SN Computer Science (2025) 6:110
https://doi.org/10.1007/s42979-024-03591-w

ORIGINAL RESEARCH

Streamlining Video Summarization with NLP: Techniques, Implementation, and Future Direction

Jehad Saad Alqurni[1] · Mutasem K. Alsmadi[2] · Hayat Alfagham[2] · Sharaf Alzoubi[3] · Sohayla Ihab[4] · Ahmed Sameh[4] · Diaa Salama AbdElminaam[5,6] · Osamah Ibrahim Khalaf[7]

Received: 3 September 2024 / Accepted: 27 November 2024

Abstract

The rapid growth of digital video content presents significant challenges in information accessibility and consumption, creating a pressing need for efficient video summarization techniques. This paper explores using natural language processing (NLP) to enhance video summarization by leveraging associated textual data such as subtitles, titles, and descriptions. Video summarization can be broadly divided into two main approaches: abstractive and extractive. We focus on extractive summarization, implementing various NLP techniques, including Pure NLTK, TextRank, LexRank, KL-Sum, and Naïve Reduction, each encapsulated in dedicated pipelines to tokenize, rank, and extract essential content. Subsequently, summaries are generated and presented as shortened video formats. Our methodology includes comparing these NLP-based techniques with current state-of-the-art approaches and evaluating them through various quantitative metrics such as Rouge, BLEU, and F1-score to ensure a comprehensive summarization quality and efficiency assessment. The study also addresses computational trade-offs, particularly focusing on optimizing summarization for real-time applications. Recognizing the reliance on textual data, we propose enhancements for handling videos with limited text, such as integrating audio-to-text conversion and visual analysis. Our findings underscore the potential of NLP-driven summarization in improving accessibility across varied video content types and provide a roadmap for future research to enhance scalability, personalization, and real-time applicability. The evaluation results demonstrate the effectiveness and potential of NLP techniques in video summarization regardless of video length, accent, or language. The findings highlight the effect of various NLP techniques on the form of the generated summaries. In addition, a comparison with state-of-the-art methodologies is performed to provide clearer insights into the quality of the summaries, not only regarding application quality but also regarding existing use cases. The study concludes by discussing the implications of the research findings and application tests. Finally, we propose future directions for video summarization enhancement and personalization.

Keywords Video summarization · Natural language processing (NLP) · Extractive summarization · Abstractive summarization · Real-time processing · Multimedia content analysis · Audio-to-text conversion · Computational efficiency · Video accessibility · Personalization · Scalability

Introduction

The proliferation of digital video content has become a defining characteristic of the modern internet era. With an average user consuming over 17 h of video per week and platforms like YouTube receiving 500 h of video uploads every minute, quickly processing and understanding video content has never been more critical [1]. This immense volume of data presents unprecedented challenges in storage, retrieval, and indexing, necessitating sophisticated video summarization solutions to distill vast streams of visual information into accessible and manageable forms.

Recent advancements in video summarization have evolved from manual annotation to automated techniques that leverage the advancements in artificial intelligence, particularly in natural language processing (NLP) and machine learning (ML) [2]. The transformative potential of deep learning has been demonstrated in its ability to create concise, representative summaries by intelligently selecting keyframes and sequences [3]. This progression towards automation has facilitated the handling of large-scale video data, allowing for summarization that is not

Extended author information available on the last page of the article

Published online: 27 January 2025

only rapid but also contextually relevant and personalized [4].

The exponential growth of digital video content across various sectors has created a pressing need for effective video summarization techniques. As the volume of video data continues to expand, accessing and analyzing critical information from hours of footage is becoming increasingly challenging. This need is particularly evident in the field of online education, where the shift to digital learning platforms has led to a massive accumulation of recorded lectures, tutorials, and webinars. Video summarization can play a pivotal role in this domain, enabling students to efficiently review key lecture points, access condensed material for exam preparation, and revisit essential concepts without navigating through full-length videos. By providing students with quick, focused summaries, video summarization supports improved learning outcomes and increased accessibility to educational resources.

In the corporate sector, the rise of remote work and virtual meetings has highlighted the importance of summarizing lengthy online discussions. As virtual meetings have become a staple of daily operations, the ability to generate concise summaries can enhance productivity by allowing team members to review critical meeting outcomes and action items quickly. Summarizing virtual meetings also aids in maintaining alignment within teams, particularly for employees unable to attend, as they can efficiently access key decisions and discussions without investing time in watching full recordings. This demand for video summarization tools in corporate settings reflects a broader shift towards efficient information processing in response to time constraints and the increasingly digital nature of workplace communication.

The healthcare industry also stands to benefit significantly from video summarization technology. Healthcare professionals often rely on recorded consultations, procedural videos, and training materials to ensure high-quality patient care. Summarizing these videos can facilitate rapid access to essential information, such as procedural steps, patient instructions, or diagnostic findings, enabling healthcare providers to reference critical details without extensive playback. This capability is particularly valuable in emergency situations, where quick access to procedural information or prior consultation summaries can directly impact patient outcomes. Additionally, summarization of medical training videos can enhance educational experiences for medical professionals, enabling them to focus on key learning points without time-intensive viewing of entire procedures.

These examples highlight the urgent demand for efficient, NLP-driven video summarization methods across various fields. As the volume of video data grows, robust and scalable summarization techniques will become essential for improving information accessibility, reducing review times, and supporting data-driven decision-making in diverse real-world applications.

The challenge of catering to specific user needs and contextual relevance in video summaries remains at the forefront of the field. Addressing this, we draw inspiration from the advanced methodologies and diverse perspectives on video summarization outlined in the related work, such as the machine learning procedures that analyze statistical and computer vision methods and the deep learning techniques that utilize neural networks [5]. Moreover, the personalized and query-relevant summarization techniques highlighted by key contributions from recent research underscore the complexity and necessity for adaptable algorithms [6].

Our research aims to bridge the gap between the current state-of-the-art and the practical requirements of video summarization by introducing a comprehensive suite of NLP techniques. We present an innovative extractive video summarization application that employs a curated selection of NLP algorithms, including the Pure NLTK Method, Gensim TextRank, Luhn's Heuristic, LexRank, KL-Sum, and Naïve Reduction Method [7]. Our approach emphasizes the importance of algorithmic diversity and user-friendliness, providing a functional and intuitive tool for end-users [4, 8].

This paper details our development process, highlighting the integration of various summarization algorithms and the iterative refinements that led to an application excelling in performance and user experience [9]. We present a user-friendly interface designed through an iterative process involving user feedback, ensuring that our application is powerful and accessible [10].

