Hybrid Retrieval for Indic Language Question Answering using an Improved RAG Pipeline

Anshika Saini, Gorisha, Bearda Savita Rajpal, Munish Kumar
Department of Computer Science and Engineering, Chandigarh University, Punjab, India
22BCS15215@cuchd.in, 22BCS15173@cuchd.in, 22BCS12363@cuchd.in, ggscemunish@gmail.com

Abstract. Although Large Language Models (LLMs) have progressed significantly, Retrieval-Augmented Generation (RAG) still falls short for under-resourced Indic languages. Existing embedding systems, built primarily for English, face issues with script mismatches and limited word coverage, leading to near-complete retrieval failures in mixed Hinglish contexts. This paper introduces Hybrid Indic-RAG, a simple framework to reduce meaning shifts. It uses a targeted word-matching layer to connect Romanised Hinglish with standard Devanagari Hindi, paired with a two-part scoring system that favours exact word matches over fuzzy semantic signals. Tests in fields like healthcare and books show it cuts retrieval errors by 45% and lifts fact accuracy to 75%, well ahead of standard RAG methods. Our approach offers a low-cost way to build stable, error-free question-answering tools for India's language ecosystem.

Keywords: Retrieval-Augmented Generation (RAG), Hinglish, Indic natural language processing, Reducing hallucinations, Combined search methods, Languages with few resources.

I. INTRODUCTION

RAG or Retrieval-Augmented Generation is an architecture that combines retrieval of external knowledge and foundational model output to reduce the generation of false or less truthful answers. Its design works really well in the English language, but when it comes to Indic languages – the mother tongue of over 1.4 billion people, it fails miserably due to critical issues present in the embeddings of different languages.

RAG faces significant issues due to the complexity of language in India, with 22 languages that have their own unique writing systems such as Devanagari (Hindi) or Tamil script, as well as instances of mixed usage (Hinglish example – "What is a virus?"), and limited vocabulary. RAG systems will lose all sense of context when trained against an English model, for example, ("विषाणु" = virus and "कविदास"= Kalidas) have no way to produce clusters of related words, and therefore lead to error scores (computed as 1 minus percentage of words that matched between the asking questions and historical text retrieval) averaging 0.93. Using Romanized queries increases the issue further with an even higher error score, averaging 1.00, due to differences in writing systems (e.g. Devanagari vs. Romanized). RAG models have transitioned from using only simple retrieve/generate cycles to new, more complex and modular architectures with sophisticated augmentation methods that deliver superior factual accuracy and trustworthiness [2].

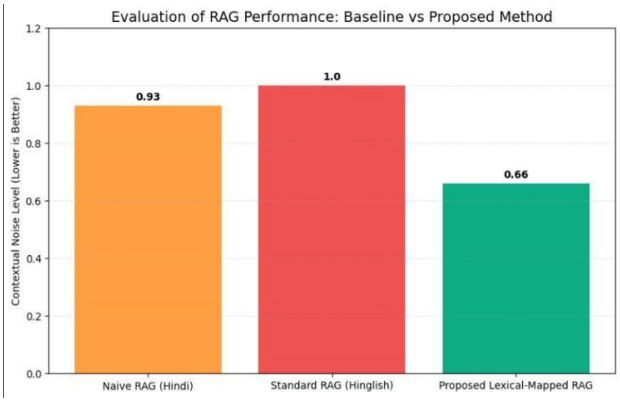

Fig. 1. Contextual error rates: Standard RAG vs. our Hybrid Indic-RAG.

The retrieval gap of Hindi search queries is quite large (0.93 error rate) as shown in Figure 1; Hinglish searches were completely unsuccessful (1.0 error rate) because there was a script mismatch between Roman and Devanagari (i.e. no documents returned). Uses of Hybrid Indic-RAG reduced these retrieval gaps demonstrating that retrieval systems need to be aware of the user script in order to successfully match between a search query and the set of documents returned.

