applied sciences

Article

Leveraging Retrieval-Augmented Generation for Swahili Language Conversation Systems

Edmund V. Ndimbo [1], Qin Luo [1], Gimo C. Fernando [1], Xu Yang [2] and Bang Wang [1,3,*]

[1] School of Information, Electronic and Communications, Huazhong University of Science and Technology, Wuhan 430074, China; ndimboeddy@gmail.com (E.V.N.); luo_qin@hust.edu.cn (Q.L.) tivane2012@outlook.com (G.C.F.)
[2] Hubei Key Laboratory of Intelligent Yangtze and Hydroelectric Science, China Yangtze Power Co., Ltd., Yichang 443000, China; yang_xu@ctg.com.cn
[3] Hubei Key Laboratory of Smart Internet Technology, Huazhong University of Science and Technology, Wuhan 430074, China
* Correspondence: wangbang@hust.edu.cn

Abstract: A conversational system is an artificial intelligence application designed to interact with users in natural language, providing accurate and contextually relevant responses. Building such systems for low-resource languages like Swahili presents significant challenges due to the limited availability of large-scale training datasets. This paper proposes a Retrieval-Augmented Generation-based system to address these challenges and improve the quality of Swahili conversational AI. The system leverages fine-tuning, where models are trained on available Swahili data, combined with external knowledge retrieval to enhance response accuracy and fluency. Four models—mT5, GPT-2, mBART, and GPT-Neo—were evaluated using metrics such as BLEU, METEOR, Query Performance, and inference time. Results show that Retrieval-Augmented Generation consistently outperforms fine-tuning alone, particularly in generating detailed and contextually appropriate responses. Among the tested models, mT5 with Retrieval-Augmented Generation demonstrated the best performance, achieving a BLEU score of 56.88%, a METEOR score of 72.72%, and a Query Performance score of 84.34%, while maintaining relevance and fluency. Although Retrieval-Augmented Generation introduces slightly longer response times, its ability to significantly improve response quality makes it an effective approach for Swahili conversational systems. This study highlights the potential of Retrieval-Augmented Generation to advance conversational AI for Swahili and other low-resource languages, with future work focusing on optimizing efficiency and exploring multilingual applications.

Keywords: retrieval-augmented generation; Swahili NLP; language model optimization; low-resource languages; model performance

check for updates

Academic Editor: Pedro Couto

Received: 6 December 2024
Revised: 28 December 2024
Accepted: 30 December 2024
Published: 8 January 2025

Citation: Ndimbo, E.V.; Luo, Q.; Fernando, G.C.; Yang, X.; Wang, B. Leveraging Retrieval-Augmented Generation for Swahili Language Conversation Systems. *Appl. Sci.* **2025**, *15*, 524. https://doi.org/10.3390/app15020524

1. Introduction

Conversational systems are advanced artificial intelligence (AI) tools designed to interact with users through natural language. These systems are widely used in applications such as virtual assistants, customer service chatbots, and tools for education and healthcare [1]. By enabling machines to understand and respond to human language, conversational systems have become an essential part of modern technology. Over the years, significant advancements in Natural Language Processing (NLP) have enhanced the ability of these systems to generate human-like responses [2]. However, building conversational systems for low-resource languages, such as Swahili, remains a considerable challenge due to the limited availability of large, high-quality datasets.

Swahili, spoken by over 100 million people in East Africa, including Tanzania, Kenya, and Uganda, is a linguistically rich and culturally significant language but is underrepresented in the field of NLP [3]. Unlike widely spoken languages such as English or Chinese, which benefit from abundant digital resources, Swahili lacks the extensive datasets necessary for training advanced language models. This scarcity of data affects various aspects of NLP, such as syntactic parsing, semantic analysis, and named entity recognition. Moreover, the complex linguistic features of Swahili, including its extensive noun class system, agglutinative morphology, and verb conjugations, add to the challenge of creating effective language models. The absence of domain-specific corpora further limits the development of conversational systems capable of generating accurate and contextually relevant responses for Swahili speakers.

As a result, Swahili speakers face significant challenges in accessing the benefits of modern conversational AI tools. These challenges include the inability to access localized customer service tools, limited access to Swahili-enabled educational applications, and a lack of support for everyday queries in digital assistants. The lack of digital inclusion restricts Swahili speakers from fully participating in technological advancements, contributing to a growing digital divide. Retrieval-Augmented Generation (RAG) offers a promising solution to these challenges by enhancing conversational systems, even in low-resource settings. RAG combines two key components: large language models (LLMs) and retrieval systems. LLMs are trained on vast amounts of text to generate fluent and coherent responses, while retrieval systems fetch relevant information from external databases [4,5]. Together, these technologies allow RAG-based systems to incorporate external knowledge dynamically, making their responses more accurate and context-aware. For example, if a user asks a question about Swahili culture, a RAG system can retrieve information from a relevant database and use it to craft a well-informed answer. This capability is particularly valuable for languages like Swahili, where training data alone are insufficient to achieve high-quality conversational performance. This paper focuses on leveraging RAG to develop a Swahili language conversation system, addressing the challenges of limited data and improving the quality of conversational AI for Swahili speakers. The study involves preparing a Swahili-specific dataset, implementing a RAG framework, and evaluating its performance using metrics such as BLEU, METEOR, Query Performance, and inference time. Four models—mT5, GPT-2, mBART, and GPT-Neo—are tested to compare the effectiveness of RAG against traditional fine-tuning methods. The results demonstrate that RAG significantly improves response quality by retrieving and utilizing external knowledge. Although RAG introduces longer response times, its benefits in accuracy and relevance make it an effective approach for building Swahili conversational systems [6,7].

This paper highlights the broader potential of RAG for other low-resource languages facing similar challenges. Many languages worldwide lack the digital resources required for developing advanced NLP systems. By showing how RAG can improve Swahili NLP, this research provides a roadmap for extending similar techniques to other underrepresented languages. This approach not only enhances the digital capabilities of Swahili but also promotes inclusive AI technologies that can benefit diverse linguistic communities. By combining modern AI techniques with existing language data, this study demonstrates how even languages with fewer resources can achieve significant advancements in NLP. Ultimately, this research aims to contribute to a future where all languages are represented in modern technology, ensuring equitable access to the benefits of conversational AI [8].

2. Related Work

2.1. Natural Language Processing in Low-Resource Languages

Natural Language Processing (NLP) for low-resource languages like Swahili presents significant challenges, primarily due to the lack of large, high-quality datasets. In contrast to languages such as English and Chinese, which benefit from vast digital resources and well-annotated corpora, Swahili remains underrepresented in NLP research [9]. This lack of resources makes it difficult to train models capable of understanding and generating accurate and contextually relevant responses in Swahili. Consequently, researchers have explored alternative approaches to maximize the utility of the limited Swahili data available. Early approaches often relied on traditional machine learning models like Naïve Bayes, decision trees, and support vector machines. These methods are computationally lightweight and work reasonably well with small datasets but struggle with the complexity and nuances of natural language [10]. For example, these models have been applied to tasks such as spam detection in Swahili SMS, where researchers adapted the algorithms to handle mixed-language inputs, as Swahili speakers often switch between Swahili and English in communication [11]. While effective for basic tasks, these models fail to capture deeper linguistic patterns such as contextual relationships and idiomatic expressions. Deep learning methods, such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, have also been applied to Swahili NLP. These models are better suited for understanding the relationships between words and their sequences. For instance, CNNs have been used to extract key features from Swahili text, while LSTMs have focused on capturing sequential dependencies [12]. However, their performance remains constrained by the small size of available Swahili datasets. Without sufficient data, even advanced models struggle to generalize effectively, limiting their applicability to real-world scenarios [13].

2.2. Retrieval-Augmented Generation (RAG) in NLP

Retrieval-Augmented Generation (RAG) is an innovative approach in NLP that combines the strengths of retrieval systems and large language models (LLMs). Unlike traditional language models that rely solely on pre-existing training data, RAG includes a retrieval component to dynamically fetch relevant information from external knowledge bases or databases. This additional step allows the model to provide responses that are more accurate, contextually relevant, and detailed [14]. RAG works by first identifying the most relevant information related to a user's query from an external source. This retrieved information is then incorporated into the language model, which uses it to generate a response. For example, if a user asks a question about Swahili culture, the retrieval system can find relevant content in a Swahili knowledge repository. The language model can then use this content to craft a well-informed response [15,16]. This approach is particularly useful for low-resource languages like Swahili, where training data alone may not cover all possible queries or topics. By leveraging external knowledge, RAG can overcome the limitations of small datasets and enhance the performance of conversational systems.

Compared to traditional fine-tuning methods, where the model's ability to answer queries is restricted by its training data, RAG offers dynamic adaptability. Fine-tuned models require retraining when new knowledge is introduced, making them less flexible for real-time or continuously evolving applications. RAG, on the other hand, can incorporate new information dynamically during retrieval, eliminating the need for frequent retraining [17]. Another alternative is knowledge-grounded conversational models, which rely on static knowledge graphs or curated databases. While these models excel in specific domains with well-defined knowledge bases, they are less effective in low-resource settings like Swahili due to the lack of comprehensive, structured knowledge repositories. RAG bridges this gap by utilizing unstructured data sources, making it more versatile in diverse domains [18]. However, RAG introduces additional computational complexity because of the retrieval step, which increases the response time compared to static models. Despite this, its ability to generate highly relevant and contextually accurate responses outweighs the computational trade-offs, especially for low-resource scenarios. For Swahili NLP, RAG has the potential to significantly improve conversational systems by dynamically retrieving missing contextual information. For instance, a RAG-based Swahili chatbot could fetch detailed explanations about historical events, cultural practices, or idiomatic expressions from a database, even if such topics were not covered in its training data. This method bridges the gap between resource availability and system performance, making it a powerful tool for building conversational systems in low-resource settings [19].

While fine-tuning and static knowledge-based approaches have their strengths, RAG emerges as an optimal choice due to its ability to integrate unstructured external data dynamically, provide adaptability, and address data limitations in low-resource languages like Swahili.

2.3. Swahili Natural Language Processing Efforts

Efforts to advance Swahili Natural Language Processing (NLP) have gained momentum in recent years as researchers recognize the need to support underrepresented languages [20]. Early work in this area focused on traditional machine learning methods for tasks like text classification and sentiment analysis. For example, algorithms like logistic regression and support vector machines were used to classify Swahili documents into categories or detect sentiment in social media posts. These models, while easy to implement, often required extensive manual feature engineering to achieve reasonable performance [21]. However, their limited ability to understand complex language patterns restricted their effectiveness. Recent advancements in multilingual models, such as multilingual BERT (mBERT) and SwahBERT, have provided new opportunities for Swahili NLP. These models are trained on large multilingual datasets, allowing them to process Swahili alongside other languages. Tasks like machine translation and context understanding have benefited from these models, especially when fine-tuned on Swahili-specific data [22]. However, challenges remain, particularly with regional dialects, idiomatic expressions, and the scarcity of parallel corpora [23]. RAG has further advanced these efforts by introducing a mechanism to integrate external knowledge into the modeling process. For Swahili, RAG-based systems can dynamically retrieve relevant text snippets from databases or knowledge repositories, supplementing the limited training data available. By doing so, these systems overcome the challenges posed by linguistic diversity and insufficient data, enabling models to handle complex conversational tasks with greater accuracy and fluency. This approach has proven effective in enhancing Swahili conversational AI, particularly for generating responses to queries that require detailed knowledge or contextual awareness [24]. These advancements highlight the importance of integrating modern techniques like RAG to overcome resource constraints in Swahili NLP. While traditional and multilingual models have laid the foundation for progress, RAG represents a significant step forward, offering a scalable and efficient way to improve Swahili conversational systems. This approach not only addresses the immediate challenges of data scarcity but also sets the stage for future innovations in NLP for other low-resource languages, contributing to a more inclusive digital landscape [25].

3. Proposed Method

This section describes the proposed method for leveraging Retrieval-Augmented Generation (RAG) to address challenges in Swahili Natural Language Processing (NLP).

The method integrates advanced retrieval techniques with generative models to deliver accurate and contextually appropriate responses to user queries [26]. The design focuses on handling diverse types of questions by organizing components into a cohesive and efficient architecture.

3.1. System Architecture

The system architecture, as illustrated in Figure 1, is designed to process user queries in Swahili and generate accurate, contextually relevant responses. At the center of the architecture is the User Chatbot Interface, which serves as the primary interaction point for users to input their questions and receive responses. Supporting this interaction is the Manager UI, which provides administrative oversight for the system's operations and knowledge base updates. The Admin Interface, through the Manager UI, ensures the quality, reliability, and relevance of the system's knowledge base. Administrators validate, update, and manage digital resources dynamically using tools like web scraping and API integration. Extracted documents are processed into smaller chunks for efficient embedding and storage in the vector database.

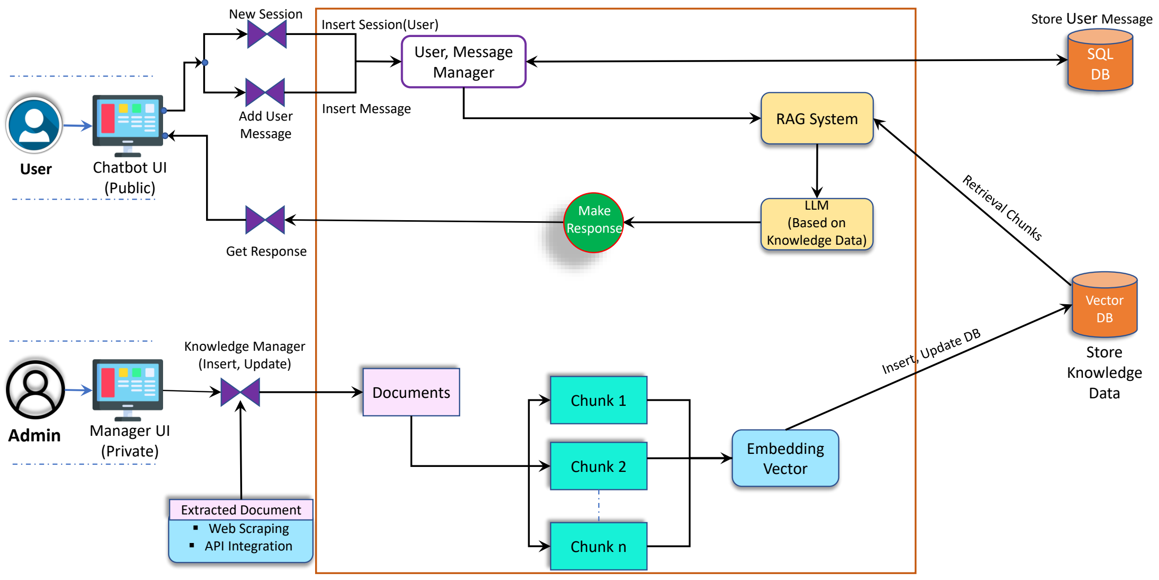


Figure 1. System architecture which illustrates the interaction between the User Chatbot Interface, Manager UI, Knowledge Manager, vector database, and the RAG system for seamless query handling and response generation.

The dataset used for this system represents Swahili's linguistic diversity by including major dialects such as Tanzanian, Kenyan, and Congolese Swahili. It spans multiple domains, including culture, healthcare, education, and everyday communication, ensuring adaptability across a variety of use cases. The dataset curation process prioritizes inclusivity, reflecting Swahili's linguistic richness and enhancing the system's applicability.

When a user submits a query, the system converts the query into vector embeddings and compares it against stored embeddings in the vector database. Relevant chunks are retrieved and passed, along with the user query, to the pre-trained large language model (LLM) for response generation. If no relevant information is found in the database, the system defaults to generating responses based on the LLM's internal knowledge, maintaining conversational continuity. The generated response is returned to the user through the User Chatbot Interface.

This architecture integrates user interaction, dynamic knowledge management, and advanced NLP techniques to provide a robust solution for Swahili conversational AI. By combining automated processes, manual validation, and comprehensive datasets, the system addresses the challenges of low-resource languages like Swahili while ensuring high standards of quality and relevance.

A key feature of the system is the automatic preprocessing workflow, which streamlines the preparation of raw Swahili data for integration into the knowledge base. This workflow, illustrated in Figure 2, is implemented using Python scripts tailored to the unique characteristics of the Swahili language. The automated process reduces manual workload while ensuring the data meet quality standards. The preprocessing begins with noise removal, where irrelevant characters, symbols, and HTML tags are filtered out to retain only meaningful content. A language detection algorithm then identifies and removes non-Swahili text, ensuring that the dataset remains focused on Swahili content. Abbreviations and shorthand terms are expanded into their full forms to improve clarity and readability. Text normalization ensures consistency in punctuation, casing, and spacing while harmonizing spelling variations. Grammar and spelling corrections are applied using predefined Swahili linguistic rules to enhance the text's linguistic accuracy. Finally, duplicate entries are detected and removed to avoid redundancy and optimize storage efficiency. These automated steps are critical for preparing high-quality data while minimizing manual intervention.

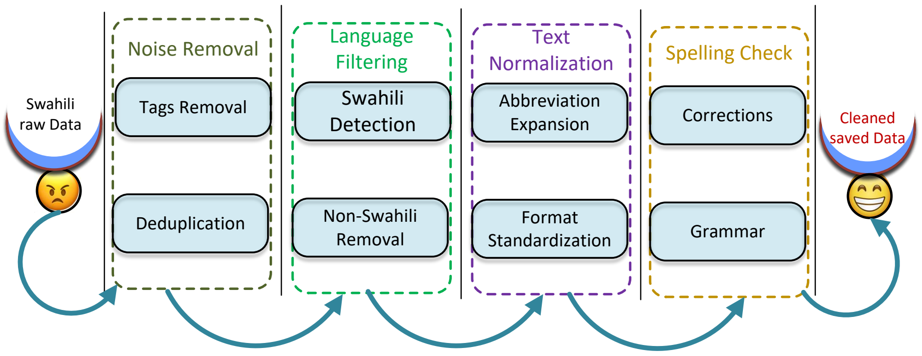


Figure 2. Validation and preprocessing workflow for Swahili data. This figure illustrates the steps for cleaning, normalizing, and validating data before their integration into the knowledge base.

Once preprocessing is complete, the cleaned data are passed to the Admin Interface for final validation. Administrators review the data for authenticity and relevance, verifying that it aligns with the system's objectives. Metadata, such as topic tags and source annotations, is added during this stage to improve retrieval accuracy and context relevance. This manual validation step complements the automated workflow, ensuring that only reliable and well-structured data are integrated into the knowledge base.

