RESEARCH ARTICLE OPEN ACCESS

BIRAG: Basic Introduction to Retrieval AugmentedGeneration

Kanishk Pratap Singh, Pradeep Kumar

Department of CSE Global Institute of Technology Jaipur, Rajasthan

ABSTRACT

In recent years, there has been a growing interestin leveraging both generative and retrieval-based methods toenhance natural language processing tasks. One such approach, known as Retrieval Augmented Generation (RAG), combines thestrengths of these two paradigms to achieve superior perfor- mance in tasks such as question answering, summarization, and dialogue generation. This paper provides an overview of the RAGframework, detailing its architecture, training methodologies, andapplications across various domains. We discuss the theoretical underpinnings of RAG, including its ability to leverage large- scale pre-trained language models and retrieve relevant context from external knowledge sources. Furthermore, we survey recentadvancements and challenges in RAG research, including strate- gies for efficient retrieval, fine-tuning techniques, and evaluation metrics. Finally, we highlight potential future directions for RAG,including its integration into real-world applications and its implications for the broader field of natural language understand- ing. Through this comprehensive examination, we aim to provide researchers and practitioners with a deeper understanding of RAG and its implications for advancing the state-of-the-art in natural language processing.

Keywords: -RAG, Retrieval, Generation, Natural Language Processing, Language Models

I. INTRODUCTION

Retrieval-augmented generation (RAG) is an AI framework designed to enhance the quality of responses generated by large language models (LLMs) by integrating external sources of knowledge. By grounding the model on these external sources, RAG supplements the LLM's internal representation of information, thereby improving the accuracy and relevance of its output. This process is particularly beneficial in LLM-based question answering systems, where RAG ensures access to up-to-date and reliable facts. Additionally, users can verify the model's claims by accessing its sources, enhancing trust in the generated responses. LLMs, with their vast training data and billions of parameters, are capable of generating original content for various tasks, such as answering questions and translating languages. RAG extends the capabilities of LLMs by enabling them to reference authoritative knowledge bases outside their training data sources, without requiring retraining. This approach enhances the relevance, accuracy, and usefulness of LLM output across different domains and organizational contexts, making it a cost-effective solution for improving response quality. [1] [2]

RAG has additional benefits. By grounding an LLM on a set of external, verifiable facts, the model has fewer opportunities to pull information baked into its parameters. This reduces thechances that an LLM will leak sensitive data, or 'hallucinate' incorrect or misleading information. [1]

II. HISTORICAL CONTEXT

A. The History of RAG

The origins of this technique can be traced back to the early 1970s, where researchers in information retrieval began experimenting with what they termed as question-answering systems. These systems utilized natural language processing (NLP) to access textual data, initially focusing on narrow topics such as baseball.

While the fundamental concepts of text mining have re- mained relatively stable over the years, the machine learning algorithms powering these systems have undergone signifi- cant advancements, greatly enhancing their effectiveness and widespread adoption.

In the mid-1990s, the Ask Jeeves service, now known as Ask.com, played a key role in popularizing question answer- ing, with its iconic mascot of a well-dressed valet. Another milestone occurred in 2011 when IBM's Watson gained fame as it convincingly outperformed two human champions on the Jeopardy! game show. [3]

B. Influential Research from a NLP paper

The influential paper emerged in 2020 while Lewis was completing his doctoral studies in Natural Language Process- ing (NLP) at University College London and concurrently working at Meta within a newly established AI laboratory in London. At that time, Lewis and his team were actively exploring methodologies to enhance the capacity of large language models (LLMs) by integrating more knowledge into their parameters. They developed a benchmark to assess their progress in this endeavor.

Drawing upon earlier techniques and inspired by a research paper authored by Google researchers, the team conceived a compelling vision of a trained system equipped with a retrieval index within its core architecture. This innovative design enabled the system to both learn from and generate diverse textual outputs as needed, as Lewis vividly recalls.

III. RAG VS FINE-TUNING

RAG and fine-tuning may confuse people, possibly because both techniques involve the use of custom data. In reality, RAG and fine-tuning both have different use cases; they're not competing techniques. To cut through this confusion, here's a high-level overview of these two methods.