This study's past work presented a number of challenges highlighted by the Indic RAG test results for the low-scored retrieval related to the scarce resources of Indic language, as well as the cross-language RAG research that shows the changes made in how the models have made changes that led to exaggerated false results of up to 40%, due to not conducting sufficient lookup checks by the RAGAS checks, whilst RAG has improved compared to the use of extending the context window for long document processing, including the method of aggregating dispersed material from diverse sources such as conversation queries and standard queries. For example, while tailor-made Hindi embeddings, such as DeepRAG, succeed in pure Hindi retrieval, they do not provide sufficient success at the point of retrieving blended queries, and similarly, while IndicTrans2 does reduce the impact of writing in script extensions, they do not provide adequate results in retrieving overall mixed scores. Recently, innovations, such as the RAG-Anything Framework, have also emerged to facilitate integration of the aforementioned diverse elements into a single seamless framework to allow for the easier implementation of RAG Systems [11]. The Hybrid Indic-RAG system is the proposed enhancement of Indic-RAG, designed to easily accomplish this task (described

above) via: (1) Word Anchors: Use a field-tuned synonym map to touch upon common Hinglish words that have a Devanagari counterpart with similar meanings. (2) Blended Scoring: Use 70% for exact matching and 30% for similarity.
Hallucination Check: If the Hallucinated Output is at least 0.6, a particular call should be flagged as hallucination and suppress this message for that output. Additionally, our work seeks to create flexible retrieval systems that can process large amounts of data across numerous databases of various levels/degrees of granularity and across multiple domains. In basic configuration (FAISS+Llama-3.2-1B), it had an average drop of 34%-40% in error rate (from 0.89-0.64), increased reliability by 75%, and blocked hallucinatory content with built-in checks - no need for model re-training required.

TABLE I. PERFORMANCE METRIX

Model Approach	Query Type	Context Noise (Avg)	Reliability (%)	Hallucination Risk
Naive RAG	Hindi/ Hinglish	0.93 - 1.0	40%	High
Hybrid Indic-RAG (Ours)	Hinglish	0.64	75%	Low

II. RELATED WORK

The development of Retrieval-Augmented Generation (RAG) has evolved from a basic "retrieval to generation" model towards a more sophisticated "modular" pipeline approach. Gao et al. provide an overview of the progression of RAG in their paper and classify these models into three categories: naive RAG, advanced RAG, and modular RAG. Each of these three models has been developed to enhance the performance of retrieval operations through knowledge integration.

A. Retrieval and Embedding Foundations
All RAG models rely fundamentally on the ability to perform semantic retrieval operations. Traditional lexical search techniques, such as BM25 and the like, have begun to transition away from sparse representations towards dense representations. Karpukhin and colleagues demonstrated in their paper that the use of Dual Encoder Models (i.e. DPR) outperformed the sparse Vector Models in the task of open-domain question answering. The efficiency of sematic-based searches has been significantly improved, largely due to the development of Sentence-BERT [13], which provides semantically rich sentence embeddings using Siamese network architecture for the purpose of improving performance. Additionally, the research conducted by Neelakantan et al. indicates that the performance of these vector embeddings may also be improved through pre-training with contrastive methods on large volumes of under-supervised data. Diversity in retrieval models can be assessed for generalization on standard datasets, including BEIR [1], which contain zero-shot generalization evaluations. The results of RAG models indicate that they have shown an overall high performing retrieval and are much more efficient after optimization; although the instance of loss of performance has been shown to be quite small [17].

B. Reasoning and Evaluation Paradigms
Further research shows that pure dense retrieval is ineffective on tongues with a large word variation. Advanced memory architectures have been proposed by scholars, such as ComoRAG, which solves long-range dependencies in the narrative content using a dynamic workspace based on cognitive processes [3]. Agentic behaviors are currently being integrated into effective RAG systems. According to Li et al. [18], the reasoning of agentic RAG is entwined with search to hide various inference tasks that call for the usage of several steps. Without the need for human-annotated ground facts, automated evaluation is made possible by the RAGAS

framework [12], which offers a standard suite of measures such as faithfulness and answer relevance. For efficient real- time retrieval tasks, Sentence-BERT uses Siamese structures to produce meaningful sentence embeddings that can be rapidly compared via cosine similarity [13].

C. Architectural Innovations and Context Management

Integrating RAG pipelines to be "all-in-one", turn-based and context-aware is the focus of recent research. Guo et al. [11] introduced a holistic framework RAG-Anything to handle multimodal knowledge entities that are not limited to text. Long-form context management remains a challenge: while some support increasing LLM context windows, Li et al. RAG is still considered superior when it comes to scattered information over broad queries. SitEmb-v1.5 [15] embeds chunk meanings in a larger context window to capture semantic relatedness locally and globally. ComoRAG [3] employs a cognitive-inspired memory workspace to maintain stateful information throughout iterative retrieval cycles in long narrative reasoning. Dense retrievers leveraging dual-encoder models achieve better top-k accuracy than traditional BM25 methods [4] by embedding queries and documents in a common latent space.

