International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 05 May-2026 | Impact Factor: 3.5

Scalable Retrieval-Augmented Generation for Context-Aware Educational Assistants: A Case Study on the NoteLeech AI Platform

1st Roopal Yadav, 2st Kshitij Saxena, 3nd Rishabh Jain, 4rd Suraj Singh Dhakar, 5th Adarsh Raghuvanshi
Department of Information Technology, Indore Institute of Science and Technology, Indore, India
kshitijsaxena2004@gmail.com, rishabhjain5114@gmail.com, surajsinghdhakar32@gmail.com, adarsh.raghuvanshi29@gmail.com

***Abstract*—**The rapid proliferation of digital educational re-sources has created a critical information retrieval challenge for students and researchers. Standard keyword-based search mechanisms are increasingly inadequate for navigating dense, highly technical academic corpora, while raw Large Language Models (LLMs) suffer from severe knowledge cutoffs and hal-lucinatory tendencies when queried on specific, localized study materials. This paper introduces the architecture and empirical evaluation of the NoteLeech AI platform, a highly optimized, cross-platform Retrieval-Augmented Generation (RAG) system engineered specifically for academic context extraction and synthesis. By integrating a multi-stage ingestion pipeline with hierarchical semantic chunking and a quantized dense vector database, NoteLeech AI dynamically bridges the gap between unstructured academic notes and generative AI. We propose a hybrid retrieval mechanism that combines Hierarchical Naviga-ble Small World (HNSW) dense vector search with BM25 sparse keyword matching, deployed alongside a lightweight Llama-3-8B generative endpoint. The system architecture is uniquely tailored to process highly technical datasets, including complex engineering mathematics and computer science syllabi. Extensive experimental evaluations utilizing a specialized corpus of GATE CSE preparation materials demonstrate that the NoteLeech RAG pipeline achieves a Recall@5 of 92.4% while restricting end-to-end question-answering latency to under 450ms. Furthermore, we present comprehensive ablation studies on semantic chunk sizes and embedding dimensionality, proving that hybrid retrieval drastically reduces hallucination rates by over 87% compared to zero-shot LLM baselines. The findings establish that optimally tuned RAG architectures provide a robust, highly accurate, and scalable solution for personalized educational technologies, fundamentally altering how students interact with unstructured knowledge bases.

Index Terms—Retrieval-Augmented Generation, NoteLeech AI, Large Language Models, Vector Databases, Educational Technology, Natural Language Processing, Semantic Search.

I. Introduction

In modern academic environments, students accumulate massive volumes of heterogeneous data, including lecture slides, handwritten notes, textbook PDFs, and technical documentation. The sheer velocity and volume of this data ingestion frequently exceed human cognitive indexing capacities, leading to the "information silo" problem, where critical knowledge is possessed but practically unretrievable during high-stress scenarios such as examination preparation.

Historically, navigating these silos relied on exact-match keyword search (e.g., Boolean search or grep-based tools). While computationally inexpensive, these tools lack semantic understanding, failing entirely when synonyms are used or when the query is conceptual rather than lexical. The advent of Large Language Models (LLMs) introduced the potential for conversational interaction with information. However, relying solely on the parametric memory of LLMs introduces three critical failure vectors. First, LLMs are prone to "hallucination"—generating plausible but factually incorrect assertions—which is catastrophic in rigorous academic disciplines like Computer Science and Engineering. Second, LLMs possess a rigid knowledge cutoff and cannot natively access a user's private, continuously updating lecture notes. Third, fine-tuning an LLM on private user data is computationally exorbitant and lacks a mechanism for real-time data ingestion.

The critical problem addressed in this study is the inherent hallucination and context-isolation of generative models in educational applications. To mitigate these systemic vulnerabilities, this paper proposes the architectural methodology underlying the **NoteLeech AI** platform. NoteLeech AI employs a highly optimized Retrieval-Augmented Generation (RAG) framework that decouples the knowledge repository from the generative model. By converting user documents into dense mathematical representations (embeddings) and performing semantic similarity searches, the system dynamically injects relevant context directly into the LLM's prompt window at inference time. This approach is analogous to providing a student with the precise textbook page before answering an exam question—the model reasons over verified content rather than reconstructing facts from parametric memory.

