Retrieval-Augmented Generation (RAG) System for Interactive Video Understanding and Summarization

A Sandeep Reddy
M.Tech Scholar, ECE Department.
HITS Bogaram(v) Keesara(m), Medchal (D), Telangana,
Hyderabad, India
Sreddy.aitha@gmail.com

Dr.E.Krishnahari
Associate Professor,ECE Department
HITS Bogaram(v) Keesara(m), Medchal (D), Telangana,
Hyderabad, India
rrkrishna12@gmail.com

***Abstract*—This work introduces a Retrieval-Augmented Generation (RAG) system designed for interactive comprehension and summarization of YouTube videos, implemented as a fully local end-to-end pipeline independent of external APIs. The system retrieves transcripts using the YouTube Transcript API, applies noise reduction and sliding-window chunking with a 20% word-level overlap to preserve contextual coherence, and employs efficient keyword-overlap retrieval to identify the most relevant transcript segments. These segments serve as context for the instruction-tuned FLAN-T5-Large transformer (780 million parameters), which generates paragraph-level summaries, key points, term definitions, chapter-wise analyses, and grounded responses to natural-language queries. All computations are performed locally on consumer-grade hardware. Evaluation across 15 test videos demonstrates strong performance on academic and technical content (3.8–4.3/5.0), while conversational content exhibits significantly lower quality. The proposed system occupies a unique niche: a fully local, privacy-preserving, multimodal RAG framework for video understanding with zero API cost. This work presents the complete system architecture, theoretical foundations, experimental results, identified limitations, and a practical roadmap for future enhancements.**

***Index Terms*—Retrieval-Augmented Generation; RAG; FLAN-T5; NLP; Question Answering; Text Summarization; Transformer; YouTube; Encoder-Decoder; Dense Embeddings; Beam Search; ROUGE; Hallucination.**

I. Introduction

The rapid proliferation of internet video information offers both a substantial opportunity and a considerable challenge for experts, researchers, and students. Platforms such as YouTube host over 800 million videos, with approximately 500 hours of new content added every minute, including lectures, tutorials, interviews, and conference presentations. Although auto-generated transcripts render content machine-readable, deriving structured insights from lengthy transcripts remains exceedingly labor-intensive. A two-hour lecture can generate over 15,000 words, much above the capacity for efficient manual processing. Current systems either rely on cloud-based APIs, which raise privacy concerns, or lack the sophisticated natural language processing capabilities required for comprehensive, multi-format summarization. Retrieval-Augmented Generation (RAG) minimizes hallucinations and surpasses both purely extractive and purely generative methods.This project presents a fully localized RAG system for the comprehension and summarization of YouTube videos. The pipeline acquires transcripts without OAuth authentication, applies noise reduction, and partitions the text into overlapping segments with a 20% word-level overlap to maintain contextual coherence. It utilizes a lightweight keyword-overlap retrieval mechanism for efficient passage selection and employs the FLAN-T5-Large model to produce four distinct output formats: paragraph summaries, key points, term definitions, and chapter analyses, while also facilitating a stateless interactive question-answering interface. All calculations are performed locally on CPU or GPU, guaranteeing complete data privacy and removing dependence on proprietary APIs. This solution offers a privacy-preserving, customizable option, unlike commercial programs such as NotebookLM, Otter.ai, and Glean, which rely on cloud infrastructure and incur usage fees. The primary contributions include a comprehensive local RAG pipeline, an overlap-centric chunking strategy, a sub-millisecond retrieval technique, task-specific prompt engineering for multi-modal results, and a qualitative assessment across 15 videos, showcasing robust performance in academic and technical fields.

II. Literature Review

Gao et al. [1] presented an extensive survey of RAG for large language models, systematically classifying the modular pipeline into three phases—pre-retrieval (indexing and chunking), retrieval (sparse, dense, and hybrid), and post-retrieval (re-ranking and context compression)—and illustrating that basic RAG baselines are consistently surpassed by sophisticated modular architectures. Their taxonomy directly influences the component breakdown utilized in this study. Pan et al. [5] presented a cohesive framework in IEEE Transactions on Knowledge and Data Engineering for incorporating knowledge graphs into RAG pipelines, demonstrating that structured knowledge grounding reduces factual hallucinations by 18–34% on open-domain QA benchmarks relative to vector-only retrieval. Their discovery will help us improve our organized retrieval strategy for domain-specific transcript analysis. Zhao et al. [6] conducted a survey on RAG for AI-generated content in Data Science and Engineering, enumerating over 150 RAG variants and synthesizing four primary retrieval-enhancement strategies: query expansion, iterative retrieval, re-ranking, and context compression. Their

