, then the sentences are selected based on the score’s parameter. The score for each sentence calculated while generating summary in CE can be passed as scores parameter to the function performing the summarization. After selection, the sentences are reordered and combined together to form a summary.

3. RESULTS AND DISCUSSION

The different text summarization methods are executed and evaluated in Google Colab notebook [28], [29]. It is generally observed that extractive summarization methods take less time to generate summary but the quality of summary generated is lower than that of abstractive models, hence we combined 3 different extractive algorithms as a single algorithm to improve the summary quality. Abstractive models such as BERT and BART take more time to generate summary but the summary quality is better than summary generated using extractive methods. To include the best of both approaches, we combined BART pre-trained model with CE method as hybrid approach and evaluated it. Additionally, we also explored how clustering algorithms such as the K-means algorithm fare against the extractive methods for text summarization tasks. We also have analyzed modern methods available for text summarization which use the AI model - Google’s Gemini models with respect to these traditional approaches. The experiments and methods were designed for summarizing YouTube video transcripts but these approaches can be extended to any other text summarization task such as news summarization, review summarization and so on.

3.1. Dataset description

The ROUGE scores were calculated for different text summarization approaches on the “cnn_dailymail” dataset from hugging face library [30]. This dataset has approximately 300k articles and corresponding highlights serving as a summary of the article.

3.2. Evaluation metrics

Recall-oriented understudy for gisting evaluation (ROUGE) [31] is one of the standard metrics used to assess the quality of the generated summary by comparing it with reference summary. There are different variants of ROUGE scores: -

a) ROUGE-1: - Calculates unigram overlap between the generated summary and the reference summary.
b) ROUGE-2: - Calculates the overlap of bigrams between generated summary and reference summary.
c) ROUGE-L: - Measures the length of the longest common subsequence present in both generated summary as well as the reference summary.

$$Rouge - N = \frac{\sum_{S\in\{RefSum\}} \sum_{gram_n \in S} Count_{match}(Gram_n)}{\sum_{S\in\{RefSum\}} \sum_{gram_n \in S} Count(Gram_n)} \tag{3}$$

where,
N: length of n-gram,
S∈{RefSum}: indicates sentence ‘S’ that belongs to reference summary (RefSum).
gram_n: refers to n-gram in the sentence and gram_n εS indicates n-grams present in sentence S.
Count (gram_n): Indicates the total number of times a specific n-gram occurs in reference summary.
Count_match (gram_n): indicates the maximum number of n-grams occurring in both candidate summary as well as reference summary.

3.3. Comparison of proposed extractive and hybrid model with abstractive and extractive models

3.3.1. Model evaluation using ROUGE scores

From Figure 4 and Table 1. we observe that among the extractive methods, summarization using BERT has a higher ROUGE score compared to combined-extractive algorithm. Among the abstractive models BART model generates a higher ROUGE scoring summary than Google Gemini Pro model. Hybrid approach has a slightly better ROUGE score when compared with extractive methods but its scores are less when compared with abstractive models. Overall, ROUGE scores are improved in CE and closer to BERT. The proposed hybrid aApproach is still better than BERT.

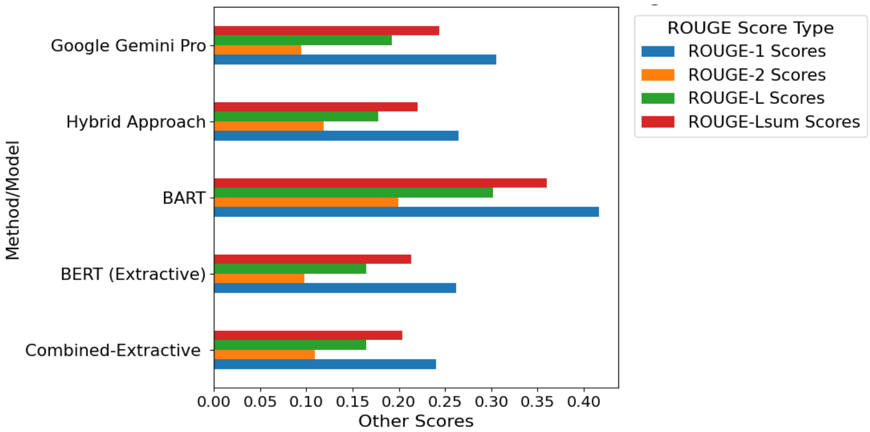


Figure 4. ROUGE scores for combined-extractive, BERT (extractive), hybrid approach, BART and Google Gemini Pro model on CNN/DailyMail dataset

