AI-DRIVEN AGENTIC FRAMEWORK FOR ENTERPRISE SYSTEM TROUBLESHOOTING WITH ENHANCED RAG MODELS

[1]P. Chandrasekhara Reddy, [2]K. Bhargavi, [3]P. Swetha

[1,2,3]Assistant Professor, Department of Computer Science and Engineering,

[1]Vemu Institute of Technology, P Kothakota, [2,3]St. Martin's Engineering College, Secunderabad, Telangana, India

*Corresponding Author

E-mail : kbhargavicse@smec.ac.in

ABSTRACT

Technical troubleshooting in enterprise settings frequently requires navigating various, heterogeneous data sources to effectively resolve intricate issues. This paper introduces an innovative agentic AI solution based on a Weighted Retrieval-Augmented Generation (RAG) Framework designed for enterprise technical troubleshooting. By dynamically weighting retrieval sources, including product manuals, internal knowledge bases, FAQs, and troubleshooting guides according to query context, the framework prioritizes the most pertinent information. It prioritizes product manuals for SKU-specific inquiries while integrating general FAQs for more extensive concerns. The system utilizes FAISS for efficient dense vector search, along with a dynamic aggregation mechanism to integrate results from various sources seamlessly. A LLaMA-based self-evaluator guarantees the contextual precision and assurance of the produced responses prior to their dissemination. This iterative process of retrieval and validation improves precision, diversity, and reliability in response generation.Initial assessments of extensive enterprise datasets indicate the framework's effectiveness in enhancing troubleshooting precision, decreasing resolution durations, and accommodating diverse technical challenges. Future research intends to improve the framework by incorporating sophisticated conversational AI functionalities, facilitating more interactive and intuitive troubleshooting experiences. Efforts will concentrate on enhancing the dynamic weighting mechanism via reinforcement learning to optimize the relevance and accuracy of retrieved information. By integrating these advancements, the proposed framework is set to transform into a holistic, autonomous AI solution, revolutionizing technical service workflows within enterprise environments.

Keywords: Artificial Intelligence, RAG

1. INTRODUCTION

Enterprise technical troubleshooting is a complex, knowledge-intensive process that often re- quires access to a diverse set of resources, including product manuals, FAQs, troubleshooting guides, and internal knowledge bases. These resources are typically scattered across various data silos, making it difficult to extract the most relevant information quickly, particularly in high-pressure, time-sensitive situations.

Traditional approaches to troubleshooting rely heavily on keyword-based search engines or static knowledge management systems that fail to capture the nuanced, context-dependent na- ture of technical issues. For instance, while a keyword-based system might retrieve a generic

FAQ entry for a SKU-specific query, it may overlook a critical entry in the corresponding product manual. Such limitations lead to increased resolution times and suboptimal service delivery.

To address these challenges, we propose a dynamically weighted Retrieval-Augmented Gen- eration (RAG) framework tailored specifically for enterprise technical troubleshooting. Unlike traditional RAG systems that apply uniform retrieval strategies, our approach dynamically ad- justs the weight of each data source based on the query's context and type. This ensures that the system emphasizes highly relevant sources for specific queries while maintaining the flexibility to consider broader resources for less focused inquiries.

1.1 Dynamic Weighting Mechanism

We introduce a dynamic weighting mechanism that assigns context-dependent weights to each data source during retrieval. This ensures that the system prioritizes the most relevant sources for a given query while maintaining the flexibility to incorporate other useful resources.

1.2 Enhanced Retrieval and Aggregation

The framework utilizes FAISS for dense vector search, incorporating a top-K selection mechanism to retrieve the most relevant results. To ensure precision, a threshold is applied to each individual FAISS index, filtering out retrievals that fall below the set threshold and eliminating irrelevant or weakly related matches. This filtering step minimizes the potential for hallucinations by the LLM, thereby improving response accuracy. Following this, dynamic aggregation techniques are employed across the filtered indices to effectively integrate information from multiple sources, ensuring comprehensive and contextually relevant outputs.

1.3 Self-Evaluation for Accuracy Assurance

A LLaMA-based self-evaluation module is integrated to assess the accuracy and contextual relevance of generated responses before presenting them to the user. By employing this self- evaluator model, the system ensures that outputs align closely with the query's intent, delivering consistently high-quality, contextually appropriate answers tailored to user needs.

