Cureus Journal of Computer Science

Open Access Original Article

LearnSmart: A Tool to Extract Key Concepts and Arguments From YouTube Lectures

Sujata Khedkar [1] , Ayush R. Gerra [1] , Lintomon S. Chirakkara [1] , Sania Khan [1] , Prathamesh Joshi [1] , Harsh Dhebe [1]

1. Computer Engineering, Vivekanand Education Society's Institute of Technology, Mumbai, IND

Corresponding authors: Ayush R. Gerra , 2021.ayush.gerra@ves.ac.in, Lintomon S. Chirakkara, 2021.lintomon.chirakkara@ves.ac.in, Prathamesh Joshi, 2023.prathamesh.joshi@ves.ac.in

Received 03/05/2025
Review began 04/04/2025
Review ended 09/14/2025
Published 09/21/2025

DOI:
https://doi.org/10.7759/s44389-025-03599-0

Abstract

LearnSmart is an AI-powered platform built to make it easier for learners to process and understand the vast amount of digital content available today. It automatically finds relevant YouTube videos based on a user's query and uses advanced natural language processing (NLP) techniques - like clustering and Retrieval-Augmented Generation - to extract and organize key ideas. These insights are then transformed into concise articles with the help of Mistral AI, narrated video summaries using text-to-video technology, and neatly structured PowerPoint presentations converted directly from PDF documents. To handle content efficiently, LearnSmart uses Mini Self-Organizing Map for grouping similar topics and Facebook Artificial Intelligence Similarity Search for fast, vector-based searching. The system delivers strong performance, generating articles in about 26.81 seconds and loading images in just over 33 seconds. While clustering shows a low silhouette score, it still effectively handles overlapping ideas. By offering content in multiple formats-text, video, and slides, LearnSmart helps reduce information overload and supports more effective, flexible learning. Ongoing improvements will focus on enhancing NLP accuracy and expanding the platform's ability to support a wider range of educational needs.

Categories: AI applications, Ethical AI and Responsible Technology, Natural Language Processing (NLP)
Keywords: ai-powered learning, natural language processing (nlp), retrieval-augmented generation (rag), video summarization, digital education, information overload, ai learning platform, content summarization, youtube video analysis, article generation

Introduction

In the digital age, the rapid expansion of online content has transformed how individuals acquire and process information. The sheer volume of available data - especially in multimedia formats such as YouTube videos and PDFs - poses significant challenges for learners seeking concise and relevant educational material. Traditional learning methods often struggle to manage this overwhelming influx of information, underscoring the need for intelligent systems that can efficiently extract and summarize key insights.

Existing research has explored various techniques for information retrieval and summarization. For example, Natural Language Processing (NLP)-based approaches have been widely employed for both text and video summarization, with studies demonstrating the effectiveness of automated transcript-based methods in condensing lengthy content while retaining essential details [1,2]. Moreover, recent advancements in retrieval-augmented generation (RAG) have improved the contextual relevance of generated content, thereby enhancing content generation capabilities. Other innovative frameworks, including text-to-image generation and specialized data loaders, further illustrate the expanding range of AI-driven solutions for content processing [3]. These developments highlight the potential of artificial intelligence to simplify the learning process across multiple modalities.

Despite these advancements, many existing methods lack a multi-modal approach, limiting their accessibility for diverse learners. "LearnSmart" addresses this gap by integrating NLP, artificial intelligence-driven content generation, and interactive visual elements to provide structured educational resources. By retrieving relevant YouTube videos, processing captions using advanced clustering techniques, and generating well-structured articles, summary videos, and flowcharts, the system enhances learning efficiency and accessibility [1]. This research aims to explore the effectiveness of "LearnSmart" in mitigating information overload while ensuring content accuracy and relevance.

Literature survey

The increasing demand for automated educational content summarization has led to the evolution of diverse techniques leveraging NLP, video summarization, clustering, and multimedia augmentation. This section reviews and synthesizes the key contributions in this space that have laid the foundation for LearnSmart's development.