Table 1. ROUGE scores for combined-extractive, BERT, hybrid approach, BART and Google Gemini Pro model on CNN/DailyMail dataset

Name of the method/model used	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
CE (Proposed)	0.2401	0.1094	0.1649	0.2040
BERT (Extractive)	0.2618	0.0976	0.1647	0.2134
BART	0.4163	0.1996	0.3017	0.3601
Hybrid (Proposed)	0.2644	0.1188	0.1776	0.2204
Google Gemini Pro	0.3051	0.0944	0.1923	0.2441

From Table 2 and Figure 5, we can infer that all the versions of the basic K-means algorithm have better ROUGE scores when compared with the Silhouette K-means algorithm. Among different basic K-means versions, basic K-means + TF-IDF + Hybrid version has the highest ROUGE-1 score (0.3227). Among different Silhouette K-means versions, Silhouette K-means + TF-IDF has the highest ROUGE-1 score (0.2648). In general, we observe that both basic K-means and Silhouette K-means versions using Word2Vec has higher ROUGE scores compared to basic K-means and Silhouette K-means versions using TF-IDF vectorizer.

Table 2. ROUGE scores for different versions of K-means algorithm on CNN/DailyMail datase

K-means version	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE LSum
Basic K-means + Word2Vec	0.2652	0.0845	0.1637	0.2154
Silhouette K-means + Word2Vec	0.2335	0.0902	0.1437	0.1875
Basic K-means + TF-IDF	0.2751	0.1070	0.1732	0.2246
Silhouette K-means + TF-IDF	0.2648	0.1063	0.1663	0.2166
Basic K-means + TF-IDF + CE	0.2473	0.1038	0.1594	0.2041
Silhouette K-means + TF-IDF + CE	0.2324	0.102	0.1525	0.1938
Basic K-means + Word2Vec + CE	0.2828	0.1021	0.1807	0.2311
Silhouette K-means + Word2Vec + CE	0.2375	0.1073	0.1613	0.2024
Basic K-means + TF-IDF + Hybrid	0.3227	0.1314	0.2114	0.2689