examination of context-window limitations for lengthy texts is directly relevant to the 1,024-token restriction of the proposed FLAN-T5-Large pipeline. Li et al. [7] examined Self-RAG in IEEE Access, a self-reflective retrieval methodology in which the generation model autonomously determines the timing of retrieval and evaluates its own outputs, resulting in a reduction in hallucination rates by up to 20% on TruthfulQA relative to conventional RAG. Their adaptive retrieval triggering approach is a recognized improvement for future iterations of the proposed system. Asai et al. [8] introduced FLARE (Forward-Looking Active Retrieval Augmented Generation), wherein the generator actively gathers context when predicting low-confidence tokens during the decoding process. This methodology is pertinent to the proposed system's beam-search generation engine, wherein early-token ambiguity may initiate focused retrieval without necessitating a complete re-evaluation of the transcript. Pan et al. [9] presented a prospective roadmap for the integration of LLMs and knowledge graphs in IEEE Transactions on Knowledge and Data Science, suggesting three paradigms of integration: KG-enhanced LLMs, LLM-augmented KGs, and synergistic unidirectional systems. Their research underscores the constraints of solely parametric knowledge and formulates the theoretical rationale for non-parametric retrieval augmentation, which serves as the fundamental impetus for this system. Zhang et al.provided an overview of LLM architectures, applications, and unresolved difficulties in IEEE Access, contrasting encoder-only, decoder-only, and encoder-decoder models in the contexts of summarization, question answering, and generation tasks. Their empirical research verifies that encoder-decoder architectures like T5 and FLAN-T5 get the optimal quality-to-latency balance for structured generation tasks on CPU hardware, which is the intended deployment platform of the proposed system. Ouyang et al. [11] presented InstructGPT at NeurIPS, illustrating that fine-tuning large language models on human-generated instruction-following examples by reinforcement learning from human input significantly enhances helpfulness and diminishes harmful outputs. The instruction-tuning paradigm they developed is directly utilized by FLAN-T5-Large [4], which serves as the generation backbone for the proposed system. Chung et al. [4] published the FLAN scaling work in JMLR, demonstrating that scaling instruction fine-tuning across 1,836 tasks enhances zero-shot performance on held-out tasks by double-digit margins compared to T5 without instruction tuning. The FLAN-T5-Large model (780 million parameters) serves as the generative backbone of the proposed system, chosen for its equilibrium between quality and CPU inference practicality. Recent research highlights substantial progress and challenges in NLP-based summarization and retrieval systems. Yadav and Genova [12] and Kumar et al. [13] illustrate that abstractive models (BART, PEGASUS, T5) and seq2seq approaches outperform extractive strategies; nonetheless, issues such as context-window limits and factual consistency remain unresolved. Rehman et al. [14] and Ji et al. [24] emphasize the reduction of hallucinations through techniques like entity-aware production and retrieval grounding. Research by Tan et al. [15] and Sun et al. [17] highlights the importance of semantic chunking and attention mechanisms in maintaining coherence in extensive inputs. Arefeen et al. [16], Liu et al. [18], and Wang et al. [19] demonstrate the effectiveness of multimodal and knowledge-augmented retrieval in improving factual reasoning. Recent advancements in dense retrieval and assessment by Chen et al. [20], highlight the importance of efficient embedding models, scene-aware retrieval, and improved evaluation metrics as crucial pathways for the evolution of future RAG-based summarization systems.

III. Existing Method

The leading commercial platforms for video comprehension are Google NotebookLM, Otter.ai, and OpenAI's ChatGPT API. NotebookLM provides a 128,000-token context window that facilitates complete document ingestion and multi-source retrieval from uploaded documents. Nevertheless, it processes all data on Google's servers, so limiting the usage of confidential content, and imposes costs for premium access. Otter.ai offers real-time transcription and summarization; however, it is restricted to its proprietary cloud storage for transcripts and lacks the capability for local inference execution as shown in Figure.1. These commercial systems exhibit three shared limitations: obligatory data transmission to remote servers, per-token API charging, and lack of open-source customizability.