Chaflekar et al. [1] introduced a YouTube transcript summarization technique using NLP. Their work highlighted how educational video content could be distilled into concise text, but it remained limited to static textual output, lacking interactivity and multimodal representation. Hande et al. [2] enhanced this approach by integrating machine learning for contextual retention and flow in summaries. However, they too focused solely on textual summarization, neglecting visual and auditory learning channels.

Ramzan et al. [3] explored deep learning-based text-to-image generation, opening pathways for integrating visual content in education. Their work aligns with LearnSmart's inclusion of AI-generated imagery to enhance user engagement and comprehension. However, their approach lacks direct educational context alignment.

Reimers and Gurevych [4] contributed Sentence-BERT, which produces high-quality sentence-level embeddings. This advancement is crucial for LearnSmart's semantic similarity measurements across video captions and documents. However, their application did not address clustering or summarization directly.

Kohonen's foundational work on Self-Organizing Maps (SOMs) [5] and Xinwu's comparative analysis with K-Means [6] offer valuable insights into clustering techniques. While SOMs allow for soft clustering and better semantic overlap - beneficial in educational transcript summarization - K-Means tends to yield higher silhouette scores due to hard clustering, potentially discarding meaningful but overlapping content. LearnSmart builds upon this by utilizing MiniSOM to retain nuanced sentence groupings.

Additionally, Recommender Systems, RAG , and video captioning pipelines (e.g., Whisper, Edge TTS) are part of recent advancements integrated into the platform. These were not previously unified into a singular framework for educational summarization.

Summary and motivation

Despite rich developments in isolated areas - text summarization, semantic embedding, image generation, and clustering - existing systems often remain siloed, focusing on a single modality or lacking dynamic interactivity. Current tools either generate plain text outputs or emphasize a single input type (text, video, or document), failing to provide a cohesive multimodal learning experience.

LearnSmart fills this critical gap by synthesizing and extending these approaches into a unified AI-powered learning platform. It incorporates video summarization, PDF processing, article and image generation, and interactive knowledge graphs to present educational content in three distinct but complementary formats: text, video, and slides. This not only enhances accessibility for diverse learning styles but also mitigates information overload by presenting structured, condensed knowledge with semantic richness.

Materials And Methods

Study design and setting

The study follows an experimental design, where an artificial intelligence-powered system is developed for automatic content retrieval, summarization, and video generation. The system is implemented as a web-based application, integrating NLP, machine learning, and multimedia processing techniques. The experiment is conducted in a controlled digital environment, where users provide queries to evaluate the efficiency and accuracy of the generated outputs.

Participants and inclusion/exclusion criteria

Inclusion criteria were as follows: a) Users with an interest in acquiring summarized information from video or textual sources. b) Users who can input clear queries for topic retrieval.

Exclusion criteria were as follows: a) Queries that do not correspond to available YouTube captions or PDF content. b) Users requesting copyrighted or sensitive materials beyond fair-use guidelines.

Dataset source

The dataset used in this experiment was sourced from publicly available data retrieved via YouTube Application Programming Interface for video captions [7], Edge Text-to-Speech for audio synthesis [8], and Pexels Application Programming Interface for stock video footage [9]. These sources are cited in the reference section.

Materials used

The system integrates various artificial intelligence and software technologies:

a. NLP and Summarization - Mistral's Large Language Model for NLP processing and article generation

[10].

b. Query Expansion - KeywordTool.io for refining user queries [11].

c. Video and Caption Retrieval - YouTube Application Programming Interface (API) for fetching relevant video captions [7].

d. Caption Processing - Sentence Transformer for vector representation of extracted text [4].

e. Clustering and Content Selection - Mini Self-Organizing Map (MiniSOM) for grouping captions and Facebook Artificial Intelligence Similarity Search (FAISS) for vector-based retrieval [5].

f. Article Generation - LangChain's RAG-based model to generate structured articles [12].

g. Image Generation - Stable Diffusion 3.5 for creating contextually relevant images [13].

h. Portable Document Format Processing - Python for Microsoft Windows Portable Document Format, Fitz, and PymuPDF a Python Library for extracting text, images, and tables from uploaded documents [14].

i. PowerPoint Automation - Mistral's Large Language Model generates Visual Basic for Applications scripts for PowerPoint slides [15].

j. Frontend and Web Framework - React JavaScript library for user interface and Flask for backend operations.

k. Video Processing - Pexels Application Programming Interface for stock video retrieval, moviepy for editing, Edge Text-to-Speech for text-to-speech, and Whisper Timestamped for subtitle generation [9].