Following validation, the data are segmented into smaller chunks and converted into vector embeddings using a dedicated embedding service. These embeddings are stored in the vector database, enabling fast and efficient similarity-based retrieval. When a user submits a query, the system retrieves relevant chunks by comparing the query embedding with stored embeddings in the vector database [27,28]. The retrieved information is then combined with the internal knowledge of a pre-trained large language model (LLM) to generate accurate and contextually appropriate responses. To further improve reliability and address potential gaps, the system incorporates a feedback loop. User interactions are logged, and flagged responses are analyzed to identify inconsistencies or errors in the knowledge base. This feedback is processed through the Admin Interface, enabling administrators to make targeted updates and refinements. This iterative process ensures continuous improvement in response quality and system reliability.

The system architecture integrates the User Chatbot Interface, Admin Interface, automated preprocessing, and retrieval mechanisms with the LLM to provide a robust solution for Swahili conversational tasks. By combining automated processes, manual validation, and user feedback, the system addresses the challenges of low-resource languages like Swahili while maintaining high standards of quality and relevance [29].

3.2. RAG Mechanism

The Retrieval-Augmented Generation framework combines two major approaches: information retrieval and generative language modeling. These two components work together to address some of the challenges found in Natural Language Processing, especially for low-resource languages like Swahili [30]. The framework retrieves relevant information from a database and uses it to guide the generation of responses. This makes the system more accurate and contextually relevant compared to traditional models, which rely solely on what they learned during training. By retrieving external information, the framework addresses situations where training data alone are not enough to provide correct answers. Swahili, for example, has limited annotated datasets available. Generating responses without access to external information can lead to errors or irrelevant answers. The Retrieval-Augmented Generation framework solves this problem by grounding its responses in a large knowledge database, ensuring that the output is both coherent and supported by real-world data [31]. The system is composed of two main components, as shown in Figure 3. The first is the retriever, which is responsible for finding pieces of information, also called chunks, from the knowledge database. These chunks are selected based on their relevance to the user's query. The second component is the generator, which uses the user's query along with the retrieved chunks to create a fluent and meaningful response. By combining these two steps, the system generates responses that are more accurate and complete compared to models that generate text without access to external knowledge [32].

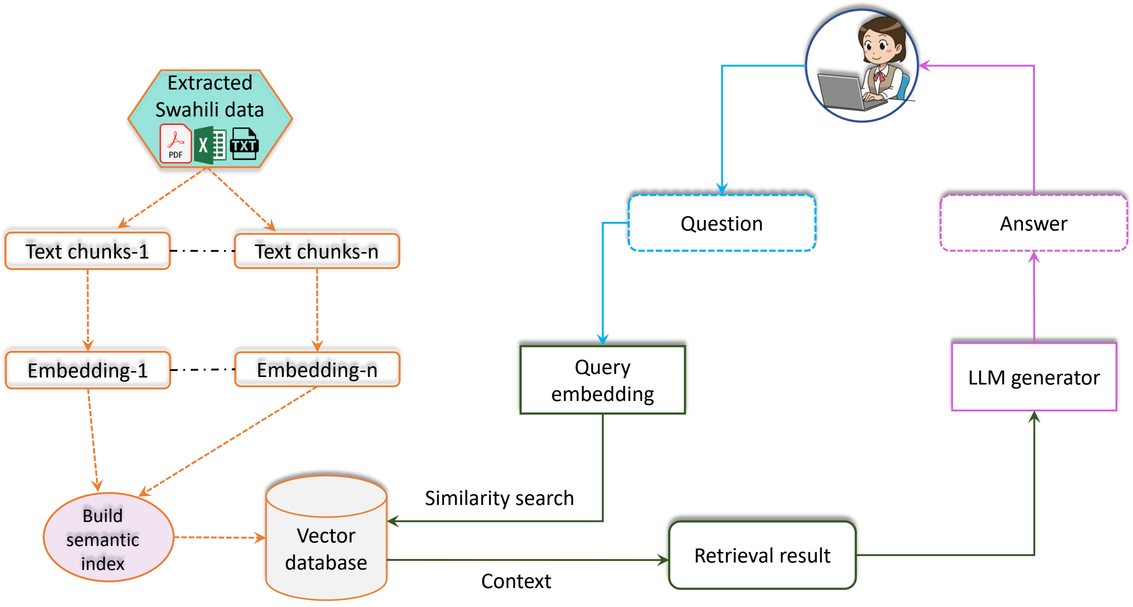


Figure 3. The Retrieval-Augmented Generation architecture for Swahili Natural Language Processing. It combines query embedding, retrieval, and contextual response generation.

3.2.1. Data Collection and Preparation

Figure 4 illustrates the workflow for data collection and preparation, which is essential for creating a reliable and organized knowledge database for Swahili Natural Language Processing (NLP) tasks. The database is generated through a systematic pipeline that combines automated tools, manual verification, and structured organization. The process begins by identifying dependable data sources, such as websites, FAQ sections, books, and media content. These sources provide diverse Swahili language data, ensuring comprehensive coverage of both general and specialized topics [33].

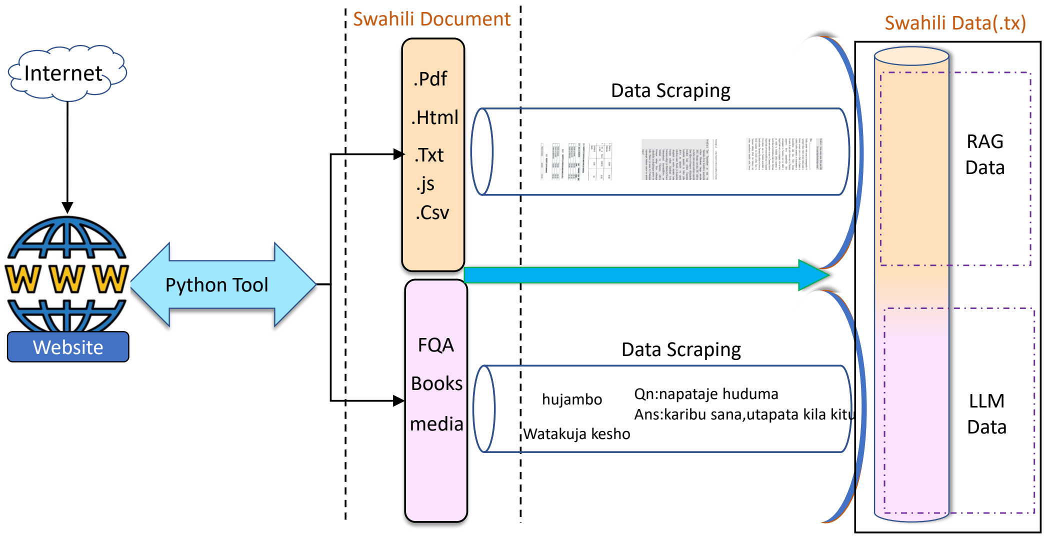


Figure 4. Workflow for data collection and preparation. This figure illustrates the pipeline for collecting, processing, and organizing Swahili data for retrieval and generative tasks.

Data extraction is automated using Python-based tools, enabling the system to efficiently collect data from multiple formats, including PDFs, HTML files, text files, JavaScript files, and CSV files. The extracted data are processed into two streams which are one optimized for retrieval tasks and the other for fine-tuning the generative model. For retrieval tasks, documents are divided into smaller chunks, which are later embedded into dense vector representations for similarity searches. For generative tasks, the data are refined to ensure they support the fine-tuning of the language model, enabling the generation of accurate and contextually relevant responses.

The organization of this database is overseen by administrators, who ensure the content remains relevant and up to date. This involves categorizing the data based on topics, domains, and linguistic variations within Swahili. Regular updates are performed using automated web scraping tools and API integrations to collect new content from trusted sources. Additionally, administrators validate the data to remove outdated or irrelevant information and to add annotations, such as metadata for improved retrieval accuracy. This structured curation ensures that the knowledge database evolves with time and remains suitable for Swahili NLP applications.

Preprocessing steps, as depicted in the workflow, ensure the quality and consistency of the dataset. Automated scripts perform tasks such as duplicate removal, normalization, and the application of Swahili-specific linguistic rules [34]. These steps are crucial for aligning the dataset with system requirements and improving its utility for both retrieval and generation tasks. For instance, normalization harmonizes spelling variations and standardizes text formats, while duplicate removal as shown in Figure 2 reduces redundancy to optimize database storage.

By integrating automation with linguistic considerations and structured processes, the workflow depicted in Figure 4 addresses the limitations of Swahili's limited annotated datasets. This pipeline enables the Retrieval-Augmented Generation framework to ground its responses in a large, dynamically updated, and well-organized knowledge database, ensuring that the system delivers accurate and coherent results supported by real-world data.

3.2.2. Embedding and Vector Representation

Figure 5 illustrates the embedding and vector representation process, which is a foundational component of the Retrieval-Augmented Generation (RAG) framework for Swahili conversational systems. This process transforms chunks of text into mathematical representations, called vectors, which the system uses to retrieve relevant information and generate accurate responses. The process begins with the input data, which could be a user query, a Swahili document, or any text-based information. The input is divided into smaller sections, known as chunks, to make processing more efficient. Each chunk typically contains 100–200 words, ensuring a balance between capturing meaningful information and maintaining computational efficiency. These chunks are then passed through an embedding model, such as Sentence-BERT (SBERT) or multilingual BERT (mBERT), which generates dense vector representations [35]. These vectors encapsulate the semantic meaning and contextual relationships within the text, enabling precise comparisons in the vector space.

Once generated, the vectors are indexed and stored in a vector database optimized for fast and efficient similarity searches. The system uses approximate nearest neighbor (ANN) algorithms, such as Hierarchical Navigable Small World graphs (HNSW), to accelerate the retrieval process while maintaining high accuracy. When a user submits a query, the query is also converted into a vector using the same embedding model. The system then compares this query vector to the stored vectors in the database using similarity measures, such as cosine similarity. This step identifies the most relevant chunks for the query [36].

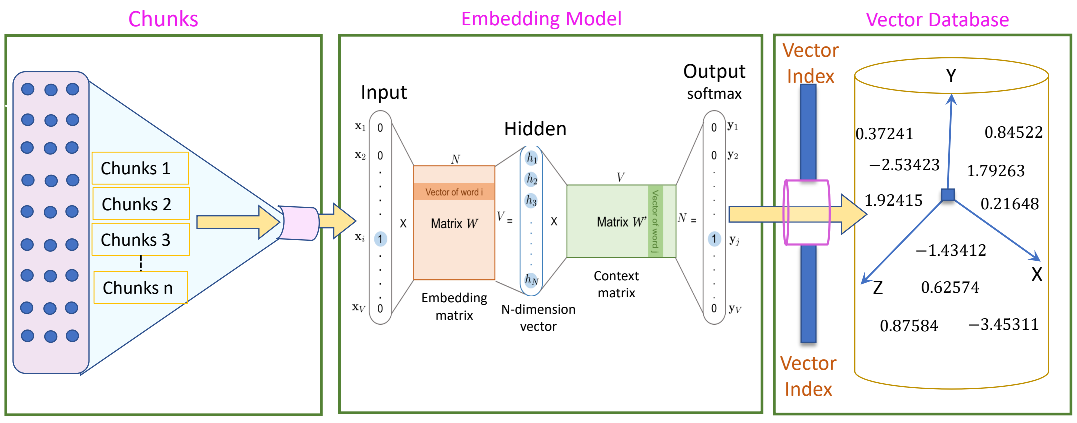


Figure 5. Embedding workflow and vector representation process for Swahili NLP. This figure highlights the key steps, from chunking input data to generating vectors, storing them in a database, and retrieving relevant information for response generation.

The retrieved chunks are then passed to the response generation model. The model synthesizes the retrieved information with its internal knowledge to generate responses that are accurate, contextually appropriate, and linguistically coherent. For example, if a user inquires about Swahili culture, the system retrieves relevant chunks from the database and incorporates them into a comprehensive answer.

This embedding and vector representation process ensures that the RAG framework can dynamically retrieve external knowledge to handle complex user queries [37]. It is especially valuable for Swahili, a low-resource language, as it mitigates the limitations of limited training data by leveraging external knowledge sources.

The vector representation of each text chunk in the embedding process, as shown in Figure 5, is mathematically defined as

$$\mathbf{v}_i = f_{\text{embed}}(T_i) \tag{1}$$

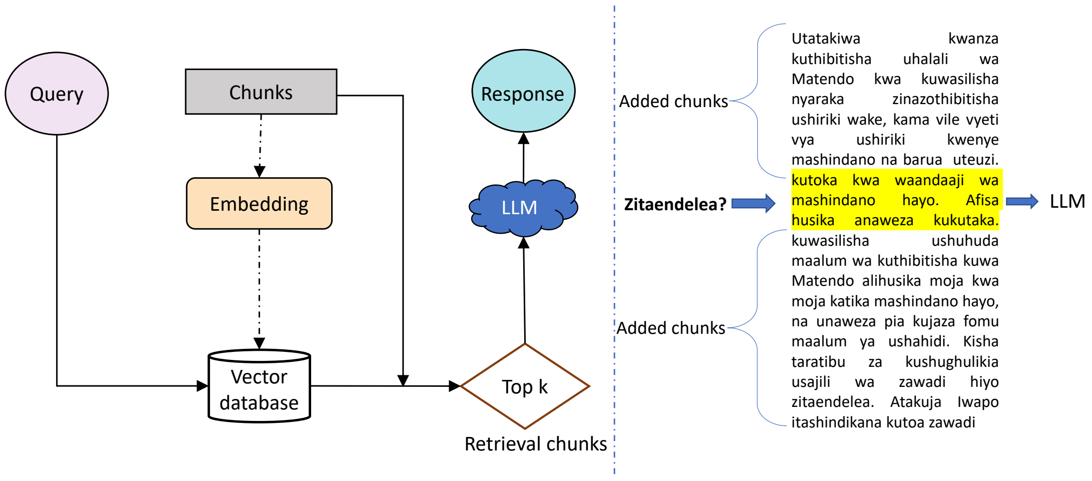
where

- $\mathbf{v}_i$ is the vector representation of the i-th chunk, capturing its semantic meaning and contextual information.
- T_i is the text content of the i-th chunk, extracted from the original input document or query.
- f_{embed} is the embedding function provided by the selected embedding model, such as Sentence-BERT (SBERT) or multilingual BERT (mBERT). This function converts the text chunk into a high-dimensional vector space.

The embedding process ensures that semantically similar chunks are positioned close to each other in the vector space. By indexing the vectors in a specialized database, the system achieves computationally efficient retrieval. This design supports fast similarity searches and enhances response generation accuracy.

3.2.3. Retriever and Query Processing

The retriever component is a critical element of the Retrieval-Augmented Generation (RAG) framework, bridging user queries with contextually relevant responses. Figure 6 illustrates this process, which combines embedding models, vector databases, and retrieval mechanisms to efficiently locate and return the most relevant information.



Utatakiwa kwanza kuthibitisha uhalali wa Matendo kwa kuwasilisha nyaraka zinazothibitisha ushiriki wake, kama vile vyeti vya ushiriki kwenye mashindano na barua uteuzi. kutoka kwa waandaaji wa mashindano hayo. Afisa husika anaweza kukutaka. kuwasilisha ushuhuda maalum wa kuthibitisha kuwa Matendo alihusika moja kwa moja katika mashindano hayo, na unaweza pia kujaza fomu maalum ya ushahidi. Kisha taratibu za kushughulikia usajili wa zawadi hiyo zitaendelea. Atakuja Iwapo itashindikana kutoa zawadi

Figure 6. Retriever and query processing workflow in the RAG framework for Swahili NLP.

The retrieval process begins when a user submits a query Q. This query is transformed into a dense vector representation $\mathbf{q}$ using an embedding model, such as Sentence-BERT (SBERT) or multilingual BERT (mBERT). The embedding captures the semantic meaning and contextual nuances of the query, enabling precise comparisons with precomputed vectors stored in the vector database. The query embedding process is expressed mathematically as

$$\mathbf{q} = f_{\text{embed}}(Q) \tag{2}$$

where

- $\mathbf{q}$ is the dense vector representation of the query.
- Q is the user's query in textual form.
- f_{embed} is the embedding function from the pre-trained model.

The vector database consists of vectors $\mathbf{c}_i$, each representing a chunk of preprocessed text T_i. These chunks are created by segmenting larger documents into coherent units and embedding them using the same model. This ensures that both the query and the database entries share a common semantic space. The vector representation for each chunk is

$$\mathbf{c}_i = f_{\text{embed}}(T_i), \quad i = 1, 2, \ldots, n \tag{3}$$

where

- T_i is the i-th chunk of text.
- $\mathbf{c}_i$ is the vector representation of the i-th chunk.
- n is the total number of chunks in the database.

The similarity between the query vector $\mathbf{q}$ and each chunk vector $\mathbf{c}_i$ is calculated using cosine similarity, defined as

$$\text{Sim}_{\text{cos}}(\mathbf{q}, \mathbf{c}_i) = \frac{\mathbf{q} \cdot \mathbf{c}_i}{\|\mathbf{q}\| \|\mathbf{c}_i\|} \tag{4}$$

Here,

- $\mathbf{q} \cdot \mathbf{c}_i$ is the dot product of the query and chunk vectors.
- $\|\mathbf{q}\|$, $\|\mathbf{c}_i\|$ are the L2 norms (magnitudes) of the vectors.

The selection of relevant information is based on the similarity scores calculated between $\mathbf{q}$ and $\mathbf{c}_i$. Chunks with the highest similarity scores are considered the most contextually relevant. The system retrieves the top-k chunks, where k is a parameter defining the number of chunks to use for answer generation. This approach ensures that the retrieved information aligns semantically with the user query while covering diverse contexts for robust responses. To provide additional clarity, Algorithm 1 outlines the detailed steps involved in the retrieval mechanism. This algorithm highlights the key processes, from query embedding to chunk selection, ensuring transparency in how the retriever identifies relevant information.