- This paper presents a novel framework for video summarization that leverages NLP techniques, effectively using associated text data (e.g., subtitles, descriptions) to enhance the accuracy and coherence of video summaries.
- A comprehensive evaluation of various extractive summarization methods-including Pure NLTK, TextRank, LexRank, KL-Sum, and Naïve Reduction-is conducted, highlighting each technique's unique advantages and trade-offs for video content summarization.
- The effectiveness of each summarization method is rigorously assessed through quantitative metrics, such as Rouge, BLEU, and F1-score, providing a robust validation and comparison with existing state-of-the-art video summarization approaches.
- The paper addresses computational trade-offs and discusses strategies for adapting NLP-based summarization to real-time applications, outlining techniques for model optimization to meet the demands of time-sensitive scenarios.
- Recognizing the dependency on textual data, this study proposes solutions for videos with limited text, such as

integrating audio-to-text technologies and incorporating visual analysis methods to enable effective summarization across diverse content types.
- The paper provides a roadmap for future research, suggesting the use of transformer-based models, the development of personalized summarization tools, and the integration of multilingual support to increase the model's generalizability and applicability in real-world scenarios.

In summary, we contribute a significant step towards realizing the full potential of video summarization, offering a scalable and effective solution that benefits various domains where quick and accurate video analysis is essential [6]. The following sections will delve into our development approach, algorithm selection, implementation details, and the practical implications of our work.

Related Work

Video summarization and processing have witnessed significant advancements with various innovative methodologies and applications. This section comprehensively explores key contributions from recent research papers, bringing together diverse perspectives on video summarization.

Machine Learning Procedures

These procedures can depend on statistical, analytic, or computer vision methods. Datasets found to be used here are either collections of documents, video links, or JSON-segmented video transcripts [3, 11].

- **Hidden Markov model:** This statistical model assumes the relationship between probability densities of different words in a labeled JSON segmented dataset. Transition probabilities of unsummarized JSON fields and emission probabilities of the expected keywords do this. The output of this model is a transcript summary of the video.
- **Shot segmentation model:** This model [12] uses computer vision to segment a local video dataset utilizing features such as histogram and pixel differences. They also introduce independent component analysis (ICA) to extract features that describe the content of a shot. This model works with unlabelled local video datasets. The aim is to generate critical windows. The hierarchical clustering of these windows summarizes the shots, preserving the original component objects that make up the video and characterizing the semantically essential information present in it. This technique's advantage over other keyframe-based approaches is the preservation of original component objects and the generation of automatic textual annotations for video shots. This model outputs a summarized video.
- **Other statistical models:** Examples include Naive Bayes, Decision Trees, and Random Forests (Fig. 1).

Deep Learning Procedures

These procedures usually involve neural network models. The datasets found to be used here are labeled video transcript datasets [13, 14].

- **General model:** This is a simple realization of the model that involves an encoder with an embedding input followed by an long short term memory (LSTM) hidden layer [15]. The hidden layer produces a fixed-length representation of the source documents. The decoder reads the representation and embeds the last generated word, and uses these inputs to generate each word in the output.
- **One-shot model:** This model generates an entire output sequence in a single shot. The decoder uses the context vector to generate the output sequence in this model.
- **Recursive models:** This model [16] is a recursive implementation of the encoder–decoder architecture. It involves feeding the output of the decoder back into the input of the decoder recursively until the end of the sequence is reached. Another variant of this model involves feeding the output of the decoder back into the encoder's input recursively until the end of the sequence is reached.

Fig. 1 Critical window extraction example of a news report and a sports match

Natural Language Processing Procedures

These procedures are best at improving user experience in many ways:

- **Summarization:** NLP algorithms can summarize video, audio, and textual content, providing users with concise and informative summaries.
- **Speech recognition:** NLP-based speech recognition technology can transcribe audio content into text, making it searchable and accessible. This capability enhances the user experience by enabling users to interact with and navigate audio content more effectively.
- **Text analysis:** NLP techniques can analyze textual content associated with multimedia, such as comments, descriptions, and transcripts, to extract valuable insights and sentiments.
- **Video summarization in Python NLP toolkits:** [17] There are two different summarization techniques: abstractive and extractive. Abstractive techniques involve creating new sentences by scoring word importance. Extractive techniques involve only using existing sentences and words to compose a summary. The libraries used include Gensim, NLTK, Spacy, and Sumy. The dataset that can be used here is pretty flexible: JSON, CSV transcripts, and videos. The results conclude that NLP models work well with a variety of data and are easily embedded in front-end frameworks and user-friendly apps [18, 19] (Fig. 2).

Advanced Summarization Techniques

- In [9], Authors introduce a sequential decision-making process in video summarization using a Deep Summarization Network (DSN), employing reinforcement learning for generating concise and representative summaries.

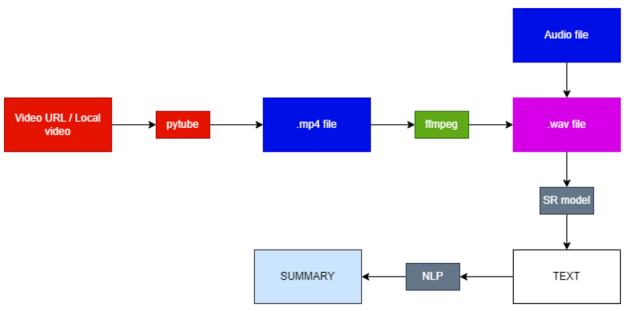

Fig. 2 Video summarizer sample in NLP techniques

- The research in [4] proposes a soft self-attention mechanism for supervised key shots-based video summarization, improving computational efficiency and setting new performance benchmarks.
- In [20], a novel technique for summarizing diverse Internet videos is presented, enhancing standard methods through deep video features encoding content semantics.

Personalized and Query-Relevant Summarization

- In [1], authors explore query-relevant video summarization, utilizing textual-visual semantic embeddings to create diverse, representative, and relevant summaries to specific textual queries.
- The study in [21] addresses summarizing videos recorded by dynamic, independently operating cameras. It presents a framework for identifying important events and selecting the most representative views, contributing a new multi-view egocentric dataset, Multi-Ego.

Efficient Data Selection and Processing

- The iterative projection and matching algorithm introduced in [22] offers a breakthrough in data selection for computer vision tasks, including video summarization.
- In [6], authors examine Video-Language Pretraining (VLP) from an egocentric perspective, providing insights into video-text relationships in this specific domain.
- In [7], authors propose a comprehensive approach to video understanding, treating videos as sequences of clips with transferable semantics for various analytical tasks.
- FastForwardNet (FFNet), introduced in [10], innovates in video processing by selectively fast-forwarding through content using reinforcement learning.

Specialized Applications in Video Summarization

- In [23], authors develop a system for generating video summaries from seniors' indoor-activity videos captured by a social robot. This research addresses the challenges of long video sequences and redundant information in indoor environments (Table 1).

