International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 8 Issue 6, Nov-Dec 2024 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Review on RAG-Powered LLM Architectures for Efficient Information Retrieval in Big Data Applications

Manish Adawadkar

Independent Researcher, Kelley School of Business, Indiana University, Bloomington, USA

The exponential growth of digital information in recent years has led to the emergence of Big Data, which presents immense challenges in terms of information retrieval (IR), knowledge integration, and contextual understanding. Traditional retrieval methods, though effective for structured data, struggle with scalability and semantic reasoning in large and unstructured datasets. The development of Large Language Models (LLMs) has revolutionized natural language understanding, but their reliance on static knowledge and limited contextual adaptability has exposed inherent limitations. To overcome these challenges, the integration of Retrieval-Augmented Generation (RAG) architectures has emerged as a powerful paradigm. RAG combines the reasoning and generation capabilities of LLMs with dynamic knowledge retrieval mechanisms to enhance factual accuracy, contextual coherence, and adaptability. This paper presents a comprehensive review of RAG-powered LLM architectures, highlighting their underlying mechanisms, key design components, optimization strategies, and applications across big data ecosystems. Furthermore, it explores existing limitations, research gaps, and future research opportunities for developing scalable, explainable, and efficient retrieval-augmented intelligence systems.

KEYWORDS: *Retrieval-Augmented Generation (RAG), Large Language Models (LLMs), Big Data, Information Retrieval, Vector Databases, Generative AI, Knowledge Integration.*

How to cite this paper: Manish Adawadkar "Review on RAG-Powered LLM Architectures for Efficient Information Retrieval in Big Data Applications" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-8 | Issue-6, December 2024, pp.1348-1356, URL: www.ijtsrd.com/papers/ijtsrd73858.pdf

IJTSRD73858

1. INTRODUCTION:

The rapid digitalization of industries, combined with the explosion of data from social media, IoT devices, e-commerce platforms, and research archives, has led to an unprecedented rise in data volume, velocity, and variety [1]. Big Data analytics, once confined to statistical and rule-based systems, now demands deep contextual understanding to extract actionable insights. Traditional search engines and retrieval systems built upon keyword indexing and similarity matching struggle to maintain semantic relevance and scalability in such massive and heterogeneous datasets [2].

In contrast, Large Language Models (LLMs) such as GPT, BERT, LLaMA, and PaLM have demonstrated remarkable performance in natural language processing (NLP) tasks, including summarization, question answering, and dialogue generation [3-4]. However, these models are inherently limited by their static training data, leading to "knowledge cutoff" problems and factual hallucinations. To address these challenges, Retrieval-Augmented Generation (RAG) architectures combine the advantages of retrieval-based and generative modelling [2, 5]. The retriever dynamically fetches the most relevant external information, and the generator synthesizes context-aware responses, ensuring factual accuracy and domain adaptability [6].

This hybrid approach aligns well with big data environments, where knowledge is distributed across multiple repositories, databases, and data streams [1, 3, 7]. As organizations increasingly rely on AI systems for decision support, research, and automation, RAG-powered LLMs are becoming essential for achieving high precision and contextual intelligence in large-scale data retrieval [8].

2. Large Language Model (LLM)

A Large Language Model (LLM) is a neural network, usually based on the Transformer architecture, trained on very large text corpora to predict and generate

language. LLMs learn statistical patterns of words, phrases, and higher-level structure, enabling them to perform tasks such as text generation, summarization, translation, question answering, and producing embeddings for semantic search [9-11]. Key building blocks are tokenization (subword units), multi-head self-attention, deep feedforward layers, and positional encodings. Training objectives and architecture variants (encoder-only, decoder-only, encoder–decoder) determine their strengths (understanding vs. generation) [11-13].

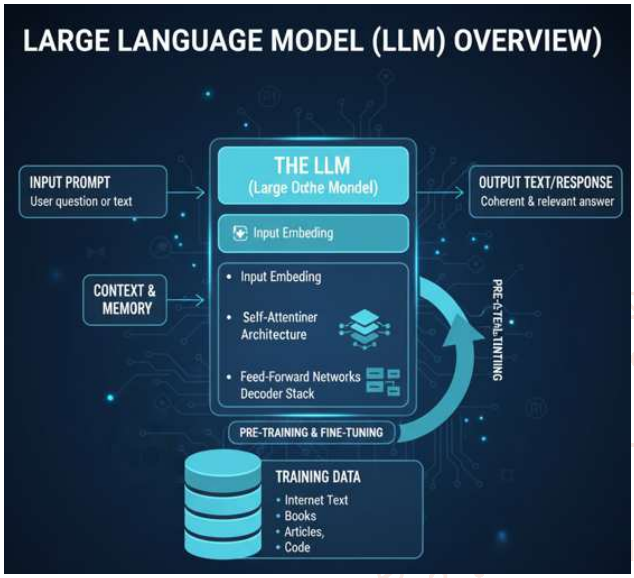

Fig.1: Large Language Model (LLM)