1.4 Scalability and Versatility

Our framework is engineered to seamlessly adapt to a wide spectrum of enterprise datasets and technical scenarios, emphasizing both scalability and versatility. By leveraging a Facade Pattern for data sources, it simplifies integration: adding a new data source requires only the creation of a dedicated data source class and attaching it to the existing facade. This design minimizes complexity while maximizing extensibility. Furthermore, the data sources can be searched in parallel, ensuring that retrieval processes are both efficient and scalable. This parallel search capability enhances the system's ability to handle large volumes of data, significantly improving performance in complex troubleshooting tasks.

This approach accelerates troubleshooting by enhancing both speed and precision, creating a robust foundation for scalable, AI-driven, and autonomous technical service systems tailored to enterprise environments. The modularity of our framework ensures rapid adaptation to evolving business needs and technological advancements.

2 RELATED WORK

Enterprise troubleshooting has traditionally relied on keyword-based search systems, such as Elasticsearch, which offer efficient solutions for querying structured and semi-structured data. These systems typically leverage ranking algorithms like BM25[Robertson et al., 1994] or TF-IDF

to retrieve results based on term frequency and inverse document frequency, providing relevance scores for exact or near-exact matches. However, while effective in certain contexts, these ap- proaches often fall short when addressing complex, multi-dimensional issues. This limitation arises from their reliance on surface-level keyword matching, without the capacity to incorpo- rate contextual understanding or semantic reasoning. As a result, traditional search mechanisms struggle to handle nuanced queries that require deeper comprehension of the underlying data and its relationships.

The emergence of neural search technologies has led to significant advancements in infor- mation retrieval. Dense retrieval methods, which leverage embeddings to capture the semantic similarity between queries and documents, have proven particularly effective in improving search performance within unstructured and heterogeneous data environments. Pre-trained transformer models, such as BERT [Devlin et al., 2019] and MiniLM [Wang et al., 2020], have been instrumen- tal in facilitating this transformation by enabling more accurate and contextually aware search capabilities. These models, grounded in the Transformer architecture [Vaswani et al., 2017], allow for a deeper understanding of query-document relationships, surpassing the limitations of traditional keyword-based approaches.

FAISS, introduced by Johnson et al. (2017), has emerged as a critical tool for scalable dense retrieval, enabling fast similarity searches across large vector spaces. Its efficiency makes it a preferred choice for systems that utilize dense embeddings. The application of transformer-based embeddings, such as those generated by Sentence-BERT (Reimers & Gurevych, 2019) and GPT- based models (Brown et al., 2020), has further enhanced the precision and effectiveness of these retrieval systems.

Retrieval-Augmented Generation (RAG) has significantly enhanced the capabilities of Large Language Models (LLMs) by integrating document retrieval into the generation process, pro- viding more accurate and contextually relevant responses. By combining dense retrieval with generative models, RAG facilitates end-to-end query resolution by retrieving relevant knowledge from large corpora and generating contextually accurate natural language responses [Lewis et al., 2020; Karpukhin et al., 2020]. Multi-head RAG (MRAG)[Besta et al., 2023]

takes this fur- ther by leveraging multiple attention heads in the Transformer architecture to capture different aspects of data, improving retrieval accuracy for multifaceted queries.

Self-evaluation mechanisms for generative models, such as iterative response refinement, have also gained attention in recent research. Techniques explored by Wang, T et al. [2024] focus on improving reliability and factual consistency. Our system leverages a self-evaluator based on the LLaMA model to assess the accuracy and relevance of the generated outputs. This evaluator suppresses low-quality or irrelevant responses, ensuring that only contextually appropriate and high-quality answers are delivered in real-time.

In parallel with advances in RAG, the application of reinforcement learning to optimize re- trieval pipelines has been explored. Adaptive query representation techniques [Koo, H et al] have shown promising results in fine-tuning retrieval models for specific domains. These meth- ods complement our proposed dynamic weighting mechanism, which adjusts retrieval weights dynamically to address enterprise-specific troubleshooting challenges.