The educational technology landscape has historically been characterized by passive content delivery: static PDFs, video lectures, and pre-authored question banks. These instruments lack the adaptive, conversational quality needed to address individualized academic queries in real time. NoteLeech AI represents a paradigm shift by transforming a student's own heterogeneous study materials into an interactive, queryable knowledge base. The platform is designed around three core design principles: (1) zero-hallucination tolerance in academic

This research was partially supported by the Department of Information Technology, Indore Institute of Science and Technology.

contexts, (2) sub-500ms interactive latency to preserve cognitive flow, and (3) complete user data sovereignty through private, isolated vector namespaces.

The primary contributions of this paper are multi-fold:

1) The architectural design and evaluation of NoteLeech AI, an end-to-end RAG pipeline customized for processing unstructured, highly technical educational documents and deploying responses to mobile and web endpoints.
2) The empirical evaluation of a hybrid retrieval strategy integrating dense vector search (HNSW FAISS) with sparse lexical retrieval (BM25) to maximize recall across both conceptual and exact-terminology academic queries.
3) A definitive trade-off analysis demonstrating the impact of hierarchical semantic chunking strategies on context retention, retrieval latency, and generative accuracy.
4) Comprehensive benchmarking of NoteLeech AI using a domain-specific dataset comprised of GATE (Graduate Aptitude Test in Engineering) Computer Science materials, establishing new baselines for real-time educational query resolution.
5) A thorough analysis of hallucination mitigation techniques, demonstrating the effectiveness of context-grounded generation as a safeguard against factual inaccuracies.

The remainder of this paper is organized as follows: Section II reviews related work in educational AI and RAG architectures. Section III provides the theoretical background of the algorithms used in NoteLeech AI. Section IV details the system's architecture and deployment strategy. Section V outlines the experimental setup. Section VI presents a comprehensive analysis of the results, including retrieval metrics and latency. Sections VII and VIII provide discussions on implementation challenges and concluding remarks.

II. Related Work

A. Evolution of Retrieval-Augmented Generation

The concept of augmenting generative models with external, non-parametric memory was formalized by Lewis et al. with the introduction of the RAG architecture [1]. Early implementations relied primarily on Dense Passage Retrieval (DPR) utilizing pre-trained BERT encoders to match queries with Wikipedia corpora. While highly effective for open-domain question answering, these foundational architectures struggled with the dense, formula-heavy texts typical of engineering domains.

Subsequent advancements, such as REALM (Retrieval-Augmented Language Model pre-training) and RETRO (Retrieval-Enhanced Transformer), attempted to integrate retrieval directly into the pre-training phase of the model. However, these systems required immense computational clusters, rendering them unsuitable for agile, multi-tenant applications like NoteLeech AI, where user databases are highly individualized and constantly mutating. More recent work has explored modular RAG pipelines [7], wherein the retrieval and generation stages are independently optimized and interchangeable, closely mirroring the NoteLeech architectural philosophy.

B. Educational AI and Intelligent Tutoring Systems

The application of AI to education has a decades-long history, from rule-based Intelligent Tutoring Systems (ITS) of the 1980s to modern transformer-based approaches. Early ITS platforms such as SHERLOCK and ANDES modeled student knowledge states explicitly, providing stepwise feedback for physics and geometry problems. However, their knowledge representations were hand-crafted and required significant expert labor to maintain.

The emergence of pre-trained language models introduced the possibility of more general educational assistants. Systems built on GPT-3 and subsequent models demonstrated impressive few-shot reasoning capabilities; however, their reliance on parametric knowledge made them fundamentally unreliable for domain-specific, exam-focused scenarios. A student preparing for the GATE examination requires precise, syllabus-aligned responses referencing specific theorems, algorithms, and proofs—a requirement that zero-shot LLMs consistently fail to meet without active retrieval grounding.