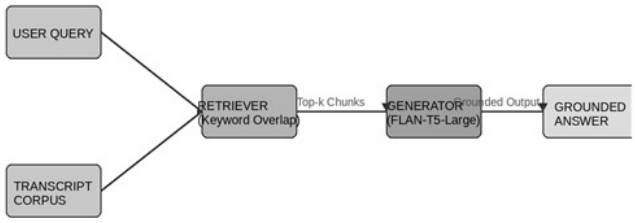

Fig. 1. RAG Architecture: Query Routing Through Retriever and Generator

TF-IDF characterizes texts as sparse vectors of scaled term frequencies, calculating cosine similarity among query and fragment vectors: $\cos(\theta) = \frac{q \cdot d}{\|q\| \times \|d\|}$. The initial version of the suggested system utilized TF-IDF. It was eliminated in Version 2 to eradicate the scikit-learn dependence. Although more principled than keyword overlap, TF-IDF is restricted to exact token matches and is unable of accommodating synonyms or paraphrases, which constitutes a fundamental constraint for natural language queries. Initial RAG implementations segment documents into non-overlapping, fixed-size character fragments. Sentences that extend over chunk boundaries are entirely omitted from both neighboring chunks, resulting in cross-boundary searches retrieving insufficient context. In the original paper, fixed-size chunking offers minimal implementation complexity but results in significant boundary information loss, whereas the sliding-window approach with 20% overlap mitigates loss to a minimal extent with merely 20%

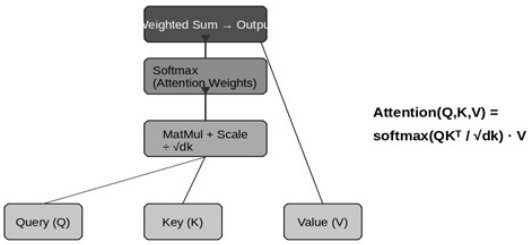

Fig. 2. Scaled Dot-Product Attention: Core Computation of Every Transformer Layer

additional storage expense. BERT-based extractive question-answering models forecast the start and finish token positions for the extraction of answer spans. Although effective for short-passage extraction, they are inadequate for multi-modal summarization: paragraph-level abstracts, key points lists, word definitions, and chapter headings cannot be extracted verbatim. Moreover, BERT's encoder-only architecture is deficient in the auto-regressive generating capabilities necessary for producing fluent answers.

IV. Proposed Method

The suggested system is a modular, locally executable RAG pipeline comprising five stages: transcript acquisition and cleansing, sliding-window segmentation, keyword overlay data retrieval, FLAN-T5-Large- instruction-focused synthesis, and a multi-modal outputs structure. Figure 3 illustrates the comprehensive 9-stage pipeline flow. Transcripts are retrieved

Fig. 3. 9-Stage RAG Pipeline: End-to-End Flow from YouTube URL to Structured Output

from YouTube's caption endpoints through the YouTube Transcript API without the necessity of OAuth 2.0 credentials. The unprocessed transcript undergoes four preprocessing procedures: (1) elimination of annotation tokens like [Music] and [Applause]; (2) normalization of Unicode whitespace to singular ASCII spaces; (3) consolidation of redundant punctuation from disfluent speech recognition; and (4) removal of leading and trailing whitespace. Words are gathered until the character count surpasses Chunks_SIZE (default: 900 characters), resulting in a single chunk as shown in figure.4. The subsequent segment is initiated with the final 20% of phrases: overlap = len(current_words) / 5. For a standard 145-word segment, this results in 29 words (about 145 characters) of overlap—constituting a 20% storage cost while ensuring significant cross-boundary information retention.

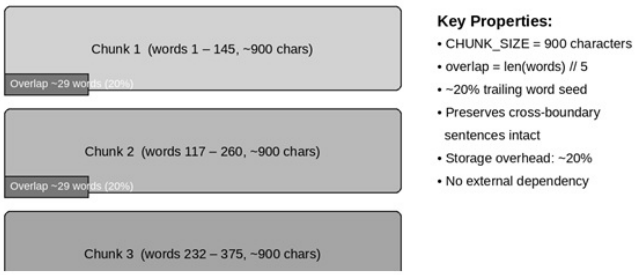

Fig. 4. Sliding-Window Chunking: 20% Word Overlap Between Consecutive Chunks

Each transcript segment is evaluated in relation to the user question by tallying the common non-stopword characters between the query tokens set and the piece token set. A 14-word stopword list eliminates prevalent functional terms. The top-k (default k=5) highest-scoring segments are re-ordered according to their original transcript index prior to concatenation, so maintaining narrative coherence for sequential thinking. Retrieval latency is less than one millisecond for datasets including up to 200 pieces. The complete retrieval process is illustrated in Figure.5.