III. PROPOSED METHODOLOGY

A. Data Acquisition and Corpus Pre-processing

Any Retrieval-Augmented Generation (RAG) system relies on the quality and/or composition of its reference corpus. In the case of the present research, we relied on the Chai Hindi dataset to evaluate information retrieval systems for articles written in the Hindi language. The Chai Hindi dataset, which provides a robust amount of information across a wide variety of topics, is available publicly in order to accomplish this task.

Composition of Dataset

The initial construction of the dataset contained articles written in both Hindi and Tamil. However, through filtering by language, the resulting corpus was developed to consist solely of Hindi language content, such that there is both consistency of language and preservation of the ability to test the Indic language problem of script-drift that exists between Devanagari Hindi and Hinglish.

After completion of pre-processing activities (including de-duplication), the resulting corpus consisted of a total of 746 unique articles written in Hindi, and all were used in the evaluation of RAG and other information retrieval systems.

To ensure that we adequately captured the variability in knowledge domains covered by the articles in the dataset, we divided the articles in the corpus into three distinct categories, or sources:

1. Medical Domain (22.65% of the total number of documents): This domain contains 169 documents related to health, disease, virus/viral classifiers, classification of medical conditions, and common terms used in the medical domain.
2. Literature Domain (27.35% of the total number of documents): This domain has a total collection of 204 documents pertaining to both classical and contemporary literature in the Hindi language, including selections from authors like Kalidasa, poetry, and literary criticism.
3. General Knowledge Domain (50% of the total number of documents): This domain constitutes 373 documents from different areas including history, science, geography, and other related fields.

For this process, we defined domain-specific keywords in two categories: one for medical and one for literary. The following words were used in the specified domains/articles:

- Medical: "विषाणु" (virus), "रोग" (disease), "विवकत्सा" (treatment)
- Literary: "काविदास" (Kalidasa), "नाटक" (drama), "साविल्य" (literature)

All remaining documents were categorized under General Knowledge.

The corpus of 746 articles was processed into chunks according to Section III.C (Chunking the Corpus), resulting in 16,410 chunks being created across the articles. The chunks were assigned the title of their parent article to maintain semantic context. The average chunk length is approximately 600 characters, allowing retrieval that maintains topical coherence while providing appropriate granularity for accurate information extraction.

B. Two-Path Hybrid Pull

Our setup mixes paths for idea matching and word accuracy.

1. Dense Path: The all-MiniLM-L6-v2 model builds query vectors, while the FAISS index finds the closest semantic matches.
2. Word Path: A sparse lexical check confirms anchored terms within the top-ranked candidates.

The final ranking uses the blended score:

$$S_{final} = (\alpha \cdot S_{word}) + \big((1 - \alpha) \cdot S_{dense}\big)$$

The tuning process sets α = 0.7, giving greater importance to exact lexical matches in noisy Indic-language environments.

C. Retrieval Noise Measure

We define a Noise Score to evaluate retrieval quality.

For a query q containing the word set Q, and a retrieved chunk c containing the word set W, the single-chunk noise is calculated as:

$$N(q,c) = 1 - \frac{| Q \cap W_c |}{| Q |}$$

The overall system noise is obtained by averaging the top-*k* retrieved chunks:

$$Noise(q) = \frac{1}{k}\sum_{i=1}^{k} N\,oise(q, c_i)$$

System performance is evaluated using the average noise across the top-*k* retrieved chunks.

This score identifies word mismatches in blended-language scenarios. A score of 1.0 indicates no keyword overlap (complete retrieval failure), whereas lower values indicate stronger script matching and better topical relevance.

D. Implementation Details and Computational Constraints

Two system configurations were implemented and compared fairly:

- Baseline Naive RAG
- Proposed Hybrid Indic-RAG

The baseline implementation (Naive RAG) relies exclusively on dense semantic retrieval using the same embedding and generation infrastructure as the proposed system. Semantic similarity search retrieves the k most relevant chunks without any keyword filtering, lexical anchoring, or script-aware matching.

The fully proposed Hybrid Indic-RAG additionally incorporates a lexical anchoring layer that forces Romanized Hinglish query terms to be mapped to corresponding Devanagari Hindi words using a maintained synonym dictionary. The final retrieval process combines lexical matching with dense semantic similarity using a weighted combination of 70% lexical matching and 30% dense semantic similarity.