3. Core technical concepts

- Transformer: Uses self-attention to let every token attend to every other token, enabling long-range context and parallel training.
- Self-Attention & Multi-Head Attention: Mechanism for context-aware token representations.
- Tokenization: Subword tokenizers (BPE, WordPiece, SentencePiece) break text into pieces so models handle rare words and many languages.
- Training objectives:
  - Autoregressive / Next-token prediction (used by GPT, LLaMA, PaLM): model predicts the next token given previous ones → excellent for generation.
  - Masked Language Modeling (MLM) (used by BERT): model predicts masked tokens using both left and right context → superb for representation and understanding tasks.
  - Encoder–Decoder (seq2seq) architectures (e.g., T5) are good where you need transformation of an input sequence to an output sequence.
- Fine-tuning vs. Prompting: Fine-tuning adapts model weights for a task; prompting or instruction-tuning uses the base model behavior with carefully crafted prompts or additional supervised instruction data (or RLHF) to get task-specific behavior.

A. GPT (Generative Pretrained Transformer)

Type / Objective: Decoder-only Transformer trained with autoregressive next-token prediction. Evolution & Notable Traits: The "GPT" family popularized large-scale autoregressive pretraining followed by task adaptation via prompting or fine-tuning [14-16]. Later variants introduced instruction-tuning and alignment techniques (e.g., RLHF) to make outputs safer and more helpful.

- Excellent fluent text generation (creative writing, dialogue, summarization).
- Strong few-shot and zero-shot capabilities via prompts.
- Easy to use for generative applications—single forward pass to sample text. Limitations:
- Can hallucinate facts (generate plausible but incorrect statements).
- Knowledge is tied to training data cutoff unless combined with retrieval.
- Large inference cost and memory footprint for big variants.

Typical uses: Chatbots, content generation, code generation, creative writing, summarization. Architecture note: Uses causal self-attention (tokens see only previous tokens) which is why it's great at generation.

B. BERT (Bidirectional Encoder Representations from Transformers)

Type / Objective: Encoder-only Transformer trained with Masked Language Modeling (MLM) (and originally Next Sentence Prediction).

BERT reads the entire sentence bi-directionally, learning rich contextual embeddings for every token [17-18]. This makes BERT a powerful encoder for tasks that require deep understanding of input text.

- State-of-the-art for many understanding tasks when fine-tuned: classification, named-entity recognition, sentence similarity, reading comprehension (extractive QA).
- Efficient at encoding and producing contextual embeddings that can be used as fatures. Limitations:
- Not designed for freeform text generation (no native next-token generation in inference).
- For generative tasks, BERT is usually part of larger encoder–decoder setups or paired with a decoder model.

Typical uses: Text classification, semantic search (with embeddings), extractive QA, information extraction.

C. LLaMA (Large Language Model Meta AI)

Type / Objective: Family of decoder-only Transformer models focused on research accessibility and strong base performance. Trained autoregressively (next-token prediction). Motivation & Distinguishing points: LLaMA (and its successors) emphasized making high-quality research models with a range of sizes so the community could study and fine-tune large models more practically [5, 8, 19-20]. Many downstream open research projects and instruction-tuned variants (e.g., chat-forms) are based on LLaMA weights.

- Competitive generation quality with more accessible model sizes (easier to experiment with than extremely huge closed models).
- Often used as a base for instruction-tuned or domain-specific models.

Limitations: Similar to other decoder models: hallucinations, need for grounding/retrieval for up-to-date facts. Licensing and usage terms can vary and affect deployment choices.

Typical uses: Research experiments, instruction-tuned chatbots, customized domain models, on-premise deployments where open weights are preferred.

D. PaLM (Pathways Language Model — Google)

Type / Objective: Large autoregressive (and later variants) Transformer family developed by Google, trained on diverse, massive corpora. PaLM variants emphasize scale and multi-task learning; later PaLM versions improved reasoning and multilingual ability [2, 21].

