Technical Customer Service Support with RAG Fine Tunned Llama 3

Jose Della Sala
jo172092@ucf.edu
University of Central Florida
Orlando, Florida, USA

Abstract

Providing effective technical customer service support is a critical challenge for organizations managing complex product ecosystems. This paper explores the application of Retrieval-Augmented Generation (RAG) using a fine-tuned LLaMA 3 model to enhance customer support workflows for Bogen's E7000 system. The project involves creating a custom dataset derived from Bogen's documentation manuals to train the model with domain-specific knowledge of the E7000 system. The objective is to assist customer service representatives by developing an LLM capable of processing technical queries, identifying potential issues within the E7000 system, and proposing solutions or troubleshooting tips. By leveraging the RAG framework, the system dynamically retrieves relevant context from an external knowledge base to augment the model's responses, ensuring scalability and precision. Results demonstrate the feasibility of deploying a fine-tuned LLM to improve query processing efficiency and response accuracy. This work highlights the transformative potential of advanced LLMs in delivering technical customer support in specialized domains.

Keywords

Retrieval-Augmented Generation (RAG), Fine-Tuned Language Models, Technical Customer Support, Bogen E7000 System, Troubleshooting Automation

ACM Reference Format:
Jose Della Sala. 2018. Technical Customer Service Support with RAG Fine Tunned Llama 3. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Technical customer service support is a critical component of delivering a seamless user experience, particularly for complex systems like Bogen's E7000. The E7000 system is an advanced IP-based voice communication and emergency response solution, designed to provide scalable and reliable audio distribution across various appliances. At its core, the system controller orchestrates communication between a wide range of devices, including VoIP speakers, analog station bridges, and other appliances, to deliver robust paging, intercom, and emergency alert functionalities.

Customer service representatives often face challenges in addressing technical queries efficiently due to the system's intricate design and the extensive technical documentation required for troubleshooting. To address these challenges, this project leverages the power of Retrieval-Augmented Generation (RAG) and a fine-tuned LLaMA 3-8B-Instruct model to create an intelligent assistance tool for technical customer support. By incorporating domain-specific knowledge from a custom dataset based on Bogen's documentation manuals, the model is designed to assist representatives in understanding customer queries, identifying potential issues, and providing actionable troubleshooting tips. This reduces the time required to resolve issues and enhances the accuracy of responses.

The system integrates RAG to dynamically retrieve relevant context from an external knowledge base, enabling the model to generate precise and context-aware responses. This approach combines the benefits of a fine-tuned LLM with the scalability of knowledge retrieval, providing a robust solution for handling technical support tasks. The primary objective of this project is to demonstrate how state-of-the-art NLP techniques can be tailored to domain-specific applications, transforming the way technical support is delivered in real-world scenarios.

As a byproduct of creating and streamlining the process for RAG fine-tuning of the LLM, a multi-agent system was developed to facilitate various stages of the pipeline. This multi-agent system consists of the following components:

- A **Question Generator Agent** to generate relevant questions from pieces of context embedded in the knowledge base.
- A **Question Answering Agent** tasked with creating answers based on the generated questions and their corresponding contexts. These two agents form the core of the dataset creation phase.
- An **Evaluation Agent** designed to grade the answers produced by the model during evaluation runs by comparing generated answers to expected ones.
- The central component, the **Fine-Tuned LLM Agent**, which underwent RAG fine-tuning and served as the primary agent for answering technical customer queries.

This multi-agent system not only enhanced the fine-tuning workflow but also contributed to creating a structured and repeatable methodology for training and evaluating large language models in domain-specific applications. Figure 1 illustrates the flowchart of the process followed, starting with the sources for documentation,

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

passing through the dataset creation, the fine tuning and the evaluation and describes how the agents interact with each other to yield the final product.

•

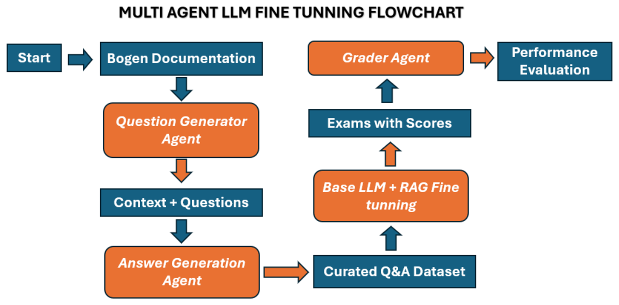

Figure 1: Diagram of the multi-agent system for fine-tuning and evaluating the LLM. The agents streamline the dataset creation, fine-tuning, and evaluation processes.