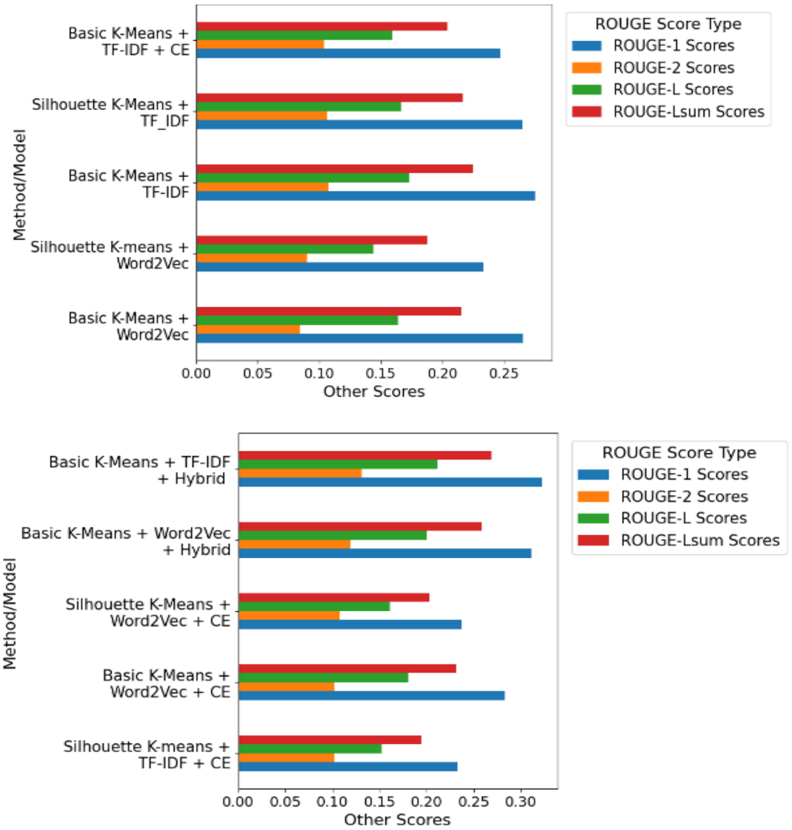


Figure 5. ROUGE scores for different versions of K-means algorithm

3.3.2. Model evaluation based on time taken

To assess the effectiveness of different text summarization approaches based on time taken to generate summary, we have considered multiple YouTube videos having transcript length ranging from 1000 to 2000 words. Based on the values in Table 3, we observe that the CE method takes a minimum amount of time to generate summary and text summarization using BERT takes significant time to generate summary as it uses a BERT based pre-trained model to generate embeddings. Similarly, the pre-trained model BART takes a maximum amount of time to generate a summary. These pre-trained models have to perform complex calculations to generate summary, hence they take more computational resources and time to generate a complete summary. The time taken by hybrid approach is slightly lower compared to pre-trained models as the intermediate summary generated through CE is already a condensed form of the transcript, hence by passing this as input to BART model, lesser number of chunks are created therefore it takes slightly lesser time for summary generation. Google Gemini Pro model generates the summary in a very short span of time when compared with pre-trained models, although it’s slightly higher than the time taken by extractive models.

Based on the values observed in Table 4, we can clearly infer that all the basic K-means versions take very less time to generate the summary when compared against Silhouette K-means versions. This is because, in Silhouette K-means versions, the clusters are initialized multiple times to determine the Silhouette score whereas in basic K-means versions the clusters are initialized only one time for each video transcript. Among basic K-means versions, basic K-means + TF-IDF takes the least amount of time and the version using Word2Vec model takes the maximum amount of time to generate a summary.

Table 3. Average time taken to generate summaries using different methods and pre-trained models

Method	Average time taken (secs)
Combined-Extractive (Proposed)	1.07
BERT (Extractive)	44.99
BART	156.50
Hybrid (Proposed)	138.09
Google Gemini Pro	6.67

Table 4. Average time taken to generate summaries using different versions of K-means algorithm

K-means version	Average time taken (secs)
Basic K-means + TF-IDF	1.54
Basic K-means + Word2Vec	1.85
CE + Basic K-means + TF-IDF	1.55
CE + Basic K-means +Word2Vec	1.65
Silhouette K-means + TF-IDF	27.40
Silhouette K-means + Word2Vec	23.08
CE + Silhouette K-means+ TF-IDF	30.61
CE + Silhouette K-means + Word2Vec	21.89
Basic K-means + Word2Vec + Hybrid	138.19
Basic K-means + TF-IDF + Hybrid	142.16

Among Silhouette K-means versions, Silhouette K-means + Word2Vec + CE takes the least amount of time and Silhouette K-means + CE + TF-IDF version takes most time to produce summary. Both the versions of basic K-means combined with hybrid approach takes a very large amount of time to generate summary as hybrid approach uses BART model in background to generate intermediate summary. In general, the time taken by basic K-means versions is slightly higher and is comparable to that of extractive algorithms. The time taken by Silhouette K-means versions is much higher when compared against extractive algorithms, but is far better than the time taken by different pre-trained models to generate summary. The Colab notebook with code regarding evaluation of these methods is made available in [28], [29].

Based on the task to perform, suitable methods can be employed. For tasks such as summarizing an educational video, user will be willing to wait for some time till the summary gets created hence in this situation we can use the hybrid approach or any of the Silhouette/Hybrid based K-means variant. For other tasks such as blog summarization or a review summarization on a website user expects the summary to be created as soon as the website loads so in this situation it is advisable to use either the CE method or any of the basic K-means variants. One of the possible methods which can be employed to reduce time constraint is parallel execution of tasks such as parallelly finding optimal ‘k’ for Silhouette K-means based variants or parallelly generating summary for multiple chunks in hybrid method. As hybrid method uses BART model which already requires higher computational resource, parallel execution of tasks might increase the time taken to generate summary hence we need to be careful before implementing this approach. In this paper we haven’t experimented with this idea yet but research can be carried out on this in future.

3.3.3. Comparative analysis with other works

To assess the performance of our hybrid and K-means based methods, we have done a comparative analysis with other works as shown in Table 5 based on the Rouge-1 scores. For this analysis we have chosen HNTSumm [12] which is similar to our hybrid approach. The authors had performed their analysis on news summary dataset. Additionally, we have also analyzed works of [32] based on TF-IDF and of [33] which is also a hybrid method using Semantic LDA with transformer-based model. From the analysis we observe that hybrid (Base) performance is almost similar to the performance of other works done. We can also observe that addition of K-means with hybrid method has further improved the performance significantly.