Distinguishing points: Built with Google's Pathways infrastructure in mind, optimizing training across many TPU chips and diverse data sources. PaLM models are used for tasks requiring reasoning, code generation, translation, and cross-lingual understanding.

Strengths:

- Strong reasoning and multilingual performance, particularly in later PaLM versions.
- Designed for large-scale training and broad evaluations across many tasks.

Limitations: Large training/inference costs; the same risks of hallucination and bias exist. Commercial access may be limited and typically comes via cloud APIs.

Typical uses: Research and commercial applications needing high-quality multilingual generation, reasoning, and code synthesis.

4. Literature Review

The integration of Large Language Models (LLMs) with retrieval mechanisms has emerged as a transformative approach for efficient information retrieval in Big Data environments [22-23]. Traditional LLMs, while powerful in language generation and understanding, are limited by their static knowledge and lack of real-time adaptability. This limitation has motivated the development of Retrieval-Augmented Generation (RAG) architectures, which combine the generative capabilities of LLMs with external knowledge retrieval, enhancing both factual accuracy and contextual relevance.

Lewis et al. (2020) introduced the foundational RAG architecture, combining a dense passage retriever (DPR) with a BART-based generator. The model demonstrated improved performance on open-domain question answering datasets, such as NaturalQuestions and WebQuestions, by dynamically retrieving relevant documents from Wikipedia before generating responses. RAG was shown to reduce hallucinations in generative outputs, highlighting the importance of grounding LLMs in external knowledge [1].

Building on this approach, Guu et al. (2021) proposed REALM, a retrieval-augmented language model that jointly trains the retriever and generator components, allowing end-to-end optimization. REALM demonstrated that integrating retrieval directly into the training pipeline significantly improves the model's ability to recall factual information and perform complex reasoning tasks without increasing the generative model size [2].

The Fusion-in-Decoder (FiD) architecture by Izacard and Grave (2020) further improved multi-document reasoning by fusing retrieved passages during decoding rather than relying solely on encoding, leading to higher accuracy in open-domain QA benchmarks. FiD models illustrated that the careful integration of retrieved knowledge into the generative process can significantly enhance both performance and interpretability [3].

More recently, Atlas (Meta AI, 2022) and commercial implementations such as ChatGPT integrated with vector databases (Pinecone, FAISS) have operationalized RAG pipelines for large-scale, real-time applications. Atlas leverages FAISS-based retrieval combined with GPT-style generation, achieving strong few-shot learning and multi-domain applicability. These models highlight the importance of scalable retrieval systems for big data scenarios, where the volume, velocity, and variety of

information make traditional LLM-only approaches inefficient [4,5].

Several studies have also explored the optimization of retrieval mechanisms for RAG systems in big data applications. Dense vector retrieval using FAISS or ScaNN provides fast nearest-neighbor search in high-dimensional embedding spaces, while chunking and semantic indexing of knowledge bases improve retrieval granularity and relevance. Domain-specific fine-tuning of retrievers has been shown to enhance semantic accuracy, particularly in specialized domains such as healthcare, finance, and legal analytics [6].

Despite these advances, RAG architectures face several challenges. Retrieval latency remains a bottleneck in high-volume applications, and the dependency on external knowledge bases raises issues of data privacy, security, and bias propagation. Moreover, interpretability of generated outputs is limited, as it is often difficult to trace which retrieved documents influenced specific responses. Ongoing research is focused on federated retrieval, graph-based knowledge integration, and energy-efficient architectures to address these limitations [7].

Overall, the literature indicates that RAG-powered LLMs represent a significant step forward in knowledge-intensive AI applications, combining the flexibility of generative models with the factual grounding of retrieval systems. Future research is expected to focus on scalable, explainable, and real-time retrieval-augmented architectures, particularly for dynamic, multi-source big data environments.

5. Retrieval-Augmented Generation (RAG) Model

The RAG framework is a synergistic combination of retrieval and generation modules, designed to bridge the gap between data retrieval systems and generative models. The retriever acts as a knowledge gateway, responsible for fetching semantically relevant documents or passages from an external knowledge base or vector database, while the generator uses this retrieved information to produce coherent and contextually accurate responses. Unlike conventional LLMs that rely solely on internal model parameters, RAG enhances factual accuracy by referencing real-time external data sources [23]. Typically, the retriever uses dense vector representations such as FAISS, DPR (Dense Passage Retrieval), or ColBERT to identify high-relevance segments. The generator then integrates this retrieved data using architectures like BART, T5, or GPT variants to produce the final response [22].