2 Related Work

The application of Large Language Models (LLMs) in technical customer support and system diagnostics has been explored in several studies, providing a solid foundation for the current work on fine-tuning LLaMA 3-8B-Instruct for Bogen's E7000 system. This section highlights relevant past research and connects it to the present study's contributions.

2.1 LLMs in Technical Customer Support

Wulf and Meierhofer [5] demonstrated how LLMs like GPT-4 could be used to automate repetitive tasks in technical customer support, such as summarizing inquiries and answering frequently asked questions. Their work highlighted the potential for LLMs to reduce the cognitive load on human agents by handling lower-level tasks efficiently. However, they also noted that advanced technical reasoning required further enhancements, such as fine-tuning and integration with domain-specific databases. This study builds on these insights by fine-tuning the LLaMA 3-8B-Instruct model specifically for Bogen's E7000 system, enhancing its ability to address complex, domain-specific queries.

2.2 LLMs for Domain-Specific Chatbots

Kumar and Srivastava [1] explored LLM-based chatbots for answering technical queries and identified limitations in handling multi-step problems and domain-specific information. This work addresses these gaps by employing a Retrieval-Augmented Generation (RAG) approach that dynamically retrieves relevant context from an external knowledge base. By incorporating this framework, the fine-tuned LLM in this study demonstrates improved capabilities in generating precise and context-aware responses for customer service scenarios.

2.3 LLMs in Debugging and Troubleshooting

Kang and Chen [4] investigated the utility of LLMs in automating debugging tasks and troubleshooting system issues. While their work focused on leveraging LLMs for generating plausible debugging hypotheses, it highlighted challenges such as hallucinations and difficulty in diagnosing multi-root causes. This study addresses similar challenges by fine-tuning the LLM using a curated dataset of technical questions and answers derived from Bogen's documentation, ensuring that the model provides reliable and accurate outputs.

2.4 Collaborative Multi-Agent Systems

Zhou and Li [7] introduced a collaborative multi-agent system, D-Bot, for diagnosing complex database systems. By enabling multiple LLMs to collaborate asynchronously, their work demonstrated significant improvements in diagnosing multi-root cause problems. Inspired by this approach, the present study employs a multi-agent system for dataset creation and evaluation. The Question Generator Agent, Question Answering Agent, and Evaluation Agent streamline the fine-tuning and evaluation processes, enhancing the reliability of the trained LLM.

2.5 Prompt Fine-Tuning and Hallucination Mitigation

Wang and He [6] emphasized the importance of prompt engineering and fine-tuning in mitigating hallucinations and improving LLM performance in software engineering tasks. Their findings underscore the value of domain-specific fine-tuning, which is a key focus of this study. By leveraging the SFTTrainer and parameter-efficient fine-tuning techniques such as Q-LoRA, this work optimizes the LLM for handling the unique requirements of Bogen's E7000 system.

2.6 Relevance to the Current Study

The research cited above highlights the growing interest in leveraging LLMs for domain-specific applications. This study extends the body of work by:

(1) Fine-tuning an LLM specifically for the Bogen E7000 system using a custom dataset derived from technical documentation.
(2) Employing a RAG framework to enhance the model's contextual understanding and response generation capabilities.
(3) Implementing a multi-agent system to streamline dataset creation and evaluation, improving the efficiency and scalability of the fine-tuning process.

These advancements address key limitations identified in previous studies, such as handling complex queries, ensuring accuracy, and reducing hallucinations. By combining fine-tuning with RAG and a multi-agent system, this study contributes to the growing field of LLMs for technical customer support and sets the stage for future research in domain-specific NLP applications.

3 Methodology

To develop a Retrieval-Augmented Generation (RAG) fine-tuned language model tailored to Bogen's E7000 system, a systematic process was employed. The methodology involved creating a custom dataset from Bogen's technical documentation, generating a question-and-answer dataset, fine-tuning a language model, and evaluating its performance. This process ensured the model was trained on high-quality, domain-specific data. Below, the steps are outlined in detail.