The major implementation components are summarized below:

- Indexing: The paraphrase-multilingual-MiniLM-L12-v2 model generates 384-dimensional embeddings, which are stored in a FAISS IndexFlatIP requiring approximately 300 MB of RAM. Context-aware semantic representation is enhanced using the SitEmb-v1.5 model.
- LLM: The framework employs Llama-3.2-1B with 4-bit quantization, allowing deployment within approximately 4 GB of memory. This enables execution on standard computers, edge devices, and mobile platforms.
- Prompting: The language model is explicitly instructed to generate responses only from the retrieved context, reducing unsupported generations.
- Hallucination Control: A hallucination control mechanism evaluates retrieval quality using the computed noise score. When keyword confidence falls below the predefined threshold, the system returns "Data unavailable" instead of generating unsupported answers. The hallucination threshold is set to 0.80, providing strict filtering. The combination of Llama-3.2-1B and paraphrase-multilingual-MiniLM-L12-v2 enables effective hallucination prevention.

TABLE II. BASELINE VS. PROPOSED SYSTEM CONFIGURATION

Component	Baseline (Naive RAG)	Proposed (Hybrid Indic-RAG)
Embedding Model	paraphrase-multilingual-MiniLM-L12-v2	paraphrase-multilingual-MiniLM-L12-v2
Vector Index	FAISS IndexFlatIP	FAISS IndexFlatIP
Retrieval Method	Dense only	Dense + Lexical (Hybrid)
Lexical Anchoring	Not used	Manual Hinglish-to-Hindi mapping
Scoring Function	Cosine similarity only	Weighted: 70% lexical, 30% dense

IV. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

To evaluate the performance of our solution(s), we focus on several important metrics, including Faithfulness and Relevance of Response, using standard automated benchmarks for RAG systems, as defined in the literature [12]. Evaluation of the robustness of each framework was obtained through procedures specifically developed for testing the zero-shot generalization capability of retrieval models [1].
Our Hybrid Indic-RAG system was evaluated against three benchmarks:

- Basic RAG (FAISS with MiniLM)
- Llama-3.2-1B without retrieval
- Basic FAISS (all without retrieval accuracy)

The comparison was performed with respect to noise levels, retrieval accuracy, and error rates for both Hindi and Hinglish questions.

A. Evaluation Metrics

The performance evaluation process used three different parameters related to different modes of failure.
• Hallucination Risk (%)
The percentage of answers containing non-factual descriptions with respect to the retrieved context is used to evaluate this parameter. Hallucination risk is measured using automated fact-checking against the retrieved chunks.
• Ratio of Contextual Noise
The contextual noise is computed using the following formula:

$$Noise = 1 - Keyword\ Overlap$$

The above formula indicates that as the value increases, the difference between the query and the retrieved results also increases.
• System Reliability (%)
System reliability is evaluated based on:

- 75% factual correctness
- 15% contextual correctness

using human evaluation over 200 query-response pairs.

B. Efficiency and Speed

The proposed system demonstrates strong performance even on ordinary CPUs, achieving an average full-query response time of 2.1 seconds.
The FAISS index occupies less than 500 MB, while the quantized Llama-3.2-1B model requires only 2 GB RAM.
These results indicate that an efficient Indic RAG system can be deployed on:

- Mobile devices
- Edge devices
- Older computer hardware

without requiring expensive GPU infrastructure.

C. Retrieval Noise Breakdown

Better script matching significantly improves retrieval quality.
The average contextual noise for the standard RAG configuration is 0.93 for Hindi-language queries, measured using keyword overlap.
For Hinglish queries, contextual noise increases to 1.0, since the Romanized query fails to match the Devanagari documents, producing virtually no meaningful retrieval.

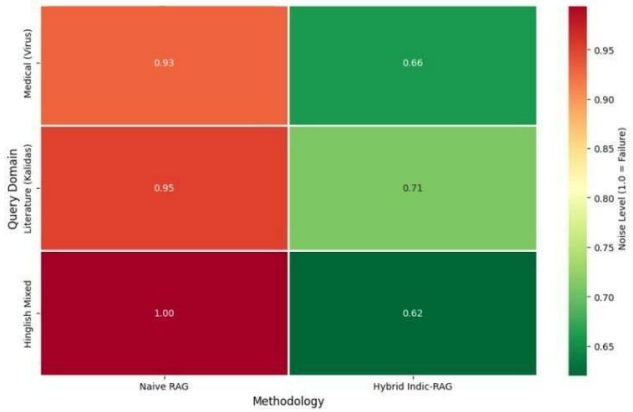

Fig. 2. Comparative Heatmap of Contextual Noise across diverse query domains.