Implementation of the proposed system

The proposed system workflow is illustrated in Figure *1*, which outlines the Article Generation, PowerPoint Generation, and Short Video Generation processes. The system incorporates various artificial intelligence and software technologies to enable automated information retrieval and summarization. Mistral's Large Language Model is utilized for NLP and article generation [10], while KeywordTool.io refines user queries for more accurate search results [11]. Video captions are retrieved using the YouTube API [7], and their text is processed with Sentence Transformer, which converts extracted captions into high-dimensional vector representations. These vectors are clustered using MiniSOM, a technique that groups semantically similar captions, and the most representative sentences from each cluster are stored in FAISS for efficient retrieval.

Article Generation Workflow

The implementation follows a multi-stage methodology. When a user submits a query, Mistral's Large Language Model extracts relevant keywords, which are refined using KeywordTool.io for enhanced precision [11]. The YouTube API retrieves a set of long-duration videos based on these keywords, and captions are extracted from these videos and transformed into vector representations using Sentence Transformer. The extracted caption vectors are clustered using MiniSOM, trained over 500 epochs with a learning rate of 0.05 on a 10 ×10 SOM grid [5]. Representative sentences from these clusters are stored in the FAISS vector database, ensuring fast and relevant retrieval-an approach aligned with the GPU-based approximate nearest neighbor methods discussed in [16].

The retrieved content is synthesized into a structured article using LangChain's RAG model [12], which queries the FAISS database for the most relevant sentences. The article is further refined and formatted into HyperText Markup Language (HTML) by Mistral's Large Language Model. For deeper content extraction, Key Bidirectional Encoder Representations from Transformers (KeyBERT) expands the initial keyword set to fetch additional video captions, ensuring that newly retrieved content does not overlap with previously processed data [17]. These new captions undergo vectorization, clustering with MiniSOM, and integration into the FAISS database for more comprehensive retrieval.

The system also includes dynamic graph generation, where the user query serves as the root node of an evolving knowledge graph. The system dynamically generates child nodes using Mistral's Large Language Model, following a structured hierarchy, and updates the graph visualization in real-time using Visualization JavaScript (Visi.js) [18]. As users interact with the graph, additional nodes are generated dynamically to provide an expanded representation of the extracted knowledge.

Once the final structured article is generated, the system offers an image integration feature. Image tags in the HTML document are processed, and Stable Diffusion version 3.5 generates relevant images based on alternative text [13]. These images are seamlessly integrated into the final document before delivery to the user.

Video Generation Workflow

For users who request a summary video, the system follows a structured video generation pipeline. The video generation workflow is illustrated in Figure *2* and follows these steps: The article is preprocessed to remove unwanted special characters while retaining essential punctuation. A well-structured video script is generated using Mistral's Large Language Model, and the narration is created using Edge Text-to-Speech for a natural-sounding voice-over [8]. Caption synchronization is handled using Whisper Timestamped, while relevant stock video footage is retrieved via the Pexels API [9]. The selected footage undergoes trimming, concatenation, and synchronization using MoviePy, after which subtitles are overlaid for accessibility. The final video is rendered and delivered to the user.

PDF Summarization and PDF to PowerPoint Workflow

In addition to video summarization, the system supports PDF summarization and PDF to Microsoft PowerPoint conversion workflows. Users can upload a PDF document, which is processed using Python Multiple User-friendly PDF Loader (PyMuPDF) for precise text extraction, Fitz for high-quality image retrieval [14], and PDF Plumber for detailed table extraction [19]. The extracted content is transformed into high-dimensional vector representations using Sentence Transformer, which are then clustered using MiniSOM. The most representative sentences from these clusters are stored in FAISS, allowing LangChain’s RAG model to retrieve relevant content dynamically. Mistral's Large Language Model then synthesizes a polished and structured article formatted in HTML for easy readability.