3.1 Data Extraction from Documentation

Textual content was extracted from various Bogen documentation sources, including the reference manual for the system controller, which sits at the heart of the E7000 system, as well as setup guides and appliance-specific reference manuals for devices such as VoIP speakers and analog station bridges. A Python script was used to process these documents, dividing the content into smaller chunks to meet the context window limitations of the LLaMA 3-8B-Instruct model, which has a maximum context length of 4,096 tokens.

The segmentation of text was designed to keep the context window as short as possible while still retaining meaningful insights about each section. Delimiters were added between chunks to maintain structure and clarity. This approach ensured that the extracted data provided comprehensive and relevant content for training while aligning with the technical constraints of the language model.

3.2 Creation of Q&A Dataset

To generate a high-quality question-and-answer (Q&A) dataset, GPT-4.0 was utilized with carefully crafted prompts. The dataset was designed to ensure relevance, uniqueness, and conciseness in both the questions and answers. Each entry in the dataset contained a unique ID, a question, and its corresponding answer, structured in JSON format to enable seamless integration with the fine-tuning process.

An example of a Q&A entry is shown below:

```
{
    "qas": [
        {
            "id": "00022",
            "is_impossible": false,
            "question": "What type of network connection
     is required for the E7000 web-based UI?",
            "answers": [
                {
                    "text": "The E7000 web-based UI
     requires a secure Hypertext Transfer Protocol Secure
      (https) type network connection to the E7000 system
      server. Users can access the UI using the Google
     Chrome web browser from compatible Windows or Mac
     operating systems, as well as Android-based devices
     ."
                }
            ]
        }
    ]
}
```

Each Q&A entry includes:

- **id:** A unique identifier for the question.
- **is_impossible:** A Boolean flag indicating whether the question has an answer within the provided context.
- **question:** The question derived from the context window.
- **answers:** A list containing the corresponding answer(s) to the question, with each answer represented as text.

The questions were generated using the following prompt:

```
Task:
Generate a set of concise and unique questions based on
     the context above.
Ensure that each question focuses on a distinct aspect of
     the context.
Avoid verbosity or similar questions.
Provide the questions in the following format:

[Question 1]
[Question 2]
[Question 3]
```

The answers were then generated using this prompt:

```
Context:
{context}

Question:
{question}

Task:
Provide a concise answer (maximum of 2-3 sentences).
     Focus on key details only.
```

This iterative process allowed for the creation of a robust dataset that provided comprehensive coverage of the Bogen E7000 system's features and troubleshooting information. The structured JSON format ensured compatibility with the fine-tuning tools and streamlined the training workflow.

3.3 Dataset Preparation

The result of the Q&A generation process was a JSON file that mapped each context to a corresponding set of questions and answers. This JSON file was then parsed into a Pandas DataFrame and transformed into a Hugging Face Dataset, which served as the input for parameter-efficient fine-tuning of the language model.

Each token in the data set followed a structured format, starting with header characters, followed by a prompt, questions, context, and finally the corresponding answer. The initial dataset contained approximately 4,800 rows. A final filtered sample of around 4,400 data points was retrieved, with a token count of less than 2,000 tokens per entry. Entries exceeding 2,000 tokens were discarded as they were considered too lengthy for efficient processing within the LLM's context window limitations. A token count parameter was added to each entry to facilitate filtering and analysis.

The final data set was divided into training, validation and test subsets for fine-tuning. The token distribution of the dataset, as well as its split for training, validation, and testing, is shown in Figure 2.

This structured and concise data set enabled the fine-tuning process to maximize the language model's efficiency and accuracy while adhering to its context window limitations.

3.4 Fine-Tuning with RAG

The fine-tuning process took advantage of the Meta-Llama-3-8B-Instruct model, available at Hugging Face, which is optimized for instruction-based tasks. Parameter-efficient fine-tuning (PEFT) techniques, including Q-LoRA, were used to adapt the model efficiently without requiring the tuning of all model parameters.