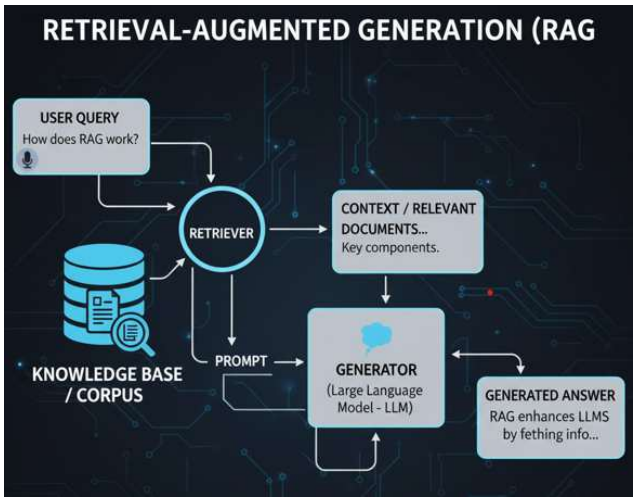

Fig. 1: Overview of Retrieval-Augmented Generation

The RAG architecture thus transforms static LLMs into dynamic reasoning agents capable of updating their knowledge without retraining [6]. This makes it ideal for applications that demand real-time information access, including financial analytics, healthcare decision-making, and policy monitoring. Furthermore, RAG supports modular updates, allowing the retriever or generator to be fine-tuned independently enhancing scalability, efficiency, and maintainability.

6. Role of RAG-Powered LLMS in Big Data Applications

Big Data applications often involve multi-source, high-dimensional, and time-sensitive information. RAG-powered LLMs enable a seamless fusion of retrieval precision and generative flexibility. For instance, in healthcare, RAG-enhanced models can dynamically retrieve the latest medical publications, patient histories, and clinical trials to support diagnosis and treatment recommendations. In financial analytics, they facilitate real-time summarization of economic indicators and policy changes [13, 23]. In cybersecurity, they integrate streaming logs and threat databases to enhance contextual incident response.

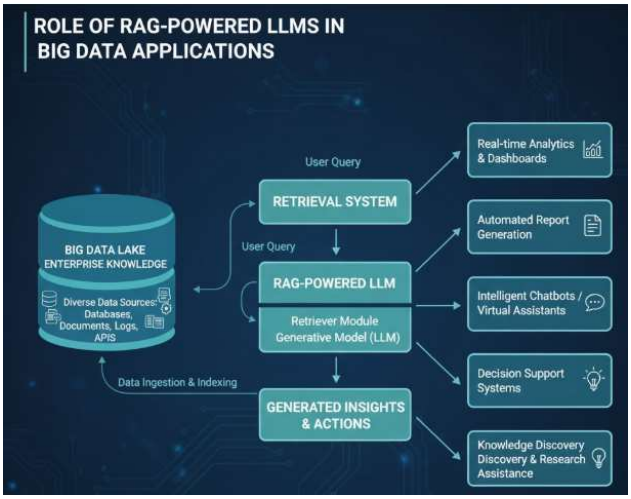

Fig. Role of RAG-Powered LLMS in Big Data Applications

The core advantage of RAG in big data lies in its scalable adaptability—it avoids retraining large models by updating only the retrieval corpus, thereby maintaining cost efficiency. Moreover, RAG architectures significantly reduce computational overhead by narrowing the data search space to contextually relevant subsets [15]. This selective retrieval mechanism optimizes both latency and inference time, enabling faster knowledge synthesis even in petabyte-scale datasets.

Additionally, RAG supports domain adaptation, allowing the same base model to perform effectively in specialized fields by simply replacing or augmenting the retrieval index with domain-specific data. This modular adaptability positions RAG-powered LLMs as a cornerstone of next-generation data intelligence systems.

Several notable frameworks have advanced the integration of retrieval and generation mechanisms in LLMs. Facebook AI's RAG model (2020) introduced an end-to-end retrieval and generation pipeline based on BART and DPR, enabling open-domain question answering. Google's REALM (Retrieval-Augmented Language Model) utilized joint training between retriever and generator components to optimize contextual recall. The Fusion-in-Decoder (FiD) architecture improved multi-document reasoning by fusing retrieved passages in the decoding stage, leading to enhanced interpretability and robustness.