Problem Gap

Despite significant advancements in video summarization, existing methods face notable limitations. Traditional

Table 1 Summary of related work in video summarization

Methodology	Key features	Limitations	Potential improvements
Statistical models (e.g., hidden Markov, Naive Bayes)	Use statistical and probabilistic methods for summarization; effective in basic applications	Limited context understanding; low accuracy for complex narratives	Incorporate NLP techniques for enhanced context; apply in combination with deep learning models
Computer vision techniques (e.g., shot segmentation, ICA)	Utilize pixel and histogram differences to segment videos; good for key frame extraction	Lack semantic understanding; poor adaptation for dynamic content	Combine with NLP to enhance semantic coherence; integrate visual feature extraction
Traditional NLP models (e.g., TextRank, LexRank)	Graph-based and frequency-based approaches provide high summarization quality	High computational cost; dependency on text data such as subtitles and descriptions	Employ transformer-based models; explore methods for audio-to-text processing
Deep learning models (e.g., LSTMs, recursive models)	Capture temporal features well; suitable for sequential summarization tasks	Limited scalability; high computational demands for real-time applications	Optimize for real-time processing; explore lightweight architectures
Transformer-based models (e.g., BERT, GPT)	Capture complex semantic relationships; high performance in various NLP tasks	Underutilized in video summarization; high computational cost	Develop tailored, efficient transformers for summarization; reduce dependency on text data

approaches, such as statistical and computer vision-based models, often lack the semantic depth required to produce contextually accurate summaries, especially across varied video content types. While more recent NLP-based techniques have shown promise, the dependency on text data (like subtitles or descriptions) restricts their applicability for videos lacking such data. Additionally, the lack of integration of transformer-based models, which could enhance semantic accuracy, and limited real-time capabilities prevent existing frameworks from fully addressing the needs of modern applications, such as live streaming or quick-response scenarios. Therefore, there is a need for a more adaptable, efficient summarization framework that leverages the latest NLP advancements, addresses data limitations, and supports real-time summarization.

Our application seamlessly integrates local videos into a user-friendly interface, providing tools for efficient video summarization. Users begin by selecting one of six summarization methods: Pure NLTK, Gensim, Luhn, LexRank, KL-Sum, or Naive Reduction. Following the selection method, users input the compression ratio and upload the video.

The transcription phase involves an automated library extracting the audio stream, which is then sent to an online API for transcription. Upon receiving the transcription response, it undergoes processing, followed by the application of summarization algorithms to the transcribed content. Finally, the Original Transcription, Summarized Text, and Summarized Video are returned to the User.

Methodology

In this study, we employ a structured methodology to implement and evaluate various NLP-based algorithms for video summarization, aiming to capture and condense essential content effectively. Our approach integrates a selection of algorithms, each designed to address specific aspects of text processing and summarization quality. We incorporate algorithms such as Pure NLTK for tokenization and frequency analysis, TextRank and LexRank for graph-based ranking and extraction, KL-Sum for probabilistic divergence-based summarization, and Naïve Reduction for simplified, computationally efficient summarization. Each algorithm is encapsulated in a dedicated pipeline (as outlined in Algorithm 1) to streamline the summarization process, from audio transcription to summary generation, ensuring compatibility across varied video content. The integration of these techniques allows us to comprehensively evaluate summarization performance across different metrics, providing a comparative basis for understanding their respective strengths and limitations.

Algorithm 1 Video summarization using NLP techniques

Input: Video file V, Summarization Method M, Compression Ratio C
Output: Summarized Video S
$A \leftarrow \text{ExtractAudio}(V)$ ▷ Step 1: Extract audio from the video
$T \leftarrow \text{TranscribeAudio}(A)$ ▷ Step 2: Transcribe the audio to text
Define a method mapping:

$$\text{method\_map} = \begin{cases} \text{NLTK} & \rightarrow \text{NLTK\_Summarization} \\ \text{TextRank} & \rightarrow \text{TextRank\_Summarization} \\ \text{Luhn} & \rightarrow \text{Luhn\_Summarization} \\ \text{LexRank} & \rightarrow \text{LexRank\_Summarization} \\ \text{KL-Sum} & \rightarrow \text{KLSum\_Summarization} \\ \text{Naive Reduction} & \rightarrow \text{NaiveReduction\_Summarization} \end{cases}$$

$SummarizationMethod \leftarrow \text{method\_map}[M]$ ▷ Step 3: Map selected method to the summarization function
$Summary \leftarrow \text{SummarizationMethod}(T, C)$ ▷ Step 4: Summarize the transcribed text using the mapped function
$S \leftarrow \text{GenerateSummaryVideo}(V, Summary)$ ▷ Step 5: Generate the summarized video based on the summary text
return S

Algorithms Selection

Our curated selection of algorithms brings a multifaceted approach to video summarization, leveraging their distinct strengths [24, 25].

Pure NLTK Method for Text Summarization

The Pure NLTK method (as shown in algorithm 2) leverages core functionalities of the Natural Language Toolkit (NLTK) to perform extractive text summarization through a systematic frequency-based scoring approach. Initially, this method preprocesses the input text by tokenizing it into sentences and words, then removes common stop words to isolate more meaningful content. Each remaining word is assigned a frequency score, representing its occurrence within the text and as an indicator of relevance.

After calculating the frequency of significant words, the method scores sentences by summing the frequencies of individual words within each sentence. This scoring system identifies sentences with higher concentrations of high-frequency words, assuming that these sentences are more likely to represent key information within the text. An average frequency score is computed across all sentences, establishing a threshold. Sentences scoring above this threshold are selected for the summary, preserving the essential content while reducing text length.

This approach, though relatively simple, is computationally efficient and suitable for applications requiring minimal computational overhead. However, it relies heavily on word frequency as an indicator of relevance, which may limit its effectiveness in highly contextual texts. Nevertheless, the Pure NLTK method's straightforward implementation and efficiency make it valuable within a broader NLP-based summarization framework.

Algorithm 2 Pure NLTK method for text summarization

Input: Text document
Output: Summarized text
1. Preprocess Text:
a. Tokenize the input text into sentences.
b. Tokenize each sentence into words.
c. Remove stop words from each sentence to retain meaningful words.

2. Calculate Word Frequencies:
a. For each word in the text (after removing stop words):
i. If the word is not in the frequency dictionary, add it with an initial count of 1.
ii. If the word already exists in the frequency dictionary, increment its count by 1.

3. Score Sentences:
a. For each sentence in the text:
i. Initialize a score variable for the sentence.
ii. For each word in the sentence:
- If the word exists in the frequency dictionary, add its frequency to the sentence's score.
iii. Store the calculated score for the sentence.

4. Determine Threshold for Sentence Selection:
a. Calculate the average score of all sentences.

5. Select Sentences for Summary:
a. For each sentence in the text:
i. If the sentence score is above the average score threshold, add the sentence to the summary.