Table 5. Comparative analysis with other works based on ROUGE score

Method name	Dataset used	Rouge-1
Hybrid (Proposed)	CNN/DailyMail	0.2644
Basic K-means + Word2Vec + Hybrid (Proposed)	CNN/DailyMail	0.3112
Basic K-means + TF-IDF + Hybrid (Proposed)	CNN/DailyMail	0.3227
BERT (Extractive) [27]	CNN/DailyMail	0.2618
HNTSumm [12]	News Summary Dataset	0.338
Supervised-FastTextDeepMLP [21]	Konkani Language Dataset	0.3305
TF-IDF based method [35]	CNN/DailyMail	0.283
Semantic LDA [36]	WikiHow	0.2710

4. CONCLUSION

As consumption of video content increases in future with more users having access to mobile device and Internet, the need for getting key information from a video, without watching the entire video is very much significant. In this paper we have designed and developed a Hybrid approach involving combination of classical text summarization methods which includes Luhn’s algorithm, TextRank and keyword-based method with transformer-based models which can generate better summary in lesser time. We have explored YouTube transcript summarization using both abstractive models and extractive methods.

Under extractive summarization we have considered two methods: combined-extractive method and text summarization using BERT. Under abstractive summarization we have evaluated BART pre-trained model and Google Gemini Pro model. Additionally, we have explored mainly 2 different variants of the K-means clustering algorithm. (Basic K-means and Silhouette K-means). Among these 2 different versions 6 different sub-versions of basic K-means and 4 different sub-versions of Silhouette K-means are analyzed.

Based on the ROUGE score results achieved it is observed that the proposed hybrid approach and CE method have Rouge-1 scores as 0.2644 and 0.2401 respectively. Furthermore, we see significant improvement in Rouge scores when K-means is combined with the hybrid approach as indicated by Rouge-1 scores of basic K-means + Word2Vec + Hybrid variant and basic K-means + TF-IDF + Hybrid variant with scores as 0.3112 and 0.3227 respectively.

By experimentation and evaluation using ROUGE scores on CNN/dailymail dataset, we have explored the strengths and weaknesses of each approach. We believe that our work contributes to ongoing efforts to improve text summarization, providing valuable insights for future innovation in this field. These approaches can be used in education domain for providing summarized video content helping students to grasp key concepts quickly. The methods discussed in this paper can also be used for other summarization related tasks such as news summarization, blog creation from video content, review summarization and so on. As part of future work, we plan to advance the text summarization in diverse accents videos considering the noise factor. Research can also be carried out to evaluate the effectiveness of the techniques discussed in this paper using additional metrics other than ROUGE scores, such as BLEU and METEOR scores. Furthermore, research can also be extended to summarize transcripts or any other textual data from other video sharing applications to create a unified approach for video content summarization.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Naidila Sadashiv	✓	✓	✓	✓	✓	✓				✓		✓	✓	
Aneesha Krishna Maiya	✓	✓			✓	✓		✓	✓	✓	✓			
Geetha Shivareddy			✓				✓	✓						
Akash Reddy			✓				✓	✓						

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The authors confirm that the data supporting the findings of this study are available within the article.

REFERENCES

[1] R. Nallapati, B. Zhou, C. dos Santos, Ç. Gulçehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," in *CoNLL 2016 - 20th SIGNLL Conference on Computational Natural Language Learning, Proceedings*, 2016, pp. 280–290, doi: 10.18653/v1/k16-1028.

[2] A. See, P. J. Liu, and C. D. Manning, "Get to the point: summarization with pointer-generator networks," in *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 2017, vol. 1, pp. 1073–1083, doi: 10.18653/v1/P17-1099.

[3] M. F. Mridha, A. A. Lima, K. Nur, S. C. Das, M. Hasan, and M. M. Kabir, “A survey of automatic text summarization: progress, process and challenges,” *IEEE Access*, vol. 9, pp. 156043–156070, 2021, doi: 10.1109/ACCESS.2021.3129786.

[4] M. Allahyari *et al.*, “Text summarization techniques: a brief survey,” *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 10, 2017, doi: 10.14569/ijacsa.2017.081052.

[5] R. Ferreira *et al.*, “Assessing sentence scoring techniques for extractive text summarization,” *Expert Systems with Applications*, vol. 40, no. 14, pp. 5755–5764, Oct. 2013, doi: 10.1016/j.eswa.2013.04.023.