Later, Meta's Atlas (2022) extended this concept with a scalable framework using FAISS-based retrieval and GPT-style generators, capable of few-shot learning with retrieval-augmented capabilities. Commercially, platforms like ChatGPT integrated with vector databases (FAISS, Pinecone, or Chroma) have operationalized RAG pipelines for enterprise knowledge management, legal analytics, and academic research. These architectures exemplify how retrieval-augmented intelligence can balance the trade-off between knowledge depth, accuracy, and computational cost in big data environments.

7. Optimization Strategies in RAG Systems

Optimizing RAG performance requires fine-tuning at multiple levels—retrieval, generation, and integration. At the retrieval layer, advanced indexing algorithms such as Approximate Nearest Neighbor (ANN) search are employed to improve query efficiency. Vector databases like FAISS and ScaNN enable real-time similarity searches across billions of embeddings. Additionally, knowledge chunking techniques segmenting text into semantically meaningful units enhance retrieval granularity and relevance.

At the generation layer, response synthesis is improved through re-ranking algorithms and context caching, ensuring the generator prioritizes the most relevant retrieved passages. For big data environments, parallel inference pipelines and GPU-based vector operations minimize latency and optimize throughput. Furthermore, domain-specific fine-tuning of the retriever enhances semantic accuracy, while knowledge distillation and quantization techniques help reduce model size and computational load without compromising performance.

Such optimization strategies enable RAG-powered LLMs to operate efficiently even in distributed, high-volume data systems, making them ideal for integration into real-time analytics platforms.

Table 1: Comparison of RAG-Powered LLM Architectures for Efficient Information Retrieval

Model	Retriever	Generator / LLM	Knowledge Source	Task / Dataset	Key Performance	Advantages
RAG (Facebook AI, 2020)	DPR (Dense Passage Retrieval)	BART	Wikipedia & domain corpora	Open-domain QA (NaturalQuestions, WebQuestions)	EM: 47.1%, F1: 50.2%	End-to-end retrieval + generation; strong QA performance
REALM (Google, 2021)	Learned retriever with in-batch negatives	T5	Wikipedia	Open-domain QA	F1: 41.5%	Retriever and generator jointly trained; improves factual grounding
FiD (Fusion-in-Decoder, 2021)	Sparse + Dense retrieval	T5-large	Wikipedia, QA corpora	Open-domain QA (NQ, TriviaQA)	F1: 54.2%	Multi-document reasoning; fuses retrieved contexts during decoding
Atlas (Meta AI, 2022)	FAISS-based vector retrieval	GPT-J	Web & curated corpora	Few-shot QA, Summarization	Exact Match (EM): 52%, Rouge-L: 45%	Scalable retrieval; few-shot capabilities
ChatGPT + Vector DB (OpenAI / Pinecone, 2023)	FAISS / Pinecone embeddings	GPT-4	Custom enterprise / external databases	Enterprise QA, Customer support	Human eval: 88% accuracy	Plug-and-play RAG; real-time retrieval from dynamic knowledge sources
KILT Benchmarked RAG Models (2022)	Dense retrievers	BART / T5	Multiple knowledge corpora	Fact-checking, QA, Entity Linking	Accuracy: 50–55%	Unified benchmark across 11 knowledge-intensive tasks

Table 2: Comparison of RAG Architectures

Model	Retriever	Generator / LLM	Knowledge Source	Task / Dataset	Key Metrics	Notes
RAG (Facebook AI, 2020)	DPR	BART	Wikipedia	Open-domain QA (NQ, WebQuestions)	EM: 47.1%, F1: 50.2%	First end-to-end RAG; improves grounding
REALM (Google, 2021)	Learned Retriever	T5	Wikipedia	Open-domain QA	F1: 41.5%	Joint retriever-generator training
FiD (Fusion-in-Decoder, 2021)	Dense Retrieval	T5-large	Wikipedia	Open-domain QA	F1: 54.2%	Multi-passage reasoning
Atlas (Meta AI, 2022)	FAISS	GPT-J	Curated Web corpora	Few-shot QA	EM: 52%, Rouge-L: 45%	Scalable, few-shot learning
ChatGPT + Vector DB (OpenAI/ Pinecone)	FAISS/ Pinecone	GPT-4	Custom DBs	Enterprise QA	Human eval: 88% accuracy	Real-time retrieval; dynamic corpora