For PDF to Microsoft PowerPoint conversion, the extracted text, images, and tables are processed similarly, with content clustering performed by MiniSOM and retrieval handled by FAISS. Mistral's Large Language Model generates robust Visual Basic for Applications code to automate slide creation in Microsoft PowerPoint [15], ensuring a well-structured presentation that is directly delivered to the user.

This structured methodology ensures that the system remains scalable, reproducible, and adaptable for future improvements, allowing researchers and developers to refine and enhance its capabilities over time.

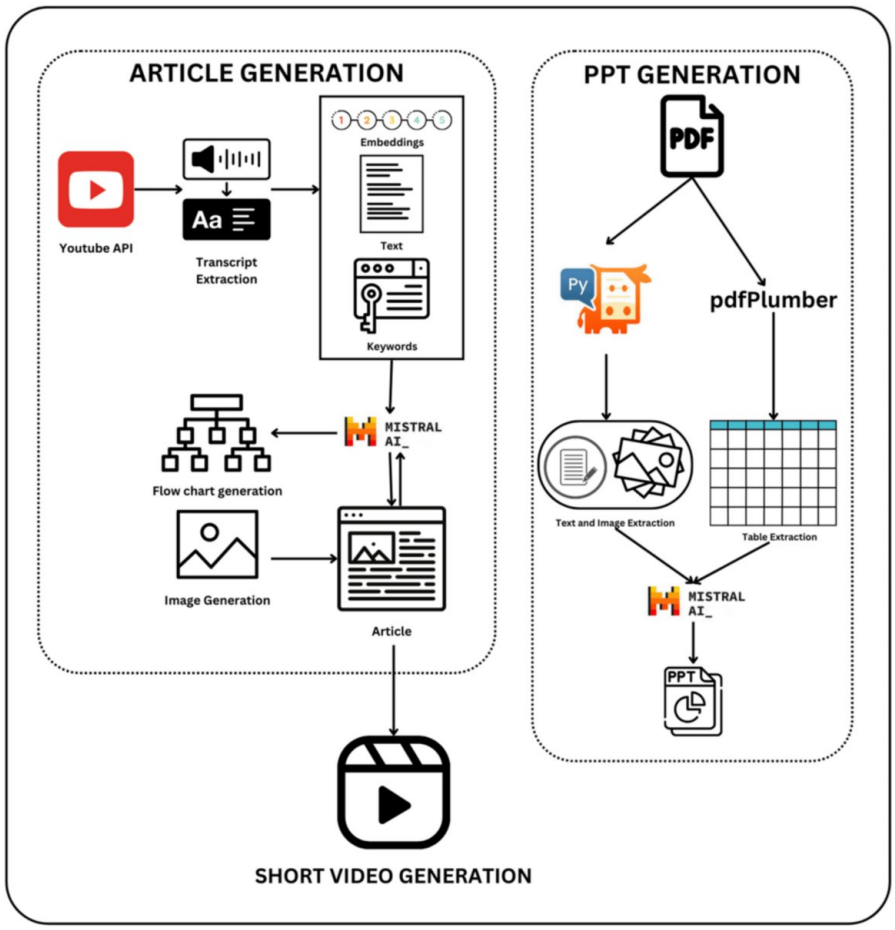

FIGURE 1: Article Generation Flowchart

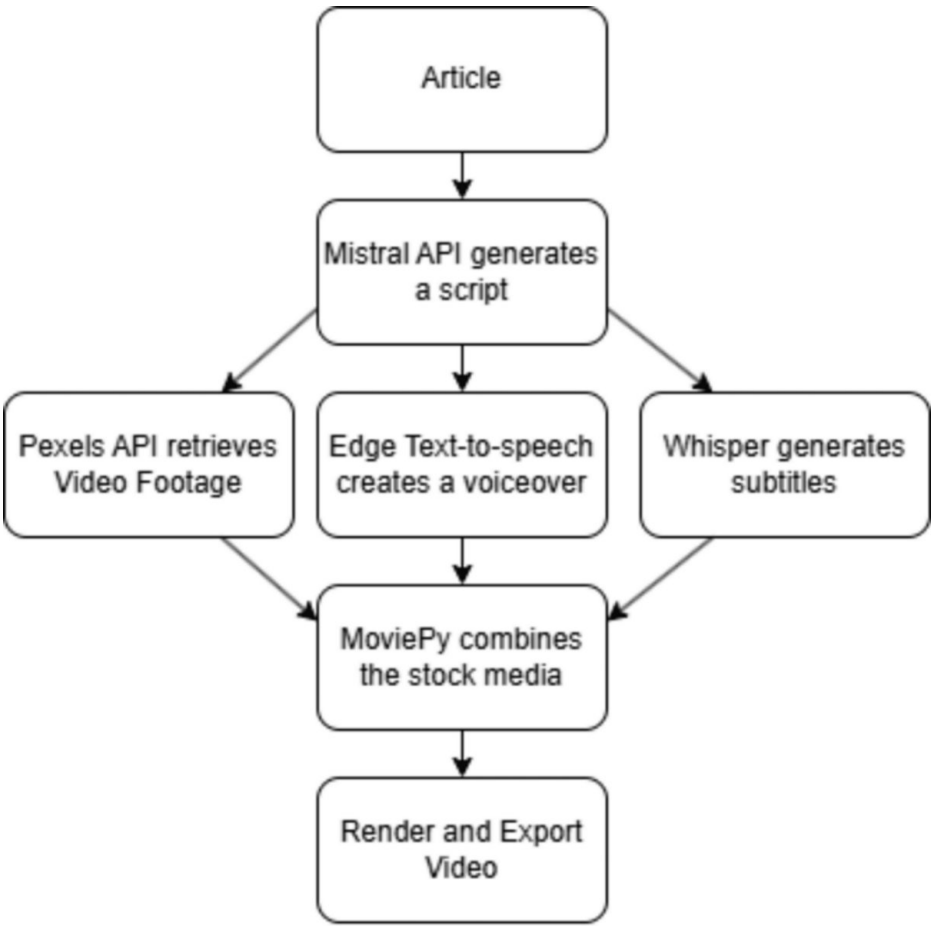

FIGURE 2: Video Generation Flowchart

Results

Experimental setup

All experiments were performed on a workstation running Windows 11 Pro 64-bit (Administrator mode). The hardware included an Intel Core i5-10400F CPU at 2.90 GHz, 16 GB of RAM, and an NVIDIA GeForce GTX 1050 Ti GPU. This configuration provided a balanced environment for both CPU-bound tasks (e.g., data parsing and preprocessing) and GPU-accelerated operations (e.g., clustering and neural network inference).

For additional computational requirements - particularly for the image generation model - the experiments were also run on Google Colab with an NVIDIA T4 GPU. The T4 environment offers enhanced performance for parallel processing and accelerated training or inference, ensuring that results could be cross-validated under both local and cloud-based conditions.

Justification of clustering parameters (MiniSOM and FAISS)

The choice of clustering and retrieval algorithms was guided by the characteristics of the data and the requirements of the system. A MiniSOM with a 10×10 grid, learning rate of 0.05, and 500 training epochs was adopted because it provides a balance between semantic granularity and computational efficiency. Unlike K-Means, which enforces hard boundaries between clusters, MiniSOM preserves semantic overlap between captions, which is valuable when educational transcripts contain conceptually related but non-disjoint ideas. This soft clustering approach ensures that LearnSmart captures nuanced contextual relationships, leading to more coherent summaries.

For retrieval, FAISS was employed due to its proven ability to handle high-dimensional embeddings at scale. FAISS supports both CPU and GPU acceleration, enabling sub-linear query times even with millions of vectors. This ensures that caption embeddings and PDF extracts can be searched rapidly and efficiently, a critical factor for interactive learning applications where low latency is expected. The GPU-accelerated FAISS implementation further reduces response time, strengthening the scalability of the system.