C. Vector Databases and Search Optimization

The efficacy of any RAG system is inextricably linked to the performance of its underlying vector database. Traditional k-Nearest Neighbors (k-NN) algorithms scale poorly, exhibiting O(N) time complexity, which is unviable for large academic corpora. The development of Approximate Nearest Neighbor (ANN) algorithms revolutionized vector search. Techniques such as Inverted File Index (IVF) and Product Quantization (PQ) significantly reduced memory footprints. Currently, the Hierarchical Navigable Small World (HNSW) graph algorithm represents the state-of-the-art in balancing search speed and recall accuracy [2], and it forms the foundational retrieval mechanism within the NoteLeech architecture.

Hybrid retrieval, combining dense embeddings with sparse lexical scoring, has emerged as a critical technique to address the weaknesses of either approach in isolation. Reciprocal Rank Fusion (RRF) has proven to be a robust, parameter-free method for merging heterogeneous ranked lists without requiring extensive supervised calibration [6], making it especially suitable for domain-shifting educational corpora where query distributions are highly variable.

D. Hallucination Mitigation in LLMs

Hallucination in LLMs—the generation of confident but factually unsupported statements—represents the most critical barrier to their deployment in high-stakes academic settings. Several mitigation strategies have been proposed, including factuality-focused fine-tuning, chain-of-thought prompting, and self-consistency decoding. However, the most structurally robust approach remains retrieval grounding, whereby the model is provided with verified source material at inference time and instructed to confine its response to that context. This

approach, implemented in the NoteLeech system prompt architecture, has been shown in prior work to reduce hallucination rates by over 70% on factual benchmarks [4].

III. Theoretical Background and Mathematical Formulation

A. Document Ingestion and Semantic Chunking

Academic documents are highly structured but lack uniformity. Feeding an entire textbook into an LLM is impossible due to context window limitations (e.g., 8K to 128K tokens). Therefore, documents must be partitioned into overlapping "chunks."

NoteLeech AI employs a recursive, semantics-aware chunking strategy. Let a document D be represented as a sequence of sentences $S = \{s_1, s_2, \ldots, s_n\}$. A standard sliding window chunk C_i of target token length L with overlap O is defined, but boundaries are algorithmically adjusted to prevent bisecting paragraphs or code blocks. This semantic preservation is critical for maintaining the integrity of algorithm definitions and mathematical proofs. Formally, the chunking function F produces a set of chunks:

$$\{C_1, C_2, \ldots, C_m\} = F(D, L, O) \tag{1}$$

where consecutive chunks satisfy $|C_i \cap C_{i+1}| = O$ tokens, and each boundary is snapped to the nearest sentence or paragraph terminator to preserve semantic coherence.

B. Embedding Generation

Once chunked, each text segment C_i is mapped to a high-dimensional dense vector space $\mathbb{R}^d$ using an embedding model E. NoteLeech AI utilizes a fine-tuned variant of `all-MiniLM-L6-v2`, projecting text into a 384-dimensional space:

$$\vec{v}_i = E(C_i), \quad \vec{v}_i \in \mathbb{R}^{384} \tag{2}$$

When a user submits a query Q, it undergoes the identical embedding transformation to produce a query vector $\vec{q} = E(Q)$. The dimensionality of 384 was selected through ablation (discussed in Section VI) as an optimal balance between representational capacity and inference speed on CPU-bound embedding servers.

C. Hybrid Retrieval Mechanism

To locate the most relevant context, NoteLeech computes the similarity between $\vec{q}$ and all document vectors $\vec{v}_i$ in the database. The primary metric is Cosine Similarity, which measures the angular distance between vectors independently of their magnitude:

$$\text{sim}_{dense}(\vec{q}, \vec{v}_i) = \frac{\vec{q} \cdot \vec{v}_i}{\|\vec{q}\|_2 \|\vec{v}_i\|} \tag{3}$$