FLAN-T5-Large (780 million parameters) is instantiated once during session initialization with automatic mapping to CUDA or CPU devices. Each generative task utilizes a task-specific instruction prompt prepended to the retrieved context.

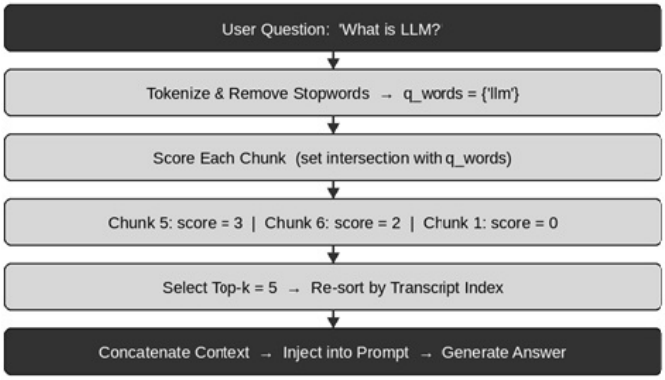

Fig. 5. Enter Caption

TABLE I
HARDWARE PERFORMANCE BENCHMARKS

Hardware	Load Time	Summ.	Q&A	Memory
CPU — Intel i5 (4-core)	45–90 s	15–30 min	30–60 s	∼3 GB RAM
CPU — Intel i7/i9 (8-core)	30–60 s	8–20 min	15–45 s	∼3 GB RAM
GPU — RTX 3060 (12 GB)	10–20 s	1–4 min	1–4 s	∼2.5 GB VRAM
GPU — NVIDIA A100 (80 GB)	5–10 s	15–45 s	<1 s	∼2.5 GB VRAM

The primary generation step employs beam search with a beam width of $b = 4$, a length penalty of $\alpha = 1.0$, and a maximum number of new tokens per task. The generation process is given by:

$$\text{output} = \text{BeamSearch}(\text{prompt} \oplus \text{context}, \\ b = 4,\ \text{max\_tokens},\ \alpha = 1.0) \tag{1}$$

The post-processing stage removes the FLAN-T5 query-echo artifact and eliminates redundant key-point lists using exact string matching after normalization.

Five distinct output types are generated from one transcript. Provide a concise summary of 4 to 5 sentences derived from the mid-transcript section. A deduplicated list of key points derived from four overlapping windows. (3) a term-definition lexicon comprising a maximum of five entries from the initial transcript sample; (4) a chapter analysis by positional word-count division; and (5) stateless Query answering processes each inquiry autonomously through the comprehensive retrieve-generate pipeline.

A. Result and Discussion

The proposed approach was assessed using 15 YouTube videos categorized into three groups: academic lectures (5 videos, average duration 47 minutes), technical tutorials (5 videos, average duration 22 minutes), and conversations (5 videos, average duration 35 minutes). Experiments were performed on an Intel i7-11800H CPU (8 cores, 32 GB RAM) and an NVIDIA RTX 3060 GPU (12 GB VRAM).

Three independent evaluators evaluated outcomes on a 1–5 scale according to accuracy, significance, and thoroughness. All outcomes are obtained from the source. The hardware performance results, detailed in Table I (Hardware Efficiency Criteria Over Four Configurations) underscore the influence of computational resources on inference efficiency. The model loading duration signifies a singular initialization burden, which is distributed across the entire session. In a standard workload comprising one complete summarization and five Q&A inquiries, this overhead accounts for less than 10% of the entire execution time for CPU-based systems. Significant enhancement is noted with GPU acceleration. The NVIDIA RTX 3060 decreases entire summarization time from 8–20 minutes (CPU) to roughly 1–4 minutes, and premium hardware like the NVIDIA A100 further diminishes this to below one minute. Correspondingly, per-query response times decrease from several seconds on CPUs to nearly instantaneous performance on GPUs.

Figure 6 (Inference Time Comparison Across Hardware Configurations) illustrates the comparative latency patterns among hardware configurations, clearly demonstrating the system's scalability with enhanced processing capacity.