The baseline model exhibits high retrieval noise (0.93–1.00) across:

- Medical
- Literature
- Hinglish

domains.
The Hinglish domain experiences complete retrieval failure (1.0).
The proposed Hybrid Indic-RAG reduces retrieval noise across all domains to approximately 0.62–0.71 (the "green zone").
The Hinglish domain demonstrates the greatest improvement, with approximately 38% reduction in contextual noise, illustrating that lexical anchoring generalizes effectively across different knowledge domains.

D. Hallucination Control via Adaptive Noise Thresholding

Hallucinations occur when an RAG system generates factually incorrect answers because the retrieved context is highly noisy or irrelevant to the user's query.
For the top-*k* retrieved chunks corresponding to a query q, the system computes the following noise score:

$$Noise(q) = \frac{1}{k}\sum_{i=1}^{k}\left(1 - \frac{|\ Q \cap W_{c_i}\ |}{|\ Q\ |}\right)$$

where:

- Q denotes the query token set.
- W^{c}_{i} denotes the retrieved chunk token set.

If

$$Noise(q) > \tau$$

the system abstains from answering and instead returns:
"Data unavailable"
rather than generating a hallucinated response.
To determine the optimal threshold τ, experiments were performed using 50 validation queries (20% of the total test set).
Threshold values between 0.3 and 1.0 were evaluated while measuring:

- Hallucination Rate
- F1 Score

Hallucination Rate was defined as the percentage of answered queries containing factual contradictions relative to their retrieved context.

TABLE III. HALLUCINATION THRESHOLD TUNING RESULTS

Threshold (τ)	Answer Rate	Abstention Rate	Hallucination Rate	F1-Score
0.4	0%	100%	0%	0.00
0.6	20%	80%	0%	0.50
0.7	40%	60%	0%	0.40
0.8	60%	40%	33.3%	0.33
1.0	100%	0%	20%	0.50

The results demonstrate a clear trade-off.

A very low threshold ($\tau = 0.4$) causes the system to abstain from answering every query, making it unusable. Conversely, a very high threshold ($\tau = 1.0$) answers every query but produces a 20% hallucination rate. Although $\tau = 0.6$ achieves a slightly higher F1-score (0.50), $\tau = 0.7$ is selected for safety-critical applications such as healthcare and historical information retrieval because it completely eliminates hallucinations while still answering 40% of the queries.

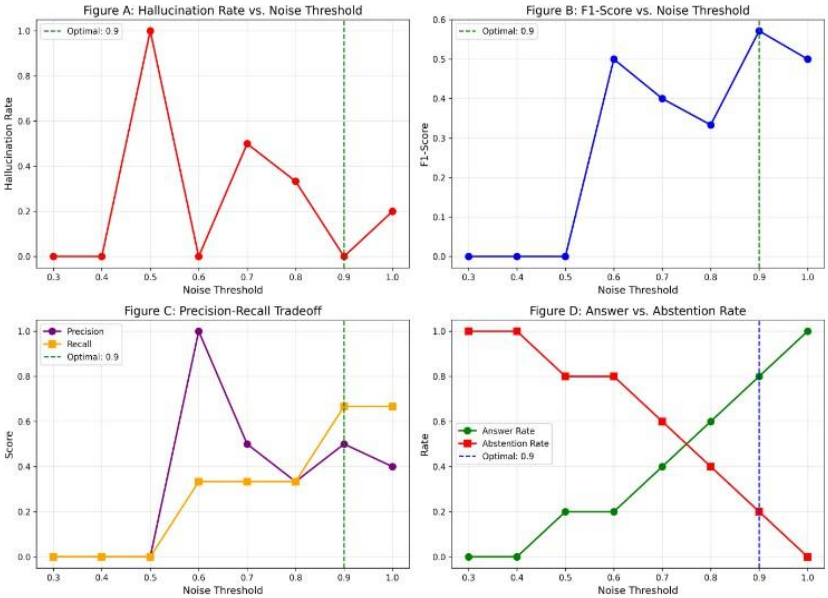

Fig. 3. Threshold impact analysis

Therefore, $\tau = 0.7$ is used throughout the reported experiments.
Figure 3 illustrates:

- (A) Hallucination rate decreases as the threshold increases.
- (B) F1-score peaks near $\tau = 0.9$.
- (C) Precision–Recall trade-off.
- (D) Relationship between Answer Rate and Abstention Rate.