Algorithm 1: Retrieval mechanism for query processing

```
1 Input: Query $Q$, Vector Database $\mathcal{C}$, Embedding Function $f_{\text{embed}}$, Number of
   Chunks $k$
2 Output: Top-$k$ Relevant Chunks $\mathcal{C}_{\text{top-k}}$
3 Generate query embedding: $\mathbf{q} \leftarrow f_{\text{embed}}(Q)$
4 foreach $\mathbf{c}_i \in \mathcal{C}$ do
5     Compute similarity score: $\text{Sim}_{\text{cos}}(\mathbf{q}, \mathbf{c}_i)$
6 end
7 Rank all chunks $\mathbf{c}_i$ by similarity scores in descending order
8 Select top-$k$ chunks: $\mathcal{C}_{\text{top-k}}$
9 return $\mathcal{C}_{\text{top-k}}$
```

The retrieved chunks are passed to a pre-trained large language model (LLM), which synthesizes the information into a coherent response. This ensures that the response is both contextually accurate and comprehensive. To refine retrieval further, a hybrid scoring mechanism combining cosine similarity with BM25, a ranking function for textual relevance, is employed. The final score is computed as

$$\text{Score}_{\text{final}} = \alpha \cdot \text{Sim}_{\text{cos}} + (1 - \alpha) \cdot \text{Score}_{\text{BM25}} \tag{5}$$

Here, α is a weight parameter balancing semantic and lexical relevance. This hybrid scoring method ensures that the system retrieves information not only aligned with the meaning of the query but also significant from a textual relevance perspective. By leveraging these criteria, the retriever ensures that the system provides accurate and contextually meaningful responses, as depicted in Figure 6. Table 1 provides a clear translation of a Swahili question and its corresponding paragraph into English to aid readers in understanding the data used in the retriever model. This table illustrates how Swahili queries are formulated and how associated paragraphs provide the contextual information necessary for retrieval. Specifically, the Swahili question, "*Je, ni hatua gani zinazofuata baada ya kuthibitisha uhalali wa Matendo*?" translates to "*What are the next steps after verifying the legitimacy of Matendo*?". The associated paragraph provides detailed procedural instructions, demonstrating the kind of data processed in the retriever workflow.

Table 1. Translation of Swahili question and associated paragraph for Figure 6.

Swahili Text	*English Translation*
Question Je, ni hatua gani zinazofuata baada ya kuthibitisha uhalali wa Matendo	*What are the next steps after verifying the legitimacy of Matendo*
Paragraph Utatakiwa kwanza kuthibitisha uhalali wa Matendo kwa kuwasilisha nyaraka zinazothibitisha ushiriki wake, kama vile vyeti vya ushiriki kwenye mashindano na barua uteuzi kutoka kwa waandaaji wa mashindano hayo. Afisa husika anaweza kukutaka kuwasilisha ushuhuda maalum wa kuthibitisha kuwa Matendo alihusika moja kwa moja katika mashindano hayo, na unaweza pia kujaza fomu maalum ya ushahidi. Kisha taratibu za kushughulikia usajili wa zawadi hiyo zitaendelea	*You will first need to verify the legitimacy of Matendo by submitting documents proving their participation, such as certificates of participation in competitions and nomination letters from the organizers of those competitions. The relevant officer may also request you to provide specific evidence showing that Matendo was directly involved in those competitions, and you may also need to fill out a special evidence form. Then, the procedures for handling the registration of the reward will proceed*

This table translates the Swahili question and its associated paragraph into English.

The response generation step utilizes the retrieved chunks $\mathcal{C}_{\text{top-k}}$ and the original query Q to create a coherent response. The generator, typically a pre-trained transformer model such as mT5 or GPT-2, processes the query vector $\mathbf{q}$ and the retrieved chunks to produce the final output. This process is mathematically expressed as

$$\hat{y} = g(\mathbf{q}, \mathcal{C}_{\text{top-k}}) \tag{6}$$

where

- $\hat{y}$ is the generated response.
- g is the response generation function implemented by the generative model.

The retrieval and generation processes work in tandem to ensure that the output is both accurate and contextually grounded. By leveraging external knowledge dynamically, the RAG framework overcomes the limitations of Swahili's low-resource setting and generates meaningful responses for complex queries.

3.2.4. Query Processing and Answer Generation Workflow

Figure 7 depicts the workflow for processing user queries and generating accurate answers using the Retrieval-Augmented Generation (RAG) framework. The process begins when a user submits a query through the Chatbot Interface. The query is first cleaned and tokenized to remove noise and prepare it for embedding generation. This preprocessing ensures the text is in a format suitable for downstream tasks. Next, the query is converted into a dense vector representation using a pre-trained embedding model fine-tuned on Swahili-specific data [38]. This embedding captures the semantic meaning of the query, enabling accurate similarity comparisons with passages stored in the RAG vector database.

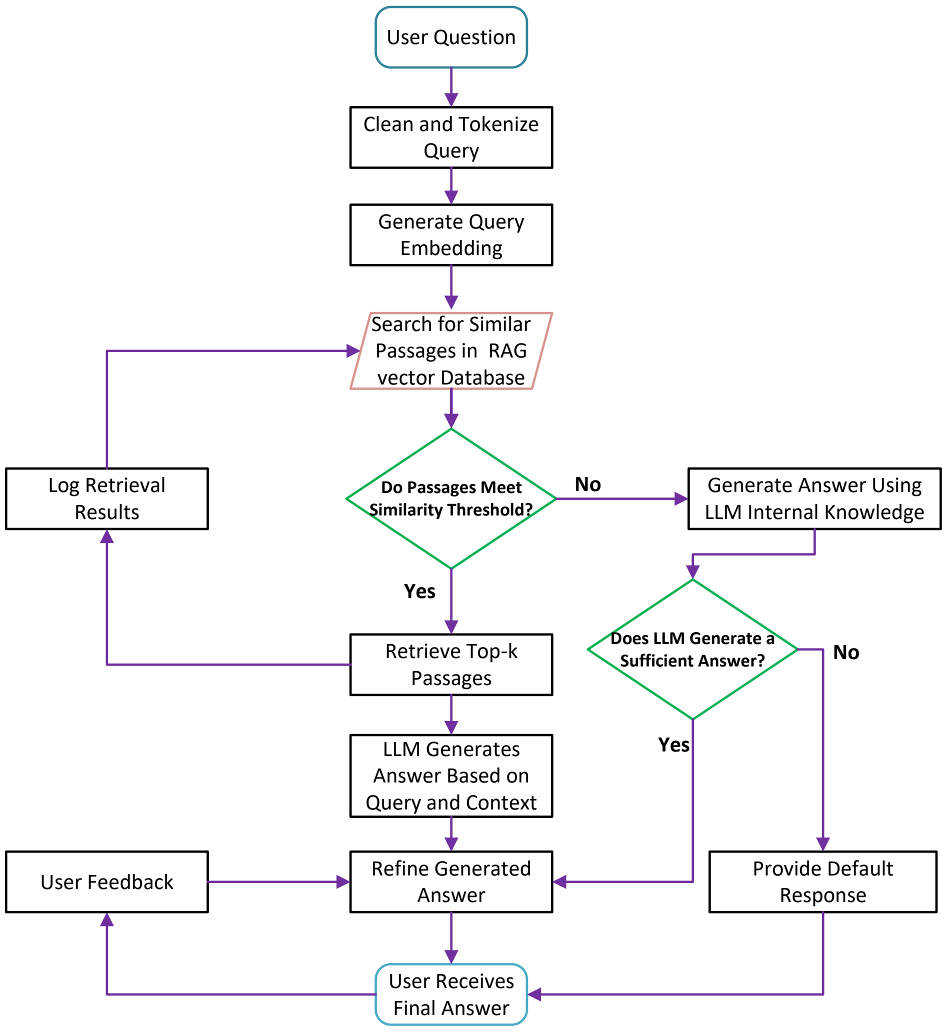

User Question
Clean and Tokenize Query
Generate Query Embedding
Search for Similar Passages in RAG vector Database
Do Passages Meet Similarity Threshold?
No
Generate Answer Using LLM Internal Knowledge
Log Retrieval Results
Yes
Retrieve Top-k Passages
Does LLM Generate a Sufficient Answer?
No
Yes
LLM Generates Answer Based on Query and Context
User Feedback
Refine Generated Answer
Provide Default Response
User Receives Final Answer

Figure 7. Query processing and answer generation workflow. This updated figure illustrates the systematic steps for handling user queries, retrieving relevant passages, and generating coherent answers in the RAG system.

Using similarity metrics such as cosine similarity, the query embedding is compared to the embeddings of passages in the database. If the retrieved passages, including those searched from the Log Retrieval Results module, meet the predefined similarity threshold, the system selects the top-k passages and forwards them, along with the user query, to the pre-trained large language model (LLM) for further processing. However, if no relevant passages are found even after searching the logs, the system skips retrieval and directly relies on the LLM's internal knowledge to generate a response. This ensures a seamless user experience while maintaining the robustness of the workflow.

The LLM generates an answer by combining the retrieved contextual information (if available) with its internal knowledge. If the generated answer meets quality standards ensuring it is relevant, coherent, and grammatically accurate it is refined further and delivered to the user. However, if the LLM fails to generate a sufficient answer, the workflow provides a default response to ensure user engagement is maintained. The system also incorporates a feedback mechanism, allowing users to provide input on the response quality. This feedback is logged and used to iteratively improve both retrieval and generative components of the system, enhancing its overall accuracy and reliability over time.

3.2.5. System Interaction Workflow

Figure 8 illustrates the interaction workflow between the user, the Retrieval-Augmented Generation (RAG) system, the vector database, and the large language model (LLM). This workflow highlights the sequence of operations that enable the system to process user queries and deliver accurate responses.

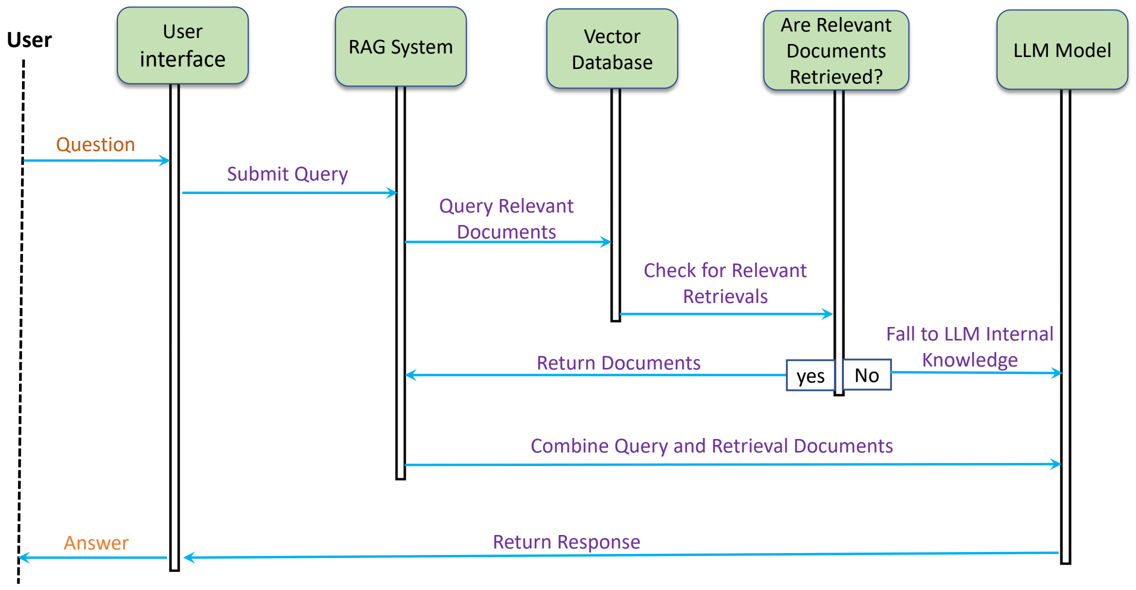


Figure 8. System interaction workflow. This figure illustrates the sequence of operations, from query submission to response generation, with fallback mechanisms in case relevant documents are not found in the vector database.

The process begins with the user submitting a query through the interface. The query is then sent to the RAG system, which acts as the central processing unit [39]. Within the RAG system, the query is processed to extract its semantic meaning and match it with relevant information stored in the vector database. The vector database, which holds embedding representations of document chunks, is queried to identify the most relevant documents based on the similarity between the query vector and stored embeddings.

If relevant documents are found, they are retrieved and combined with the original user query to create a unified context. However, if no relevant documents are identified, the system falls back to utilizing the LLM's internal knowledge, ensuring that the user query is addressed even in the absence of external data. This context, either enriched with retrieved documents or relying solely on internal knowledge, is then passed to the pre-trained LLM, which uses the provided information to generate a coherent and contextually relevant answer [40].

The LLM-generated answer is returned to the RAG system, where it is prepared for delivery. The final response is sent back to the user interface, completing the interaction loop. This structured workflow, including fallback mechanisms, ensures robust handling of user queries and highlights the seamless integration of retrieval and generation components. Figure 8 clearly demonstrates the collaboration between the various components, emphasizing the seamless flow of information that enables the system to meet user needs in Swahili Natural Language Processing applications [41].

4. Experimentation

4.1. Dataset

The dataset used to evaluate Retrieval-Augmented Generation (RAG) for Swahili Natural Language Processing (NLP) tasks was carefully curated to address the unique challenges of low-resource languages like Swahili [42]. It includes a diverse range of components designed to test RAG's ability to retrieve and generate accurate responses for conversational systems. The dataset composition is detailed in Table 2.

Table 2. Dataset composition for Retrieval-Augmented Generation (RAG) in Swahili NLP.

Dataset Component	Train Entries	Validation Entries	Test Entries
Swahili Queries (Fine-Tuning)	4500	1000	1500
Paraphrased Questions (Fine-Tuning)	1500	500	500
Noisy Text (Fine-Tuning)	500	250	250
External Context Chunks (RAG)	3000	–	–
Total Entries	7000	1750	2250

The dataset comprises Swahili queries, paraphrased questions, multi-turn conversations, noisy text, and a set of external context chunks. Swahili queries include straightforward user-posed questions designed to evaluate the retrieval and generation processes. Paraphrased questions test semantic understanding and the system's ability to generalize across similar queries. Multi-turn conversations assess the ability to maintain context across multiple exchanges, while noisy text introduces real-world challenges like typos and informal syntax. The external context chunks serve as the knowledge base for retrieval, containing 5000 preprocessed chunks derived from Swahili documents. These chunks provide the external knowledge needed to generate contextually accurate responses. Unlike other components, the external chunks are not split into training, validation, or test sets since they are dynamically retrieved during the response generation process [43]. To ensure consistency and usability, the dataset was preprocessed by removing non-informative symbols, correcting orthographic errors, and segmenting longer text entries into smaller chunks suitable for retrieval. This preprocessing step enhances retrieval accuracy and ensures that the generator produces coherent and relevant outputs. This dataset highlights the distinction between fine-tuning and RAG. Fine-tuning focuses solely on direct mappings between Swahili queries and responses, while RAG leverages external knowledge to enrich the generator's ability to provide accurate and context-aware answers [44].

4.2. Experiment Settings

The experiments were conducted to evaluate the integration of retrieval and generation within the Retrieval-Augmented Generation framework for Swahili Natural Language Processing. Four pre-trained models—mT5, GPT-2, mBART, and GPT-Neo—were tested to generate accurate, fluent, and contextually grounded responses to Swahili queries. These models were carefully selected for their strong performance in multilingual and low-resource settings, as they provide diverse architectural advantages for tasks involving Swahili NLP. The pre-trained models, listed in Table 3, were fine-tuned on the curated dataset with an 80-10-10 split for training, validation, and testing. These splits ensured sufficient examples for training while maintaining independent datasets for unbiased evaluation. Additionally, preprocessing steps specific to Swahili were employed to harmonize linguistic variations and enhance data quality, thereby improving the models' effectiveness.

Hyperparameter tuning was performed systematically to achieve an optimal balance between computational efficiency and model performance. A batch size of 16 was used to optimize memory usage while ensuring stable gradient updates. The learning rate was set at 2×10^{-5}, a value derived from empirical trials to stabilize convergence and minimize overfitting. Training was conducted over five epochs, which provided sufficient coverage of the dataset without introducing redundancy.The experiments were conducted on NVIDIA Tesla V100 GPUs with 32 GB of memory (Nvidia, Santa Clara, CA, USA), as shown in Table 3. This hardware facilitated efficient training of large models such as mT5 and GPT-Neo, which require significant computational resources. Fast SSD storage with a capacity of 2 TB was employed to ensure rapid access to datasets and embeddings during both training and retrieval operations.

FAISS was employed as the retrieval backend for vector similarity search, using cosine similarity to measure the relevance of retrieved text chunks. FAISS was chosen for its scalability and efficiency in managing high-dimensional vector data, making it an optimal choice for the Swahili knowledge database. The cosine similarity metric was applied to ensure semantic relevance between query embeddings and document chunks. The experiments evaluated three critical challenges in Swahili NLP: handling rephrased questions, noisy text, and multi-turn dialogues. Rephrased questions tested the models' ability to generalize across variations in query phrasing. Noisy text introduced typographical errors and informal syntax to simulate real-world scenarios. Multi-turn dialogues assessed the models' capacity to maintain context over multiple exchanges. This experimental design ensured a comprehensive evaluation of the models' capabilities and highlighted the effectiveness of the Retrieval-Augmented Generation framework in addressing the unique challenges of low-resource languages like Swahili [45–47].

Table 3. Experiment settings for Retrieval-Augmented Generation in Swahili NLP.

Experiment Component	Parameter	Value	Description	Justification
Model Architecture	Pre-trained Models	mT5, GPT-2, mBART, GPT-Neo	Transformer-based models fine-tuned for Swahili tasks	Suitable for multilingual and low-resource settings
Data Splits	Training Data	80%	Dataset for training fine-tuning and RAG models	Provides sufficient examples for model learning
	Validation Data	10%	Dataset for hyperparameter tuning and evaluation	Reduces risk of overfitting
	Test Data	10%	Dataset for independent model evaluation	Ensures fair performance assessment
Hardware Configuration	GPU	NVIDIA Tesla V100 (32 GB)	High-performance GPU for training and retrieval tasks	Capable of handling large-scale computations
	Storage	SSD (2 TB)	Fast storage for datasets and embeddings	Enhances data access speed during experiments
Training Parameters	Batch Size	16	Number of examples per training step	Balances memory usage and model updates
	Learning Rate	2×10^{-5}	Step size for updating model weights	Ensures stable training convergence
	Epochs	5	Complete passes through the training data	Sufficient for dataset size

Table 3. *Cont.*

Experiment Component	Parameter	Value	Description	Justification
Retriever Mechanism	Backend	FAISS	Vector similarity search for retrieval	Enables efficient retrieval from large datasets
	Similarity Metric	Cosine Similarity	Metric for measuring semantic relevance	Suitable for high-dimensional embeddings
Optimization	Algorithm	Adam Optimizer	Adaptive learning rate algorithm	Stabilizes gradient updates
	Loss Function	Cross-Entropy Loss	Measures error between predictions and ground-truth responses	Effective for supervised learning tasks
Frameworks	Libraries	TensorFlow 2.x, Hugging Face Transformers	Modern libraries for model training and evaluation	Provide robust tools for implementing RAG

4.3. Experimental Results

This section presents the experimental results comparing the performance of fine-tuned models and RAG-enhanced models across multiple key metrics, including BLEU score, METEOR score, Query Performance, Multi-Turn Coherence, inference time, ROUGE-L score, Swahili Dialect Testing, and perplexity. The results highlight the improvements achieved by integrating retrieval mechanisms with generation techniques in low-resource Swahili NLP tasks. Table 4 provides a detailed comparison [48].