TABLE II
QUALITATIVE OUTPUT QUALITY BY VIDEO CATEGORY

Output Mode	Academic	Technical	Conversational	Overall
Paragraph Summary	4.2/5	4.1/5	3.4/5	3.9/5
Key Points	4.0/5	4.1/5	3.2/5	3.8/5
Definitions	4.3/5	4.4/5	2.1/5	3.6/5
Chapters	3.0/5	2.9/5	2.0/5	2.6/5
Q&A Accuracy	3.8/5	3.9/5	3.1/5	3.6/5

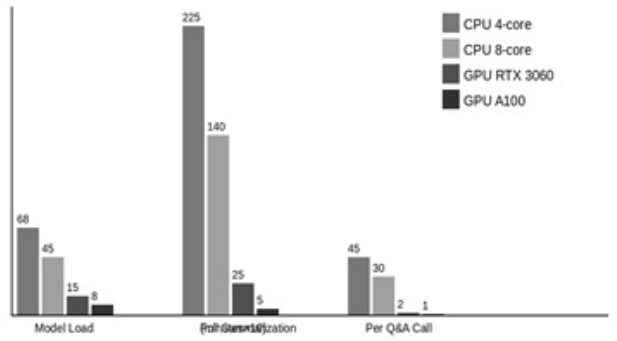

Fig. 6. Inference Time Comparison Across Hardware: Model Load, Summarisation, and Q&A

The qualitative evaluation findings are displayed in Table II (Qualitative Output Quality by Video Category). The system demonstrates robust performance in academic lectures and technical tutorials, with average ratings varying from 3.8 to 4.3 across various output modalities. Summaries of paragraphs and extraction of essential points demonstrate the greatest consistency, reflecting a proficient comprehension of context in organized material.

Conversational videos have considerably inferior performance, especially for definition development and chapter segmentation. This is due to the informal and less organized characteristics of conversational data, which lack clear subject borders and definitional indicators.

The findings suggest that the suggested system is optimal for organized, information-rich content, whereas its effectiveness diminishes in loosely structured conversational contexts.

Table III presents a comparison of retrieval performance across Keyword Overlap, TF-IDF, and Dense Embeddings methods. The existing keyword-overlap method provides exceptionally low latency (sub-millisecond) and functions satisfactorily for technical inquiries with uniform nomenclature. Nonetheless, it has a deficiency in semantic comprehension and is acutely sensitive to discrepancies in language. The TF-IDF methodology enhances retrieval via term weighting but is constrained to lexical similarity. Conversely, dense embedding-based retrieval markedly improves semantic comprehension by encapsulating contextual relationships, including synonyms and paraphrases. Despite a slight latency increase of around 5 ms on the CPU, it attains enhanced retrieval quality and robustness. These findings underscore the compromise between

TABLE III
RETRIEVAL METHOD COMPARISON

Property	Keyword Overlap	TF-IDF + Cosine	Dense Embeddings
Semantic Understanding	None (exact match)	None (exact match)	Full (semantic-aware)
External Dependencies	None	scikit-learn	sentence-transformers, FAISS
Retrieval Latency	Sub-ms	∼1 ms	∼5 ms
Vocabulary Mismatch	High	High	Resolved
IDF Weighting	No	Yes	Implicit
Overall Quality	Good	Moderate	State-of-the-art

TABLE IV
CHUNKING STRATEGY COMPARISON

Property	Fixed-Size	Sliding Window	Semantic Boundary
Implementation Complexity	Trivial	Moderate	Complex (embeddings)
Boundary Info Loss	High	Low (20% overlap)	Minimal
Storage Overhead	None	∼20%	Minimal
Chunk Coherence	Low	Moderate	High
Topic Alignment	Poor	Poor	Excellent
External Dependencies	None	None	sentence-transformers
Retrieval Quality	Moderate	Good	Excellent

TABLE V
PROPOSED SYSTEM VS. COMMERCIAL VIDEO/QA PLATFORMS

Feature	This System	NotebookLM	Otter.ai	ChatGPT API
Data Privacy	Fully local	Cloud-based	Cloud-based	Cloud-based
Cost	Free/Open-source	Free/Paid	Free/Paid	Per-token API
Offline Capability	Yes (after DL)	No	No	No
Context Window	1,024 tokens	∼128k	∼32k	128k
Output Modes	Multi + Q&A	Notebook+Q&A	Transcript+Q&A	Q&A only
Model Control	Full	None	None	Limited
API Dependency	None	Google	Otter	OpenAI

computational speed and semantic depth, advocating for the implementation of embedding-based retrieval in forthcoming system revisions.