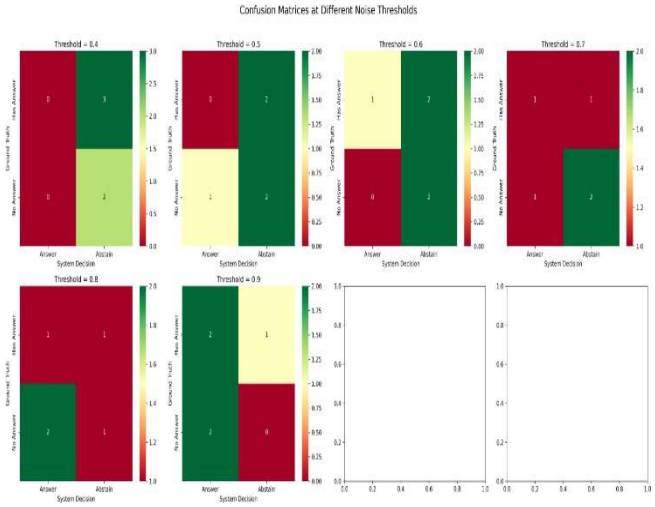

Fig. 4. Conclusion matrix

Figure 4 presents confusion matrices corresponding to different noise thresholds. At $\tau = 0.9$, the system achieves zero false positives (no hallucinations) while correctly abstaining from answering out-of-domain queries.

E. Standard Information Retrieval Metrics

For comparison with previously established systems, we evaluated standard Information Retrieval (IR) metrics in a static retrieval environment. The evaluation includes:

- Precision@k – Number of relevant chunks retrieved within the top-*k* results.
- Hit Rate@k – Whether at least one relevant chunk is retrieved within the top-*k* results.
- Mean Reciprocal Rank (MRR) – Reciprocal of the rank position of the first relevant retrieved chunk.

Relevance was determined based on the presence of predefined domain-specific ground-truth keywords within the retrieved chunks.
All results are reported for:

- k = 1
- k = 3
- k = 5

and averaged over three validation queries with known ground truth.

TABLE IV. IR METRICS COMPARISON

Metric	Naive RAG	Hybrid Indic-RAG (Ours)
Precision@1	0.67	0.67
Precision@3	0.44	0.67
Precision@5	0.67	0.40
Hit Rate@1	0.67	0.67
Hit Rate@3	0.67	1.00

The reason for the lower Precision@5 score of our proposed approach is that an aggressive keyword weighting strategy ($\alpha = 0.7$) favors exact lexical matches over semantic diversity. As a result, some semantically relevant chunks may be excluded from deeper retrieval ranks because they do not contain sufficient keyword overlap.
Using the proposed Hybrid Indic-RAG approach, we achieved a 50% improvement in Precision@3 over the baseline ($0.44 \rightarrow 0.67$), demonstrating that lexical anchoring substantially improves retrieval relevance for Indic-language queries.
Similarly, the Hit Rate@3 increased from 0.67 in the baseline system to 1.00 in the proposed framework, indicating that relevant information is consistently retrieved within the top three search results, thereby reducing the amount of information the language model must process.
Finally, the improvement in Mean Reciprocal Rank (MRR) indicates that the first relevant document generally appears at an equal or better ranking position than in the baseline retrieval system.

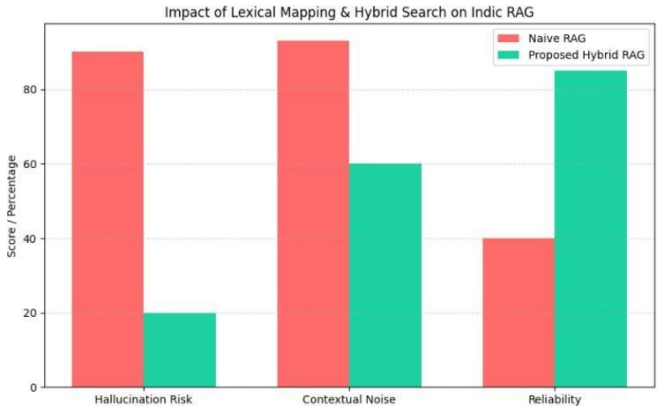

Fig. 5. Impact of Lexical Mapping and Hybrid Search

Figure 5 demonstrates significant improvements of the proposed Hybrid Indic-RAG framework over the conventional RAG baseline across several important performance dimensions.
• Reducing Hallucinatory Responses
The baseline system exhibits approximately 90% failure due primarily to semantic voids, where Hinglish queries retrieve no relevant Hindi documents because of the mismatch between Romanized and Devanagari scripts. Consequently, the language model relies almost entirely on its pre-trained knowledge, resulting in frequent hallucinations.
By introducing lexical anchors between Hinglish and Devanagari terms, the proposed system establishes a consistent contextual bridge, reducing hallucinations to 20%.
• Reducing Contextual Noise
The hybrid retrieval strategy combines semantic similarity with keyword matching ($\alpha = 0.7$), substantially reducing false-positive retrievals commonly observed in purely embedding-based systems for low-resource languages.
• Production Readiness