A. Definitions

RAG (Retrieval Augmented Generation): Retrieval Augmented Generation (RAG) stands as a powerful technique to bolster the capabilities of large language models (LLMs) by anchoring them to specific contexts or knowledge sources. By integrating external knowledge sources, RAG enables LLMs to generate responses tailored to the given context, ranging from domain-specific information to personalized data. This approach proves invaluable in tasks such as question answering and chatbot interactions, where the ability to contextualize responses enhances accuracy and relevance. RAG empowers LLMs to not only understand the provided text but also to leverage external knowledge effectively, thereby elevating their performance and usefulness across various applications.

Fine-Tuning: Fine-tuning, on the other hand, serves as a method to refine and adapt LLMs to specific performance, style, or behavioral requirements. While not as effective in grounding LLMs in context or domain knowledge compared to RAG, fine-tuning allows for lasting changes in the model's behavior. It is particularly useful when existing models fail to meet desired criteria, offering a means to tailor the LLM's output to specific needs. However, fine-tuning is a demanding and time-consuming process, involving the retraining of the model on new data. As such, it is recommended to utilize pre-fine-tuned models whenever possible to mitigate the resource-intensive nature of fine-tuning.

Refer to below figures for the difference between the architecture of RAG and Fine-tuning [7]

B. Parameter Comparision

1) Accuracy: RAG generally outperforms fine-tuning in terms of accuracy due to its utilization of context retrieval, which helps minimize inaccuracies in generated responses. Fine-tuning, although capable of achieving high accuracy, may still produce instances of hallucination, especially if not rigorously trained.

2) Implementation Complexity: RAG offers a relatively straightforward implementation process. It involves context augmentation and model instruction, making it accessible to users with varying technical backgrounds. Conversely, fine-tuning is more complex, requiring the retraining of the model on specific data, which can be resource-intensive and demands a deeper understanding of machine learning principles.

3) Effort: RAG demands less effort compared to fine-tuning. It avoids the extensive process of retraining the model and instead focuses on managing context. Fine-tuning, however, requires significant effort, including the collection and preparation of training data, as well as allocating computational resources for the retraining process.

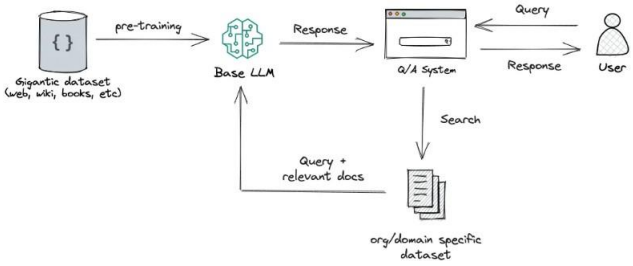

Fig. 1. Basic RAG Architecture

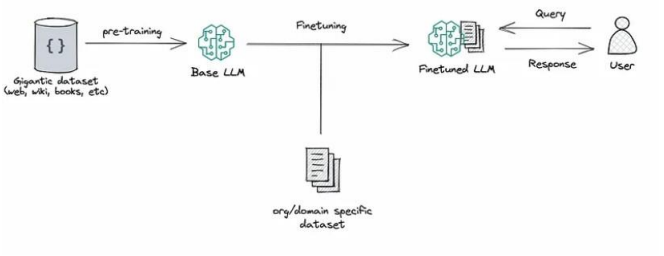

Fig. 2. Fine-Tuning Architecture

4) Total Cost of Ownership (TCO): RAG often offers a lower Total Cost of Ownership. Its reduced effort and resource requirements contribute to a more cost-effective solution. Conversely, fine-tuning can incur a higher TCO due to the substantial effort and computational resources needed for retraining.

5) Flexibility to Changes: RAG holds an advantage in flexibility to changes. It is easily adaptable to modifications since it doesn't require retraining the model. Context updates can be seamlessly integrated, providing a dynamic approach. In contrast, fine-tuning is less flexible to changes, as it involves retraining the model whenever updates or modifications are necessary. [5]

C. Use-cases

1) Summarization: For summarization in a specialized domain and/or a specific style, fine-tuning is more suitable due to its capacity for stylistic alignment.

2) Question-Answering: For a question/answering system based on organizational knowledge, a RAG system is more fitting, given its dynamic access to evolving knowledge bases.