To supervise the fine-tuning process, we utilized the SFTTrainer from the Hugging Face trl library. The SFTTrainer simplifies the process of supervised fine-tuning for open large language models (LLMs), making it highly effective for our use case. The LoRA (Low-Rank Adaptation) configuration used during fine-tuning was set

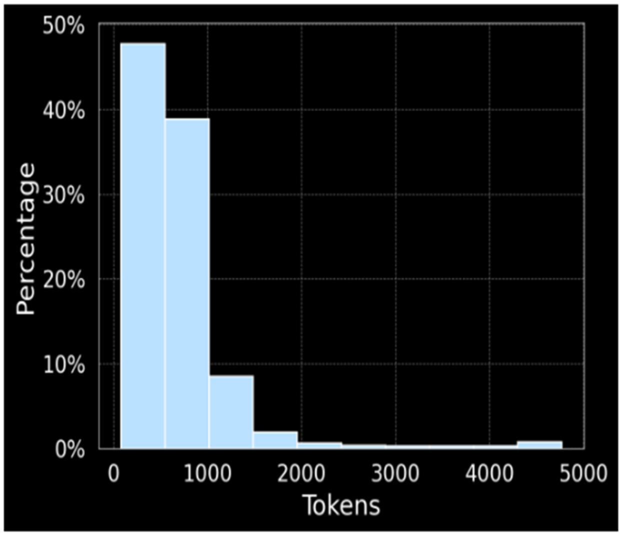

Figure 2: Token distribution in the Hugging Face Dataset and dataset splits.

with a rank of 32, allowing efficient adaptation of the model to the custom dataset while conserving computational resources.

The fine-tuned model was uploaded to Hugging Face Repository for reproducibility and further experimentation.

4 Evaluations

The evaluation of the fine-tuned Meta-Llama-3-8B-Instruct model was performed using two distinct tasks: a quantitative assessment and a qualitative analysis. These evaluations aimed to measure the model's performance on both knowledge-based tasks and real-world customer service scenarios.

4.1 Quantitative Evaluation

The quantitative evaluation focused on testing the model's knowledge of the E7000 system. A comprehensive test set was created, comprising 48 questions based on Bogen's final examination and additional questions generated by a Bogen expert to ensure extensive coverage of the system's features. These questions were presented in various formats, including true/false, multiple-choice (single answer), and multiple-choice (select one or more).

A pipeline was configured to generate answers with specific token limits: 5 tokens for true/false questions and 15 tokens for completion tasks. These limits were set to encourage concise and precise responses, avoiding unnecessary elaboration. The temperature parameter was kept low, with a top_p value of 0.9, ensuring deterministic and focused output.

To evaluate the performance, three different LLMs were tested:

(1) **Base LLM**: The unmodified language model.
(2) **Fine-Tuned LLM (1 epoch)**: A version of the model fine-tuned for one epoch.
(3) **Fine-Tuned LLM (3 epochs)**: A model fine-tuned with additional training for optimized performance.

A total of 7 runs were conducted. Each run consisted of 5 independent experiments or tests, with each experiment consisting of a randomly sampled set of 20 questions from the 48-question battery. This random sampling ensured diversity in the evaluation process and robustness in assessing the models' knowledge.

The generated answers for each test, from each LLM, were graded using another agent with GPT 4.0 as the core LLM of this agent. The grading process involved supplying the grading LLM with a JSON file containing:

- The *generated answers* produced by the LLM.
- The *expected answers* from the test set.

The grading agent was configured to assign a score of 1 for a correct answer (when the generated answer matched the expected answer) and `0` for an incorrect one. The scores for each test were averaged across the 5 experiments within a run, resulting in an overall grade for that run. This grade was a floating-point number between `0.0` and `1.0`, which was then converted to a percentage scale (0–100%) for better interpretability and tabulated for analysis.

The evaluation of the LLM-generated answers was conducted using the following prompt:

```
Given the following:
Generated Answer: {generated}
Expected Answer: {expected}

Grade the answer: Return only '1' if the generated answer
      matches the expected answer, otherwise return '0'.
Instructions: if the generated answer contains the
     correct letter, count it as a good score.
For example:

    "question": "What is the TFTP Server IP address in
     the Appliance Network Setup?",
    "generated_answer": "b. The IP address",
    "expected_answer": "b. The IP address of the Nyquist
     System Controller or Server"
    is Correct because both answers are b.
```

This systematic evaluation framework ensured a rigorous comparison of the base model and the fine-tuned models, providing a detailed quantitative assessment of their respective performance in understanding and reasoning about the E7000 system.