Despite these advancements, the application of RAG to enterprise troubleshooting remains underexplored. Previous research has primarily concentrated on general-purpose information retrieval or open-domain question answering [Alberti et al., 2019; Kwiatkowski et al., 2019]. This work extends existing research by introducing a dynamic weighting mechanism specifically designed for enterprise contexts, integrating diverse data sources, and incorporating threshold- based filtering to mitigate hallucinations in generated responses. The effectiveness of the pro- posed framework is rigorously validated through comprehensive experimentation on technical troubleshooting tasks.

3 METHODOLOGY

3.1 Embedding Model and Retrieval Infrastructure

For generating document and query embeddings, we employ the all-MiniLM-L6-v2 model. This model is selected due to its efficiency and semantic accuracy, which allows for the generation of high-quality, compact embeddings that effectively capture the meaning of the textual data. The all-MiniLM-L6-v2 model excels in encoding both short and long texts into fixed-size vectors, preserving their semantic relationships for effective downstream retrieval.

For the retrieval process, we use FAISS (Facebook AI Similarity Search) as the infrastructure for dense vector search. FAISS enables fast and scalable retrieval by indexing the embeddings generated by the all-MiniLM-L6-v2 model and performing nearest neighbor searches within a high-dimensional space. This system allows for efficient querying of large datasets, ensuring that relevant documents are retrieved quickly and accurately based on their semantic similarity to the input query.

3.2 Dynamic Weighting Strategy

To prioritize the most relevant data sources during retrieval, we implement a dynamic weighting mechanism. For each FAISS index corresponding to a specific data source (e.g., product manuals,

FAQs, troubleshooting guides, internal knowledge bases), a dynamic weight w_k is assigned based on query type and context. This weight adjusts the effective distance $\tilde{D}_{k,i}$ for document i in index k, computed as:

$$\tilde{D}_{k,i} = w_k \cdot D_{k,i}$$

where $D_{k,i}$ is the original FAISS distance score.

Weights w_k are determined using domain knowledge and patterns observed in query behavior. For example, product manuals are assigned higher weights for queries containing SKU identifiers, whereas FAQs may be emphasized for general troubleshooting queries.

3.3 Top-K Selection and Aggregation

The retrieval process begins by computing the adjusted distances for each data source k. Prior to aggregation, a threshold-based filtering mechanism is applied to each data source to remove results with weak matches. This filtering step ensures that only results that meet a minimum rel- evance threshold are retained, effectively reducing the risk of hallucinations in the large language model (LLM). The filtered results for each data source are defined as:

$$T_k = \{i \mid \tilde{D}_{k,i} \text{ satisfies the threshold condition for index } k\}$$

After filtering, the top K results are selected for each data source k:

Tk = {i | D˜k,i is among the top K for index k}

These top results from all sources are then aggregated into a global pool G as follows:

G = [Tk

k=1

Finally, the top K results from the global pool G are selected based on the smallest adjusted distances:

Tfinal = Top-Kmin(G)

This multi-stage selection process ensures both diversity and contextual relevance in the retrieved results, preventing any single data source from dominating and ensuring that under- represented but critical sources are not neglected.

3.4 Response Generation and Self-Evaluation with LLaMA

The final set of retrieved document chunks is provided to a LLaMA-based generative model, along with the user's query, to produce a contextually relevant response. The model synthe- sizes the answer by leveraging the retrieved information, ensuring comprehensive coverage of the query. The generated response is then passed to a LLaMA-based self-evaluator model, which assesses its accuracy and relevance. This evaluation process ensures that the response aligns with both the query and factual expectations. Responses that meet a predefined confidence threshold for accuracy and relevance are considered valid and presented to the user. By incorpo- rating this self-evaluation step, the system minimizes the risk of hallucinations and enhances the overall quality and reliability of the generated output, particularly in technical troubleshooting scenarios.

3.5 SYSTEM ARCHITECTURE

The proposed framework consists of three main components:

1. Preprocessing and Indexing: The data is first preprocessed and chunked based on the data's granularity. The resulting chunks are then embedded using the all-MiniLM-L6-v2 model, which generates high-quality vector representations. These embeddings are indexed using FAISS, a high-performance library for fast and scalable nearest neighbor search, ensuring efficient retrieval based on query similarity.