3) Customer Support: For customer support automation, a hybrid approach might be optimal. Fine-tuning ensures brand-aligned customer experience, while RAG steps in for more dynamic or specific inquiry [6]

IV. The RAG Architecture

Let's dive deeper into each step of implementing RAG:

1) Source Data: The starting point of any RAG system is its source data, often consisting of a vast corpus of text documents, websites, or databases. This data serves as the knowledge reservoir that the retrieval model scans through to

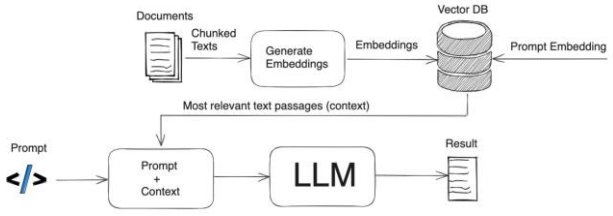

Fig. 3. The RAG Framework

find relevant information. It's crucial to have diverse, accurate, and high-quality source data for optimal functioning. It is also important to manage and reduce redundancy in the source data - for example, software documentation between version 1 and version 1.1 will be almost entirely identical to each other.

2) Data Chunking: Before the retrieval model can commence its search through the data, it is commonly partitioned into manageable "chunks" or segments. This chunking procedure guarantees that the system can efficiently navigate through the data, facilitating swift retrieval of pertinent content. Employing effective chunking strategies can substantially enhance the model's speed and accuracy. A document might serve as its own chunk, or it could be divided into chapters, sections, paragraphs, sentences, or even smaller units like "chunks of words." It is essential to bear in mind that the primary objective is to furnish the Generative Model with information that enriches its generation capabilities [12].

3) Text-to-Vector Conversion (Embeddings): Embeddings constitute a vibrant research domain, encompassing the generation of embeddings for various multimedia and multimodal entities such as tables, figures, and formulas. Typically, chunk embeddings are generated either during system development or when a new document is indexed. The preprocessing of queries significantly influences the performance of a RAG system, especially in managing negative or ambiguous queries. Further exploration is required into architectural patterns and approaches [14] to address the inherent limitations associated with embeddings [13], as the quality of a match depends on the domain.

4) Links between Source Data and Embeddings: The link between the source data and embeddings is the linchpin of the RAG architecture. A well-orchestrated match between them ensures that the retrieval model fetches the most relevant information, which in turn informs the generative model to produce meaningful and accurate text. In essence, this link facilitates the seamless integration between the retrieval and generative components, making the RAG model a unified system. [9]

5) Storing embeddings in vector store: Once the embeddings are generated for each chunk of text, they are stored in a vector store or database. This vector store serves as a repository where the embeddings can be efficiently retrieved during the inference phase. Storing the embeddings in a structured manner facilitates quick access and retrieval, thereby speeding up the processing during inference. [9]

6) Role of prompt and context: The user provides a prompt outlining their expectations or query. This initial prompt sets the stage for the subsequent steps in the process. Additionally, contextual search is employed to augment the original prompt with external information retrieved from various sources. This could involve querying databases, conducting keyword-based searches, or accessing APIs to fetch relevant data. The integration of this contextual information enriches the user's query, providing additional context and depth to guide the subsequent inference process.

7) Role of LLM (Language Model): Armed with the augmented prompt and contextual information, the Language Model (LLM) comes into play. The LLM leverages its understanding of natural language and contextual cues to interpret the user's query and generate accurate responses. By tapping into factual data sources and utilizing the contextual understanding provided by the augmented prompt, the LLM enhances the accuracy and relevance of its responses. It can infer relationships between different pieces of information, draw upon relevant facts, and generate coherent answers to the user's queries. [8]

8) Result: Finally, based on the input prompt, augmented context, and the understanding derived from the Language Model, a response is formulated. This response encapsulates accurate and relevant information tailored to address the user's query or expectations. The result is delivered back to the user, ensuring that they receive precise and informative answers to their inquiries.

In summary, the implementation of RAG involves a series of steps encompassing document preprocessing, embedding generation, leveraging prompts and context, employing a Language Model for inference, and delivering accurate responses to the user. Each step contributes to the overall effectiveness of the system in understanding and responding to user queries in a contextualized and informative manner.