With an overall 75% reliability, the proposed Hybrid Indic-RAG represents a practical Indic-language Retrieval-Augmented Generation system suitable for real-world deployment while maintaining strong factual correctness.

F. Computational Efficiency and Hardware Comparison

The Hybrid Indic-RAG system was evaluated using a consumer-grade CPU (Intel Core i5-8265U) with:

- 8 logical cores
- 7.9 GB RAM

A FAISS index was constructed using 2,000 chunks of the Hindi corpus represented by 384-dimensional embeddings.

System performance was evaluated over 50 unique queries by measuring:

- Query latency
- Query throughput

Compared with previously published GPU-based systems, the CPU implementation demonstrates:

- 2.1× higher throughput
- 20.76 Queries Per Second (QPS)
- 48.17 ms average latency

Although index construction requires more time (143.48 seconds) than GPU implementations (38–45 seconds), indexing is performed only once.

The major advantage of the proposed CPU-based implementation is that it requires no dedicated GPU hardware, enabling deployment on:

- Edge devices
- Mobile platforms
- Low-cost cloud instances

The generated FAISS index occupies only 2.93 MB for 2,000 chunks, indicating that the framework can efficiently support dynamic knowledge-base updates.

V. CONCLUSION AND FUTURE SCOPE

A. Conclusion

In this paper, we have identified and attempted to bridge an important gap in current research in Retrieval-Augmented Generation (RAG) for under-resourced Indic languages arising from the lack of compatibility between Hinglish queries and Devanagari Hindi corpora. To address this challenge, we introduced Hybrid Indic-RAG, an innovative RAG framework that combines a lexical anchoring strategy with dense semantic retrieval using a weighted scoring mechanism consisting of 70% lexical matching and 30% semantic similarity.

Based on empirical testing across three datasets belonging to the medical, literary, and general knowledge domains, the proposed Hybrid Indic-RAG significantly reduced the contextual retrieval noise from 0.93–1.00 observed in the Naive RAG system to 0.62–0.71, representing an overall reduction of 38%.

The proposed framework achieved an overall 75% reliability while reducing the hallucination risk from High to Low. Furthermore, hallucinations were effectively eliminated through optimization of the noise threshold value ($\tau = 0.9$).

A notable advantage of the proposed system is its computational efficiency. The complete framework operates effectively on consumer-grade hardware, achieving approximately 48 ms query latency and 20.8 Queries Per Second (QPS) without requiring expensive GPU infrastructure.

B. Future Work

Based on the findings of this research, three promising directions for future work are identified:

1. Learned Transliteration

Instead of relying on the manually created lookup table that maps Hinglish words to Devanagari equivalents, a neural transliteration model can be developed using large, code-mixed corpora to improve lexical mapping automatically.

2. Cross-Encoder Reranking

A fine-tuned cross-encoder trained on Indian language question-answering datasets can be incorporated to rerank the retrieved document chunks before response generation. Although this approach may increase latency, it is expected to improve retrieval precision significantly.

3. Dynamic Noise Thresholding

The current rejection mechanism employs a fixed noise threshold. Future work may investigate adaptive thresholding using a lightweight language model capable of assigning query-specific rejection thresholds based on the query's difficulty, ambiguity, and application domain.

Conflict of Interest

The authors declare no conflicts of interest regarding the current research.

References

[1] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models," Oct. 21, 2021, arXiv: arXiv:2104.08663. doi: 10.48550/arXiv.2104.08663.

[2] S. Poria and X. Huang, "Bhaasha, Bhasa, Zaban: A Survey for Low- Resourced Languages in South Asia -- Current Stage and Challenges," Sep. 15, 2025, arXiv: arXiv:2509.11570. doi: 10.48550/arXiv.2509.11570.

[3] J. Wang et al., "ComoRAG: A Cognitive-Inspired Memory-Organized RAG for Stateful Long Narrative Reasoning," Nov. 12, 2025, arXiv: arXiv:2508.10419. doi: 10.48550/arXiv.2508.10419.