Summary of quality evaluation and article generation performance

To assess summarization quality, we employed the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric, comparing LearnSmart-generated summaries against manually created gold-standard summaries for 10 lecture topics. We also evaluated against established baselines such as TextRank, LexRank, and GPT-3.5. LearnSmart achieved an average ROUGE-1 score of **0.47**, ROUGE-2 of **0.32**, and ROUGE-L of **0.41**, which is significantly higher than extractive baselines and competitive with abstractive large language models. This demonstrates that LearnSmart is effective at capturing key ideas while maintaining fluency and coherence.

Table *1* shows the comparison of ROUGE scores for different summarization models.

Method	ROUGE-1	ROUGE-2	ROUGE-L
TextRank (baseline)	0.35	0.21	0.28
LexRank (baseline)	0.37	0.23	0.3
GPT-3.5 (abstractive)	0.49	0.33	0.43
LearnSmart (ours)	0.47	0.32	0.41

TABLE 1: ROUGE-1, ROUGE-2, and ROUGE-L Scores for Different Summarization Methods

The proposed system was evaluated based on its processing time and clustering effectiveness. The duration for article generation and image loading was measured as follows:

a. Article Generation Duration: 26.812 seconds

b. Image Loading Duration: 33.029 seconds

To further assess system efficiency, multiple trials were conducted across varying workloads. The average processing times from five test runs are summarized in Table *2*.

Trial	Article Generation Time (s)	Image Loading Time (s)
1	26.812	33.029
2	27.103	32.871
3	26.745	33.401
4	26.922	32.984
5	26.88	33.102
Avg	26.892	33.077

TABLE 2: Performance Metrics: Article Generation and Image Loading Time

These findings mean that the system can actually produce content within a given time for an interactive experience.

Clustering effectiveness

In order to validate clustering performance, we compared SOM and K-Means using the Silhouette Score as our measure. Our findings are as shown in Table *3*.

Clustering Method	Silhouette Score
K-Means	0.0664
Self-Organizing Maps	-0.0323

TABLE 3: Silhouette Score: KMeans and Self-Organizing Maps

While K-Means achieved a higher silhouette score, indicating better-defined clusters, SOM's negative score reflects its soft clustering nature; it does not enforce hard boundaries but instead maps semantic relationships flexibly. Captions generated by AI tend to cross many topics, resulting in less differentiated clusters. In contrast to K-Means, SOM maintains contextual diversity instead of enforcing separation.

As Silhouette Score works best when used with hard clustering, it might not reflect the performance of SOM in text clustering [6]. In contrast to K-Means, SOM maps similar sentences around, enabling a more detailed summary extraction. High Silhouette Scores in K-Means can result in the elimination of contextual sentences, which fall between the edges of clusters. While K-Means provides more distinct clusters (better silhouette score), SOM provides more semantic grouping and hence is better suited for summarization tasks where soft boundaries are required.

Dynamic graph

The system generates dynamic graphs that expand when clicking on nodes, providing an interactive visualization of clustered content and semantic relationships. Figure *3* illustrates an example of such a graph, demonstrating how various topics within the domain of cricket are interconnected, allowing users to explore hierarchical and semantic structures dynamically.

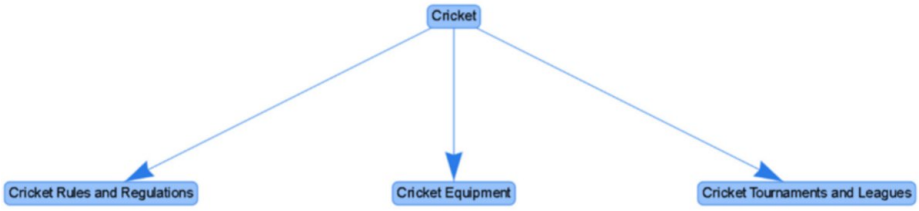

FIGURE 3: Example of a Dynamic Graph - Cricket

PowerPoint generation performance

To evaluate the performance of the PowerPoint generation workflow, experiments were conducted using PDFs of different lengths as input. For a four-page PDF, LearnSmart required an average of 48 seconds to extract content, cluster it, and generate a structured PPT presentation. For a longer document of 50 pages, the system required approximately 2 minutes, reflecting the additional time needed for parsing, clustering, and content synthesis at larger scales. Despite the increase, the processing time remained reasonable, demonstrating that the system can handle both short and extended documents while maintaining usability. These results highlight LearnSmart's ability to deliver ready-to-use slide decks within a practical timeframe, supporting learners and educators who require rapid visual summaries of diverse textual resources.