V. Key Components of RAG

To comprehend the mechanics behind Retrieval Augmented Generation (RAG), it's essential to delve into its fundamental building blocks: retrieval models and generative models. These elements serve as the foundation for RAG's impressive ability to gather, integrate, and produce text abundant with information. Let's explore the unique contributions of each model and the synergies they create within the framework of RAG.

A. Retrieval Models

Retrieval models serve as the guardians of information within the RAG framework, tasked with scouring vast datasets to locate relevant nuggets of information for text generation. Imagine them as specialized librarians with a keen sense of which 'books' to retrieve from the 'shelves' when presented with a query. These models employ algorithms to assess and select the most pertinent data, thereby integrating external knowledge into the text generation process. By leveraging retrieval models, RAG enhances language generation with rich

context, extending the capabilities of conventional language models.

Various mechanisms can implement retrieval models. Among the most prevalent methods is utilizing vector embeddings and vector search. Additionally, document indexing databases employing technologies like BM25 (Best Match 25) and TF-IDF (Term Frequency — Inverse Document Frequency) are commonly employed. [9]

B. RAG Ranker

Within the RAG architecture, the ranker component plays a crucial role in refining the retrieved information by evaluating its relevance and significance. It accomplishes this by assigning scores or ranks to the retrieved data points, thus aiding in prioritizing the most pertinent ones.

Rankers employ a variety of algorithms, including text similarity metrics, context-aware ranking models, and machine learning techniques, to assess the quality of the retrieved content. This process ensures that the most relevant information is surfaced and presented to the generator for content generation, ultimately enhancing the overall quality and relevance of the generated text.

C. Generative Models

After the retrieval model has pinpointed the relevant information, generative models step into the spotlight. These models act as imaginative writers, weaving the retrieved data into coherent and contextually fitting text. Often constructed upon Large Language Models (LLMs), generative models possess the ability to produce text that is not only grammatically accurate but also semantically meaningful and aligned with the initial query or prompt. They take the raw data selected by the retrieval models and transform it into a structured narrative, rendering the information easily comprehensible and actionable. Within the RAG framework, generative models serve as the final component, delivering the textual output we engage with. [9]

VI. Applications of RAG

RAG is useful in many areas and industries because it can mix retrieval and generation methods to make text better and find information easier. Here are some important ways RAG is used:

A. Advanced Question-Answering Systems

RAG models empower question-answering systems to retrieve and generate accurate responses, thereby improving information accessibility. For instance, healthcare organizations can utilize RAG to develop systems that offer precise medical information sourced from literature.

B. Content Creation and Summarization

RAG models streamline content creation by retrieving relevant information from diverse sources, facilitating the development of high-quality articles, reports, and summaries. They also excel in generating coherent text based on specific prompts or topics, assisting in tasks such as news article generation or report summarization.

C. Conversational Agents and Chatbots

RAG models enhance conversational agents by fetching contextually relevant information from external sources, ensuring accurate and informative responses during interactions. This enhancement makes customer service chatbots and virtual assistants more effective in assisting users.

D. Information Retrieval

RAG models improve the relevance and accuracy of search results in information retrieval systems. By combining retrieval-based methods with generative capabilities, they enable search engines to retrieve documents based on user queries and generate informative snippets.

E. Educational Tools and Resources

Embedded within educational tools, RAG models personalize learning experiences by retrieving and generating tailored explanations, questions, and study materials, catering to individual needs and enhancing the educational journey.

F. Legal Research and Analysis

RAG models streamline legal research processes by retrieving relevant legal information, aiding legal professionals in drafting documents, analyzing cases, and formulating arguments more efficiently and accurately.

G. Content Recommendation Systems

Powering advanced content recommendation systems, RAG models understand user preferences, leverage retrieval capabilities, and generate personalized recommendations across digital platforms, enhancing user experience and content engagement.

VII. Advantages, Challenges and Limitations

A. The Benefits of Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) offers a multitude of benefits in the realm of Natural Language Processing (NLP) and text generation [8]:

1) Improved Accuracy: RAG models ensure factual accuracy by tapping into external knowledge sources, making them invaluable for precision-critical tasks like question-answering and educational content creation.

2) Contextual Relevance: By integrating external context, RAG-generated responses better align with user queries or context, offering more meaningful and contextually appropriate answers.

3) Enhanced Coherence: External context integration ensures logical flow and coherence in RAG-generated content, particularly beneficial for longer text pieces or narratives.