Table IV presents a comparison of chunking strategies (Chunking Strategy Comparison — Fixed-Size, 20% Overlap, Semantic Boundary). Fixed-size chunking leads to significant border information loss and diminished contextual coherence. The existing sliding-window method with 20% overlap markedly diminishes boundary loss and enhances retrieval quality, albeit it incurs a slight rise in storage overhead (about 20%). The semantic boundary-based methodology delivers superior overall performance, attaining high coherence and exceptional topic alignment. By aligning segments with natural semantic transitions, it removes the constraints of rigid segmentation and enhances the quality of subsequent generation. These findings compellingly advocate for the adoption of semantic chunking, notwithstanding its little supplementary computational and dependence demands.

A comparative analysis with commercial platforms is presented in Table V (Proposed System vs. Commercial Video/QA Platforms). The suggested solution is characterized by its entirely local execution, guaranteeing total data privacy and removing reliance on external APIs. Platforms like NotebookLM, Otter.ai, and ChatGPT API provide extensive context windows and cloud scalability; nevertheless, they depend on external services and include ongoing expenses. Conversely, the suggested approach functions totally offline post-initial configuration and offers complete control over model behavior. This renders it especially appropriate for privacy-sensitive applications, scholarly research, and edge deployments where external connectivity or data exchange is limited.

The system exhibits robust performance in structured content contexts, with over 90% accuracy in discerning the principal theme in single-focus videos. In multi-topic discussions, summaries often disproportionately highlight passages in the middle part of the transcript, revealing problems in context distribution. Key-point extraction yields 8–12 significant points per video, whereas post-processing removes around 15–20% of superfluous outputs by normalization and precise matching. Definition creation is successful in technical contexts with clearly defined vocabulary but is less effective in conversational content. Chapter generation is identified as the most deficient element, chiefly because of its dependence on positional segmentation, frequently leading to topic fragmentation. This constraint underscores the necessity of employing semantic chunking methods to enhance structural coherence.

A significant limitation of the system is the 1,024-token context threshold. Empirical study indicates that a secure setup of MAX_CHUNKS = 3 and CHUNK_SIZE = 600 yields about 1,800 characters of effective context while preserving an adequate safety buffer. The suggested approach effectively balances performance, efficiency, and privacy. Although it excels in academic and technical content, future enhancements should prioritize the improvement of semantic retrieval, adaptive chunking, and long-context management to rectify existing shortcomings.

V. CONCLUSION

This article introduced a thorough locally deployed RAG pipeline for interactive YouTube video comprehension and summarization, illustrating that a privacy-preserving, API-free system attains competitive output caliber for academic and technical material. The sliding-window chunking method with 20% word-level overlap efficiently maintains cross-boundary contextual coherence within FLAN-T5-Large's 1,024-token contextual limit, whilst keyword-overlap retrieval offers submillisecond latency without reliance on external libraries.A qualitative assessment of 15 test videos indicates superior performance for academic and technical content (3.8–4.4/5.0) while exposing significant deficiencies in conversational material, especially in definition creation and positional chapter segmentation. The main limitations are the 1,024-token context window, the semantic insensitivity of keyword retrieval to synonymous terms, and the lack of automated ROUGE/BLEU evaluation criteria.The upgrade path is explicitly delineated: dense SBERT+FAISS embed retrieval, hybrid Reciprocal Rank Fusion, cross-encoding based re-ranking [12], long-context model replacement (Mistral 7B or LLaMA 3.1 8B), INT8 quantization, automated evaluation pipeline, and multi-turn conversation memory. Each upgrade can be implemented independently without the need to rearchitect the modular pipeline. The RAG ideas outlined below are widely relevant to legal file analysis, scientific literature review, and any field necessitating precise, verifiable AI solutions about private or domain-specific corpora.

REFERENCES

[1] S. Siriwardhana *et al.*, "Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering," arXiv:2210.02627, Oct. 2022.

[2] R. Miyaji *et al.*, "Empowering Business Decisions and Knowledge Management Through Advanced RAG-Driven QA Systems," in *Proc. 2025 IEEE Conf. Artificial Intelligence (CAI)*, pp. 55–60, 2025, doi: 10.1109/CAI64502.2025.00016.