Table 4. Comparison of metrics between fine-tuned and RAG models for Swahili NLP.

Metric	mT5		GPT-2		mBART		GPT-Neo	
	Fine-Tune	**RAG**	**Fine-Tune**	**RAG**	**Fine-Tune**	**RAG**	**Fine-Tune**	**RAG**
BLEU Score (%)	45.34	56.88	44.12	50.43	47.24	52.36	43.76	49.87
METEOR Score (%)	62.12	72.72	59.48	66.13	63.87	68.54	60.32	65.98
ROUGE-L Score (%)	66.21	81.54	70.67	76.43	51.42	63.89	53.34	61.28
Query Performance (%)	68.12	84.34	65.62	76.51	70.14	78.93	67.15	75.98
Multi-Turn Coherence (%)	75.43	88.25	72.38	81.23	78.12	82.46	73.24	80.98
Swahili Dialect Testing (%)	64.31	78.24	61.54	71.42	66.12	74.38	62.87	70.45
Perplexity (%)	24.32	18.12	26.45	20.34	22.87	19.54	25.12	21.45
Inference Time (s)	0.93	1.05	0.85	0.97	1.01	1.12	0.78	0.89

To assess the statistical significance of the observed improvements, paired t-tests were conducted to compare the performance metrics of fine-tuned models and RAG-enhanced models. The results confirm that the improvements in BLEU, METEOR, ROUGE-L, Query Performance, and Multi-Turn Coherence scores are statistically significant with $p < 0.01$. This statistical validation underscores the robustness of the enhancements achieved by integrating retrieval mechanisms. The results in Table 4 demonstrate that the RAG-enhanced models consistently outperform their fine-tuned counterparts across most metrics. For example, mT5 with RAG achieved a BLEU score of 56.88% and a METEOR score of 72.72%, reflecting superior translation accuracy and fluency. Similarly, the Query Performance metric shows significant improvements, with RAG-enhanced models achieving up to 84.34%, compared to 68.12% for the fine-tuned models. The ROUGE-L score and Swahili Dialect Testing results further highlight the advantages of integrating retrieval, as these metrics measure contextual alignment and handling of linguistic variations, respectively [49]. The perplexity values, which indicate the confidence of the model in generating responses, are notably lower for RAG-enhanced models, showcasing improved fluency and reduced uncertainty in responses.

The choice of metrics was driven by their relevance to evaluating text generation quality, contextual relevance, and linguistic diversity in Swahili NLP. BLEU, METEOR, and ROUGE-L were selected for their ability to measure lexical overlap, semantic alignment, and contextual coherence, respectively. BLEU evaluates precision by calculating n-gram overlap between generated and reference texts, making it useful for assessing fluency. METEOR incorporates synonym matching and paraphrasing, capturing linguistic variations beyond exact matches. ROUGE-L focuses on sequence alignment, measuring the overlap of the longest common subsequences between the generated and reference texts. These three metrics collectively address the generative quality of the models. Additionally, Query Performance and Multi-Turn Coherence metrics were introduced to evaluate conversational relevance and the model's ability to maintain context over multiple exchanges. Swahili Dialect Testing specifically measures the adaptability of the model across various dialects, ensuring linguistic inclusivity. Perplexity was included to quantify the model's uncertainty in generating responses, with lower values indicating higher confidence and fluency. Metrics like F1 score and accuracy were not used because they are primarily suited for classification or prediction tasks, where exact matches are critical. Similarly, semantic similarity measures, while relevant for retrieval tasks, do not directly evaluate the quality of generative outputs. The selected metrics collectively provide a comprehensive evaluation framework tailored to the needs of conversational Swahili NLP.

Inference time, reported in seconds, shows a slight increase for RAG-enhanced models due to the additional retrieval step. However, this trade-off is justified by the substantial gains in accuracy and relevance, as seen in the BLEU and Multi-Turn Coherence scores. To address this, future optimization strategies include caching frequently accessed query embeddings, reducing retrieval latency through approximate nearest neighbor search, and leveraging more efficient embedding models to balance accuracy and speed. The inclusion of statistical tests adds further rigor to the analysis, validating that these improvements are not coincidental but rather a consistent outcome of the RAG framework. These results underscore the effectiveness of the RAG framework in addressing the challenges of low-resource languages like Swahili and highlight its potential for other underrepresented languages.

The results in Table 5 highlight the performance differences among retrieval models when integrated into the RAG framework for Swahili NLP. Each dimension of the table provides insights into the strengths and weaknesses of the retrieval mechanisms. Semantic matching shows that FAISS achieved the highest BLEU score of 56.88% and ROUGE-L score of 71.54%, indicating its superior ability to understand and generate semantically aligned responses. BM25 performed moderately with BLEU and ROUGE-L scores of 53.72% and 69.34%, while TF-IDF achieved lower scores of 50.43% and 67.23%. For Swahili-specific handling, FAISS excelled with Query Performance at 84.34% and Dialect Testing at 78.24%, demonstrating its strength in addressing Swahili linguistic nuances. BM25 achieved 80.12% for Query Performance and 74.89% for Dialect Testing, while TF-IDF scored lower at 77.34% and 72.12%. Multi-Turn Coherence was highest with FAISS at 88.25%, reflecting its ability to maintain conversational context effectively. BM25 followed with 84.23%, and TF-IDF scored 82.12%. In terms of computational efficiency, FAISS had the lowest perplexity score of 18.12%, indicating confidence in generating coherent responses. However, its inference time was slightly higher at 1.05 s compared to BM25 at 0.97 s and TF-IDF at 0.92 s. These results demonstrate the trade-offs between semantic richness and computational speed. This comparison underlines the advantages of FAISS in delivering accurate, contextually relevant, and semantically rich responses, making it the most effective retrieval mechanism for Swahili NLP in the RAG framework.

Table 5. Performance of retrieval models in RAG for Swahili NLP.

Dimension	FAISS (%)	BM25 (%)	TF-IDF (%)
Semantic Matching (BLEU, ROUGE-L)	BLEU 56.88, ROUGE-L 71.54	BLEU 53.72, ROUGE-L 69.34	BLEU 50.43, ROUGE-L 67.23
Swahili-Specific Handling (Query Performance, Dialect Testing)	Query Performance 84.34, Dialect Testing 78.24	Query Performance 80.12, Dialect Testing 74.89	Query Performance 77.34, Dialect Testing 72.12
Multi-Turn Coherence	88.25	84.23	82.12
Perplexity (Lower is Better)	18.12	19.78	20.34
Inference Time	1.05	0.97	0.92

Table 6 highlights the differences in performance across retrieval mechanisms and models. FAISS and Hybrid Retrieval consistently achieved the highest retrieval accuracy and relevance scores across all the models. For example, mT5 with Hybrid Retrieval achieved a retrieval accuracy of 92.15 percent, outperforming the same model with BM25 at 88.50 percent and TF-IDF at 84.45 percent. Response fluency scores followed a similar trend, with Hybrid Retrieval producing the most natural responses for all the models. For instance, GPT-Neo with Hybrid Retrieval achieved a response fluency score of 88.34 percent, which was higher than its performance with FAISS at 86.23 percent and TF-IDF at 80.34 percent. Relevance scores indicate how well the generated responses align with the query's context. Hybrid Retrieval led in relevance, as demonstrated by mBART paired with Hybrid Retrieval achieving a relevance score of 89.34 percent, surpassing its performance with BM25 at 84.78 percent and TF-IDF at 81.45 percent. In terms of efficiency, TF-IDF demonstrated the fastest response times due to its computational simplicity, achieving a low inference time of 0.92 s with GPT-2. However, this speed comes at the cost of lower accuracy and relevance, highlighting the trade-offs between computational efficiency and model performance. The results show that Hybrid Retrieval is the most effective mechanism for Swahili NLP tasks, providing the highest scores in most metrics, but FAISS offers a strong alternative for cases where efficiency is prioritized. The trends also reveal that mT5 consistently performs better across all retrieval mechanisms compared to the other models, making it a strong candidate for Swahili conversational systems. These findings provide valuable insights for selecting the best retrieval mechanism and model combination based on specific application needs and resource constraints.

The results in Table 7 compare the diversity, redundancy, and query overlap metrics for various retrieval models in Swahili NLP tasks. Hybrid Retrieval achieved the highest percentage of unique chunks retrieved, at 83.56 percent, reflecting its ability to provide more diverse and contextually rich responses compared to other models. In contrast, TF-IDF had the lowest diversity, retrieving only 68.45 percent unique chunks, which indicates a greater reliance on repetitive or less varied content. This limitation may hinder its effectiveness in addressing complex or nuanced queries. Redundant chunks, which represent repeated or semantically identical content, were minimized in Hybrid Retrieval at 8.67 percent, showcasing its superior capability in filtering out irrelevant repetitions. Conversely, TF-IDF exhibited the highest redundancy rate of 22.34 percent, followed by BM25 at 18.89 percent. These results highlight the importance of advanced retrieval mechanisms in improving content diversity and reducing redundancy for more effective responses. Query overlap measures the alignment between retrieved chunks and the semantic content of the input query. Hybrid Retrieval led with an overlap of 89.45 percent, indicating that it retrieved the most relevant and contextually appropriate information [50]. FAISS followed closely with 85.67 percent overlap, while BM25 and TF-IDF trailed behind, showing their relatively lower alignment with the query intent. In terms of retrieval speed, TF-IDF was the fastest with an average retrieval time of 92 milliseconds, making it a suitable choice for scenarios where speed is critical. However, this efficiency comes at the cost of lower diversity and higher redundancy. On the other hand, Hybrid Retrieval took the longest retrieval time at 120 milliseconds due to its more comprehensive search and filtering mechanisms. Despite this, its superior performance across diversity, redundancy, and query overlap metrics made it the most effective retrieval mechanism overall. These findings demonstrate the trade-offs between speed and retrieval quality across different models. While simpler models like TF-IDF are faster, more sophisticated approaches such as Hybrid Retrieval offer significant advantages in terms of content diversity and relevance, particularly for low-resource languages like Swahili. These insights emphasize the need to balance retrieval accuracy and efficiency based on application requirements.

Table 6. Comparison of retrieval mechanisms across models for Swahili NLP.

Model + Retrieval	Retrieval Accuracy (%)	Response Fluency (%)	Relevance (%)	Efficiency (s)
mT5 + FAISS	88.00	87.34	86.45	1.05
mT5 + BM25	85.67	84.12	83.45	1.10
mT5 + TF-IDF	82.45	80.34	81.78	0.95
mT5 + Hybrid Retrieval	88.00	88.00	88.00	1.20
GPT-2 + FAISS	77.45	75.34	76.12	1.10
GPT-2 + BM25	73.56	71.34	72.89	1.14
GPT-2 + TF-IDF	68.34	66.23	67.45	0.92
GPT-2 + Hybrid Retrieval	79.45	77.78	78.45	1.18
mBART + FAISS	82.12	80.67	81.45	1.08
mBART + BM25	78.45	76.34	77.12	1.14
mBART + TF-IDF	74.34	72.89	73.45	0.98
mBART + Hybrid Retrieval	85.67	84.45	83.78	1.25
GPT-Neo + FAISS	79.45	77.34	78.12	1.10
GPT-Neo + BM25	75.67	73.12	74.45	1.16
GPT-Neo + TF-IDF	72.34	70.45	71.89	0.96
GPT-Neo + Hybrid Retrieval	83.45	81.78	82.12	1.22

Table 7. Diversity and overlap in retrieved content across retrieval models.

Retrieval Model	Unique Chunks (%)	Redundant Chunks (%)	Query Overlap (%)	Average Retrieval Time (ms)
FAISS	78.34	12.45	85.67	105
BM25	72.12	18.89	82.34	98
TF-IDF	68.45	22.34	79.23	92
Hybrid Retrieval	83.56	8.67	89.45	120

The results presented in Table 8 show the performance of fine-tuned models and RAG-enhanced models across three key Swahili NLP tasks: translation, summarization, and question answering. These tasks are vital for building effective Swahili conversational systems, and the integration of retrieval mechanisms like RAG plays a significant role in improving model performance. For Translation Accuracy, which measures how well each model translates Swahili text while preserving meaning and context, mT5 outperformed the other models. The fine-tuned mT5 achieved a translation accuracy of 83.45%, but when enhanced with RAG, this increased to 88.34%, demonstrating how retrieval can improve the translation process by providing additional context. In contrast, GPT-Neo performed the weakest in both the fine-tuned (72.89%) and RAG-enhanced (76.45%) versions, indicating its limitations in handling Swahili translation tasks. In Summarization Coherence, which evaluates how logically and fluently the models generate summaries, mT5 again leads. The fine-tuned version of mT5 achieved 82.12% coherence, but this improved to 87.12% with RAG, showing the benefits of retrieval in producing contextually relevant and coherent summaries. GPT-2 and GPT-Neo also showed improvements when enhanced with RAG, but their performance remained lower than that of mT5. Notably, GPT-Neo had the lowest summarization coherence at 71.12% in its fine-tuned form, further highlighting its struggles in this task. Finally, for Question Answering Accuracy, which measures the ability of models to generate accurate answers to Swahili queries, mT5 excelled with 80.67% accuracy in the fine-tuned version. When enhanced with RAG, its performance increased to 85.67%, showing that RAG enhanced the model's ability to provide more relevant and contextually appropriate answers by retrieving additional information. While GPT-2 showed moderate improvements from 72.34% (fine-tuned) to 75.23% (RAG), GPT-Neo remained the weakest performer, with an accuracy of 69.45% in the fine-tuned version and 72.12% in the RAG-enhanced version [51].

Table 8. Adaptability of fine-tuned and RAG-enhanced models across Swahili NLP tasks.

Model + Approach	Translation Accuracy (%)	Summarization Coherence (%)	Question Answering Accuracy (%)
mT5 (Fine-Tune)	83.45	82.12	80.67
mT5 (RAG)	88.34	87.12	85.67
GPT-2 (Fine-Tune)	75.23	74.12	72.34
GPT-2 (RAG)	79.45	77.89	75.23
mBART (Fine-Tune)	80.12	79.45	77.23
mBART (RAG)	83.56	82.34	80.45
GPT-Neo (Fine-Tune)	72.89	71.12	69.45
GPT-Neo (RAG)	76.45	74.78	72.12

4.3.1. Layer-Freezing Performance Analysis

Figure 9 presents a comparison of layer-freezing performance across four models, mT5, mBART, GPT-2, and GPT-Neo, evaluated on Swahili data. The comparison is made between fine-tuned and RAG models, showing how freezing layers affects model performance for different techniques. The graph clearly demonstrates that the RAG models consistently outperformed their fine-tuned counterparts. For example, mT5 with RAG exhibited a substantial improvement in performance as more layers were frozen, suggesting that RAG is particularly effective in enhancing translation and contextual accuracy for Swahili tasks. For mBART, the performance increase with RAG was more moderate, indicating that while RAG does provide some improvements, the gains are less significant than those seen with mT5. GPT-2, however, showed a marked improvement with RAG, outperforming its fine-tuned version in all frozen layers. This reflects GPT-2's capability to benefit significantly from the retrieval mechanism, which provides additional context during response generation. GPT-Neo showed a more stable and gradual improvement with RAG, suggesting that while it does not outperform GPT-2, RAG still enhances its performance when freezing layers. This pattern of improvement across all models suggests that RAG's retrieval mechanism adds significant value to Swahili NLP tasks, particularly when multiple layers are frozen to optimize computational resources. This analysis underscores the advantages of using RAG for Swahili NLP, where freezing layers is an effective strategy to reduce computational costs while still improving model performance. The findings support the idea that Retrieval-Augmented Generation is a strong approach for enhancing low-resource language tasks, making it more efficient and contextually aware [52].

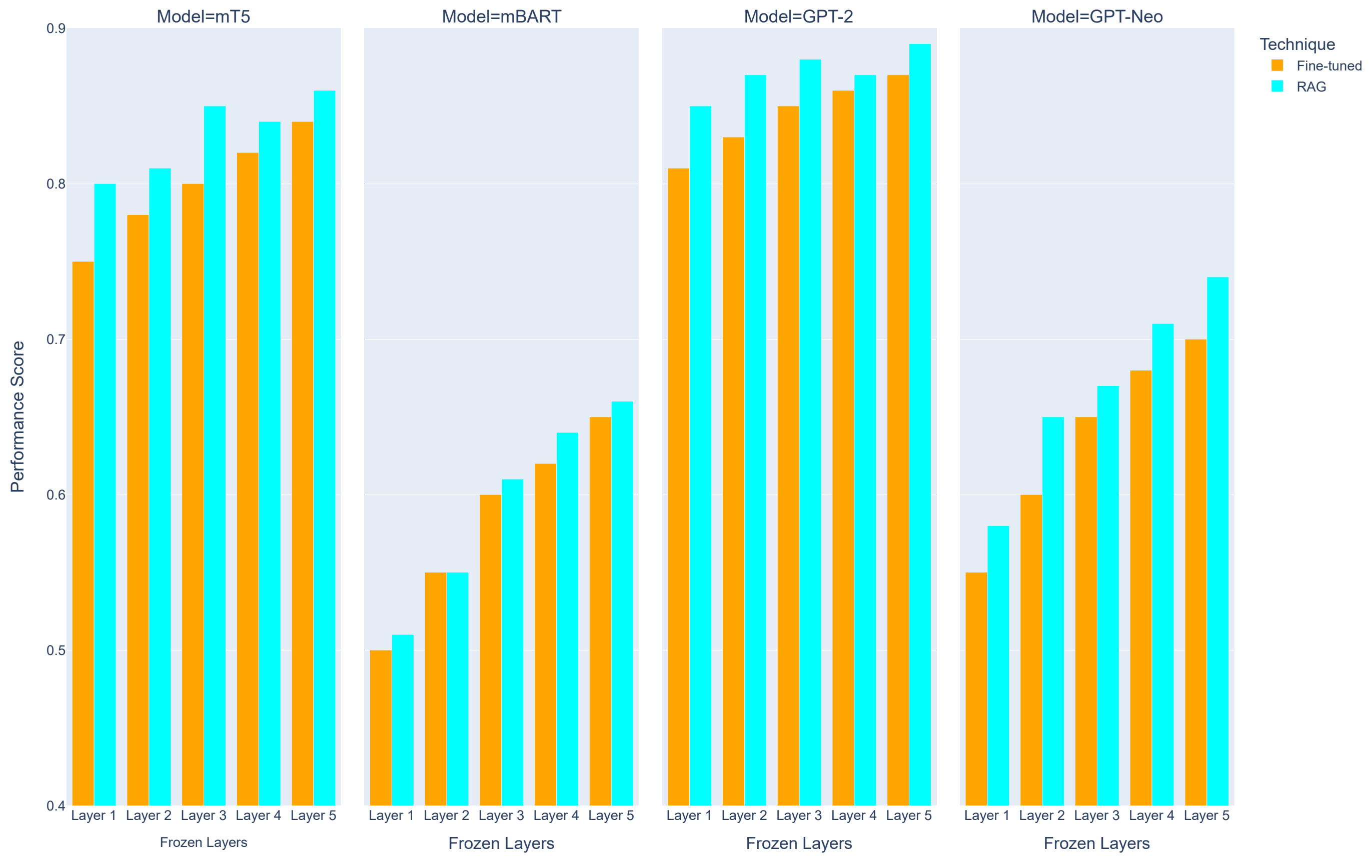


Figure 9. Layer-freezing performance comparison for Swahili data across models and techniques (fine-tuned and RAG).