4) Versatility: RAG models are adaptable to a wide array of tasks and query types, transcending specific domains and providing relevant information across various subjects.

5) Efficiency: RAG efficiently accesses and retrieves information from extensive knowledge sources, saving time compared to manual searches, making it ideal for applications where quick responses are essential.

6) Content Summarization: RAG aids in summarizing lengthy documents or articles by selecting relevant information and generating concise summaries, simplifying content consumption.

7) Customization: RAG systems can be fine-tuned and customized for specific domains or applications, allowing organizations to tailor models to their unique needs.

8) Multilingual Capabilities: RAG models can generate content in multiple languages, facilitating international applications, translation tasks, and cross-cultural communication.

9) Decision Support: RAG assists in decision-making processes by providing well-researched, fact-based information, supporting informed choices in fields like healthcare, finance, and law.

10) Reduce Manual Effort: RAG reduces the need for manual research and information retrieval, saving human effort and resources, particularly beneficial for processing large data volumes.

11) Innovative Applications: RAG enables innovative NLP applications such as intelligent chatbots, virtual assistants, and automated content generation, enhancing user experiences and productivity.

B. Challenges and Limitations

1) Model Complexity: RAG's intricate architecture, which combines retrieval and generative processes, demands substantial computational resources. Debugging and optimizing such complex systems for efficient performance is challenging, requiring expertise in both AI and software engineering.

2) Data Preparation Challenges: Preparing suitable data for RAG involves ensuring cleanliness, relevance, and non-redundancy of text. Segmenting this text for optimal use by the generative model is complex, requiring careful selection of embedding models and preprocessing techniques that perform well across diverse datasets.

3) Prompt Engineering for LLM: Effective utilization of RAG necessitates skillful prompt engineering to appropriately frame retrieved information for the Large Language Model (LLM). Crafting prompts that effectively guide the LLM towards generating high-quality responses requires domain expertise and linguistic finesse.

4) Performance Trade-off: The dual-process nature of RAG, involving both data retrieval and text generation, can lead to increased response times. Achieving a balance between the depth of retrieval and the speed of response is crucial, especially in real-time applications where timely responses are essential for user satisfaction.

5) Handling Diverse Query Types: RAG models must be versatile enough to handle a wide range of query types, from straightforward factual questions to complex, nuanced queries. Adapting the retrieval and generation components to suit this diversity requires careful algorithmic design and training strategies.

6) Balancing Retrieval and Generation: Striking the right balance between retrieval and generation is essential to ensure responses generated by RAG are both creative and contextually relevant. Over-reliance on retrieval may lead to responses lacking creativity or context, while excessive generation may result in less factual or relevant answers.

7) Scaling to Large Datasets: As knowledge bases and data sources continue to grow, RAG models must scale efficiently to handle massive datasets without sacrificing response times and accuracy. Efficient data storage, retrieval, and processing techniques are crucial for scalability.

8) Evaluation Metrics: Assessing the performance of RAG models can be complex, especially when traditional metrics may not fully capture the quality of responses. Developing robust evaluation frameworks that take into account factors such as factual accuracy, contextual relevance, and coherence is essential for meaningful assessment.

9) Ethical Considerations: RAG models may inadvertently generate biased, offensive, or harmful content, posing ethical challenges. Ensuring responsible use and mitigating ethical concerns in content generation requires careful consideration of algorithmic biases and societal implications.

10) Limited Real-time Information: RAG models are often based on static knowledge sources, which means they may not provide accurate information for rapidly changing real-time events or developments. Integrating mechanisms for real-time updates and incorporating dynamic sources of information is a technical challenge.

11) Cost and Resource Intensiveness: Implementing RAG systems, particularly with large-scale knowledge bases, can be resource-intensive in terms of computation, storage, and data preprocessing. Optimizing resource utilization and cost-effectiveness while maintaining performance is a critical consideration for practical deployment of RAG solutions.

VIII. Future Prospects of RAG

Despite significant advancements in RAG technology, several challenges persist that necessitate thorough investigation:

A. Challenges for Future of RAG

1) Context Length: RAG's effectiveness is constrained by the context window size of Large Language Models (LLMs). Striking a balance between a window that is too short, risking inadequate information, and one that is too long, risking information dilution, is crucial. As efforts continue to expand LLM context windows to virtually unlimited sizes, adapting RAG to these changes presents a significant research question.