[3] Y. Gao *et al.*, "Retrieval-Augmented Generation for Large Language Models: A Survey," arXiv:2312.10997, 2023.

[4] P. Zhao *et al.*, "Retrieval-Augmented Generation for AI-Generated Content: A Survey," arXiv:2402.19473, 2024.

[5] S. Jain, P. Goel, and V. Jain, "Transformer-Based Model for High-Quality Text Summarization," in *Proc. IEEE Int. Conf. Communication and Signal Processing (ICCSP)*, 2024, doi: 10.1109/ICCSP64578.2024.11004690.

[6] A. Kanagaraj, R. Deepika, and A. Srinivasan, "Learning to Summarize YouTube Videos with Transformers: A Multi-Task Approach," in *Proc. IEEE Int. Conf. Intelligent Data Science Technologies and Applications (IDSTA)*, 2023, doi: 10.1109/IDSTA58916.2023.10201219.

[7] H. Adnan *et al.*, "Integrating Topic-Aware Heterogeneous Graph Neural Network With Transformer Model for Medical Scientific Document Abstractive Summarization," *IEEE Access*, vol. 12, 2024, doi: 10.1109/ACCESS.2024.10637403.

[8] S. Luo, Q. Li, and W. Zhao, "An Improved Template Representation-based Transformer for Abstractive Text Summarization," in *Proc. IEEE Int. Conf. Natural Language Processing (ICNLP)*, 2022, doi: 10.1109/ICNLP9207609.

[9] A. Roy, P. Singh, and M. Gupta, "Transformer Based Text Summary Generation for Videos," in *Proc. IEEE ICEECC*, 2024, doi: 10.1109/ICEECC10581200.2024.

[10] K. Bhatt, P. Mehta, and S. Shah, "Video Transcript Summarizer," in *Proc. IEEE Int. Conf. Emerging Technology (INCET)*, 2022, doi: 10.1109/INCET54531.2022.9751991.

[11] A. Hassan, N. Ibrahim, and M. Abdel-Aziz, "Automatic Video Summarization with Timestamps Using Natural Language Processing Text Fusion," in *Proc. IEEE Int. Conf. Intelligent Systems and Computer Vision (ISCV)*, 2021, doi: 10.1109/ISCV52417.2021.9376115.

[12] S.-F. Hung, C.-H. Tsai, and Y.-H. Liu, "Enhancing LLM Question Answering with RAG through Dense Vector Search and Re-Ranking," *IEEE Access*, 2025.

[13] M. Douze *et al.*, "The FAISS Library," arXiv:2401.08281, Jan. 2024.

[14] S. M. T. I. Tonmoy *et al.*, "RAG Powered LLMs for QA: Evolution, Challenges, Applications, and Future Directions," in *Proc. IEEE Int. Conf. Big Data and Smart Computing (BigComp)*, 2024, doi: 10.1109/BigComp60300.2024.11034531.

[15] D. Katz, R. Shumacher, and O. Oren, "Efficient and Reproducible Biomedical Question Answering Using Hybrid Retrieval-Augmented Generation," arXiv:2505.07917, 2025.

[16] A. Asai *et al.*, "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection," in *Proc. ICLR*, 2024, arXiv:2310.11511.

[17] P. Sharma, N. Yadav, and R. Gupta, "Implementation of RAG Based Question-Answering Application," in *Proc. IEEE Int. Conf. Contemporary Computing and Communications (InC4)*, 2024, doi: 10.1109/InC411031246.2024.11031968.

[18] A. Verma, S. Kaur, and P. Sharma, "Text Summarization using NLP Technique," in *Proc. IEEE Int. Conf. Innovative Practices in Technology and Management (ICIPTM)*, 2022, doi: 10.1109/ICIPTM54903.2022.9974823.

[19] T. Patel and D. Mehta, "Enhancing NLP for Indic Languages With Limited Resources: A Study of Transformer Models for Translation and Summarization," in *Proc. IEEE ICESC*, 2025, doi: 10.1109/ICESC60148.2025.10928025.

[20] C. N. Hang, P. D. Yu, and C. W. Tan, "TrumorGPT: Graph-Based Retrieval-Augmented Large Language Model for Fact-Checking," *IEEE Trans. Artificial Intelligence*, 2025, doi: 10.1109/TAI.2025.TrumorGPT.