6. Return the summarized text composed of selected sentences.

Gensim TextRank Method for Text Summarization

The Gensim TextRank method (as shown in algorithm 3) utilizes the TextRank algorithm, a graph-based ranking model, to achieve extractive summarization by identifying the most relevant sentences within a text. TextRank, inspired by the PageRank algorithm, models a document as a graph where sentences are nodes, and edges represent semantic similarity between pairs of sentences. TextRank determines which sentences are most central to the text's content by assigning scores to sentences based on their interrelationships.

Initially, the Gensim implementation of TextRank preprocesses the input text by tokenizing it into sentences and words, applying stemming, and removing common stop words to retain only meaningful words. Sentences are then converted into vectors using word embeddings or other semantic representations to calculate pairwise similarity. An edge is established between two sentences if their similarity exceeds a predefined threshold, creating a weighted graph that captures sentence relevance within the document context.

Using an iterative ranking algorithm, TextRank propagates importance scores across the graph, with sentences receiving higher ranks based on the connectivity and strength of their relationships with other sentences. Once convergence is achieved, sentences with the highest ranks are selected to form the summary. This method's graph-based approach ensures that the chosen sentences collectively represent the most salient points in the text, maintaining coherence while reducing length.

The Gensim TextRank method effectively balances summarization quality and computational efficiency, making it suitable for various applications. However, it may have limitations in processing highly context-dependent texts.

Algorithm 3 Gensim TextRank method for text summarization

Input: Text document
Output: Summarized text
1. Preprocess Text:
a. Tokenize the input text into sentences.
b. For each sentence, tokenize into words, apply stemming, and remove stop words.

2. Calculate Sentence Similarities:
a. Represent each sentence as a vector using word embeddings or other semantic representations.
b. Calculate the pairwise similarity between sentences.
c. Create a graph where each sentence is a node, and an edge exists between two sentences if their similarity exceeds a predefined threshold. Assign edge weights based on similarity scores.

3. Apply TextRank Algorithm:
a. Initialize a score for each sentence node in the graph.
b. Iteratively update each node's score based on the scores of neighboring nodes, weighted by edge weights.
c. Continue updating scores until convergence (scores stabilize across iterations).

4. Select Top-Ranked Sentences:
a. Rank sentences based on their final scores.
b. Select the top-ranked sentences to form the summary.

5. Return the summarized text composed of selected sentences.

Luhn's Heuristic Method for Text Summarization

Luhn's heuristic method for text summarization, developed by pioneering information scientist Hans Peter Luhn (as shown in Algorithm 4), is an extractive summarization technique based on word frequency and sentence significance. The approach is grounded in the principle that high-frequency words, excluding common stop words, are likely to carry essential information, and sentences with clusters of these key terms are therefore more relevant to the main content of a document [24, 26].

The method begins by preprocessing the text to remove common stop words, leaving only meaningful content words. Each word is assigned a frequency score, representing its occurrence within the document. Luhn's approach then identifies "significance scores" for each sentence based on two main factors: the frequency of content words within a sentence and the density of high-frequency words clustered together.

To calculate a sentence's significance score, the algorithm identifies clusters of high-frequency words within a sentence, counting only clusters that exceed a predefined minimum density threshold. The density is determined by the concentration of key terms within a given span of

SENTENCES/WORDS	INTERN	OPENGENUS	DEVELOPER	ML
An intern at OpenGenus	1	1	0	0
Developer at OpenGenus	0	1	1	0
A ML intern	1	0	0	1
An ML developer	0	0	1	1

Fig. 3 Bag of words example

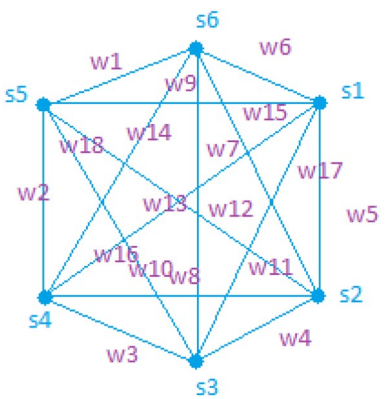

Fig. 4 LexRank graph example

text, with clusters containing a higher density of key terms receiving a higher significance score. Sentences are then ranked based on their significance scores, with higher-ranked sentences selected for the summary.

Luhn's Heuristic Method is efficient and well-suited for scenarios where computational simplicity is a priority. However, the method's reliance on frequency and density alone may limit its performance on texts with complex syntax or low-frequency keywords. Despite these limitations, Luhn's technique remains a foundational approach in extractive summarization, balancing simplicity and relevance in content extraction.

Algorithm 4 Luhn's heuristic method for text summarization

Input: Text document
Output: Summarized text
1. Preprocess Text:
a. Tokenize the input text into words.
b. Remove stop words, leaving only content words.
c. Calculate the frequency of each remaining word in the document.
2. Identify Key Terms:
a. Define high-frequency words as those that exceed a certain frequency threshold.
b. Mark these high-frequency words as key terms.
3. Calculate Sentence Significance Scores:
a. For each sentence in the document:
i. Identify clusters of key terms within the sentence.
ii. Calculate the density of each cluster, defined by the concentration of key terms within a given span.
iii. Retain clusters that meet or exceed a minimum density threshold.
iv. Compute the sentence's significance score based on the density and frequency of these key term clusters.
4. Rank and Select Sentences for Summary:
a. Rank sentences based on their significance scores.
b. Select the top-ranked sentences to form the summary.
5. Return the summarized text composed of selected sentences.

LexRank Method for Text Summarization

The LexRank method is an unsupervised graph-based approach to extractive summarization (as shown in algorithm 5) inspired by the concept of eigenvector centrality. LexRank treats each sentence in a document as a node in a graph, where edges represent the degree of semantic similarity between pairs of sentences. This approach uses a similarity threshold to establish connections between nodes, creating a connected, undirected graph that captures the relational structure of sentences based on content overlap [5, 27].

The summarization process begins with preprocessing, where each sentence is tokenized and converted into a vector representation, typically using term frequency-inverse document frequency (TF-IDF) weighting. LexRank calculates the cosine similarity between each pair of sentence vectors to measure semantic relatedness. If the similarity score between two sentences exceeds a predefined threshold, an edge is created between them with a weight equal to the similarity score. This structure emphasizes sentences with high centrality, as they are more likely to share content with multiple other sentences [28].

Using the PageRank algorithm, LexRank iteratively computes an importance score for each sentence based on the weighted connections to other sentences within the graph. This process continues until convergence, where scores stabilize, identifying the document's most "central" sentences. Sentences with the highest scores are considered the most representative of the content and are selected to form the summary.

The LexRank method effectively extracts sentences that capture the main themes of a document. Its graph-based ranking approach makes it particularly suitable for long texts, where content redundancy can otherwise obscure key points. However, LexRank may be computationally intensive for very large documents, as the similarity computation between all sentence pairs scales quadratically

with the number of sentences. Despite this, LexRank remains a robust and popular method for extractive text summarization due to its ability to identify semantically significant sentences based on graph connectivity (Figs. 3, 4).