4.3.2. BLEU Score Analysis

Figure 10 shows the comparison of BLEU scores between the Fine-Tune and RAG phases for the four models: mT5, GPT-2, mBART, and GPT-Neo. The graph clearly indicates that all four models perform better in the RAG phase compared to the Fine-Tune phase, with improvements in BLEU scores observed across the board. From the graph, it is evident that mT5 achieved the greatest improvement in BLEU score when switching from the Fine-Tune to the RAG phase. The increase in BLEU score for mT5 was substantial, suggesting that the RAG approach significantly enhanced the model's ability to generate more accurate translations [53]. GPT-2 also showed an improvement in BLEU score, though the change was smaller compared to mT5. This indicates that while the RAG technique improved GPT-2's performance, the effect was less pronounced than in mT5. Similarly, both mBART and GPT-Neo showed steady but smaller improvements in BLEU scores under the RAG phase. While the improvements were not as large as those seen in mT5, the RAG approach still resulted in a noticeable enhancement in translation quality for these models. The results presented in Figure 10 suggest that the RAG approach consistently led to better performance across all the models tested. This improvement in BLEU scores indicates that RAG helps the models generate more fluent and accurate translations. The ability of RAG to incorporate relevant external information likely contributed to this increase in translation quality. These findings demonstrate that RAG is an effective method for improving the performance of models in Swahili NLP tasks [54].

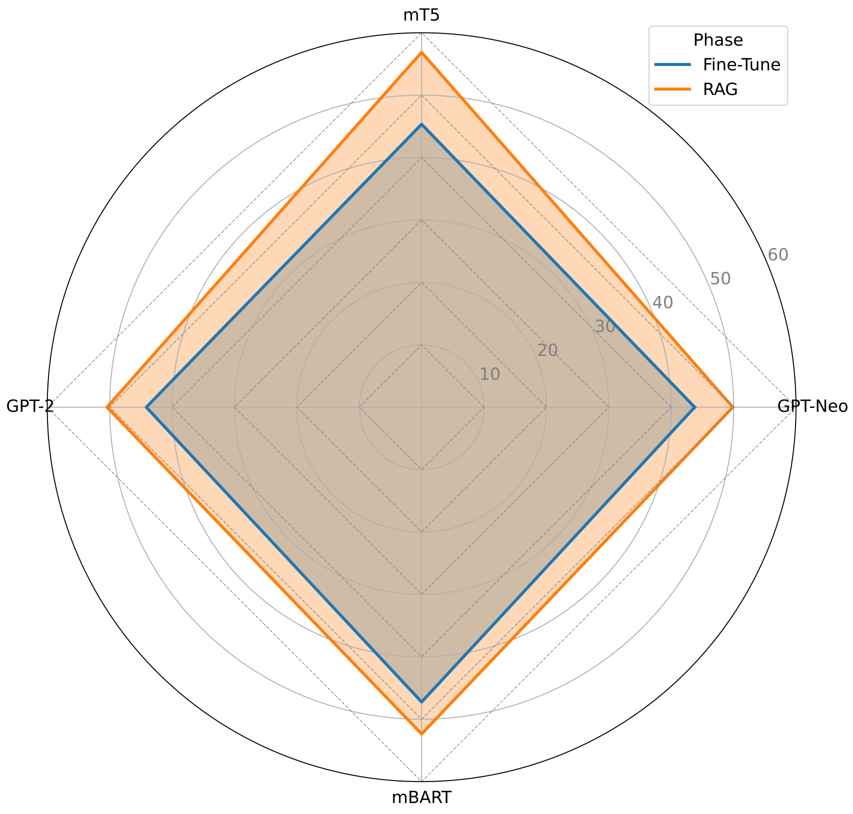


Figure 10. BLEU score comparison between Fine-Tune and RAG models.

4.3.3. METEOR Score Analysis

Figure 11 presents the comparison of METEOR scores between the Fine-Tune and RAG phases for four models: mT5, GPT-2, mBART, and GPT-Neo. The graph highlights the differences in METEOR scores when the models are trained using the RAG method compared to the Fine-Tune method. All four models show improved METEOR scores when using RAG, as opposed to fine-tuning. This indicates that the RAG method leads to more accurate translations. mT5 demonstrates the most significant improvement in METEOR score when switching from Fine-Tune to RAG. The METEOR score for mT5 increases considerably in the RAG phase, suggesting that RAG had a strong positive effect on the model's performance. This increase implies that RAG enabled mT5 to generate more accurate translations, particularly for Swahili language tasks. GPT-2 also showed an increase in METEOR score with RAG, but the improvement was smaller compared to mT5. While the change wa less dramatic, the results indicate that RAG still enhanced GPT-2's performance. mBART and GPT-Neo both show improvements in METEOR scores as well, though the changes were less pronounced than those observed in mT5. Nevertheless, RAG still produced better results for these models, resulting in more fluent and accurate translations compared to fine-tuning. The graph confirms that RAG consistently improved the METEOR scores for all models tested. This suggests that RAG enhanced the models' ability to produce more accurate and coherent translations. The observed improvements in METEOR scores were likely due to RAG's ability to retrieve and incorporate relevant external information, improving the overall translation quality for Swahili NLP tasks [55].

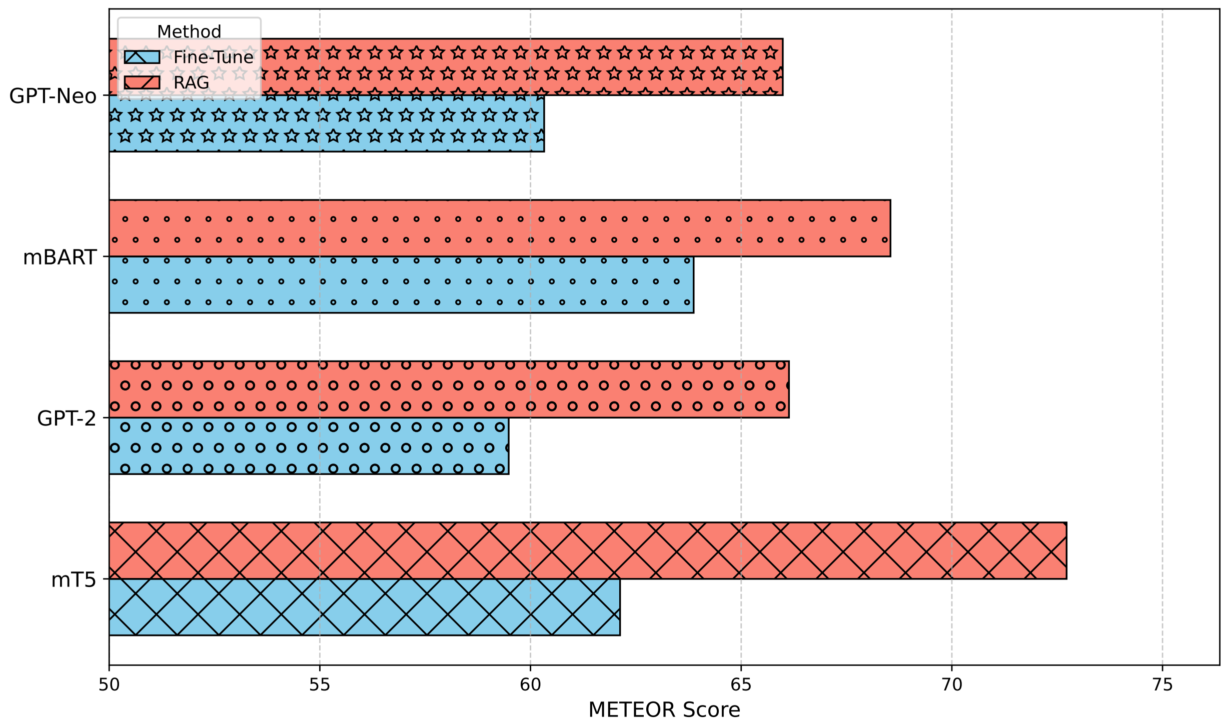


Figure 11. METEOR score comparison between Fine-Tune and RAG models.

4.3.4. ROUGE-L Score Analysis

The graph in Figure 12 presents a comparison of ROUGE-L scores for four different models: mT5, mBART, GPT-2, and GPT-Neo. The scores are shown under two conditions: fine-tuned and Retrieval-Augmented Generation (RAG). ROUGE-L is a widely used metric to evaluate the performance of text generation tasks by comparing the generated text to reference text. A higher ROUGE-L score signifies greater similarity, indicating better performance. As observed from the figure, the RAG models outperformed the fine-tuned models in all four models tested. Notably, mT5, GPT-2, and GPT-Neo showed significant improvements in their ROUGE-L scores when integrated with the RAG framework. This improvement suggests that the addition of retrieval-based information, which enhances the generation process, leads to more accurate and contextually relevant outputs. mT5 exhibited the most pronounced improvement, with the highest ROUGE-L score in the RAG condition, while GPT-2 and GPT-Neo also benefited notably from the retrieval mechanism [56]. These improvements indicate that the RAG framework helped boost the performance of these models by leveraging external information to generate more fluent and relevant responses. On the other hand, mBART shows a relatively smaller improvement under the RAG condition. This could be due to its initial performance, which already aligns closer to the RAG-enhanced results, suggesting that the impact of RAG might vary depending on the model's baseline performance. The graph clearly demonstrates that RAG provides a clear advantage in enhancing the performance of language models for Swahili NLP tasks. By enabling the retrieval of additional information, RAG helps models produce more accurate and contextually grounded responses, especially in low-resource settings where data may be limited.

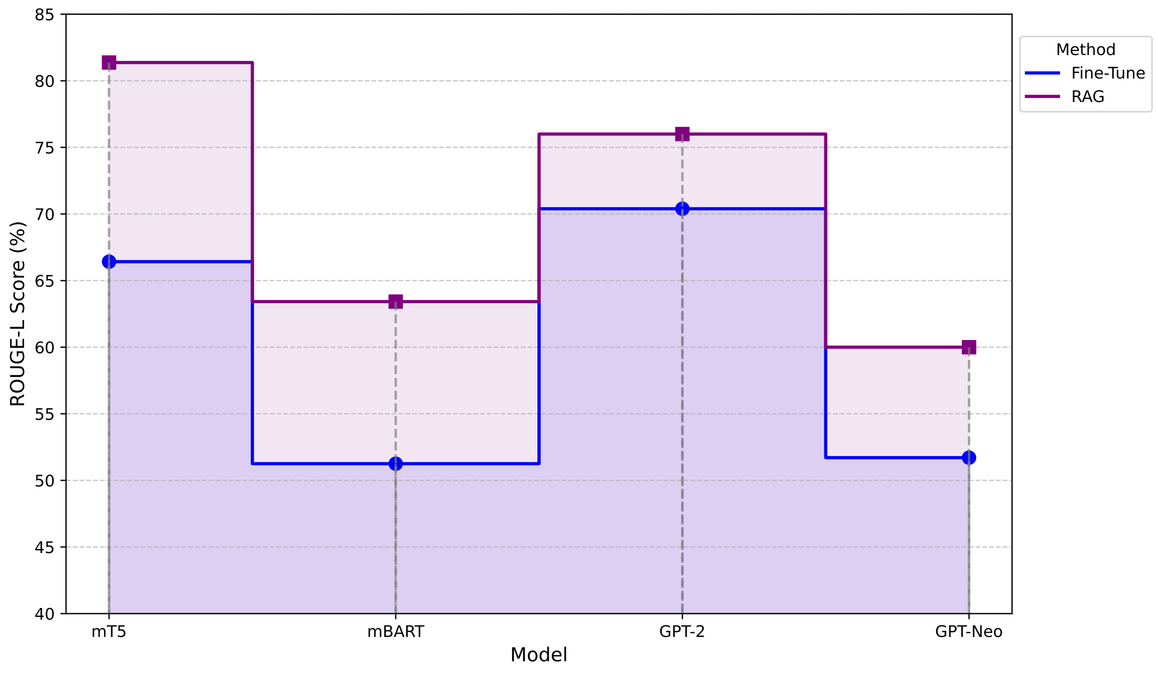


Figure 12. ROUGE-L score comparison between fine-tuned and RAG models. The graph compares the ROUGE-L scores for four models, mT5, mBART, GPT-2, and GPT-Neo, under both fine-tuned and Retrieval-Augmented Generation (RAG) conditions.

4.3.5. Query Performance Analysis

The parallel categories diagram in Figure 13 compares the Query Performance across different models (mT5, mBART, GPT-2, and GPT-Neo) and techniques (fine-tuned and RAG) and Table 9 provides the Swahili queries and their corresponding English translations used in Figure 13. These queries were analyzed to evaluate model performance under different techniques, including fine-tuning and Retrieval-Augmented Generation (RAG). The table establishes the linguistic context and demonstrates the types of queries processed by the models in the RAG framework. The diagram shows how the performance changes when applying these two techniques to different types of queries and Swahili language questions. The results suggest that the RAG method generally leads to an improvement in model performance compared to the fine-tuned method. This indicates that RAG enhanced the models' ability to generate more accurate and relevant responses. The diagram highlights that RAG has a significant positive effect on the performance of all the models. For example, the performance of mT5 increased from 68.12 percent with the fine-tuned method to 84.34 percent with the RAG technique, which represents a substantial improvement. Other models such as mBART, GPT-2, and GPT-Neo also showed improvements with RAG, although the magnitude of these improvements varies across the models. For GPT-2, the improvement was particularly noteworthy. The performance of GPT-2 increased from 65.62 percent with the fine-tuned method to 76.51 percent with RAG. This demonstrates that RAG had a significant impact on the performance of GPT-2, making it more effective at handling the queries. The diagram also indicates that the type of query—whether Original or Variation—affects the performance, with Variation queries generally producing better performance across all models. The improvement in performance with RAG is likely due to its ability to retrieve and incorporate relevant external information, which allows the models to generate more accurate and contextually appropriate responses. The diagram effectively demonstrates that RAG enhances the models' ability to process Swahili language queries and produce higher-quality outputs [57].

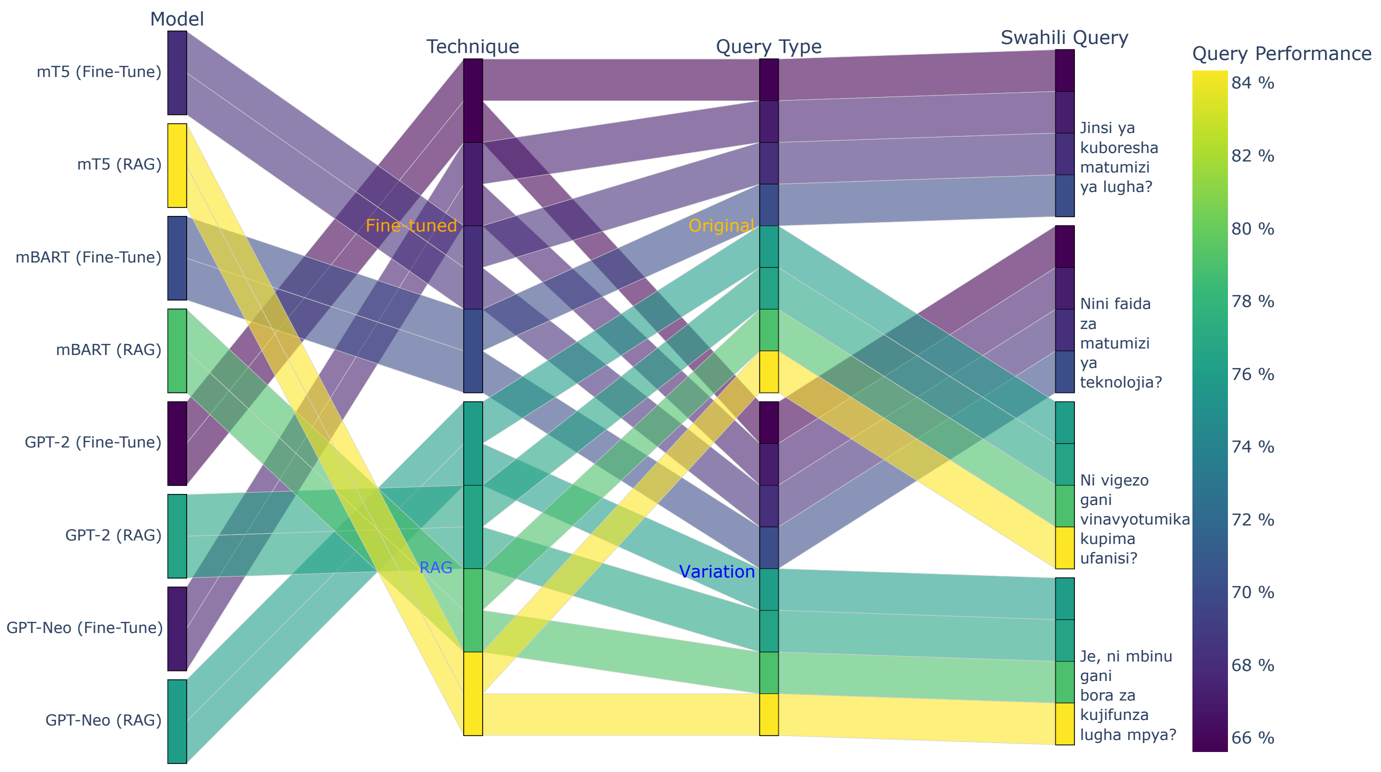


Figure 13. Performance comparison across models and techniques for Query Performance.