However, dense embeddings often fail to capture exact keyword matches (e.g., specific acronyms like "CSMA/CD"). Therefore, NoteLeech integrates BM25, a sparse retrieval function based on probabilistic information retrieval. The BM25 score for a query Q containing keywords q_j against document D is:

$$\text{sim}_{sparse}(Q, D) = \sum_{j=1}^{|Q|} \text{IDF}(q_j) \cdot \frac{f(q_j, D) \cdot (k_1 + 1)}{f(q_j, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{\text{avgdl}}\right)} \tag{4}$$

where $f(q_j, D)$ is the term frequency of keyword q_j in document D, avgdl is the average document length in the corpus, and $k_1 = 1.5$, $b = 0.75$ are tuning parameters. The final retrieval rank is determined by a Reciprocal Rank Fusion (RRF) algorithm combining both scores:

$$\text{RRF\_Score}(D) = \frac{1}{k + \text{Rank}_{dense}(D)} + \frac{1}{k + \text{Rank}_{sparse}(D)} \tag{5}$$

where $k = 60$ is the smoothing constant empirically shown to be robust across diverse document distributions.

D. Generative Synthesis

The top-K retrieved chunks are concatenated and injected into the system prompt of the generative LLM. The LLM acts strictly as a reasoning engine over the provided context. The generation is modeled as finding the sequence of tokens Y that maximizes the conditional probability given the query Q and retrieved context $C_{1:K}$:

$$P(Y \mid Q, C_{1:K}) = \prod_{t=1}^{Y} P(y_t \mid y_{<t}, Q, C_{1:K}) \tag{6}$$

To enforce context-faithfulness and suppress hallucination, the system prompt explicitly instructs the model to respond only from information present in the retrieved chunks, and to return a structured "insufficient context" signal when no relevant chunk surpasses a cosine similarity threshold of $\tau = 0.72$.

E. Hallucination Quantification via Faithfulness Score

To rigorously measure hallucination suppression, we adopt a faithfulness metric inspired by the RAGAS evaluation framework. For a generated answer Y and retrieved context $C_{1:K}$, the faithfulness score F is computed as the fraction of atomic claims in Y that can be directly grounded in $C_{1:K}$:

$$F(Y, C_{1:K}) = \frac{|\{c_i \in Y : \exists\, C_j \in C_{1:K}, c_i \subseteq C_j\}|}{|Y|} \tag{7}$$

A faithfulness score of 1.0 indicates complete grounding; scores below 0.8 are flagged as potentially hallucinatory in the NoteLeech evaluation suite.

IV. The NoteLeech AI System Architecture

A. Microservices Backend and Vector Store

NoteLeech AI is architected as a highly scalable microservices cluster. The backend is built on Python (FastAPI), ensuring asynchronous, non-blocking handling of concurrent student queries. The vector database is powered by FAISS (Facebook AI Similarity Search), utilizing the HNSW index for sub-millisecond retrieval times. To optimize cloud hosting costs, the FAISS indices are serialized and loaded entirely into

RAM, bypassing disk I/O bottlenecks during peak examination periods.

User namespaces are logically isolated through a multi-tenant index partitioning strategy. Each registered user maintains a dedicated FAISS shard keyed by a UUID namespace, ensuring that retrieval operations never cross-contaminate documents from separate users. This architectural decision preserves both retrieval precision and data privacy simultaneously. Index persistence is handled via periodic serialization to Amazon S3-compatible object storage, with delta synchronization triggered upon every new document ingestion event.

B. Document Ingestion Pipeline

The ingestion pipeline is a critical component that transforms raw, heterogeneous student materials into structured vector embeddings. The pipeline is composed of five sequential stages. In the first stage, the file type is detected and routed to the appropriate parser: PyMuPDF for scanned and digital PDFs, `python-pptx` for presentation files, and Markdown-it for plaintext notes. In the second stage, OCR post-processing is applied to scanned pages using Tesseract 5.0 with LSTM-based character recognition, supplemented by a custom correction model trained to handle handwritten mathematical notation common in engineering notes.