2. Query Handling: Upon receiving a query, it is embedded using same all-MiniLM-L6-v2 model employed for document embeddings. The resulting query embedding is then matched against the indexed embeddings through a weighted retrieval mechanism. This mechanism adjusts the retrieval process by dynamically prioritizing relevant data sources, taking into account the query's context, type, and specific requirements. By incorporating this dy- namic weighting, the system ensures the retrieval of the most relevant and contextually appropriate information, optimizing the accuracy of the response generation.

3. Response Generation and Validation: The top retrieved chunks are passed to a LLaMA-based generative model, which synthesizes a response based on the query and retrieved information. To ensure the response's accuracy and relevance, it is evaluated by a self-evaluator model, which scores the response against predefined confidence thresholds. Only responses that meet these thresholds are presented to the user, ensuring reliable and contextually accurate outputs.

4 EXPERIMENTAL SETUP

4.1 Dataset Preparation

To evaluate the proposed framework, we constructed a comprehensive dataset comprising diverse enterprise troubleshooting resources:

• Product Manuals: A collection of 1200 detailed technical manuals for various systems and motherboards, indexed based on generations.

• FAQs: 40,000 frequently asked questions covering common troubleshooting scenarios.

• Troubleshooting Guides: Step-by-step resolutions for common issues and error codes.

- Internal Knowledge Bases: Proprietary resources capturing internal expertise and practices.

Each document was preprocessed, chunked into relevant sections, error code tables, or other appropriate segments, tokenized, and embedded. These embeddings were indexed in separate FAISS indices, each representing a unique data source.

4.2 EXPERIMENTAL ENVIRONMENT

The experiments were conducted on a system equipped with:

- Hardware: A server with four NVIDIA A100 GPUs(80GB VRAM each), and two 40-core Intel Xeon processors.

- Software: Python 3.10, PyTorch 2.0, Hugging Face Transformers, and FAISS 1.7.3.

- LLaMA Model: LLaMA-3.1(70B FP16) inference and LLaMA-3.1(70B) self evaluator The framework was deployed as a service using FastAPI, enabling real-time query handling

and response generation.

4.3 Evaluation Metrics

To evaluate the performance of the system, we used the following metrics:

- Accuracy: The percentage of responses that contain correct and contextually relevant information, ensuring that the generated answers align with the user's query.

- Relevance Score: Measures how well the retrieved information aligns with the context of the query, evaluating the pertinence of the returned results.

4.4 Baseline Systems

To benchmark the effectiveness of our proposed framework, we compared two baseline systems:

- Keyword-Based Search: A traditional retrieval system utilizing BM25 to match query keywords against indexed documents. This baseline represents conventional search method- ologies commonly employed in troubleshooting systems.

- Standard RAG: A Retrieval-Augmented Generation (RAG) framework that assigns equal weights to all data sources during the retrieval process. This approach ensures a static and unbiased retrieval mechanism, serving as a comparison point to highlight the benefits of dynamic weighting.

The comparison with these baseline systems highlights the improvements offered by our framework, particularly the enhanced adaptability of the dynamic weighting mechanism, the increased reliability introduced through the LLaMA-based self-evaluation component, and the effectiveness of threshold-based filtering for refining retrieved results.

5 RESULTS AND DISCUSSION

5.1 Performance Analysis

Table 1 provides a comparative summary of system performance across key evaluation met- rics. The proposed framework consistently outperformed the baseline systems, achieving higher accuracy and relevance scores while maintaining competitive efficiency.

Table 1: Performance Comparison of Baseline Systems and Proposed Framework

System	Accuracy (%)	Relevance Score
Keyword-Based Search	76.1	0.61
Standard RAG	85.2	0.75
Proposed Framework	90.8	0.89

5.2 Impact of Dynamic Weighting and Threshold-Based Filtering

The dynamic weighting mechanism and threshold-based filtering significantly enhanced the sys- tem's performance. Dynamic weighting emphasized relevant data sources for specific queries, resulting in higher

accuracy and relevance compared to the standard RAG system. For instance, queries involving SKU identifiers benefited from increased weighting of product manuals, yielding more precise results.

Threshold-based filtering further improved the system's reliability by eliminating low-confidence matches during retrieval. This approach ensured that only highly relevant embeddings were considered, reducing noise and enhancing the quality of the final response. Together, these mechanisms contributed to the framework's robust performance across diverse query scenarios.