[6] S. Sharma, G. Aggarwal, and B. Kumar, “A survey on the dataset, techniques, and evaluation metric used for abstractive text summarization,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 22, no. 3, pp. 681–689, Jun. 2024, doi: 10.12928/TELKOMNIKA.v22i3.25512.

[7] H. P. Luhn, “The automatic creation of literature abstracts,” *IBM Journal of Research and Development*, vol. 2, no. 2, pp. 159–165, Apr. 2010, doi: 10.1147/rd.22.0159.

[8] R. Mihalcea and P. Tarau, “TextRank: bringing order into text,” *ACL Anthology*, 2004, [Online]. Available: https://aclanthology.org/W04-3252.

[9] S. A. Babar and P. D. Patil, “Improving performance of text summarization,” *Procedia Computer Science*, vol. 46, pp. 354–363, 2015, doi: 10.1016/j.procs.2015.02.031.

[10] N. H. Gabriela, R. Siautama, C. I. A. Amadea, and D. Suhartono, “Extractive hotel review summarization based on TF/IDF and adjective-noun pairing by considering annual sentiment trends,” *Procedia Computer Science*, vol. 179, pp. 558–565, 2021, doi: 10.1016/j.procs.2021.01.040.

[11] S. Pawar, H. M. Gururaj, and N. N. Chiplunar, “Text summarization using document and sentence clustering,” *Procedia Computer Science*, vol. 215, pp. 361–369, 2022, doi: 10.1016/j.procs.2022.12.038.

[12] Y. Du and H. Huo, “News text summarization based on multi-feature and fuzzy logic,” *IEEE Access*, vol. 8, pp. 140261–140272, 2020, doi: 10.1109/ACCESS.2020.3007763.

[13] M. Tomer and M. Kumar, “Multi-document extractive text summarization based on firefly algorithm,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6057–6065, Sep. 2022, doi: 10.1016/j.jksuci.2021.04.004.

[14] K. Veningston, P. V. V. Rao, and M. Ronalda, “Personalized multi-document text summarization using deep learning techniques,” *Procedia Computer Science*, vol. 218, pp. 1220–1228, 2022, doi: 10.1016/j.procs.2023.01.100.

[15] P. Muniraj, K. R. Sabarmathi, R. Leelavathi, and S. Balaji B, “HNTSumm: Hybrid text summarization of transliterated news articles,” *International Journal of Intelligent Networks*, vol. 4, pp. 53–61, 2023, doi: 10.1016/j.ijin.2023.03.001.

[16] Y. Liu and M. Lapata, “Text summarization with pretrained encoders,” in *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, 2019, pp. 3730–3740, doi: 10.18653/v1/D19-1387.

[17] J. Kumar, R. Vashistha, R. Lal, and D. Somanir, “YouTube transcript summarizer,” in *2023 14th International Conference on Computing Communication and Networking Technologies, ICCCNT 2023*, Jul. 2023, pp. 1–4, doi: 10.1109/ICCCNT56998.2023.10308325.

[18] R. Sudhan, D. R. Vedhaviyassh, and G. Saranya, “Learning to summarize youtube videos with transformers: a multi-task approach,” in *Proceedings of the 2nd IEEE International Conference on Advances in Computing, Communication and Applied Informatics, ACCAI 2023*, May 2023, pp. 1–6, doi: 10.1109/ACCAI58221.2023.10201219.

[19] R. Jadhav, P. Damre, A. Hire, P. Gosavi, and S. Deshmukh, “YouTube video summarizer in regional language,” in *Proceedings - 2024 3rd International Conference on Sentiment Analysis and Deep Learning, ICSADL 2024*, Mar. 2024, pp. 301–305, doi: 10.1109/ICSADL61749.2024.00055.

[20] M. Awais and R. M. A. Nawab, “Abstractive text summarization for the urdu language: data and methods,” *IEEE Access*, vol. 12, pp. 61198–61210, 2024, doi: 10.1109/ACCESS.2024.3378300.

[21] M. Ulker and A. B. Ozer, “Abstractive summarization model for summarizing scientific article,” *IEEE Access*, vol. 12, pp. 91252–91262, 2024, doi: 10.1109/ACCESS.2024.3420163.

[22] S. Chaurasia, D. Dasgupta, and R. Regunathan, “T5LSTM-RNN based text summarization model for behavioral biology literature,” *Procedia Computer Science*, vol. 218, pp. 585–593, 2022, doi: 10.1016/j.procs.2023.01.040.