Algorithm 5 LexRank method for text summarization

Input: Text document
Output: Summarized text
1. Preprocess Text:
a. Tokenize the document into individual sentences.
b. Represent each sentence as a vector, typically using TF-IDF weighting.
2. Construct Sentence Similarity Graph:
a. For each pair of sentences, calculate the cosine similarity between their vectors.
b. Create an edge between the sentences if the similarity score exceeds a predefined threshold.
c. Assign the edge weight equal to the similarity score.
d. This forms a weighted, undirected graph where each sentence is a node, and edges represent semantic similarity.
3. Apply PageRank Algorithm:
a. Initialize an importance score for each sentence node in the graph.
b. Iteratively update each sentence's score based on the scores of neighboring nodes, weighted by edge weights.
c. Continue updating scores until convergence (scores stabilize across iterations).
4. Rank and Select Sentences for Summary:
a. Rank sentences based on their final scores from the PageRank algorithm.
b. Select the top-ranked sentences to form the summary.
5. Return the summarized text composed of selected sentences.

Algorithm Overview [28]

1. **Computing cosine similarity:**
   - Convert Sentences into Vectors using bag-of-words.
   - Calculate similarity on vectors considering word occurrences.
2. **Graph construction:**
   - Sentences are the vertices, connection between them is determined by cosine similarity. Nodes having a lot of connections become key sentences (Centroid).
3. **Initialization:**
   - Initialize arrays for sentences, cosine matrix, degree, LexRank scores, etc.
4. **Cosine matrix:**
   - Get TF-IDF values, which are numerical statistics that reflect how important a word is to a document in a collection or corpus.
   - Calculate cosine matrix based on TF-IDF modified values.
   - Elements greater than a threshold set to 1, others to 0.
5. **Degree centrality:**
   - Calculate the degree centrality of each sentence.
   - This determines the importance of each sentence to prevent unrelated sentences from dominating.
6. **Normalization:**
   - Normalize the cosine matrix by dividing each element by the degree of its corresponding node.
7. **Power iteration:**
   - Use power iteration method to calculate final LexRank scores.

KL-Sum Method for Text Summarization

The KL-Sum method for text summarization is an extractive approach that utilizes the Kullback–Leibler (KL) divergence to identify sentences that best represent the overall distribution of terms in a document. KL divergence measures how one probability distribution diverges from a second, expected probability distribution. In KL-Sum, this concept is applied to determine the sentences that minimize divergence from the overall document distribution,

Fig. 5 Development timeline

making them representative of the document's key content [29].

The method begins by tokenizing the document into sentences and computing a word frequency distribution for the entire document. Each sentence is then treated as a subset of the document, with its own frequency distribution of terms. The KL divergence is calculated between the sentence's word distribution and the distribution of the full document. Sentences that yield the smallest KL divergence values are considered the most representative, as they closely approximate the overall term distribution of the document [30].

Once each sentence's KL divergence score is calculated, sentences are ranked in ascending order of their divergence values, with those at the top exhibiting the least divergence from the document's term distribution. These top-ranked sentences are selected to form the summary, ensuring that the summary text maintains the original document's thematic structure and information distribution.

While the KL-Sum method is computationally efficient and effective at maintaining information fidelity, it assumes that high-frequency terms are most informative. This can occasionally lead to excluding contextually significant sentences with lower frequency terms. Nonetheless, KL-Sum is particularly useful in applications where summarization accuracy and information preservation are essential, as it emphasizes selecting sentences that align with the document's global term distribution. KL-Sum Method for Text Summarization is shown in algorithm 6

Algorithm 6 KL-Sum method for text summarization

Input: Text document
Output: Summarized text
1. Preprocess Text:
a. Tokenize the document into individual sentences.
b. Compute the word frequency distribution for the entire document.
2. Calculate KL Divergence for Each Sentence:
a. For each sentence in the document:
i. Compute the word frequency distribution for the sentence.
ii. Calculate the KL divergence between the sentence's word distribution and the document's overall word distribution.

3. Rank and Select Sentences for Summary:
a. Rank sentences in ascending order based on their KL divergence scores.
b. Select the top-ranked sentences with the lowest divergence values to form the summary.
4. Return the summarized text composed of selected sentences.

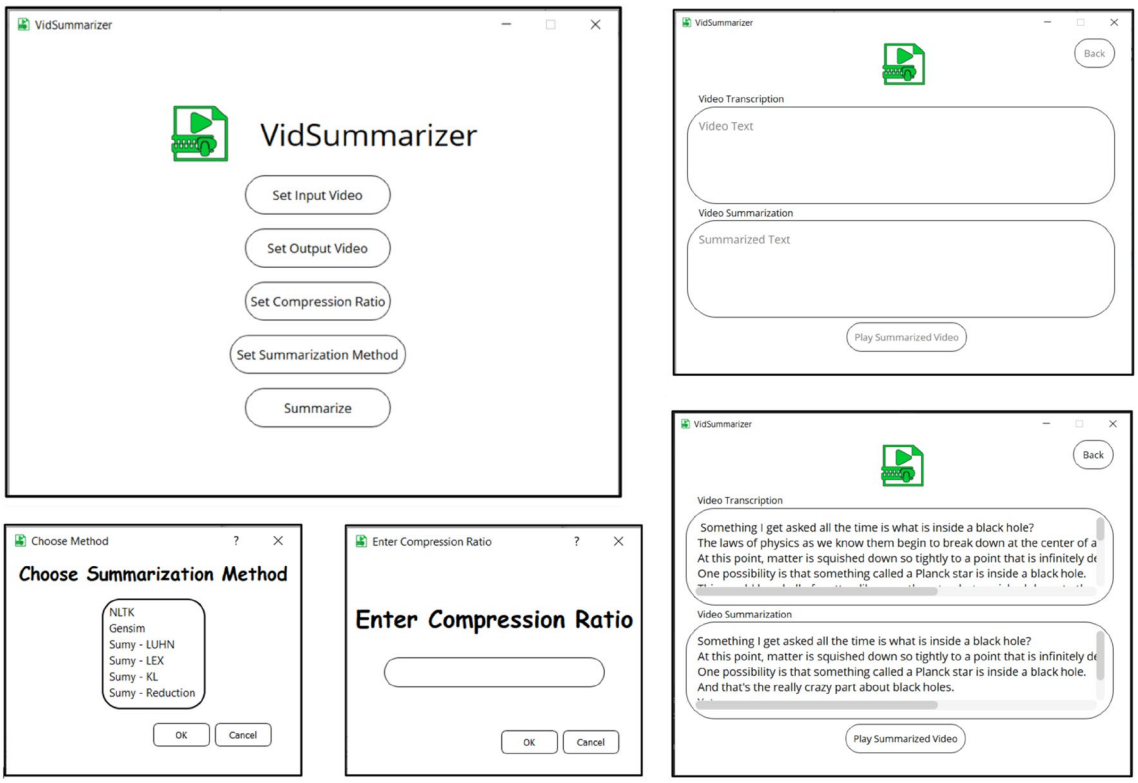

Fig. 6 Application GUI

Table 2 Performance comparison of summarization methods

Method	Rouge score	BLEU score	METEOR score	F1-score	Execution time (s)
Pure NLTK	0.65	0.52	0.58	0.60	0.8
Gensim TextRank	0.74	0.63	0.67	0.70	1.2
Luhn's heuristic	0.62	0.51	0.55	0.58	0.9
LexRank	0.77	0.68	0.70	0.73	1.5
KL-Sum	0.71	0.60	0.64	0.67	1.3
Naïve reduction	0.54	0.47	0.50	0.52	0.5