Table 9. Swahili query and corresponding English translation for Figure 13.

Swahili Query	English Query
Jinsi ya kuboresha matumizi ya lugha?	How to improve language usage?
Nini faida za matumizi ya teknolojia?	What are the benefits of using technology?
Ni vigezo gani vinavyotumika kupima ufanisi?	What are the criteria used to measure success?
Je, ni mbinu gani bora za kujifunza lugha mpya?	What are the best methods to learn a new language?

4.3.6. Multi-Turn Coherence Scores Analysis

Figure 14 presents the comparison of Multi-Turn Coherence scores across various models (MT5, mBART, GPT-2, and GPT-Neo) using two techniques: fine-tuned and Retrieval-Augmented Generation (RAG). The plot uses color coding to represent the coherence score, which indicates the relevance and consistency of the responses generated by the models across multiple conversational turns. From the plot, it is evident that the RAG technique consistently outperformed the fine-tuned technique across all models. Specifically, for the MT5 model, the coherence score showed a substantial improvement from 75.43 in the fine-tuned method to 88.26 with RAG. Similarly, other models such as mBART, GPT-2, and GPT-Neo exhibited increased coherence scores with RAG, confirming its effectiveness in improving multi-turn response consistency. Notably, GPT-2 showed a significant improvement in coherence when switching from fine-tuned to RAG, with the score increasing from 72.39 to 81.24, highlighting the effectiveness of RAG for this model. Furthermore, the plot illustrates that the performance of the fine-tuned models tended to decline as the conversation progressed, particularly from Turn 1 to Turn 5. On the other hand, the RAG technique maintained a relatively stable performance across all turns, emphasizing the technique's ability to generate more consistent and coherent responses throughout the conversation. The enhanced performance with RAG was likely due to its ability to retrieve relevant information from external knowledge sources, which helped the models generate more contextually appropriate and relevant responses. These results underscore the benefits of using RAG for improving the quality of responses in multi-turn conversations, as it allows for better coherence and relevance over time.

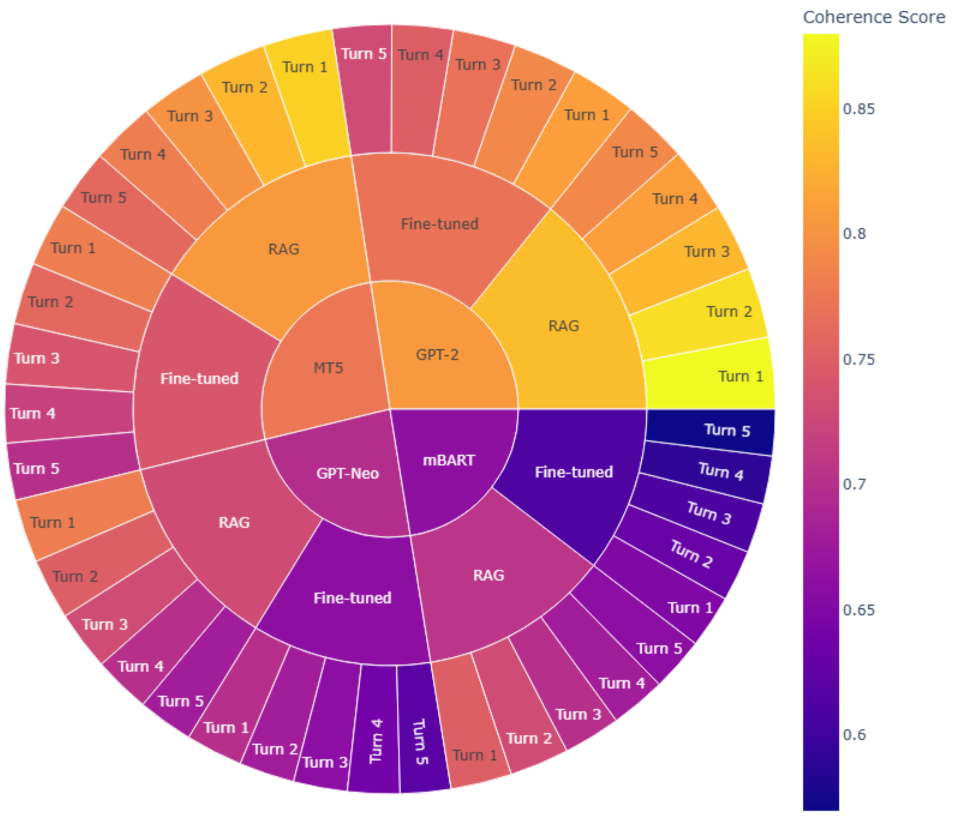


Figure 14. Comparison of Multi-Turn Coherence scores across models and techniques.

4.3.7. General Swahili Queries Analysis

The graph in Figure 15 compares the performance of fine-tuned and Retrieval-Augmented Generation (RAG) models on general Swahili queries. As shown in the graph, the RAG models consistently outperformed the fine-tuned models across all the queries tested. This improvement can be attributed to the ability of the RAG framework to incorporate external retrieval-based information, enhancing the models' capacity to generate more accurate and contextually relevant responses. Specifically, the performance of models like GPT-2, GPT-Neo, and mT5 showed a marked increase when integrated with the RAG technique, indicating that retrieval mechanisms play a significant role in improving the fluency and relevance of generated answers. Among the models, mT5 exhibited the most pronounced improvement, achieving the highest performance under the RAG condition. These results underscore the value of RAG in boosting the performance of language models, particularly in low-resource languages like Swahili, where access to extensive training data may be limited. Table 10 provides the Swahili queries and their corresponding English translations, which are directly analyzed in Figure 15. These queries were used to evaluate the performance of fine-tuned and Retrieval-Augmented Generation (RAG) models on general Swahili queries.

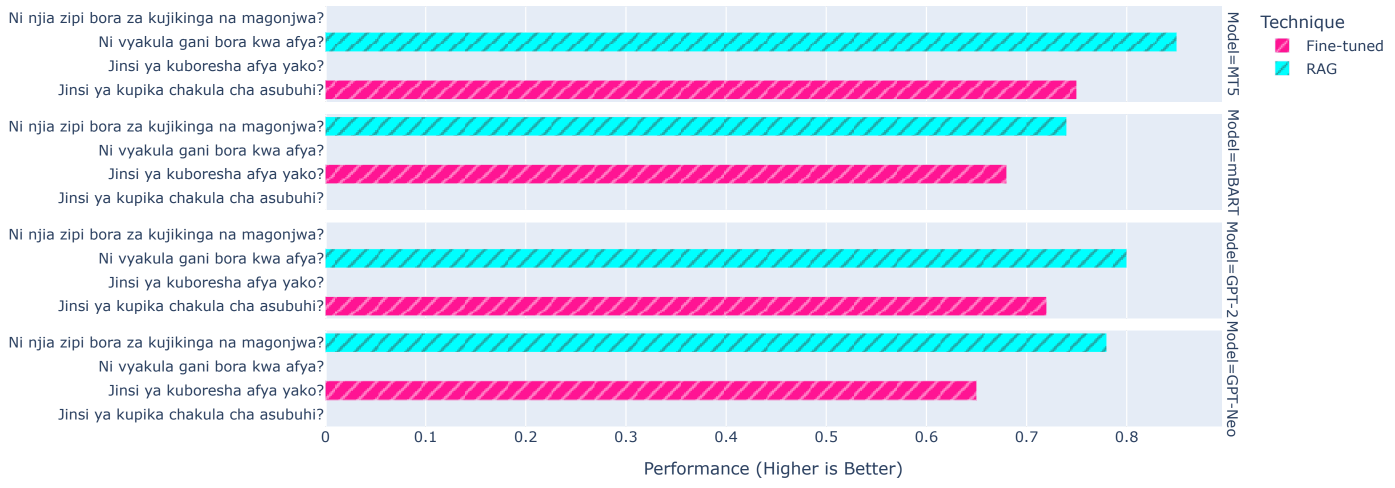


Figure 15. Performance comparison between fine-tuned and RAG models on general Swahili queries. The graph compares the performance of four models, mT5, mBART, GPT-2, and GPT-Neo, under both fine-tuned and Retrieval-Augmented Generation (RAG) conditions.

Table 10. Swahili query and corresponding English translation for Figure 15.

Swahili Query	English Query
Ni njia zipi bora za kujikinga na magonjwa?	What are the best ways to protect yourself from diseases?
Ni vyakula gani bora kwa afya?	What are the best foods for health?
Jinsi ya kuboresha afya yako?	How to improve your health?
Jinsi ya kupika chakula cha asubuhi?	How to cook breakfast?

4.3.8. Inference Time Analysis

Figure 16 shows the comparison of inference times in milliseconds for four models: mT5, mBART, GPT-2, and GPT-Neo. The models were tested with two techniques: fine-tuned and Retrieval-Augmented Generation (RAG). The horizontal bars show the time each model took to process the tasks. The bars for the fine-tuned models are on the left, and the bars for the RAG models are on the right. For example, the mT5 model took 932.4 ms when fine-tuned, and this time increased to 1045.2 ms when using RAG. This pattern of longer inference times with RAG is observed for all the models. GPT-2 took 854.1 ms with fine-tuning and 976.5 ms with RAG. GPT-Neo showed a smaller increase, from 782.3 ms to 895.6 ms. The X-axis is set to a range of 3500 ms to make sure that the longest bars fit in the plot. A vertical line at 0 ms divides the fine-tuned and RAG bars. This figure shows the trade-off between longer inference times for RAG and better performance in generating accurate and relevant responses. Even though RAG takes more time, it can help the model generate better responses.

4.3.9. Perplexity Score Analysis

Figure 17 presents the perplexity score comparison for four models—mT5, mBART, GPT-2, and GPT-Neo—evaluated under fine-tuned and Retrieval-Augmented Generation (RAG) conditions. Perplexity is a metric commonly used to assess the performance of language models, with lower values indicating better performance and greater alignment with the reference data. As illustrated in the graph, the RAG models consistently achieved lower perplexity scores compared to their fine-tuned counterparts across all tested models. This highlights the effectiveness of incorporating retrieval mechanisms in improving the quality of language model outputs. Among the models, GPT-Neo demonstrated the most significant reduction in perplexity under the RAG condition, followed by GPT-2 and mT5. On the other hand, mBART exhibited a moderate decrease in perplexity with RAG integration, indicating that the baseline performance of the model may influence the extent of improvement. These results emphasize the advantage of the RAG approach in enhancing the generative capabilities of language models, particularly in tasks involving Swahili language processing. The lower perplexity scores achieved by RAG models suggest more accurate and contextually relevant responses, addressing challenges associated with low-resource languages.

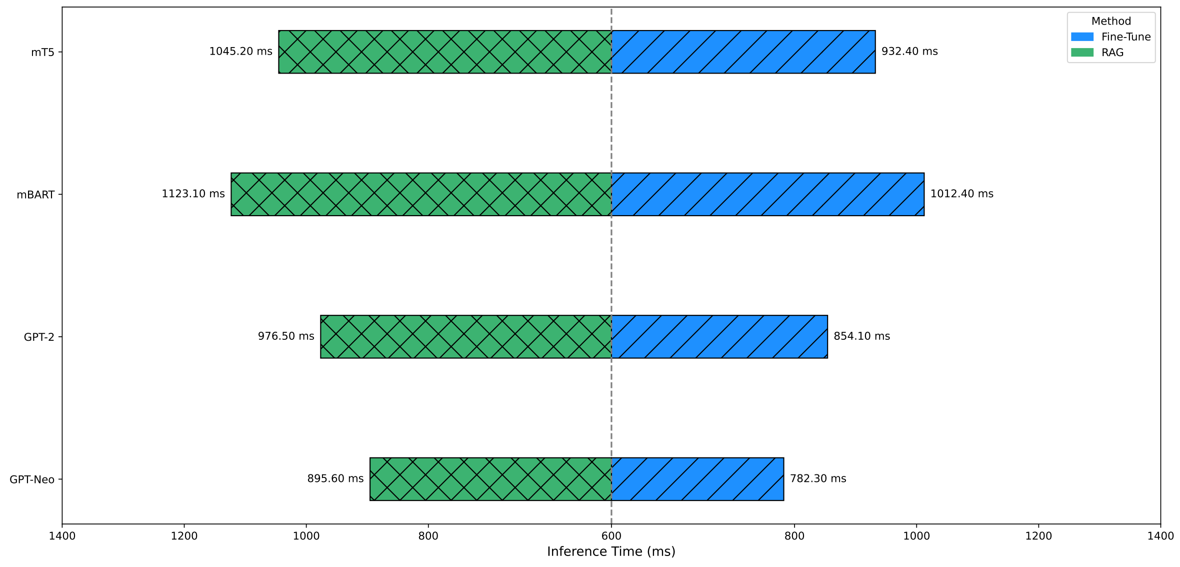


Figure 16. Comparison of inference times across models for Fine-Tune and RAG techniques.

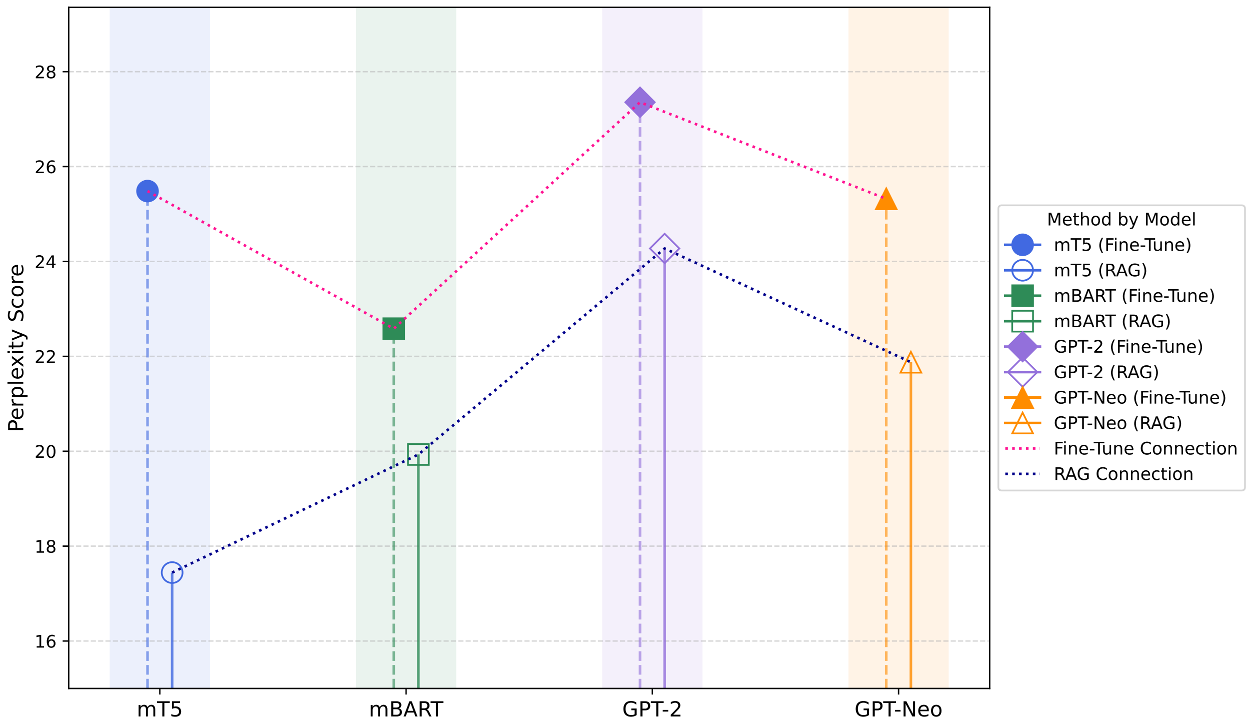


Figure 17. Perplexity score comparison by model for fine-tuned and RAG conditions. The graph shows perplexity scores for mT5, mBART, GPT-2, and GPT-Neo under both fine-tuned and Retrieval-Augmented Generation (RAG) conditions. Lower perplexity scores indicate better performance.

5. Implementation

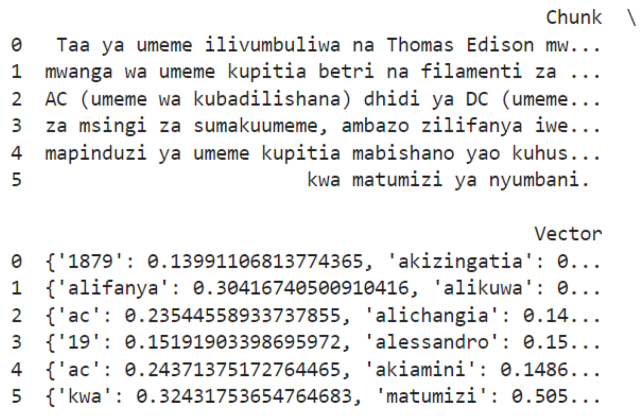
This section provides an overview of the implementation process for the Retrieval-Augmented Generation (RAG) framework. The primary objective is to preprocess the text by dividing it into smaller, meaningful segments, followed by the vectorization of these segments. These vectors are then used in the RAG framework, which integrates a retrieval mechanism to enhance the quality and relevance of generated responses. The implementation involves transforming textual data into a format suitable for efficient retrieval, ensuring that the system can generate accurate and contextually appropriate answers based on user queries.

5.1. Document Chunking and Vectorization

Table 11 and Figure 18 illustrate the process of document chunking and vectorization within the Retrieval-Augmented Generation (RAG) framework. Table 11 presents a side-by-side comparison of Swahili text chunks and their English translations, ensuring clarity for bilingual evaluation. Figure 18 shows how these text chunks are embedded into vector representations for efficient retrieval.

Table 11. Swahili text and its English translation for retrieval purposes. This table presents a segment of Swahili text alongside its English translation, both of which are organized for efficient retrieval within the Retrieval-Augmented Generation (RAG) framework. Each chunk of information is clearly delineated to facilitate accurate comparison and retrieval during the information generation process.