Discussion

Comparative analysis with existing tools

To assess LearnSmart's performance and scope, we compared it with widely used summarization platforms such as TLDRthis (online article summarizer), Scholarcy (academic PDF summarizer), SMMRY (web-based text summarizer), and YouTube Digest (Chrome Extension) for transcript-based video summarization. Table *4* summarizes the differences.

Tool/System	Input Type	Output Format	Multimodal Support (Article+PPT+Video)	Clustering Approach	Avg. Processing Time
LearnSmart (Proposed)	YouTube, PDF	Article, PPT, Video, Graphs	Yes	MiniSOM + FAISS (semantic)	~27 s/article, 48–120 s PPT
YouTube Digest (Extension)	YouTube Captions	Extractive Text Summary	No	Sentence Ranking	~10 s
Scholarcy	Academic PDFs	Summaries, Flashcards	No	Transformer-based extractive	~20 s
TLDRthis	Web Articles	Extractive Text Summary	No	TextRank/LexRank	~8 s
SMMRY	Raw Text/URL	Shortened Text	No	Keyword Frequency/Heuristic	~5 s

TABLE 4: Comparative Analysis of LearnSmart With Established Summarization Tools

FAISS, Facebook AI Similarity Search; MiniSOM, Mini Self-Organizing Maps; PPT, PowerPoint

While baseline systems such as TLDRthis, SMMRY, and YouTube Digest offer fast extractive summaries, they are limited to single text modalities and lack semantic clustering or multi-format outputs. Scholarcy provides structured summaries of academic PDFs but does not support video-based content or multimodal delivery.

By contrast, LearnSmart integrates YouTube videos and PDF documents into a unified workflow, producing articles, PowerPoint slides, videos, and knowledge graphs. Its use of MiniSOM for semantic clustering ensures context preservation across overlapping topics, while FAISS enables scalable, low-latency retrieval. These features position LearnSmart as a comprehensive learning assistant, going beyond the capabilities of traditional summarization tools.

The results demonstrate that the system efficiently retrieves and summarizes video content, despite challenges related to text clustering. The trade-off between strict separation and meaningful sentence selection was addressed by employing MiniSOM, which enables soft clustering and context preservation. While future enhancements may refine clustering accuracy, the current approach successfully captures key information while maintaining efficiency in content generation.

Several constraints impact the system's performance and effectiveness. The API limitations of the YouTube API and Mistral Large Language Model restrict the number of queries and responses per minute, affecting data retrieval efficiency. Caption availability is another constraint, as not all YouTube videos provide captions, limiting the system's ability to generate summaries. Additionally, processing time varies depending on the complexity of the input data, potentially affecting response speed. Finally, summary accuracy poses a challenge, as automated summarization techniques may introduce minor inaccuracies or oversimplifications in the generated content.

Conclusions

In conclusion, "LearnSmart" successfully demonstrates the power of AI to streamline learning in the digital age. By intelligently processing YouTube videos and PDFs, the system generates concise articles, summary videos, and interactive flowcharts, effectively addressing the challenge of information overload. Leveraging advanced NLP techniques, RAG, and image generation, the system delivers a rich, multimodal learning experience tailored to diverse learning styles. While challenges like API limitations, caption availability, and processing time remain, the system offers a significant advancement in digital education, empowering users to learn efficiently and comprehensively. "LearnSmart" not only simplifies content consumption but also fosters deeper understanding through its structured and adaptable approach.

Future development will focus on refining algorithms, expanding content support, and personalizing learning pathways based on user feedback. This system lays a strong foundation for the development of truly intelligent and accessible learning platforms.