5.3 Self-Evaluation Effectiveness

The LLaMA-based self-evaluator played a pivotal role in enhancing accuracy and relevance by suppressing less accurate responses based on a predefined confidence threshold. This mechanism, combined with dynamic weighting and individual index-level threshold filtering, resulted in a 5.6% improvement in accuracy over the baseline RAG framework. These results highlight the effectiveness of integrating generative models with robust validation and filtering strategies to ensure reliability and precision in response generation.

6 CONCLUSION AND FUTURE WORK

This study introduces a dynamically weighted Retrieval-Augmented Generation (RAG) frame- work tailored for enterprise technical troubleshooting. By integrating a context-sensitive re- trieval mechanism with generative self-evaluation, our approach achieves superior accuracy, in- terpretability, and robustness compared to traditional static pipelines. The dynamic weighting strategy ensures retrieval is fine-tuned for query-specific contexts, enhancing adaptability and relevance in diverse troubleshooting scenarios involving heterogeneous data sources.

Our results demonstrate that the proposed system effectively retrieves and synthesizes knowl- edge to deliver high-quality, actionable solutions for enterprise users. The addition of a self- evaluation component further enhances response reliability, enabling iterative refinement and ensuring consistent performance.

Future work will explore:

• Real-Time Learning and User Adaptation: Developing mechanisms for real-time feedback to continuously refine retrieval and generation processes, dynamically adjusting weights and model parameters based on user interactions.

• Conversational Troubleshooting: Extending the framework to support multi-turn conver- sational workflows, enabling iterative problem-solving while managing context and query history.

By focusing on these enhancements, the framework can evolve into a comprehensive solution for complex enterprise technical troubleshooting, adapting to diverse user needs and advancing intelligent enterprise support systems.

REFERENCES

1.   Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., & Gatford, M. (1994). Okapi at TREC-3. Proceedings of the Third Text REtrieval Conference (TREC-3), 109-126.

2.   Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805v2

3.   Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep Self- Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. arXiv preprint arXiv:2002.10957.

4.   Johnson, J., Douze, M., & Jegou, H. (2017). Billion-scale similarity search with GPUs. arXiv preprint arXiv:1702.08734.

5.   Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Lan- guage Processing (EMNLP), 3982–3992.

6.   Brown, T. B., Mann, B., Ryder, N., et al. (2020). Language Models are Few-Shot Learners.

Advances in Neural Information Processing Systems (NeurIPS), 33.

7.   Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., Kiela, D. (2021). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv:2005.11401v4.

8.   Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Petroni, F., Yih, W. (2020). Dense Passage Retrieval for Open-Domain Question Answering. arXiv preprint arXiv:2004.04906.

9.   Besta, M., Kubicek, A., Niggli, R., Gerstenberger, R., Weitzendorf, L., Chi, M., Iff, P., Gajda, J., Nyczyk, P., Müller, J., Niewiadomski, H., Chrapek, M., Podstawski, M., & Hoefler, T. (2023). Multi-Head RAG: Solving Multi-Aspect Problems with LLMs.

10.   Wang, T., Kulikov, I., Golovneva, O., Yu, P., Yuan, W., Dwivedi-Yu, J., Pang, R., Fazel- Zarandi, M., Weston, J., Li, X. (2024). Self-Taught Evaluators. arXiv:2408.02666v1.

11.   Koo, H., Kim, M., & Hwang, S. J. (2023). Optimizing Query Generation for Enhanced Document Retrieval in RAG. arXiv preprint arXiv:2407.12325v1

12.   Alberti, C., Lee, K., & Collins, M. (2019). A BERT Baseline for the Natural Questions.arXiv preprint arXiv:1901.08634v3

13.   Kwiatkowski, T., Palomaki, J., Rhinehart, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Kelcey, M., Devlin, J., Lee, K., Toutanova, K. N., Jones, L., Chang, M., Dai, A., Uszkoreit, J., Le, Q., & Petrov, S. (2019). Natural Questions: A Benchmark for Question Answering Research. Transactions of the Association for Computational Linguis- tics, 7, 129–141.

14.   Vaswani, A., Shazeer, N., & Parmar, N. (2017). Attention is all you need. Proceedings of NeurIPS 2017, 30, 5998–6008.