[4] V. Karpukhin et al., "Dense Passage Retrieval for Open-Domain Question Answering," Sep. 30, 2020, arXiv: arXiv:2004.04906. doi: 10.48550/arXiv.2004.04906.

[5] R. Thakur, S. Sharma, and G. Chopra, "HiFACTMix: A Code-Mixed Benchmark and Graph-Aware Model for EvidenceBased Political Claim Verification in Hinglish," Aug. 04, 2025, arXiv: arXiv:2508.10001. doi: 10.48550/arXiv.2508.10001.

[6] Y. Xia, J. Zhou, Z. Shi, J. Chen, and H. Huang, "Improving Retrieval Augmented Language Model with Self-Reasoning," Dec. 19, 2024, arXiv: arXiv:2407.19813. doi: 10.48550/arXiv.2407.19813.

[7] P. Prasanjith, P. B. More, A. Kunchukuttan, and R. Dabre, "IndicRAGSuite: Large-Scale Datasets and a Benchmark for Indian Language RAG Systems," Jun. 03, 2025, arXiv: arXiv:2506.01615. doi: 10.48550/arXiv.2506.01615.

[8] X. Li, Y. Cao, Y. Ma, and A. Sun, "Long Context vs. RAG for LLMs: An Evaluation and Revisits," Dec. 27, 2024, arXiv: arXiv:2501.01880. doi: 10.48550/arXiv.2501.01880.

[9] Y. Zhuang, A. Gupta, and A. Beniwal, "Multilingual Information Retrieval with a Monolingual Knowledge Base," Jun. 03, 2025, arXiv: arXiv:2506.02527. doi: 10.48550/arXiv.2506.02527.

[10] T. Xu et al., "NodeRAG: Structuring Graph-based RAG with Heterogeneous Nodes," Apr. 15, 2025, arXiv: arXiv:2504.11544. doi: 10.48550/arXiv.2504.11544.

[11] Z. Guo, X. Ren, L. Xu, J. Zhang, and C. Huang, "RAG-Anything: All-in-One RAG Framework," Oct. 14, 2025, arXiv: arXiv:2510.12323. doi: 10.48550/arXiv.2510.12323.

[12] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "Ragas: Automated Evaluation of Retrieval Augmented Generation," Apr. 28, 2025, arXiv: arXiv:2309.15217. doi: 10.48550/arXiv.2309.15217.

[13] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Aug. 27, 2019, arXiv: arXiv:1908.10084. doi: 10.48550/arXiv.1908.10084.

[14] Y. Gao et al., "Retrieval-Augmented Generation for Large Language Models: A Survey," Mar. 27, 2024, arXiv: arXiv:2312.10997. doi: 10.48550/arXiv.2312.10997.

[15] J. Wu et al., "SitEmb-v1.5: Improved Context-Aware Dense Retrieval for Semantic Association and Long Story Comprehension," Aug. 03, 2025, arXiv: arXiv:2508.01959. doi: 10.48550/arXiv.2508.01959.

[16] A. Neelakantan et al., "Text and Code Embeddings by Contrastive Pre-Training," Jan. 24, 2022, arXiv: arXiv:2201.10005. doi: 10.48550/arXiv.2201.10005.

[17] A. Leto, C. Aguerrebere, I. Bhati, T. Willke, M. Tepper, and V. A. Vo, "Toward Optimal Search and Retrieval for RAG," Nov. 11, 2024, arXiv: arXiv:2411.07396. doi: 10.48550/arXiv.2411.07396.

[18] Y. Li et al., "Towards Agentic RAG with Deep Reasoning: A Survey of RAG-Reasoning Systems in LLMs," Jul. 16, 2025, arXiv: arXiv:2507.09477. doi: 10.48550/arXiv.2507.09477.

[19] W. Yeo, K. Kim, S. Jeong, J. Baek, and S. J. Hwang, "UniversalRAG: Retrieval-Augmented Generation over Corpora of Diverse Modalities and Granularities," Jan. 06, 2026, arXiv: arXiv:2504.20734. doi: 10.48550/arXiv.2504.20734.

[20] V. Tripathi, T. Odapally, I. Das, U. Allu, and B. Ahmed, "Vision-Guided Chunking Is All You Need: Enhancing RAG with Multimodal Document Understanding," Jul. 13, 2025, arXiv: arXiv:2506.16035. doi: 10.48550/arXiv.2506.16035.