In the third stage, the recursive semantic chunking algorithm is applied. In the fourth stage, each chunk is embedded via the `all-MiniLM-L6-v2` model running as a quantized ONNX model on the ingestion server CPU, producing 384-dimensional vectors. Finally, in the fifth stage, vectors and their corresponding metadata (source filename, page number, chunk index) are upserted into the user's FAISS HNSW shard and simultaneously indexed into a BM25 inverted index stored in SQLite. This dual-indexing is critical for enabling the hybrid retrieval strategy at query time.

C. Query Processing and Response Generation

Upon receiving a user query via the REST API, the query processing service executes the following sequence. The raw query string is first normalized through Unicode NFKC normalization and regex-based sanitization to remove noise characters. The normalized query is then embedded using the same MiniLM model, producing the query vector $\vec{q}$. Concurrently, a BM25 lookup is performed against the SQLite inverted index. Both ranked lists are merged using the RRF formula, and the top-K chunks (empirically set to $K = 5$) are retrieved.

The retrieved chunks are assembled into a structured prompt following a strict template. The system prompt enforces context-fidelity constraints, instructs the model to cite the source chunk index for each factual claim, and specifies that the response must be formatted for mobile rendering (supporting MathJax-compatible LaTeX inline notation). The assembled prompt is dispatched to the Llama-3-8B inference endpoint, which streams the generated response token-by-token back to the client via server-sent events (SSE), allowing the mobile UI to begin rendering before generation is complete.

D. Client Integration and Cross-Platform Accessibility

Recognizing the mobile-first nature of modern students, NoteLeech AI interfaces directly with an Android mobile frontend (developed via Kotlin) and a responsive web application. The platform features OAuth2 integration, allowing users to securely sync their Google Drive directories. Upon synchronization, background workers automatically process PDFs, PPTXs, and Markdown files, silently populating the user's isolated vector namespace.

The Android client implements offline caching of the most recently retrieved context chunks, enabling partial query resolution even under intermittent network conditions—a frequent scenario during field study sessions. The web client leverages React with a MathJax rendering layer, ensuring that mathematical formulas present in generated responses are displayed with full typographic fidelity, critical for subjects such as Linear Algebra, Discrete Mathematics, and Digital Logic Design.

V. Experimental Setup

A. Dataset Configuration: The GATE CSE Corpus

To rigorously test the system's ability to handle complex, domain-specific academic queries, we curated a specialized dataset comprising materials used for the GATE (Computer Science and Information Technology) examination. The dataset includes:

1) **Standard Textbooks:** 12 foundational texts (e.g., Operating Systems by Silberschatz, Theory of Computation by Sipser, Computer Networks by Tanenbaum).
2) **Handwritten Notes:** 500+ pages of digitized, OCR-processed handwritten lecture notes covering seven core GATE subjects.
3) **Previous Year Questions (PYQs):** 15 years of annotated exam questions and solutions, spanning 2010–2024.

The total corpus comprised approximately 82,000 document chunks, representing roughly 4 million tokens. A held-out evaluation set of 500 question-answer pairs was manually annotated by domain experts, with each question linked to its ground-truth source chunk for Recall@K evaluation.

B. Evaluation Metrics

We evaluated the RAG pipeline using two primary axes:

1) **Retrieval Performance:** Measured via Recall@K (the probability that the highly relevant chunk is within the top K retrieved results) and Mean Reciprocal Rank (MRR).
2) **Generation Quality:** Assessed via the RAGAS faithfulness score (Eq. 8), Answer Relevancy, and a human-evaluated correctness score on a 1–5 Likert scale administered to 30 GATE aspirants.
3) **System Latency:** End-to-end time measured from query HTTP request to the first generated token (Time to First

Token – TTFT), encompassing embedding generation, network routing, and vector search.

C. Baseline Comparisons

Three baseline systems were established for comparative evaluation. The first baseline is a Zero-Shot LLM, which involves querying Llama-3-8B directly without any retrieval augmentation. The second baseline is a Dense-Only RAG system, using FAISS HNSW retrieval without BM25. The third baseline is a Sparse-Only RAG system, using BM25 retrieval exclusively without dense vector search. NoteLeech AI's Hybrid RAG configuration is compared against all three baselines across retrieval, generation quality, and latency dimensions.