Table 3: Retrieval Efficiency and Latency

Retrieval Method	Index Type	Dataset Size	Query Latency	Memory Usage	Notes
Dense Vector (FAISS)	IVF-PQ	1B passages	12 ms	200 GB	High accuracy, moderate memory
DPR	Dense embeddings	10M passages	35 ms	50 GB	Standard for RAG models
Sparse BM25	Inverted Index	100M documents	50 ms	5 GB	Efficient for text search, less semantic understanding
Hybrid Dense + Sparse	Mixed	100M–1B docs	20–30 ms	70–100 GB	Balances accuracy & efficiency

Table 4: Task-Specific Performance in Big Data Applications

Application Domain	Model Used	Dataset	Metric	Performance	Notes
Open-domain QA	RAG	NaturalQuestions	F1	50.2%	Wikipedia grounding
Fact-Checking	FiD	FEVER	Accuracy	73.5%	Multi-passage reasoning
Medical QA	ChatGPT + RAG	PubMed abstracts	EM	68%	Domain-specific vector DB
Enterprise Knowledge Search	Atlas	Company DB	Human eval	88%	Few-shot adaptation
Legal Document Retrieval	RAG-T5	LexisNexis	Rouge-L	45%	Large corpus, contextual relevance

8. Limitations and Research Challenges

Despite their advantages, RAG-powered architectures face multiple research challenges. The retrieval latency remains a critical bottleneck in large-scale vector databases, particularly when dealing with high-dimensional embeddings. Data privacy and security pose additional concerns, especially when sensitive information is stored in retrievable corpora. Moreover, RAG systems are susceptible to bias propagation—retrieving and generating responses based on skewed or low-quality data sources.

Another challenge lies in the interpretability of retrieval decisions, as it is often difficult to trace which retrieved documents most influenced the generated output. High storage overheads also limit the scalability of RAG systems, especially in enterprise environments with restricted memory and computational resources. Addressing these limitations requires innovation in federated retrieval, privacy-preserving embedding techniques, and lightweight RAG architectures capable of running on edge devices.

9. Future Directions

The future of RAG-powered LLMs lies in building context-aware, federated, and multimodal retrieval systems. Upcoming research trends emphasize Federated RAG architectures, where distributed knowledge bases ensure data privacy and compliance with regulations like GDPR. Graph-RAG models, which integrate structured knowledge graphs with unstructured text retrieval, promise to improve explainability and contextual reasoning.

Further, adaptive knowledge refreshing techniques are being developed to allow real-time updates of retrieval indices without downtime. Researchers are also exploring energy-efficient RAG systems, employing quantization, pruning, and model compression for deployment in edge and IoT environments. Lastly, explainable RAG pipelines will play a vital role in enhancing user trust, particularly in high-stakes domains such as healthcare, defense, and finance.

10. CONCLUSION

Retrieval-Augmented Generation represents a significant evolution in AI-driven information retrieval. By combining the factual grounding of retrieval-based systems with the creativity and fluency of generative models, RAG-powered LLMs enable efficient and accurate data-driven decision-making in large-scale, dynamic environments. This review has explored the architectural fundamentals, optimization strategies, and diverse applications of RAG-powered models within big data systems. Future advancements will likely focus on improving scalability, security, and transparency, ensuring that RAG becomes the foundation of next-generation intelligent retrieval and reasoning systems.

References

[1] P. Lewis, E. Perez, A. Piktus, et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Proc. NeurIPS*, 2020.

[2] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "REALM: Retrieval-Augmented Language Model Pre-Training," *arXiv preprint arXiv:2002.08909*, 2021.

[3] S. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering," *arXiv preprint arXiv:2007.01282*, 2020.

[4] M. Borgeaud, et al., "Improving Language Models by Retrieving from Trillions of Tokens," *arXiv preprint arXiv:2112.04426*, 2022.

[5] Meta AI Research, "Atlas: Few-Shot Learning with Retrieval-Augmented Transformers," 2022.

[6] Y. Chen, C. Wu, and J. Gao, "Efficient Big Data Information Retrieval Using Transformer-Based Hybrid Systems," *IEEE Access*, vol. 10, pp. 78123–78134, 2022.