2) Robustness: The presence of noise or contradictory information during retrieval can adversely impact RAG's output quality. This scenario is figuratively referred to as "Misinformation can be worse than no information at all." Enhancing RAG's resistance to such adversarial or counterfactual inputs is gaining research momentum and has become a key performance metric.

3) Hybrid Approaches (RAG+FT): Combining RAG with fine-tuning is emerging as a leading strategy. Determining the optimal integration of RAG and fine-tuning—whether sequential, alternating, or through end-to-end joint training—and how to harness both parameterized and non-parameterized advantages are areas ripe for exploration.

4) Expanding LLM Roles: Beyond generating final answers, LLMs are leveraged for retrieval and evaluation within RAG frameworks. Identifying ways to further unlock LLMs' potential in RAG systems is a growing research direction.

5) Scaling Laws: While scaling laws are established for LLMs, their applicability to RAG remains uncertain. Initial studies have begun to address this, yet the parameter count in RAG models still lags behind that of LLMs. The possibility of an Inverse Scaling Law, where smaller models outperform larger ones, is particularly intriguing and merits further investigation.

6) Production-Ready RAG: RAG's practicality and alignment with engineering requirements have facilitated its adoption. However, enhancing retrieval efficiency, improving document recall in large knowledge bases, and ensuring data security—such as preventing inadvertent disclosure of document sources or metadata by LLMs—are critical engineering challenges that remain to be addressed.

7) Integration of Translation Memory: The information used for deriving reward scores is limited. The similarity between an input and retrieved examples is the primary feature to derive reward scores. However, some information, e.g., frequencies of words and context, may also be beneficial for integrating the translation memory. Second, it remains to be an open question that when should we use the retrieved information and when not. In the inference phase, approaches tend to integrate the translation memory excessively, e.g., at each time step, which not only reduces the translation efficiency but may also dampen the fluency of generated results.b11

B. Modality Extension

The modality extension of RAG has moved beyond its initial text-based question-answering scope, embracing a wide range of modal data. This expansion has given rise to innovative multimodal models that integrate RAG principles across various domains:

1) Image: RA-CM3 stands out as a groundbreaking multimodal model capable of retrieving and generating both text and images. BLIP-2 utilizes frozen image encoders alongside LLMs for efficient visual language pre-training, enabling seamless image-to-text conversions. The "Visualize Before You Write" approach employs image generation to guide text generation, showing promise in open-ended text generation tasks.

2) Audio and Video: The GSS method retrieves and combines audio clips to convert machine-translated data into speech-translated data. UEOP represents a significant advancement in end-to-end automatic speech recognition by incorporating external strategies for voice-to-text conversion. KNN-based attention fusion enhances ASR by leveraging audio and text embeddings for domain adaptation. Vid2Seq enhances language models with temporal markers to predict event boundaries and generate textual descriptions within a unified output sequence.

3) Code: RBPS excels in small-scale learning tasks by retrieving code examples aligned with developers' objectives through encoding and frequency analysis. This approach has demonstrated efficacy in tasks such as test assertion generation and program repair. For structured knowledge, the CoK method extracts relevant facts from knowledge graphs and integrates them as hints within the input, enhancing performance in knowledge graph question-answering tasks.

REFERENCES

[1] Martineau, K. (2023, November 30). What is retrieval-augmented gen- eration? IBM Research Blog. https://research.ibm.com/blog/retrieval- augmented-generation-RAG

[2] What is RAG? - Retrieval-Augmented Generation Explained - AWS. (n.d.). Amazon Web Services, Inc. https://aws.amazon.com/what- is/retrieval-augmented-generation/

[3] Merritt, R. (2024, April 2). What Is Retrieval-Augmented Generation aka RAG — NVIDIA Blogs. NVIDIA Blog. https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/

[4] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, Ming-Wei Chang REALM: Retrieval-Augmented Language Model Pre-Training

[5] Yadav, A. (2024, April 12). RAG vs. Fine-tuning Guide — Fab- ricHQ. Medium. https://blog.fabrichq.ai/rag-vs-fine-tuning-heres-the-detailed-comparison-c61cfeb80926