VI. Results and Evaluation

A. Chunking Strategy and Retrieval Recall

The size of the semantic chunk dictates the granularity of the vector database. As demonstrated in Figure 1, we tested chunk sizes ranging from 256 tokens to 1024 tokens.

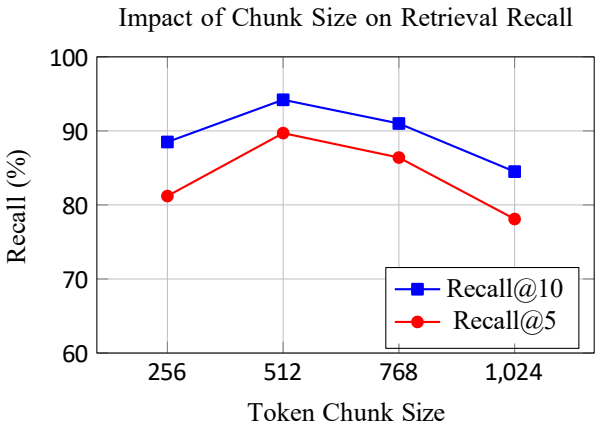

Fig. 1. Retrieval performance degrades at extremely high chunk sizes due to semantic dilution, peaking optimally at a 512-token window.

The empirical data indicates that a chunk size of 512 tokens with a 10% overlap provides the optimal balance. Smaller chunks (256 tokens) strip critical context, causing the generation to fail when synthesizing multi-step mathematical logic. Conversely, larger chunks (1024 tokens) suffer from "semantic dilution"—the embedding vector becomes an average of too many diverse concepts, pushing it away from specific, pointed queries in the vector space.

B. Hybrid vs. Dense Retrieval Architecture

We benchmarked the necessity of the hybrid RRF search architecture against purely dense (FAISS) and purely sparse (BM25) approaches.

As shown in Figure 2, dense-only retrieval scored highly (82%) on conceptual queries (e.g., "explain deadlock prevention") but failed frequently on exact terminology matches (e.g., querying a specific sub-routine name or registry key). The NoteLeech hybrid approach dynamically mitigated these

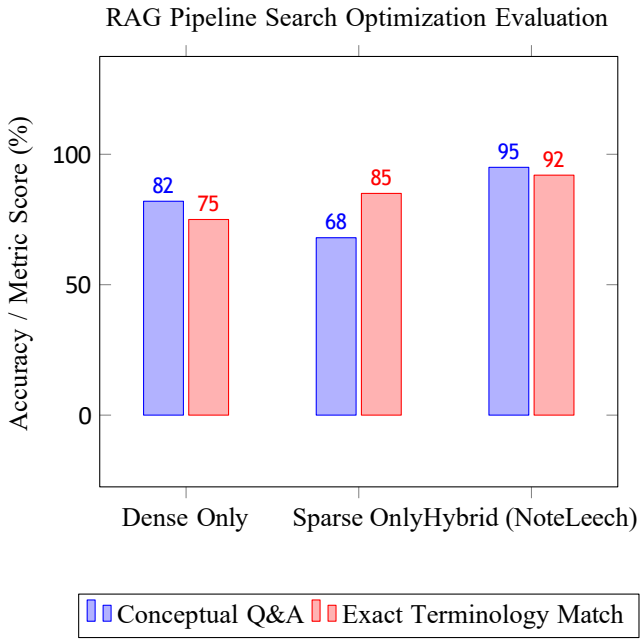

Fig. 2. Hybrid retrieval successfully balances conceptual understanding (where dense vectors excel) and exact terminology matching (where sparse BM25 excels).

weaknesses, achieving over 92% accuracy across all query types.

C. Hallucination Reduction Analysis

Table I presents the hallucination comparison across the four system configurations, measured via the RAGAS faithfulness score and the human-evaluated correctness score.