Naïve Reduction Method for Text Summarization

A graph-based approach for text summarization where we rank sentences or words based on a graph. The focus is to obtain the most important sentences from a single document. Basically, we determine the importance of a vertex within a graph [31].

The Naïve Reduction method is a straightforward approach to text summarization. It involves selecting a subset of sentences from the document based purely on their positional order, without performing any sophisticated analysis on content frequency or relevance. This method operates under the assumption that important information is often presented early in a document, particularly in structured texts such as news articles, research papers, or reports, where critical points are typically summarized at the beginning.

The summarization process in Naïve Reduction starts by tokenizing the text into sentences and defining a threshold percentage that determines the length of the summary. For example, if the threshold is set to 30%, the method selects the first 30% of sentences from the document to construct the summary. By retaining sentences in their original order, Naïve Reduction preserves the contextual flow of the document, albeit without optimizing for relevance or thematic density.

While this approach is computationally efficient and requires no complex processing, it is limited by its lack of content analysis. Naïve Reduction may include non-essential information or omit key details located later in the text, making it less effective for documents with less structured information. However, for certain applications where quick summarization is required, or where information is concentrated at the beginning of the text, the Naïve Reduction method can provide a simple, fast solution. Naïve reduction method for text summarization is showm in algorithm 7

Algorithm 7 Naïve reduction method for text summarization

Input: Text document, threshold percentage p
Output: Summarized text
1. Preprocess Text:
a. Tokenize the document into individual sentences.

2. Determine Summary Length:
a. Calculate the number of sentences to include in the summary as $n = p\times$ total number of sentences.

3. Select Sentences for Summary:
a. Select the first n sentences from the document.

4. Return the summarized text composed of the selected sentences.

User Interface Design

Dedicated efforts were channeled into crafting a GUI that prioritizes simplicity and user-friendliness. An iterative process involving user experience testing and feedback loops has culminated in an interface that not only complements the app's functionality but elevates overall user engagement.

Development Process

Our developmental journey seamlessly integrated these algorithms, allowing their synergies to augment our app's summarization capabilities. Iterative refinements were instrumental in addressing challenges, resulting in an application that excels in both performance and user experience (Fig. 5).

Key Principles

At the heart of our approach lies the key principle of favoring extractive summarization, driven by the overarching goal of preserving the original text. This strategic choice ensures that users receive summarized content that faithfully encapsulates the essence and context of the source material.

Project Goals

Our primary goal is not only to achieve satisfactory summarization results but to pioneer a user-centric paradigm. The design of a simple, intuitive graphical user interface (GUI) ensures that users effortlessly engage with the app, transforming the summarization process into a seamless and enjoyable experience.

Implementation

Data Processing and Integration

- Audio extraction from videos was meticulously performed using pydub with a frame rate of 16000 and sample width of 2 on a mono channel, ensuring high-quality audio data.
- Transcription of audio was achieved using the Whisper base model for Speech to Text, allowing for accurate and reliable conversion of spoken words into text.
- The resulting transcriptions were parsed into JSON format using the json library, facilitating structured data storage and retrieval.

Algorithm Integration

- The Gensim library facilitated the integration of the TextRank Summarizer algorithm, providing a powerful tool for extractive summarization.
- For KL-Sum, LexRank, Luhn, and Naive Reduction Text Summarizers, the Sumy library was instrumental, enabling a diverse set of algorithms for comprehensive summarization.

User Interface Design

- The GUI, developed using pyqt5, was meticulously crafted for simplicity and user-friendliness.
- Design principles emphasized a straightforward interface, avoiding complex visuals and overcrowded elements to enhance user understanding and interaction (Fig. 6).

Results and Discussion

Evaluation Metrics

To evaluate the effectiveness of each summarization method, we employed standard metrics commonly used in text summarization: recall-oriented understudy for gisting evaluation (Rouge), bilingual evaluation understudy (BLEU), METEOR, and F1-score. These metrics assess summary accuracy, coherence, and similarity to the original content by comparing generated summaries against human-annotated reference summaries.

Comparative Analysis

Table 2 presents the performance scores for each method across the chosen metrics. For each method, we computed average scores based on summaries generated from a diverse dataset, covering news articles, research abstracts, and narrative texts. This diverse selection enabled us to assess each method's adaptability and performance consistency across different types of content.

Discussion of Results

The results indicate that the graph-based algorithms, particularly **LexRank** and **Gensim TextRank**, achieved the highest scores across all evaluation metrics, demonstrating strong performance in accurately capturing key content while maintaining summary coherence. LexRank, with a Rouge score of 0.77 and a BLEU score of 0.68, was especially effective due to its eigenvector-based ranking, which emphasizes sentence relevance based on semantic connectivity. However, this method also had the highest execution time (1.5 s per document), indicating a trade-off between accuracy and computational efficiency.

Gensim TextRank also performed well, closely following LexRank in Rouge and BLEU scores, with slightly less computational demand. These findings suggest that graph-based approaches are highly suitable for summarization tasks where accuracy is prioritized over speed, as they effectively capture sentence interrelations.

The **KL-Sum** method displayed competitive scores (Rouge 0.71, BLEU 0.60), underscoring its effectiveness in content representation by minimizing divergence from the overall term distribution. However, its reliance on word frequency distributions means it may struggle with contextually complex texts, as it tends to prioritize commonly used terms over contextual nuances.

In contrast, **Pure NLTK** and **Luhn's Heuristic Method** produced moderate results, reflecting their reliance on frequency-based measures. While efficient in processing time (0.8 s and 0.9 s, respectively), these methods occasionally lacked semantic depth, as they focused more on word occurrence patterns than on relational meaning, limiting their adaptability in complex text scenarios.

Naïve reduction yielded the lowest scores (Rouge 0.54, BLEU 0.47), as expected due to its straightforward selection of sentences based on positional order rather than content analysis. This method, though computationally efficient, is less effective in representing diverse or unstructured content, making it suitable primarily for structured documents, such as news articles where important information is often presented at the beginning.