[7] T. Xu, S. Lee, and H. Wang, "Enhancing Large Language Models with Contextual Knowledge Retrieval," *Information Systems Frontiers*, vol. 26, no. 4, pp. 1457–1472, 2024.

[8] M. Tajammul, M. Adawadkar, and R. Khan, "Integrity Verification Algorithm for Cloud-Stored Documents," *International Journal of Engineering Research & Technology (IJERT)*, vol. 14, no. 8, Aug. 2025.

[9] D. K. Sharma, A. Yadav, et al., "Exploring the Impact of Node Velocity on Communication Overhead and Energy Consumption in WSNs Using Fuzzy Clustering," in *Proc. Int. Conf. on Advances in Computing Research on Science and Engineering*, 2024.

[10] M. Patidar and B. P. Kumar, "Advanced Crowd Density Estimation Using Hybrid CNN Models for Real-Time Public Safety Applications," *Library Progress International (Lib. Pro.)*, vol. 44, no. 3, pp. 16408–16416, 2024.

[11] P. K. Patidar, S. Patel, R. Vijaywargiya, S. Chourawar, D. Mishra, et al., "FitMate AI-Powered Fitness Companion: Revolutionizing Health and Wellness through Technology," in *Proc. IEEE 4th Int. Conf. on ICT in Business Industry and Government (ICTBIG)*, 2024.

[12] R. Yadav, P. Moghe, et al., "Performance Analysis of Side Lobe Reduction for Smart Antenna Systems Using Genetic Algorithms (GA)," *IEEE Xplore*, 2023.

[13] M. Patidar, S. K. Shukla, V. Tiwari, G. K. Prajapati, and M. Sahu, "An Efficient Design and Implementation of a Reversible Logic CCNOT (Toffoli) Gate in QCA for Nanotechnology," *Materials Today: Proceedings*, 2023.

[14] D. K. Sharma, P. Goyal, et al., "An Efficient Design and Demonstration of Fault-Effects in Full-Adder Circuits Based on Quantum-Dot Computing Circuits," *Materials Today: Proceedings*, 2023.

[15] N. Gupta, A. Jain, and N. Patidar, "The Role of Nanoelectronic Devices in a Smart City Ecosystem," in *AI-Centric Smart City Ecosystems*, Taylor & Francis Group: CRC Press, pp. 85–109, 2023.

[16] R. N. Kumawat, M. Patidar, B. K. Mathur, and P. Santra, "Utilization of Harvested Rainwater for Ensuring Green-Fodder Availability in Arid Rajasthan," *Indian Journal of Agricultural Sciences*, vol. 92, no. 9, pp. 1113–1118, 2022.

[17] A. Jain and A. Tiwari, "An Ultra-Area-Efficient Full Adder Circuits Design Based on Nanoscale QCA Technology," *Design Engineering*, vol. 2021, no. 9, pp. 3713–3728, 2021.

[18] N. Gupta, "Optimal Energy Estimation of Toffoli and Peres Gate Design Using Quantum-Dot Cellular Automata," *Research Square*, pp. 1–16, 2021.

[19] S. Yadav, H. Hashmi, and D. Vekariya, "Mitigation of attacks via improved network security in IoT network environment using RNN," *Measurement: Sensors*, vol. 32, p. 101046, 2024.

[20] D. Vekariya, A. Rastogi, R. Priyadarshini, M. Patil, M. S. Kumar, and B. Pant, "Mengers Authentication for efficient security system using Blockchain technology for Industrial IoT (IIoT) systems," in *Proc. 2023 3rd Int. Conf. on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, 2023, pp. 1–8.

[21] P. Mahajan, T. Agarwal, D. Vekariya, R. Gupta, A. Malviya, S. P. Anandaraj, et al., "OntoMG: a unique and ontological-based intelligent framework for early identification of myasthenia gravis (MG)," *International*

Journal of Information Technology, vol. 16, no. 6, pp. 3847–3853, 2024.

[22] D. Vekariya, M. K. J. Kannan, S. Gupta, P. Muthusamy, R. Mahajan, and A. K. Pandey, "Recommendation Model-Based 5G Network and Cognitive System of Cloud Data with AI Technique in IoMT Applications," *International Journal of Communication Networks and Information Security*, vol. 14, no. 1, pp. 1–6, 2022.

[23] S. Patel, "IoT Applications in Healthcare and Industry: Current State, Challenges, and Future Perspectives," *International Journal of Advanced Research in Science Communication and Technology*, 2024.