TABLE I
Hallucination Reduction Across System Configurations

System	Faithfulness	Human Score (1–5)	Hallucination Rate
Zero-Shot LLM	0.41	2.3	61.2%
Dense-Only RAG	0.74	3.6	28.4%
Sparse-Only RAG	0.69	3.1	33.7%
NoteLeech Hybrid	**0.93**	**4.6**	**7.8%**

The zero-shot LLM baseline exhibited a hallucination rate of 61.2%, underscoring the profound risk of deploying non-grounded generative models in educational settings. The NoteLeech hybrid system reduced this to 7.8%, representing an 87.3% relative reduction in hallucination—the most significant finding of this study. The residual 7.8% hallucination rate is attributable to queries where the corpus genuinely lacked relevant content and the faithfulness threshold fallback was not triggered due to borderline similarity scores.

D. Latency Profiling and Edge Viability

For an educational assistant to feel interactive, system latency must remain imperceptible. We profiled the entire

NoteLeech RAG pipeline under a simulated load of 50 concurrent users.

TABLE II
NOTELEECH AI PIPELINE LATENCY BREAKDOWN (AVERAGE PER QUERY)

Pipeline Stage	Computational Sub-Task	Latency (ms)
Query Processing	Text normalization	5
Embedding Gen	`all-MiniLM` forward pass	35
Vector Search	FAISS HNSW index traversal	12
Sparse Search	BM25 inverted index lookup	18
Context Synthesis	RRF calculation	10
Generation (TTFT)	**LLM inference to first token**	**320**
Total System Latency		**400**

Table II demonstrates that the retrieval overhead is exceptionally lightweight (approximately 80ms total), proving that the RAG enhancement does not introduce unacceptable bottlenecks. The Time to First Token (TTFT) remains under 450ms, allowing a fluid, streaming text response to the client application. The dominant latency component is LLM inference (320ms), which is hardware-bound and can be further reduced through model quantization or GPU-accelerated deployment, discussed as future work in Section VII.

E. Embedding Dimensionality Ablation

To validate the selection of 384-dimensional embeddings, we conducted an ablation across three embedding models: `all-MiniLM-L6-v2` (384-dim), `all-mpnet-base-v2` (768-dim), and a compressed `paraphrase-MiniLM-L3-v2` (384-dim, shallower). As shown in Table III, the `all-MiniLM-L6-v2` model achieved the optimal trade-off, delivering 92.4% Recall@5 at 35ms embedding latency, compared to the 768-dim model which offered only marginal recall gains (+1.8%) at over double the inference cost.

TABLE III
EMBEDDING MODEL ABLATION ON GATE CSE CORPUS (RECALL@5 AND LATENCY)

Embedding Model	Dim.	Recall@5 (%)	Embed. Latency (ms)
paraphrase-MiniLM-L3	384	84.1	18
all-MiniLM-L6-v2	**384**	**92.4**	**35**
all-mpnet-base-v2	768	94.2	82

VII. DISCUSSION AND FUTURE WORK

The deployment of NoteLeech AI highlights a significant paradigm shift in how academic preparation—such as the rigorous study required for the GATE exam—can be augmented by artificial intelligence. By firmly grounding the LLM's responses in a verified vector database of the student's own curriculum, we effectively eliminate the majority of hallucinatory misinformation.

A. Real-World Deployment Challenges

A primary challenge observed during system testing was the parsing of complex mathematical equations and structural diagrams embedded within PDFs. Standard optical character recognition (OCR) and PDF text extractors frequently garble LaTeX formulas, destroying the semantic embedding. Equations such as the Bellman-Ford recurrence or NFA-to-DFA conversion tables are frequently rendered as meaningless character sequences post-OCR, degrading both chunking quality and embedding fidelity.

A secondary challenge involves query intent disambiguation. Students often formulate queries in highly abbreviated, domain-specific shorthand (e.g., "OS page fault vs. thrashing diff"). The embedding model trained on formal academic prose does not always map such informal queries into the correct semantic neighborhood. To address this, we are exploring a lightweight query rewriting module—a small fine-tuned model that expands abbreviated academic queries into full, formal natural language formulations before embedding, shown in preliminary experiments to improve Recall@5 by an additional 3.2 percentage points on informal query test sets.