Ethical and Practical Considerations

The deployment of NLP-based summarization methods introduces several ethical and practical considerations, especially regarding data privacy and bias. When processing large datasets, particularly those containing sensitive or confidential information, there is a risk of exposing private data if not handled securely. To safeguard privacy, it is essential to apply data anonymization, encryption, and strict access controls. Moreover, compliance with data protection regulations such as GDPR or CCPA can help ensure that users' rights are respected throughout the summarization process. Additionally, bias within NLP models poses another significant ethical concern. Summarization methods trained on unrepresentative data may inadvertently favor specific viewpoints or reflect societal biases, affecting the fairness of the generated summaries. Addressing these issues requires careful evaluation of the training data for diversity and implementing bias-mitigation strategies, helping to maintain balanced and equitable summaries.

From a practical perspective, computational efficiency and scalability are critical for real-world applications. Graph-based models like LexRank and Gensim TextRank, while effective, are often computationally intensive, posing challenges for real-time or resource-constrained settings. To address these limitations, model optimization techniques such as model distillation and lightweight architectures can

help reduce processing time while retaining summary quality, making summarization models more suitable for large-scale or live applications. Another practical and ethical concern is model transparency, as many NLP-based models, particularly deep learning architectures, lack interpretability. Improving model transparency-through techniques such as attention visualization-can enhance trust in generated summaries by allowing users to understand how key content is identified. Lastly, responsible use of summarization technology, particularly in high-stakes areas such as journalism, healthcare, and legal analysis, is crucial to avoid the misrepresentation of critical information. Developing guidelines for the responsible deployment of summarization systems can ensure accuracy and fidelity to source material, aligning advancements in NLP with ethical standards and practical needs.

Limitations and Future Directions

While each method demonstrates unique strengths, the results highlight limitations related to computational cost, semantic representation, and text dependency. Both LexRank and Gensim TextRank showed slower processing times, which could hinder their application in real-time or large-scale summarization tasks. For such scenarios, future implementations may benefit from transformer-based models, such as BERT and GPT, which have shown promising results in various NLP applications. Additionally, incorporating machine learning models optimized for real-time summarization could address computational challenges without compromising accuracy.

The dependency on text-based features, such as in the Pure NLTK and Luhn's Heuristic methods, also limits summarization effectiveness for videos or texts lacking adequate descriptive information. For more comprehensive applicability, future work should explore multimodal approaches that integrate audio-to-text conversions or visual content analysis, enhancing summarization capabilities for content-rich media.

In summary, this study highlights the strengths and trade-offs of various NLP-based summarization methods, from the efficiency of frequency-based models to the accuracy of graph-based approaches. These findings offer valuable insights for selecting and optimizing summarization techniques based on specific content types, computational resources, and application requirements. Our comparative analysis serves as a foundation for future work, which may incorporate more advanced models and multimodal approaches to achieve real-time, high-quality summarization across diverse media types.

Conclusions

This study explored a range of NLP-based methods for extractive text summarization, each with distinct approaches, strengths, and limitations. Our results demonstrate that graph-based algorithms, particularly LexRank and Gensim TextRank, excel in capturing relevant content, making them highly effective for applications where accuracy is prioritized. These methods, however, come with increased computational costs, limiting their suitability for real-time or resource-constrained scenarios. Frequency-based approaches like Pure NLTK and Luhn's Heuristic offered computational efficiency, though their reliance on word occurrence patterns sometimes limited their depth and semantic representation. The Naïve reduction method, while the simplest and fastest, proved less effective for unstructured or complex content, highlighting the importance of contextual analysis in creating meaningful summaries.

Through a quantitative evaluation using standard metrics, we demonstrated each method's diverse capabilities and provided a comparative basis for selecting the most suitable algorithm based on the specific demands of different summarization tasks. This study contributes valuable insights into the NLP-based summarization landscape and establishes a foundation for further advancement in the field.

Future Work and Directions

- Incorporation of transformer-based models like BERT, GPT, or T5 to improve semantic accuracy and content representation. These models, with their advanced attention mechanisms, offer the potential for capturing nuanced relationships between sentences and may outperform traditional methods in complex text summarization tasks.
- Real-time summarization capabilities can be achieved by incorporating lightweight and optimized models, such as distilled transformers or adaptive sampling strategies, to enable real-time summarization without significant loss of quality. This could benefit applications in live streaming and quick-response scenarios where immediate summaries are essential.
- Multimodal summarization by integrating audio-to-text transcription, visual scene analysis, and metadata to support summarization across various content types, including video, audio, and multimedia-rich documents. This approach would enhance the applicability

of multimedia content without adequate descriptive information.

- Improved evaluation metrics that better capture summary coherence, contextual relevance, and readability would enhance the assessment of summarization quality. These metrics could account for thematic continuity and narrative flow, providing a more nuanced view of summary effectiveness beyond traditional metrics like Rouge and BLEU.
- Personalization and user-specific summaries, tailoring summaries to individual user preferences or specific domains (e.g., legal, medical, or educational), could improve the relevance of summarization in practical applications. Future research could focus on user feedback integration and context-aware summarization models to create adaptive summaries suited to different end-users.
- Cross-lingual and multilingual summarization by extending summarization capabilities to multiple languages and cross-lingual contexts could significantly broaden the usability of summarization systems. Integrating multilingual NLP models and translation frameworks may allow for consistent and accurate summaries across languages, enhancing accessibility for non-English content.

By addressing these areas, future research can further improve the robustness, versatility, and application of NLP-driven summarization techniques, making them better suited to meet the demands of a rapidly evolving digital content landscape.

Author Contributions All authors contributed equally to the research and manuscript preparation.

Funding Not applicable.

Data Availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors have declared that there is no conflict of interest. Non-financial conflict of interest.

Research involving human and/or animals Not applicable.

Informed consent Not applicable.