4.2 Qualitative Evaluation

For the qualitative analysis, the model was tested with real-life customer service queries which included issues with the system and general questions about the system. For the system issues the model was instructed to act as a customer service support agent, tasked with diagnosing customer-reported issues and proposing solutions. The following prompt was used to guide the model:

```
You are a customer service representative expert in the
     Bogen E7000 system.\n
Your task is to provide a possible explanation for the
     issue described below and suggest a possible fix.\n\
     n
Customer Query: {query}\n\n
Explanation and Fix:
```

For queries corresponding to general customer questions about the system, seeking guidance on installations or design choices, the LLM was instructed to act as a customer service expert using the following prompt:

```
You are a knowledgeable customer service representative
     specializing in the Bogen E7000 system.
Your task is to provide clear, concise, and helpful
     answers to customer training queries,
offering step-by-step instructions or guidance as needed.

Customer Query: {query}

Response:
- Provide a detailed explanation relevant to the query.
- Include step-by-step instructions, if applicable.
- Mention any key features or tips related to the query.
- Avoid unnecessary information, and ensure clarity and
     precision.

Answer:
```

This evaluation leveraged the main strength that the LLama 3 model has which is text generation. The pipeline was configured for a max of 125 tokens for the output. Empirical experimentation was the method used to arrive to this length of response as optimal since larger values caused the LLM to ramble and hallucinate.

5 Results

5.1 Quantitative Evaluation Results

The results from the quantitative evaluation demonstrate a clear improvement in the performance of the fine-tuned models compared to the base model. The base model achieved an average score of approximately 53.14% across all runs and experiments, which served as the benchmark for comparison. After fine-tuning the model for one epoch, there was a slight improvement, with an average score of 54.71%. While this increase is modest, it indicates that even minimal fine-tuning positively impacts the model's ability to handle domain-specific tasks.

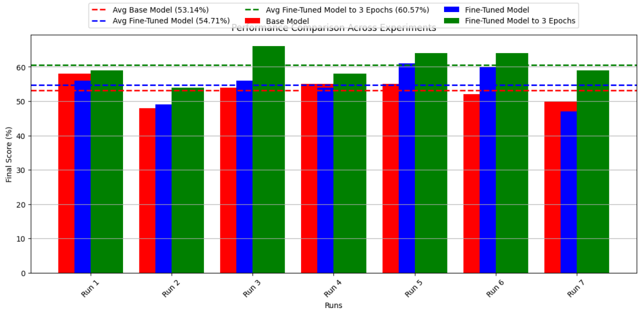

Figure 3: Performance comparison of the Base LLM, Fine-Tuned LLM (1 epoch), and Fine-Tuned LLM (3 epochs) across all runs.

A significant improvement was observed with the fine-tuned model trained for three epochs. This model achieved an average score of 60.57%, representing a considerable performance increase over both the base model and the one-epoch fine-tuned model. These results highlight the benefits of extended fine-tuning, which allows the model to better internalize domain-specific knowledge and patterns. A summary of these results, including the average scores for each model, is provided in Figure 3.

In addition to average performance, the performance differential between the base model and the best-performing model (fine-tuned to three epochs) was analyzed.

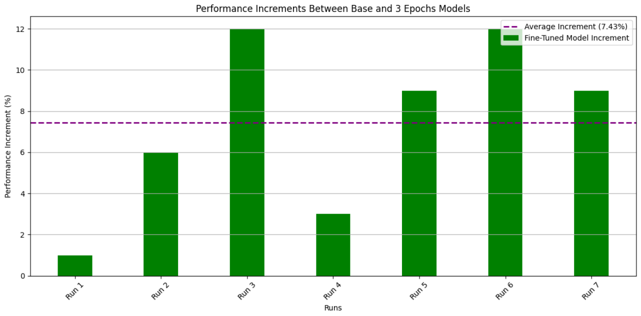

Figure 4: Performance increments between the Base Model and the Fine-Tuned Model (3 epochs) across different runs.

Figure 4 illustrates the performance increments observed across different runs. The largest performance increase in a single run was approximately 12%, while the average increase across all runs and experiments was 7.43%. These findings demonstrate that fine-tuning significantly enhances the model's ability to answer questions accurately, particularly for complex and domain-specific queries.

The results underscore the effectiveness of fine-tuning for optimizing model performance in specific application areas. The quantitative evaluation, as visualized in Figures 3 and 4, provides strong evidence of the improvements gained through iterative fine-tuning and highlights the practical advantages of adapting language models to domain-specific requirements.