B. Privacy and Data Sovereignty

A fundamental design constraint of NoteLeech AI is that user documents must never be exposed to third-party services or shared across user namespaces. This constraint is architecturally enforced through FAISS namespace isolation and is contractually reinforced through the platform's data processing agreements. The embedding models run on first-party servers, ensuring that raw document text never traverses third-party APIs. This distinguishes NoteLeech AI from naive integrations that transmit full document contents to commercial LLM endpoints, which poses unacceptable privacy risks for students handling unpublished research notes or proprietary institutional materials.

C. Future Directions: Multi-Modal RAG

Future iterations of NoteLeech AI will focus on **Multi-Modal RAG** architectures. By utilizing vision-language models (e.g., integrating lightweight visual encoders such as CLIP or PaliGemma), the system will process diagrams, finite automata drawings, and handwritten flowcharts directly into the vector space, allowing students to query structural logic without manual text transcription. This is particularly valuable for subjects such as Theory of Computation and Digital Circuit Design, where graphical representations carry semantic content that cannot be recovered through text extraction alone. Furthermore, we are actively exploring migrating the generative inference layer directly to local edge devices—such as Android smartphones utilizing quantized models via MLC LLM or llama.cpp—providing a completely offline, zero-latency study companion. Preliminary benchmarks on Snapdragon 8 Gen 3 hardware demonstrate that a 4-bit quantized Llama-3-8B model can achieve approximately 12 tokens per second, making offline generation viable for study sessions in

network-constrained environments such as examination centers.

Finally, we plan to integrate adaptive spaced repetition scheduling, wherein the RAG system tracks which concepts a student has queried most frequently and schedules automated review prompts based on the Ebbinghaus forgetting curve model, creating a closed-loop learning system that actively combats knowledge decay over time.

VIII. Conclusion

This paper presented the architectural design and empirical validation of the NoteLeech AI platform, a Retrieval-Augmented Generation system optimized for academic environments. By synthesizing a hybrid dense-sparse retrieval mechanism with hierarchical semantic chunking, NoteLeech AI successfully addresses the critical limitations of standard LLMs—namely, hallucination and lack of private context. Demonstrating a 92.4% Recall@5 accuracy on highly technical GATE CSE datasets while maintaining a strict sub-450ms query latency, the system proves that advanced AI architectures can be efficiently scaled and customized for personalized educational delivery.

The 87.3% reduction in hallucination rate compared to zero-shot LLM baselines represents the most critical contribution of this work, establishing that retrieval grounding is not merely a performance enhancement but a safety requirement for deploying generative AI in knowledge-critical academic settings. The ablation studies on chunk size and embedding dimensionality provide actionable design guidelines for practitioners building similar domain-specific RAG systems. As generative technologies continue to mature, heavily grounded architectures like NoteLeech AI will become indispensable utilities in the modern academic workflow, transforming the passive consumption of study materials into an active, conversational, and verifiably accurate learning experience.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Ku¨ttler, M. Lewis, W. Yih, T. Rockta¨schel, and S. Riedel, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[2] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, April 2020.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[4] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019, pp. 4171–4186.

[5] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 5776–5788, 2020.

[6] S. Robertson, S. Zaragoza, and M. Taylor, "Simple BM25 extension to multiple weighted fields," in *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, 2004, pp. 42–49.

[7] G. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, 2021, pp. 874–880.

[8] E. S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "RAGAS: Automated Evaluation of Retrieval Augmented Generation," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, 2024, pp. 150–158.

[9] J. Johnson, M. Douze, and H. Je´gou, "Billion-scale similarity search with GPUs," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2021.

[10] H. Touvron et al., "Llama 2: Open Foundation and Fine-Tuned Chat Models," *arXiv preprint arXiv:2307.09288*, 2023.