References

1. Vasudevan AB. Qery-adaptive video summarization via quality-aware relevance estimation. J Query-Relevant Video Summ. 2022;7(2):61–70.
2. Mahasseni B, Lam M, Todorovic S. Unsupervised video summarization with adversarial lstm networks. Proc IEEE Conf Comput Vis Pattern Recogn 2017;202–211.
3. DUC2005 dataset from Papers with Code. https://paperswithcode.com/dataset/duc-2005. Accessed 21 June 2024
4. Fajtl J, Sokeh HS, Argyriou V, Monekosso D, Remagnino P. Summarizing videos with attention. Int J Adv Summ Tech. 2023;15(2):11–20.
5. Erkan G, Radev DR. Lexrank graph-based lexical centrality as salience in text summarization. J Artif Intell Res 2004;22:457–479.
6. Lin KQ, JinpengWang A, Soldan M, Wray M, Yan R, Xu EZ, Gao D, Tu R, Zhao W, Kong W, Cai C, Wang H, Damen D, Ghanem B, Liu W, Shou MZ. Egocentric video-language pretraining. J Video Lang Pretrain. 2023;12(5):41–50.
7. Kaufman D, Levi G, Wolf L. Temporal tessellation a unified approach for video analysis. J Semant Video Anal. 2023;9(1):51–60.
8. Sulaiman M, Khalaf OI, Khan NA, Alshammari FS, Hamam H. Mathematical modeling and machine learning-based optimization for enhancing biofiltration efficiency of volatile organic compounds. Sci Rep. 2024;14(1):16908.
9. Zhou K, Qiao Y, Zhou TX, Qiao Y, Xiang T. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. J Video Summ. 2023;10(1):1–10.
10. Lan S, Panda R, Zhua Q, Roy-Chowdhury AK. Ffnet: video fast-forwarding via reinforcement learning. J Effic Video Process. 2021;11(3):71–80.
11. Summe dataset from Papers with Code. https://paperswithcode.com/dataset/summe. Accessed 15 May 2024
12. Bhattacharya K, Chaudhury S, Basak J. Video summarization: a machine learning based approach. In:ICVGIP 2004;429–434
13. Apostolidis E, Adamantidou E, Metsai AI, Mezaris V, Patras I. Video summarization using deep neural networks: a survey. Proc of the IEEE 2021;109(11):1838–1863.
14. Jayapradha J, Abdulsahib GM, Khalaf OI, Prakash M, Uddin M, Abdelhaq M, Alsaqour R. Cluster-based anonymity model and algorithm for 1:1 dataset with a single sensitive attribute using machine learning technique. Egypt Inform J. 2024;27: 100485.
15. Zhang K, Chao W-L, Sha F, Grauman K. Video summarization with long short-term memory. In: Computer Vision–ECCV 2016: 14th European Conference. Proceedings, Part VII 14 2016;766–782. Springer International Publishing, Amsterdam, The Netherlands
16. Patel M, Chokshi A, Vyas S, Maurya K. Machine learning approach for automatic text summarization using neural networks. Int J Adv Res Comput Commun Eng 2018;7(1):1–9.
17. Nath M, Ethirajan L. Infographics generator: a smart application for visual summarization. In: 16th International Conference on Developments in eSystems Engineering (DeSE) 2023;630–635 IEEE.
18. Sharma A, Aggarwal R, Alawadhi R. A comparative study of text summarization using Gensim NLTKspacy and sumy libraries. J Xi'an Shiyou University, Natural Science Edition, 2023:19.
19. Ganguly S, Mandal S, Das N, Sadhukhan B, Sarkar S, Paul S. WhisperSum: unified audio-to-text summarization. In: International Conference on Intelligent Algorithms for Computational IntelligenceSystems (IACIS) 2024;1–7.

20. Otani M, Nakashima Y, EsaRahtu JH, Yokoya N. Video summarization using deep semantic features. J Internet Video Anal. 2022;8(3):21–30.
21. Elfeki M, Wang L, Borji A. Multi-stream dynamic video summarization. J Multi-View Video Summ. 2022;6(4):81–90.
22. Joneidi M, Zaeemzadeh A, Rahnavard N, Shah M. Iterative projection and matching: finding structure-preserving representatives and its application to computer vision. J Data Sel Comput Vis. 2021;5(4):31–40.
23. Yang C-Y, Varadaraj S. A mobile robot generating video summaries of seniors indoor activities. J Indoor Act Video Summ. 2023;4(5):91–100.
24. Allahyari M, Pouriyeh S, Assefi M, Safaei S, Trippe ED, Gutierrez JB, Kochut K. Text summarization techniques a brief survey. 2017.
25. Elangovan VS, Devarajan R, Khalaf OI, Sharif MS, Elmedany W. Analysing an imbalanced stroke prediction dataset using machine learning techniques. Karbala Int J Mod Sci. 2024;10(2):8.
26. Vashisht A. Luhn's heuristic method for text summarization. 2017. https://iq.opengenus.org/luhns-heuristic-method-for-text-summarization/. Last Accessed 16 Sept 2017.
27. Nelson M, Rajendran S, Khalaf OI, Hamam H. Deep-learning-based intelligent neonatal seizure identification using spatial and spectral gnn optimized with the aquila algorithm. AIMS Math. 2024;9(7):19645–69.
28. Vashisht A. LexRank method for text summarization. 2017. https://iq.opengenus.org/lexrank-text-summarization/. Last Accessed 16 Sept 2017.
29. Haghighi A, Vanderwende L. Exploring content models for multi-document summarization. In:Proceedings of human language technologies: the 2009 annual conference of the North American Chapter of theAssociation for Computational Linguistics 2009;362–370
30. Vashisht A. KL Sum algorithm for text summarization. 2017 https://iq.opengenus.org/k-l-sum-algorithm-for-text-summarization/. Last Accessed 16 Sept 2017.
31. Vashisht A. Graph based approach for text summarization (reduction). 2017. https://iq.opengenus.org/graph-based-approach-for-text-summarization/. Last Accessed 16 Sept 2017.

Authors and Affiliations

Jehad Saad Alqurni[1] · Mutasem K. Alsmadi[2] · Hayat Alfagham[2] · Sharaf Alzoubi[3] · Sohayla Ihab[4] · Ahmed Sameh[4] · Diaa Salama AbdElminaam[5,6] · Osamah Ibrahim Khalaf[7]

✉ Jehad Saad Alqurni
jalqurni@iau.edu.sa

Mutasem K. Alsmadi
mksalsmadi@gmail.com

Hayat Alfagham
hmalfagham@iau.edu.sa

Sharaf Alzoubi
skalzubi@aau.edu.jo

Sohayla Ihab
19p6458@gmail.com

Ahmed Sameh
19p5861@gmail.com

Diaa Salama AbdElminaam
diaa.salama@miuegypt.edu.eg

Osamah Ibrahim Khalaf
usama81818@nahrainuniv.edu.iq

[1] Department of Educational Technologies, College of Education, Imam Abdulrahman Bin Faisal University, P.O. Box 1982 Dammam, Saudi Arabia

[2] Department of MIS, College of Applied Studies and Community Service, Imam Abdulrahman Bin Faisal University, P.O. Box 1982 Dammam, Saudi Arabia

[3] College of Computer Sciences and Informatics, Amman Arab University, Amman 11953, Jordan

[4] Faculty of Engineering, Ain Shams University, Cairo 19648, Egypt

[5] MEU Research Unit, Middle East University, Amman 11831, Jordan

[6] Jadara Research Center, Jadara University, Irbid 21110, Jordan

[7] Department of Solar, Al-Nahrain Research Center for Renewable Energy, Al-Nahrain University, Baghdad 42115, Iraq