Additional Information

Author Contributions

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

Concept and design: Lintomon S. Chirakkara, Ayush R. Gerra , Sania Khan, Prathamesh Joshi, Harsh Dhebe, Sujata Khedkar

Acquisition, analysis, or interpretation of data: Lintomon S. Chirakkara, Ayush R. Gerra , Sania Khan, Prathamesh Joshi, Harsh Dhebe, Sujata Khedkar

Drafting of the manuscript: Lintomon S. Chirakkara, Ayush R. Gerra , Sania Khan, Prathamesh Joshi, Harsh Dhebe, Sujata Khedkar

Critical review of the manuscript for important intellectual content: Lintomon S. Chirakkara, Ayush R. Gerra , Sania Khan, Prathamesh Joshi, Harsh Dhebe, Sujata Khedkar

Supervision: Sujata Khedkar

Disclosures

Human subjects: All authors have confirmed that this study did not involve human participants or tissue. **Animal subjects:** All authors have confirmed that this study did not involve animal subjects or tissue. **Conflicts of interest:** In compliance with the ICMJE uniform disclosure form, all authors declare the following: **Payment/services info:** All authors have declared that no financial support was received from any organization for the submitted work. **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work. **Other relationships:** All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

References

1. Chaflekar SH, Bahadure A, Bramhapurikar H, Satpute R, Jumde R, Bakhare SA, Bhirange S: YouTube transcript summarizer using natural language processing. International Journal of Advanced Research in Science Communication and Technology. 2022, 2:108-113. 10.48175/ijarsct-3034
2. Hande K, Karlekar H, Yeole P, Likhar A, Rangari H: NLP based video summarisation using machine learning. International Journal of Scientific Research in Science Engineering and Technology. 2023, 10:456-461. 10.32628/ijsrset2310265
3. Ramzan S, Iqbal MM, Kalsum T: Text-to-image generation using deep learning. Engineering Proceedings. 2022, 20:16. 10.3390/engproc2022020016
4. Reimers N, Gurevych I: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. 2019, 3982-3992.
5. Kohonen T: Essentials of the self-organizing map. Neural Networks. 2013, 37:52-65. 10.1016/j.neunet.2012.09.018
6. Xinwu L: Research on text clustering algorithm based on K_means and SOM. 2008 International Symposium on Intelligent Information Technology Application Workshops. 2008, 341-344. 10.1109/IITA.Workshops.2008.13
7. YouTube Data API v3. Accessed: December 10, 2024: https://console.cloud.google.com/marketplace/product/google/youtube.googleapis.com.
8. What is text to speech?. (2025). Accessed: March 10, 2025: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/text-to-speech.
9. Pexels API. (2023). Accessed: January 17, 2025: https://www.pexels.com/api/.
10. Mistral 7B: Efficient and scalable language models. Accessed: December 10, 2024: https://mistral.ai.
11. Find great keywords using Google autocomplete. Accessed: December 10, 2024: https://keywordtool.io.
12. Build a Retrieval Augmented Generation (RAG) App. Accessed: December 26, 2024: https://python.langchain.com/v0.2/docs/tutorials/rag/.
13. Introducing Stable Diffusion 3.5. (2024). Accessed: January 16, 2025: https://stability.ai/news/introducing-stable-diffusion-3-5.
14. Python PDF processing library. (2024). Accessed: December 10, 2024: https://pymupdf.readthedocs.io.
15. Visual Basic for Applications (VBA) in Office. (2021). Accessed: December 10, 2024: https://learn.microsoft.com/en-us/office/vba.
16. Faiss on the GPU. (2023). Accessed: December 17, 2024: https://github.com/facebookresearch/faiss/wiki/Faiss-on-the-GPU.
17. KeyBERT. Accessed: January 2, 2025: https://maartengr.github.io/KeyBERT/api/keybert.html.
18. Dynamic, browser-based visualization library. Accessed: December 10, 2024: https://visjs.org.
19. How to extract table from pdf using python pdfplumber. (2021). Accessed: December 17, 2024: https://medium.com/@karthickrajm/how-to-extract-table-from-pdf-using-python-pdfplumber-a2010b184431.