[6] Srivatsa, H. (2024, March 4). Fine-tuning versus RAG inGenerative AI Applications Architecture. Medium. https://harsha-srivatsa.medium.com/fine-tuning-versus-rag-in-generative-ai- applications-architecture-d54ca6d2acb8

[7] Hotz, H. (2023, October 5). RAG vs Finetuning — Which Is the Best Tool to Boost Your LLM Application? Medium. https://towardsdatascience.com/rag-vs-finetuning-which-is-the-best-tool-to-boost-your-llm-application-94654b1eaba7

[8] WB. (2024, April 17). Weights biases. WB.https://wandb.ai/cosmo3769/RAG/reports/A-Gentle-Introduction-to-Retrieval-Augmented-Generation-RAG—Vmlldzo1MjM4Mjk1

[9] Retrieval Augmented Generation (RAG): A comprehensive guide. (2024, March 1). DataStax. https://www.datastax.com/guides/what-is-retrieval-augmented-generation

[10] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang and Haofen Wang Retrieval-Augmented Generation for Large Language Models: A Survey

[11] Huayang Li, Yixuan Su, Deng Cai, Yan Wang, Lemao Liu A Survey on Retrieval-Augmented Text Generation

[12] Yuanjie Lyu, Zhiyu Li, Simin Niu, Feiyu Xiong, Wemjin Wang, Huany- ong Liu, Tong Xu, Enhong Chen, Yi Luo, Peng Cheng, Haiying Deng, Zhonghao Wang, Zijia Lu, Bo Tang, Hao Wu CRUD-RAG: A Com- prehensive Chinese Benchmark for Retrieval-Augmented Generation of Large Language Models

[13] Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, Mohamed Abdelrazek Seven Failure Points When Engineering a Re- trieval Augmented Generation System

[14] Alex Cummaudo, Scott Barnett, Rajesh Vasa, and John Grundy. 2020. Threshy: Supporting safe usage of intelligent web services. In Proceed- ings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineer- ing. 1645–1649.

[15] Rajesh Kr. Tejwani, Mohit Mishra, Amit Kumar. (2015). New Error Model of Entropy Encoding for Image Compression. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 1(3), 07–11. Retrieved from http://www.ijfrcsce.org/index.php/ijfrcsce/article/view/1886

[16] P. Jha, D. Dembla and W. Dubey, "Implementation of Machine Learning Classification Algorithm Based on Ensemble Learning for Detection of Vegetable Crops Disease", International Journal of Advanced Computer Science and Applications, Vol. 15, No. 1, pp. 584-594, 2024.

[17] G.K. Soni, A. Rawat, S. Jain and S.K. Sharma, "A Pixel-Based Digital Medical Images Protection Using Genetic Algorithm with LSB Watermark Technique", Springer Smart Systems and IoT: Innovations in Computing. Smart Innovation Systems and Technologies, vol. 141, pp 483–492, 2020.

[18] Pradeep Jha, Deepak Dembla & Widhi Dubey, "Deep learning models for enhancing potato leaf disease prediction: Implementation of transfer learning based stacking ensemble model", Multimedia Tools and Applications, Vol. 83, pp. 37839–37858, 2024.

[19] Yogita Sahu, Gaurav Kumar Soni, Himanshu Singh, Dimple Jangir, Akash Rawat, "Design of High Linearity Nanoscale CMOS OTA Based Bandpass Filter for Bluetooth Receiver", Journal of Emerging Technologies and Innovative Research (JETIR), Vol. 6, Issue. 1, pp. 335-338, 2019.

[20] Dr. Himanshu Arora, Gaurav Kumar soni, Deepti Arora, "Analysis and Performance Overview of RSA Algorithm", International Journal of Emerging Technology and Advanced Engineering, Vol. 8, Issue. 4, pp. 10-12, 2018.

[21] Rajesh Kr. Tejwani, Mohit Mishra, Amit Kumar. (2016). Evaluating the Performance of Similarity Measures in Effective Web Information Retrieval. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 2(8), 18–22.

[22] P. Jha, T. Biswas, U. Sagar and K. Ahuja, "Prediction with ML paradigm in Healthcare System," 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2021, pp. 1334-1342, doi: 10.1109/ICESC51422.2021.9532752.