[23] Z. H. Ali, A. K. Hussein, H. K. Abass, and E. Fadel, “Extractive multi document summarization using harmony search algorithm,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 19, no. 1, pp. 89–95, Feb. 2021, doi: 10.12928/TELKOMNIKA.V19I1.15766.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” *arXiv.Org*, 2018, [Online]. Available: https://arxiv.org/abs/1810.04805v2.

[25] U. A. K. Maiya, V. A. Reddy, S. Geetha, and C. N. Sadashiv, “YouTube transcript summarization using abstractive and extractive approaches,” in *2nd IEEE International Conference on Advances in Information Technology, ICAIT 2024 - Proceedings*, Jul. 2024, pp. 1–7, doi: 10.1109/ICAIT61638.2024.10690492.

[26] M. Lewis *et al.*, “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880, doi: 10.18653/v1/2020.acl-main.703.

[27] M. M. Haider, M. A. Hossin, H. R. Mahi, and H. Arif, “Automatic text summarization using gensim Word2Vec and K-means clustering algorithm,” in *2020 IEEE Region 10 Symposium, TENSYMP 2020*, 2020, vol. 2, pp. 283–286, doi: 10.1109/TENSYMP50017.2020.9230670.

[28] “Google Colab,” [Online]. Available: https://colab.research.google.com/drive/1N4w0FiQQv5oN9XG1oOX6NoqzpWSuFfH_?usp=sharing.

[29] “Google Colab,” [Online]. Available: https://colab.research.google.com/drive/18-VHg6148DmhjmdH_0KKNx7ce5k0k542?usp=sharing.

[30] “Datasets at Hugging Face,” 2001, [Online]. Available: https://huggingface.co/datasets/cnn_dailymail.

[31] C.-Y. Lin, “ROUGE: a package for automatic evaluation of summaries,” *ACL Anthology*, 2004, [Online]. Available: https://aclanthology.org/W04-1013.

[32] R. A. Albeer, H. F. Al-Shahad, H. J. Aleqabie, and N. D. Al-Shakarchy, “Automatic summarization of YouTube video transcription text using term frequency-inverse document frequency,” *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 26, no. 3, pp. 1512–1519, Jun. 2022, doi: 10.11591/ijeecs.v26.i3.pp1512-1519.

[33] B. M. Gurusamy, P. K. Rengarajan, and P. Srinivasan, “A hybrid approach for text summarization using semantic latent Dirichlet allocation and sentence concept mapping with transformer,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 6, pp. 6663–6672, Dec. 2023, doi: 10.11591/ijece.v13i6.pp6663-6672.

BIOGRAPHIES OF AUTHORS

Naidila Sadashiv received her Ph.D. degree in Computer Science and Engineering from University Visvesvaraya College of Engineering, Bangalore University, M.Tech. from Dr. Ambedkar Institute of Technology, VTU, and B.E. from B.V.B.C.E.T, Karnatak University, India. She is currently a professor in the Department of Information Science and Engineering at JSS Academy of Technical Education, Bangalore, India. She has a total twenty-two years of experience which includes teaching U.G. and P.G. students, research, guiding projects, handling University examinations, organizing seminars, conferences and workshops. Her area of research interest includes cloud computing, applications of AI and Blockchain. She has published papers in IEEE Conferences and Journals with more than 350 citations. She has won two best paper awards from IEEE Conferences. She is the recipient of Seed Money to Young Scientists for Research award from Vision Group on Science and Technology, Bangalore. She is reachable at naidila@jssateb.ac.in.

Aneesha Krishna Maiya received a B.E. in computer science and Engineering (CSE) from JSS Academy of Technical Education, Bangalore. He is currently working as an Associate Software Engineer in Sasken Technologies. His current research interests include machine learning, AI, and NLP. He is reachable at anishakmaiya@gmail.com.

Geetha Shivareddy received a B.E. in computer science and Engineering (CSE) from JSS Academy of Technical Education, Bangalore. She is currently working as a Software engineer in Capgemini. Her current research interests include machine learning, AI, and NLP. She is reachable at ggeethashivareddy@gmail.com.

Akash Reddy received a B.E. in computer science and engineering (CSE) from JSS Academy of Technical education, Bangalore. He is passionate about learning big data, machine learning. His research interests include machine learning and big data. He is reachable at akashreddyv572@gmail.com.