Swahili Text	**English Translation**
Taa ya umeme ilivumbuliwa na Thomas Edison mwaka wa 1879, ingawa kulikuwa na uvumbuzi wa awali unaohusiana na taa za umeme. Edison aliunda mfumo wa mwanga wa umeme unaoweza kutumika kwa vitendo, akizingatia mwendelezo wa uvumbuzi wa awali. Hata hivyo, kabla ya Edison, wanasayansi wengine kama Humphry Davy waligundua mwanga wa umeme kupitia betri na filamenti za kaboni. Edison alifanya uvumbuzi wake kuwa bora zaidi kwa kuongeza maisha ya balbu na kutengeneza mtandao wa umeme wa vitendo kwa miji. Nikola Tesla, ambaye alifanya kazi kwa muda chini ya Edison, alikuwa mpinzani mkubwa wa Edison katika maendeleo ya mifumo ya AC (umeme wa kubadilishana) dhidi ya DC (umeme wa moja kwa moja). Tesla alichangia kwa kiwango kikubwa teknolojia ya umeme wa AC, ambayo hatimaye ikashinda kwa matumizi ya umbali mrefu.	*The electric light bulb was invented by Thomas Edison in 1879, although there were earlier innovations related to electric lights. Edison created a practical electric light system, building on prior inventions. However, before Edison, other scientists like Humphry Davy had discovered electric light using batteries and carbon filaments. Edison improved his invention by extending the bulb's lifespan and creating a practical electric grid for cities. Nikola Tesla, who worked briefly under Edison, became a major opponent of Edison in the development of AC (alternating current) systems versus DC (direct current). Tesla contributed significantly to the development of AC technology, which ultimately prevailed for long-distance use.*
Historia ya nishati ya umeme ilianzia na uvumbuzi wa betri na majaribio ya Michael Faraday kuhusu sumakuumeme. Faraday aligundua kanuni za msingi za sumakuumeme, ambazo zilifanya iwezekane kuzalisha umeme kwa kutumia jenereta. Katika karne ya 19, wanasayansi na wavumbuzi walifanya majaribio makubwa ya kuboresha teknolojia ya umeme. Alessandro Volta aligundua betri ya kwanza ya voltaic, ambayo ilifanikisha uzalishaji wa umeme wa mara kwa mara. Nikola Tesla na Thomas Edison waliongoza mapinduzi ya umeme kupitia mabishano yao kuhusu mifumo ya AC na DC. Tesla alihimiza matumizi ya mfumo wa AC kwa sababu ya uwezo wake wa kusafirisha umeme kwa umbali mrefu bila hasara kubwa ya nishati. Kwa upande mwingine, Edison alisisitiza matumizi ya mfumo wa DC, akiamini kuwa ulikuwa salama zaidi kwa matumizi ya nyumbani.	*The history of electric power began with the invention of the battery and the experiments of Michael Faraday on electromagnetism. Faraday discovered the basic principles of electromagnetism, which made it possible to generate electricity using a generator. In the 19th century, scientists and inventors carried out significant experiments to improve electrical technology. Alessandro Volta discovered the first voltaic battery, which enabled the production of continuous electricity. Nikola Tesla and Thomas Edison led the electrical revolution through their debates on AC and DC systems. Tesla advocated for the use of the AC system due to its ability to transmit electricity over long distances with minimal energy loss. On the other hand, Edison emphasized the use of the DC system, believing it was safer for household use.*



```
                                              Chunk  \
0   Taa ya umeme ilivumbuliwa na Thomas Edison mw...
1  mwanga wa umeme kupitia betri na filamenti za ...
2  AC (umeme wa kubadilishana) dhidi ya DC (umeme...
3  za msingi za sumakuumeme, ambazo zilifanya iwe...
4  mapinduzi ya umeme kupitia mabishano yao kuhus...
5                        kwa matumizi ya nyumbani.

                                             Vector
0  {'1879': 0.13991106813774365, 'akizingatia': 0...
1  {'alifanya': 0.30416740500910416, 'alikuwa': 0...
2  {'ac': 0.23544558933737855, 'alichangia': 0.14...
3  {'19': 0.15191903398695972, 'alessandro': 0.15...
4  {'ac': 0.24371375172764465, 'akiamini': 0.1486...
5  {'kwa': 0.32431753654764683, 'matumizi': 0.505...
```

Figure 18. Text chunk and corresponding vector representation in RAG model. The raw text chunk is embedded into a dense vector space, where it is later used for efficient retrieval during the generation process.

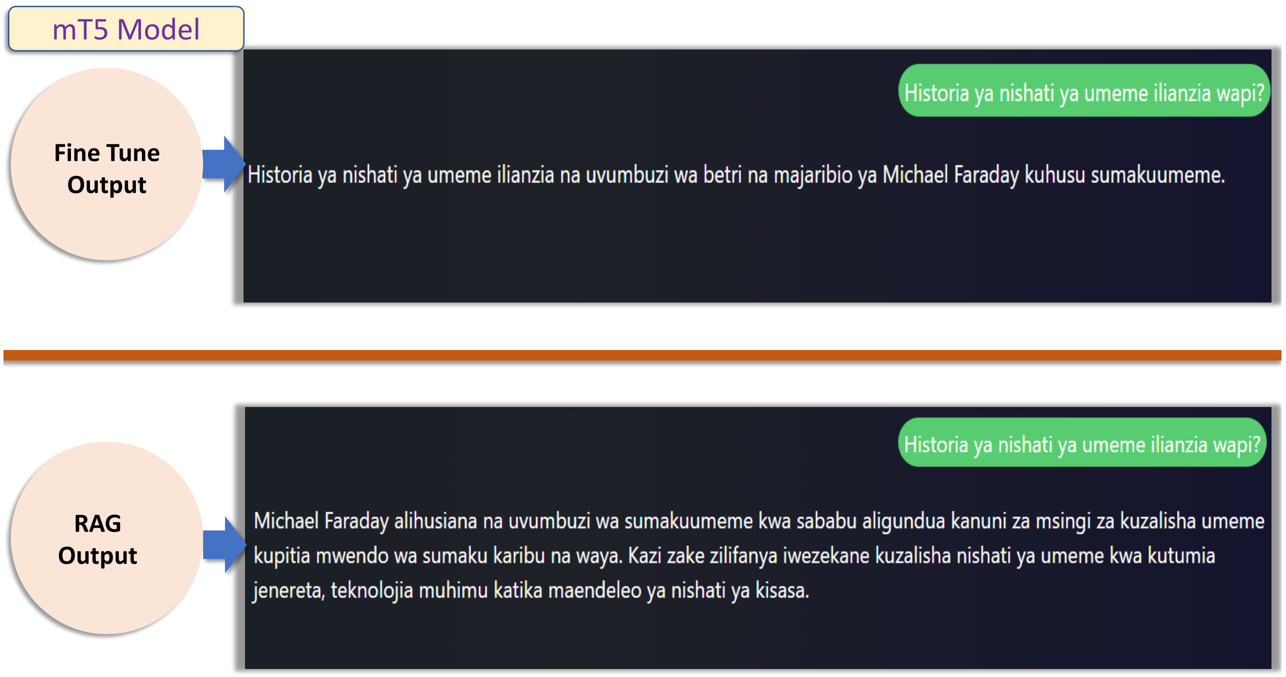
5.2. Output Response

5.2.1. mT5 Model Response

The mT5 model was evaluated on its ability to generate responses for Swahili queries under two distinct conditions: Fine-Tune and Retrieval-Augmented Generation (RAG). The evaluation revealed a substantial enhancement in response quality with the RAG approach compared to Fine-Tune. Table 12 highlights an illustrative example of a Swahili question alongside its English translation, accompanied by the respective responses generated under both conditions. The RAG responses were notably more comprehensive and contextually enriched, demonstrating a superior understanding of the query by integrating external knowledge. Figure 19 further underscores the significant advancements achieved with RAG, illustrating consistently higher accuracy and depth in the generated outputs. Unlike the Fine-Tune approach, which often provided concise but superficial responses, the RAG framework enabled the mT5 model to produce detailed and contextually relevant answers, effectively addressing the nuances of complex queries. These results affirm the efficacy of the RAG methodology in dramatically enhancing the mT5 model's performance, making it a robust solution for handling sophisticated and resource-intensive tasks in Swahili language processing.

Table 12. Swahili question and answer with English translation for Figure 19.

Language	Question	Fine-Tune Answer	RAG Answer
Swahili	Historia ya nishati ya umeme ilianzia wapi?	Historia ya nishati ya umeme ilianzia na uvumbuzi wa betri na majaribio ya Michael Faraday kuhusu sumakuumeme.	Michael Faraday alihusiana na uvumbuzi wa sumakuumeme kwa sababu aligundua kanuni za msingi za kuzalisha umeme kupitia mwendo wa sumaku karibu na waya. Kazi zake zilifanya iwezekane kuzalisha nishati ya umeme kwa kutumia jenereta, teknolojia muhimu katika maendeleo ya nishati ya kisasa.
English	*Where did the history of electricity energy begin?*	*The history of electricity began with the invention of the battery and Michael Faraday's experiments on electromagnetism.*	*Michael Faraday is associated with the discovery of electromagnetism because he discovered the fundamental principles of generating electricity through the movement of magnetism near a wire. His work made it possible to generate electrical energy using generators, a crucial technology in the development of modern energy.*



Historia ya nishati ya umeme ilianzia wapi?

Historia ya nishati ya umeme ilianzia na uvumbuzi wa betri na majaribio ya Michael Faraday kuhusu sumakuumeme.

Historia ya nishati ya umeme ilianzia wapi?

Michael Faraday alihusiana na uvumbuzi wa sumakuumeme kwa sababu aligundua kanuni za msingi za kuzalisha umeme kupitia mwendo wa sumaku karibu na waya. Kazi zake zilifanya iwezekane kuzalisha nishati ya umeme kwa kutumia jenereta, teknolojia muhimu katika maendeleo ya nishati ya kisasa.

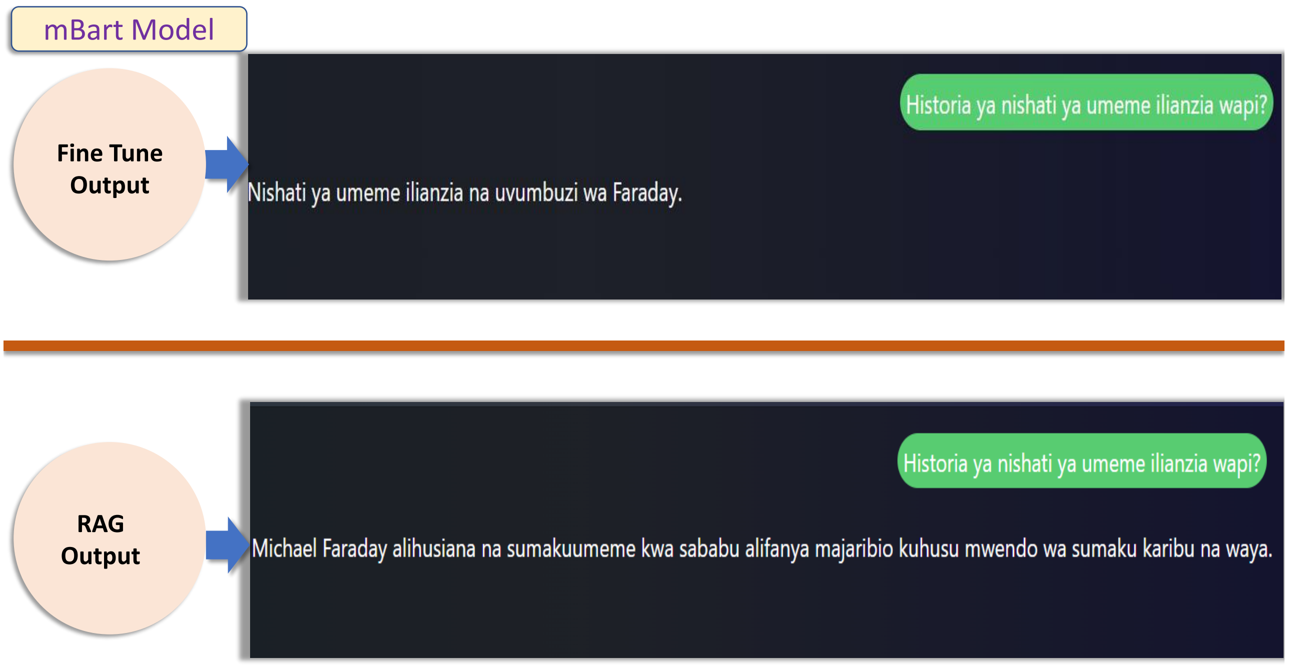
Figure 19. Response created by mT5 model in both cases for Fine-Tune and RAG, with RAG showing a good response compared to Fine-Tune.

5.2.2. mBart Model Response

The mBart model's performance was assessed using Swahili queries under both Fine-Tune and Retrieval-Augmented Generation (RAG) conditions. Table 13 demonstrates the differences in response quality between these two approaches. For the Fine-Tune method, the model provided brief and partially accurate answers, often lacking context and depth. In contrast, the RAG approach significantly improved the responses by incorporating detailed external information. For instance, while the Fine-Tune response mentioned only Faraday's name, the RAG output elaborated on Michael Faraday's experiments with electromagnetism, describing his work on the movement of magnetism near a wire. This added context highlights the enhanced capabilities of RAG in providing comprehensive answers. Figure 20 further illustrates this comparison, showing that the RAG-generated outputs consistently outperform Fine-Tune responses in quality and relevance. These results underscore the effectiveness of the RAG framework in enhancing the mBart model's ability to handle complex queries, especially for low-resource languages like Swahili.

Table 13. Swahili question and answer with English translation for Figure 20.

Language	Question	Fine-Tune Answer	RAG Answer
Swahili	Historia ya nishati ya umeme ilianzia wapi?	Nishati ya umeme ilianzia na uvumbuzi wa Faraday.	Michael Faraday alihusiana na sumakuumeme kwa sababu alifanya majaribio kuhusu mwendo wa sumaku karibu na waya.
English	*Where did the history of electricity energy begin?*	*The history of electricity began with the invention of Faraday.*	*Michael Faraday is associated with electromagnetism because he conducted experiments on the movement of magnetism near a wire.*



Historia ya nishati ya umeme ilianzia wapi?

Nishati ya umeme ilianzia na uvumbuzi wa Faraday.

Historia ya nishati ya umeme ilianzia wapi?

Michael Faraday alihusiana na sumakuumeme kwa sababu alifanya majaribio kuhusu mwendo wa sumaku karibu na waya.

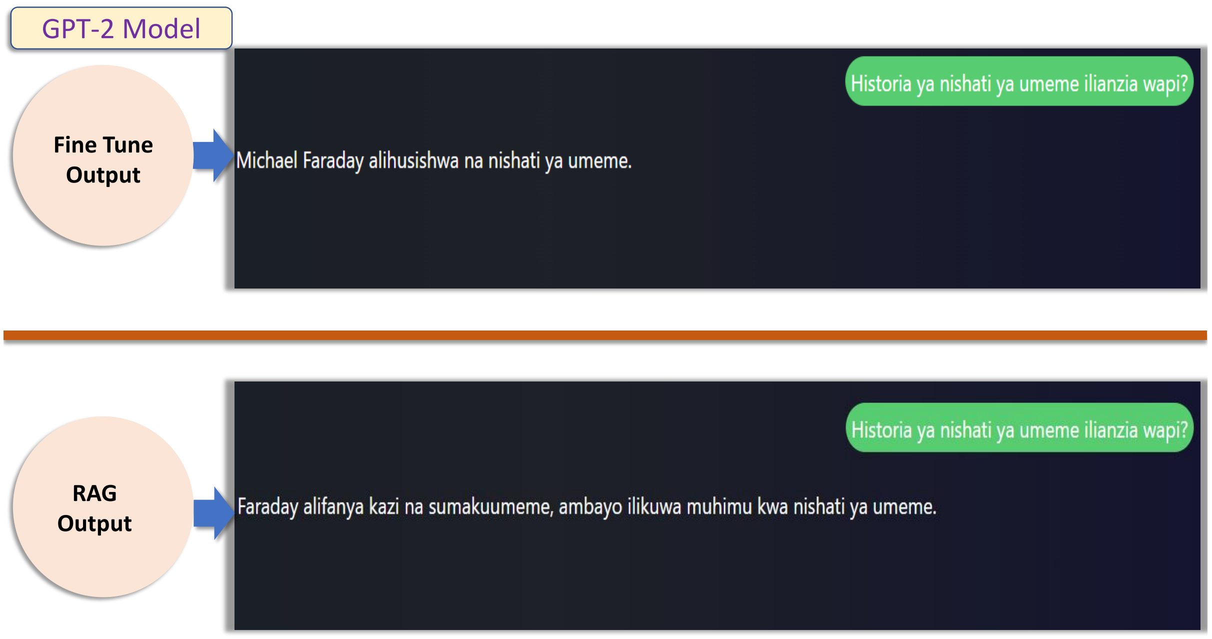
Figure 20. Response created by mBart model in both cases for Fine-Tune and RAG, with RAG showing an improvement response compared to Fine-Tune.

5.2.3. GPT-2 Model Response

The GPT-2 model was evaluated on its ability to generate responses for Swahili questions under both Fine-Tune and Retrieval-Augmented Generation (RAG) conditions. Table 14 presents a Swahili question and its English translation, alongside the responses generated under both methods. While the RAG approach improved the quality of responses compared to Fine-Tune, the overall performance of GPT-2 was less comprehensive than that of the mT5 and mBart models. For instance, the Fine-Tune response for GPT-2 only briefly associates Michael Faraday with electricity energy, while the RAG response provides slightly more context by mentioning Faraday's work with electromagnetism. However, the depth and specificity of the answers still fall short of the detailed explanations generated by mT5 and mBart under the RAG framework. Figure 21 further illustrates this comparison, showing that while RAG enhances GPT-2's ability to retrieve external information, its outputs remain relatively less detailed and contextually enriched compared to the other models. These findings indicate that while RAG improves GPT-2's performance, its capabilities in handling complex Swahili queries are more limited, highlighting the superiority of mT5 and mBart for tasks requiring detailed and nuanced responses.

Table 14. Swahili question and answer with English translation for Figure 21.

Language	Question	Fine-Tune Answer	RAG Answer
Swahili	Historia ya nishati ya umeme ilianzia wapi?	Michael Faraday alihusishwa na nishati ya umeme.	Faraday alifanya kazi na sumakuumeme, ambayo ilikuwa muhimu kwa nishati ya umeme.
English	*Where did the history of electricity energy begin?*	*Michael Faraday is associated with electricity energy.*	*Faraday worked with electromagnetism, which was crucial for electricity energy.*



Historia ya nishati ya umeme ilianzia wapi?

Michael Faraday alihusishwa na nishati ya umeme.

Historia ya nishati ya umeme ilianzia wapi?

Faraday alifanya kazi na sumakuumeme, ambayo ilikuwa muhimu kwa nishati ya umeme.

Figure 21. Response created by GPT-2 model in both cases for Fine-Tune and RAG, with RAG showing an improvement response compared to Fine-Tune.