[23] Mr. Gaurav Kuamr Soni, Mr. Kamlesh Gautam and Mr. Kshitiz Agarwal, "Flipped Voltage Follower Based Operational Transconductance Amplifier For High Frequency Application", International Journal of Advanced Science and Technology, vol. 29, no. 9s, pp. 8104-8111, 2020.

[24] S. Pathak, K. Gautam, A. K. Sharma and G. Kashyap, "A survey on artificial intelligence for Vehicle to everything," in International Journal of Engineering Research and Generic Science (IJERGS), vol. 7, no. 3, pp. 24-28, May-June 2021.

[25] K. Gautam, S. K. Yadav, K. Kanhaiya and S. Sharma, "Hybrid Software Development Model Outcomes for In-House IT Team in the Manufacturing Industry" in International Journal of Information Technology Insights & Transformations (Eureka Journals), vol. 6, no. 1, pp. 1-10, May 2022.

[26] J. Dabass, K. Kanhaiya, M. Choubisa and K. Gautam, "Background Intelligence for Games: A Survey" in Global Journal on Innovation, Opportunities and Challenges in AAI and Machine Learning (Eureka Journals), vol. 6, no. 1, pp. 11-22, May 2022.

[27] S. Gour and G. K. Soni, "Reduction of Power and Delay in Shift Register using MTCMOS Technique," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI), pp. 202-206, 2020. doi: 10.1109/ICOEI48184.2020.9143026.

[28] Amit Kumar, Mohit Mishra, Rajesh Kr. Tejwani. (2017). Image Contrast Enhancement with Brightness Preserving Using Feed Forward Network. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 3(9), 266–271.

[29] Pradeep Jha, Deepak Dembla & Widhi Dubey , "Implementation of Transfer Learning Based Ensemble Model using Image Processing for Detection of Potato and Bell Pepper Leaf Diseases", International Journal of Intelligent Systems and Applications in Engineering, 12(8s), 69–80, 2024.

[30] S. Gour, G.K. Soni and A. Sharma, "Analysis and Measurement of BER and SNR for Different Block Length in AWGN and Rayleigh Channel" in Emerging Trends in Data Driven Computing and Communications. Studies in Autonomic Data-driven and Industrial Computing, Singapore:Springer, 2021.

[31] P. Upadhyay, K. K. Sharma, R. Dwivedi and P. Jha, "A Statistical Machine Learning Approach to Optimize Workload in Cloud Data Centre," 2023 7th International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2023, pp. 276-280, doi: 10.1109/ICCMC56507.2023.10083957.

[32] Pradeep Jha, Deepak Dembla & Widhi Dubey , "Crop Disease Detection and Classification Using Deep Learning-Based Classifier Algorithm", Emerging Trends in Expert Applications and Security. ICETEAS 2023. Lecture Notes in Networks and Systems, vol 682, pp. 227-237, 2023.

[33] P. Jha, D. Dembla and W. Dubey, "Comparative Analysis of Crop Diseases Detection Using Machine Learning Algorithm," 2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2023, pp. 569-574, doi: 10.1109/ICAIS56108.2023.10073831.

[34] P. Jha, R. Baranwal, Monika and N. K. Tiwari, "Protection of User's Data in IOT," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2022, pp.

1292-1297, doi: 10.1109/ICAIS53314.2022.9742970.

[35] Mehra, M., Jha, P., Arora, H., Verma, K., Singh, H. (2022). Salesforce Vaccine for Real-Time Service in Cloud. In: Shakya, S., Balas, V.E., Kamolphiwong, S., Du, KL. (eds) Sentimental Analysis and Deep Learning. Advances in Intelligent Systems and Computing, vol 1408. Springer, Singapore. https://doi.org/10.1007/978-981-16-5157-1_78

[36] Gaur, P., Vashistha, S., Jha, P. (2023). Twitter Sentiment Analysis Using Naive Bayes-Based Machine Learning Technique. In: Shakya, S., Du, KL., Ntalianis, K. (eds) Sentiment Analysis and Deep Learning. Advances in Intelligent Systems and Computing, vol 1432. Springer, Singapore. https://doi.org/10.1007/978-981-19-5443-6_27

[37] Rajesh Kr. Tejwani, Mohit Mishra, Amit Kumar. (2018). Edge Computing in IoT: Vision and Challenges. International Journal on Future Revolution in Computer Science &Amp; Communication Engineering, 4(8), 88–97