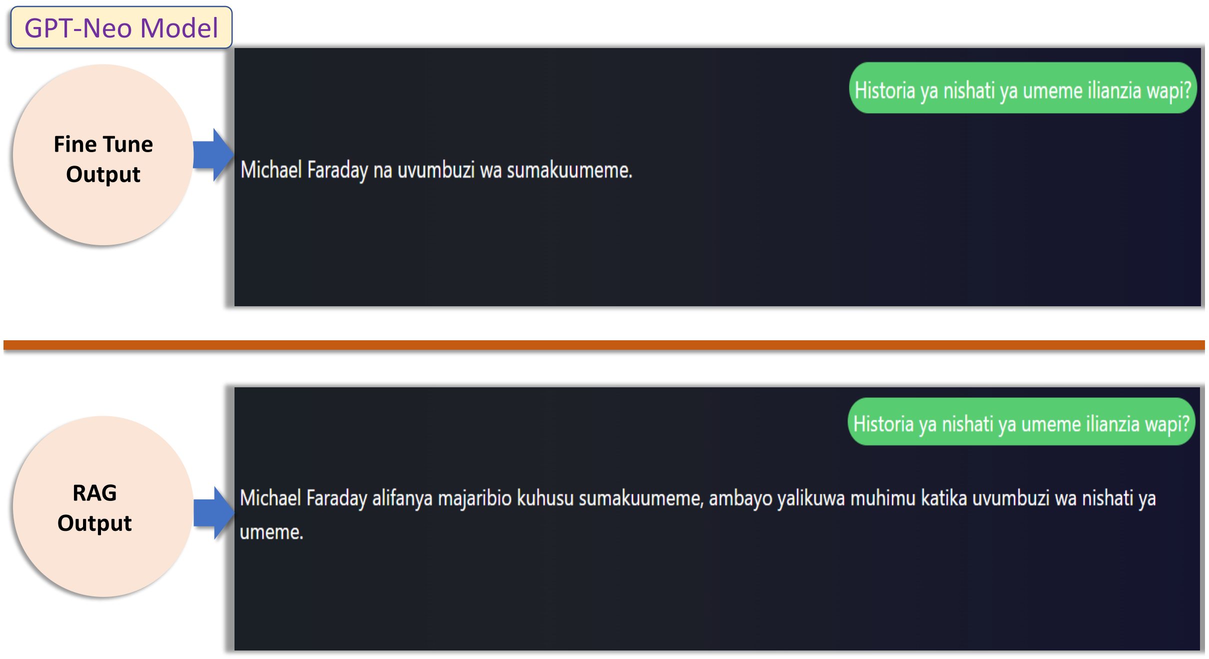
5.2.4. GPT-Neo Model Response

The GPT-Neo model was evaluated on its ability to generate responses for Swahili queries under both Fine-Tune and Retrieval-Augmented Generation (RAG) conditions. Table 15 provides an example of a Swahili question and its English translation, along with the corresponding responses generated by the GPT-Neo model under both approaches. While the RAG method improved response quality compared to the Fine-Tune approach, GPT-Neo's outputs remained less detailed and comprehensive than those generated by the mT5 model but showed noticeable improvement over GPT-2. For instance, the Fine-Tune response for GPT-Neo briefly mentioned Michael Faraday and electromagnetism, whereas the RAG response expanded on this by describing Faraday's experiments and their significance in the discovery of electricity.

Figure 22 further illustrates the comparative performance, demonstrating that RAG enables GPT-Neo to generate more contextually relevant and informative responses compared to Fine-Tune. However, the model still lacks the depth and nuance observed in the outputs of mT5, particularly when addressing complex and detailed queries. These results suggest that while GPT-Neo benefits significantly from the RAG framework, it remains a less robust option than mT5 for tasks requiring high levels of accuracy and contextual understanding in Swahili language processing.

Table 15. Swahili question and answer with English translation for Figure 22.

Language	Question	Fine-Tune Answer	RAG Answer
Swahili	Historia ya nishati ya umeme ilianzia wapi?	Michael Faraday na uvumbuzi wa sumakuumeme.	Michael Faraday alifanya majaribio kuhusu sumakuumeme, ambayo yalikuwa muhimu katika uvumbuzi wa nishati ya umeme.
English	*Where did the history of electricity energy begin?*	*Michael Faraday and the discovery of electromagnetism.*	*Michael Faraday conducted experiments on electromagnetism, which were crucial in the discovery of electricity energy.*



Historia ya nishati ya umeme ilianzia wapi?

Michael Faraday na uvumbuzi wa sumakuumeme.

Historia ya nishati ya umeme ilianzia wapi?

Michael Faraday alifanya majaribio kuhusu sumakuumeme, ambayo yalikuwa muhimu katika uvumbuzi wa nishati ya umeme.

Figure 22. Response created by GPT-Neo model in both cases for Fine-Tune and RAG, with RAG showing an improvement response compared to Fine-Tune.

6. Summary and Conclusions

This study explored the application of Retrieval-Augmented Generation (RAG) in enhancing Swahili language conversation systems. By integrating RAG with models such as mT5, GPT-2, mBART, and GPT-Neo, the research aimed to address the challenge of limited Swahili data, which has traditionally hindered the development of effective Natural Language Processing (NLP) systems for the language. The results demonstrated that RAG significantly improved model performance, as evidenced by metrics such as BLEU, METEOR, and Query Performance. Among the models tested, mT5 showed the best results, outperforming others in terms of these metrics. This suggests that mT5, when paired with RAG, is particularly effective in generating accurate, contextually relevant responses for Swahili language tasks. Despite the increased inference times associated with RAG, the improvements in response quality highlight its value for applications where accuracy is a top priority. The study also highlighted the trade-off between performance and inference time, which should be carefully considered when selecting models for real-time Swahili NLP applications. Future work will focus on optimizing the retrieval and generation processes within RAG to reduce inference times without compromising performance. The insights from this research could extend to other low-resource languages, contributing to the broader development of inclusive AI technologies.

Author Contributions: Conceptualization, B.W.; Methodology, E.V.N. and Q.L.; Software, E.V.N. and G.C.F.; Validation, E.V.N.; Formal analysis, E.V.N.; Investigation, Q.L., G.C.F. and X.Y.; Writing—original draft, E.V.N.; Writing—review & editing, X.Y. and B.W.; Supervision, B.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: Author Xu Yang was employed by the company China Yangtze Power Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this paper:

RAG	Retrieval-Augmented Generation
NLP	Natural Language Processing
BLEU	Bilingual Evaluation Understudy
METEOR	Metric for Evaluation of Translation with Explicit Ordering
ROUGE-L	Recall-Oriented Understudy for Gisting Evaluation - Longest Common Subsequence
FAISS	Facebook AI Similarity Search
BM25	Best Matching 25
TF-IDF	Term Frequency-Inverse Document Frequency
BM25 GPU	Best Matching 25 with GPU Acceleration
HTML	Hypertext Markup Language
LLM	Large Language Model
BERT	Bidirectional Encoder Representations from Transformers
LSTMs	Long Short-Term Memory Networks

References

1. Silvera-Tawil, D. Robotics in Healthcare: A Survey. *SN Comput. Sci.* **2024**, *5*, 189. [CrossRef]
2. Topol, E.J. High-performance medicine: The convergence of human and artificial intelligence. *Nat. Med.* **2019**, *25*, 44–56. [CrossRef] [PubMed]
3. Toukmaji, C.; Tee, A. Retrieval-Augmented Generation and LLM Agents for Biomimicry Design Solutions. In Proceedings of the AAAI Spring Symposium Series (SSS-24), Stanford, CA, USA, 25–27 March 2024.
4. Zeng, F.; Gan, W.; Wang, Y.; Liu, N.; Yu, P.S. Large Language Models for Robotics: A Survey. *arXiv* **2023**, arXiv:2311.07226.
5. Vaswani, A. Attention Is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017.
6. Jiang, A.Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D.S.; de las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. Mistral 7B. *arXiv* **2023**, arXiv:2310.06825.
7. Ni, J.; Qu, C.; Lu, J.; Dai, Z.; Ábrego, G.H.; Ma, J.; Zhao, V.Y.; Luan, Y.; Hall, K.B.; Chang, M.-W.; et al. Large Dual Encoders Are Generalizable Retrievers. *arXiv* **2021**, arXiv:2112.07899.
8. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv* **2019**, arXiv:1908.10084.
9. Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K.Q.; Artzi, Y. BERTScore: Evaluating Text Generation with BERT. *arXiv* **2020**, arXiv:1904.09675.
10. Wolfe, C.R. LLaMA-2 from the Ground Up. 2023. Available online: https://cameronrwolfe.substack.com/p/llama-2-from-the-ground-up (accessed on 7 June 2024).
11. Driess, D.; Xia, F.; Sajjadi, M.S.M.; Lynch, C.; Chowdhery, A.; Ichter, B.; Wahid, A.; Tompson, J.; Vuong, Q.; Yu, T.; et al. PaLM-E: An Embodied Multimodal Language Model. In Proceedings of the 40th International Conference on Machine Learning (ICML'23), Honolulu, HI, USA, 23–29 July 2023; Volume 202, pp. 8469–8488.
12. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv* **2023**, arXiv:2201.11903.
13. Béchard, P.; Ayala, O.M. Reducing hallucination in structured outputs via Retrieval-Augmented Generation. *arXiv* **2024**, arXiv:2404.08189.
14. Banerjee, S.; Agarwal, A.; Singla, S. LLMs Will Always Hallucinate, and We Need to Live with This. *arXiv* **2024**, arXiv:2409.05746.
15. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, M.; Wang, H. Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv* **2024**, arXiv:2312.10997.
16. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *arXiv* **2021**, arXiv:2005.11401.
17. Gholami, A.; Kim, S.; Dong, Z.; Yao, Z.; Mahoney, M.W.; Keutzer, K. A Survey of Quantization Methods for Efficient Neural Network Inference. *arXiv* **2021**, arXiv:2103.13630.
18. Bajwa, J.; Munir, U.; Nori, A.; Williams, B. Artificial intelligence in healthcare: Transforming the practice of medicine. *Future Healthc. J.* **2021**, *8*, e188–e194. [CrossRef] [PubMed]

19. Pal, A.; Umapathi, L.K.; Sankarasubbu, M. MedMCQA: A Large-Scale Multi-Subject Multi-Choice Dataset for Medical Domain Question Answering. *arXiv* **2022**, arXiv:2203.14371.
20. Gu, Y.; Tinn, R.; Cheng, H.; Lucas, M.; Usuyama, N.; Liu, X.; Naumann, T.; Gao, J.; Poon, H. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Trans. Health Inform.* **2022**, *3*, 1–23. [CrossRef]
21. Bedi, S.; Liu, Y.; Orr-Ewing, L.; Dash, D.; Koyejo, S.; Callahan, A.; Fries, J.A.; Wornow, M.; Swaminathan, A.; Lehmann, L.S.; et al. A Systematic Review of Testing and Evaluation of Healthcare Applications of Large Language Models (LLMs). *medRxiv* **2024**. [CrossRef]
22. Ge, J.; Sun, S.; Owens, J.; Galvez, V.; Gologorskaya, O.; Lai, J.C.; Pletcher, M.J.; Lai, K. Development of a Liver Disease-Specific Large Language Model Chat Interface Using Retrieval Augmented Generation. *medRxiv* **2023**. [CrossRef] [PubMed]
23. Ramjee, P.; Sachdeva, B.; Golechha, S.; Kulkarni, S.; Fulari, G.; Murali, K.; Jain, M. CataractBot: An LLM-Powered Expert-in-the-Loop Chatbot for Cataract Patients. *arXiv* **2024**, arXiv:2402.04620.
24. Liévin, V.; Hother, C.E.; Motzfeldt, A.G.; Winther, O. Can Large Language Models Reason About Medical Questions? *Patterns* **2024**, *5*, 100943. [CrossRef]
25. Jovanovic´, M.; Baez, M.; Casati, F. Chatbots as Conversational Healthcare Services. *IEEE Internet Comput.* **2021**, *25*, 44–51. [CrossRef]
26. Zhou, H.; Liu, F.; Gu, B.; Zou, X.; Huang, J.; Wu, J.; Li, Y.; Chen, S.S.; Zhou, P.; Liu, J.; et al. A Survey of Large Language Models in Medicine: Progress, Application, and Challenge. *arXiv* **2024**, arXiv:2311.05112.
27. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv* **2023**, arXiv:1910.10683.
28. Chung, H.W.; Hou, L.; Longpre, S.; Zoph, B.; Tay, Y.; Fedus, W.; Li, Y.; Wang, X.; Dehghani, M.; Brahma, S.; et al. Scaling Instruction-Finetuned Language Models. *arXiv* **2022**, arXiv:2210.11416.
29. Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv* **2023**, arXiv:2307.09288.
30. Gao, Y.; Liu, Y.; Zhang, H.; Li, Z.; Zhu, Y.; Lin, H.; Yang, M. Estimating GPU Memory Consumption of Deep Learning Models. In Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual, 8–13 November 2020.
31. Jeon, H.; Kim, Y.; Kim, J.-J. L4Q: Parameter Efficient Quantization-Aware Fine-Tuning on Large Language Models. *arXiv* **2024**, arXiv:2402.04902.
32. Dettmers, T.; Pagnoni, A.; Holtzman, A.; Zettlemoyer, L. QLoRA: Efficient Finetuning of Quantized LLMs. *arXiv* **2023**, arXiv:2305.14314.
33. Xu, Y.; Xie, L.; Gu, X.; Chen, X.; Chang, H.; Zhang, H.; Chen, Z.; Zhang, X.; Tian, Q. QA-LoRA: Quantization-Aware Low-Rank Adaptation of Large Language Models. *arXiv* **2023**, arXiv:2309.14717.
34. Christophe, C.; Kanithi, P.K.; Munjal, P.; Raha, T.; Hayat, N.; Rajan, R.; Al-Mahrooqi, A.; Gupta, A.; Salman, M.U.; Gosal, G.; et al. Med42—Evaluating Fine-Tuning Strategies for Medical LLMs: Full-Parameter vs. Parameter-Efficient Approaches. *arXiv* **2024**, arXiv:2404.14779v1.
35. Han, T.; Adams, L.C.; Papaioannou, J.-M.; Grundmann, P.; Oberhauser, T.; Löser, A.; Truhn, D.; Bressem, K.K. MedAlpaca—An Open-Source Collection of Medical Conversational AI Models and Training Data. *arXiv* **2023**, arXiv:2304.08247.
36. Jin, Q.; Dhingra, B.; Liu, Z.; Cohen, W.W.; Lu, X. PubMedQA: A Dataset for Biomedical Research Question Answering. *arXiv* **2019**, arXiv:1909.06146.
37. Abacha, A.B.; Demner-Fushman, D. A Question-Entailment Approach to Question Answering. *BMC Bioinform.* **2019**, *20*, 511.
38. Hu, T.; Zhou, X.-H. Unveiling LLM Evaluation Focused on Metrics: Challenges and Solutions. *arXiv* **2024**, arXiv:2404.09135.
39. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. BLEU: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 6–12 July 2002; Association for Computational Linguistics: Stroudsburg, PA, USA, 2002; pp. 311–318.
40. Lin, C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*; Association for Computational Linguistics: Barcelona, Spain, 2004; pp. 74–81.
41. Banerjee, S.; Lavie, A. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, 29 June 2005; pp. 65–72.
42. Zhou, J. QOG: Question and Options Generation Based on Language Model. *arXiv* **2024**, arXiv:2406.12381.
43. Wu, J.; Zhu, J.; Qi, Y. Medical Graph RAG: Towards Safe Medical Large Language Model via Graph Retrieval-Augmented Generation. *arXiv* **2024**, arXiv:2408.04187.
44. Singhal, K.; Tu, T.; Gottweis, J.; Sayres, R.; Wulczyn, E.; Hou, L.; Clark, K.; Pfohl, S.; Cole-Lewis, H.; Neal, D.; et al. Towards Expert-Level Medical Question Answering with Large Language Models. *arXiv* **2023**, arXiv:2305.09617.

45. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. A Survey of Large Language Models. *arXiv* **2023**, arXiv:2303.18223.
46. Mhatre, A.; Warhade, S.R.; Pawar, O.; Kokate, S.; Jain, S.; Emmanuel, M. Leveraging LLM: Implementing an Advanced AI Chatbot for Healthcare. *Int. J. Innov. Sci. Res. Technol.* **2024**, *9*, 3144–3151. [CrossRef]
47. Singhal, K.; Azizi, S.; Tu, T.; Mahdavi, S.S.; Wei, J.; Chung, H.W.; Scales, N.; Tanwani, A.; Cole-Lewis, H.; Pfohl, S.; et al. Large Language Models Encode Clinical Knowledge. *Nature* **2023**, *620*, 172–180. [CrossRef]
48. Khalid, S.; Wu, S.; Zhang, F. A Multi-Objective Approach to Determining the Usefulness of Papers in Academic Search. *Data Technol. Appl.* **2021**, *55*, 734–748. [CrossRef]
49. Wang, S.; Jiang, J.A. A Compare-Aggregate Model for Matching Text Sequences. In Proceedings of the International Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017; pp. 1–15.
50. Kumar, L.; Sarkar, S. ListBERT: Learning to Rank E-Commerce Products with Listwise BERT. *arXiv* **2022**, arXiv:2206.15198.
51. Adhikari, A.; Ram, A.; Tang, R.; Lin, J. DocBERT: BERT for Document Classification. *arXiv* **2019**, arXiv:1904.08398.
52. Briskilal, J.; Subalalitha, C.N. An Ensemble Model for Classifying Idioms and Literal Texts Using BERT and RoBERTa. *Inf. Process. Manag.* **2022**, *59*, 102756. [CrossRef]
53. González-Carvajal, S.; Garrido-Merchán, E.C. Comparing BERT against Traditional Machine Learning Text Classification. *arXiv* **2020**, arXiv:2005.13012.
54. Anaby-Tavor, A.; Carmeli, B.; Goldbraich, E.; Kantor, A.; Kour, G.; Shlomov, S.; Tepper, N.; Zwerdling, N. Do Not Have Enough Data? Deep Learning to the Rescue! *AAAI Conf. Artif. Intell.* **2020**, *34*, 7383–7390. [CrossRef]
55. Mass, Y.; Carmeli, B.; Roitman, H.; Konopnicki, D. Unsupervised FAQ Retrieval with Question Generation and BERT. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 807–812.
56. Nogueira, R.; Yang, W.; Lin, J.; Cho, K. Document Expansion by Query Prediction. *arXiv* **2019**, arXiv:1904.08375.
57. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models Are Unsupervised Multitask Learners. *OpenAI Blog* **2019**, *1*, 9.