5.2 Qualitative Evaluation Results

The qualitative evaluation of the fine-tuned LLaMA-3-8B-Instruct model provided insightful evidence of its improved domain-specific understanding compared to the base LLM. The analysis demonstrated that the fine-tuned model not only delivered accurate responses but also showcased a nuanced understanding of the Bogen E7000 system's components and functionality. Several queries from the test set exemplify this:

- **Enhanced System Diagnosis:** In response to Query 8, *"My DCS attached to the GA10PV is flashing blue, and I cannot make calls. What does this mean?"*, the fine-tuned model correctly identified the issue as a failure of the DCS to register with the E7000 system. It provided actionable steps, such as verifying the network connection and checking the E7000 server logs for error messages. This level of detail and accuracy was absent in the base model's response, which only generalized the issue as a communication failure.
- **Specific Recommendations:** For Query 4, *"How can I set up an E7000 system if I have a school with 35 classrooms? I need to make calls and emergency announcements. What type of appliances do you recommend?"*, the fine-tuned model demonstrated domain-specific expertise by recommending VoIP Speaker Stations for classrooms and VoIP Phone Stations for administrative areas. This guidance aligns with

practical system configurations, illustrating the model's ability to offer relevant and context-aware advice.

- **Reference to Documentation:** In Query 9, *"How do I set up a routine to configure weather alerts to the WBC?"*, the fine-tuned model enhanced its response by pointing the user to the relevant section of the E7000 Server Configuration manual for detailed instructions. This feature highlights the model's ability to leverage external resources effectively, a critical skill for handling technical queries.

The results indicate that the fine-tuned model successfully internalized key concepts and terminology specific to the Bogen E7000 system. This capability, as demonstrated through its responses, makes it a viable tool for improving customer service support by delivering precise and actionable insights. Further qualitative analysis could expand the scope to assess the model's ability to handle more complex scenarios or multi-step instructions.

For a the complete set of code used to generate the results and the result files for both evaluation methods you can refer to the GitHub at the following address: [3]

6 Discussion and Next Steps

The results of this study underscore the effectiveness of fine-tuning a large language model (LLM) using Retrieval-Augmented Generation (RAG) techniques to address domain-specific tasks. The quantitative evaluation highlights that the fine-tuned model trained for three epochs demonstrated a significant improvement over the base model, with an average performance increase of approximately 7.43 percentage points across all runs. This measurable magnitude confirms the impact of the fine-tuning process, as the model consistently outperformed the base model and the one-epoch fine-tuned model in every test. Importantly, the three-epoch model never underperformed compared to the other models, emphasizing the reliability of this training configuration.

Conversely, the model fine-tuned for only one epoch provided minimal performance gains, suggesting that such a shallow fine-tuning process is insufficient for substantial improvement. This outcome supports discarding configurations with insufficient epochs and exploring longer training durations. Given that the training and validation loss after three epochs stabilized around 0.15–0.2, further fine-tuning for five epochs or more could potentially yield additional gains in model performance. These results demonstrate the importance of balancing fine-tuning depth to maximize the model's learning potential without overfitting.

A critical factor in the fine-tuning process lies in the quality of the dataset. The generation of high-quality question-and-answer (Q&A) pairs is both a cornerstone of success and a significant challenge. Writing a dataset that encapsulates comprehensive, accurate knowledge about the Bogen E7000 system requires substantial effort, including expert input and manual review. While using automated agents to generate Q&A pairs is efficient, it often necessitates human oversight to ensure the relevance and precision of the data. One observed challenge was dividing lengthy PDFs into coherent sections for context generation. The complexity of these documents, compounded by irrelevant sections such as trademark disclaimers, frequently resulted in contamination of the input context. This highlights the need for careful manual intervention to produce meaningful, non-redundant context chunks that facilitate the generation of high-quality questions and answers.

The qualitative results further reinforce the effectiveness of the fine-tuned model in handling customer service scenarios. The three-epoch model demonstrated domain knowledge, such as recognizing key issues in system functionality and providing actionable guidance. For example, in Query 8, the model correctly identified a station registration issue as the source of the problem. In Query 4, it recommended appropriate appliances like VoIP speakers and phones for a school setting, and in Query 9, it directed users to specific documentation for additional guidance. These examples highlight the model's capacity to integrate domain-specific knowledge into practical, customer-oriented responses.

Moreover, the three-epoch fine-tuned model shows immense potential as a guidance tool for Bogen's customer service representatives. Its ability to provide quick, accurate answers to queries, perspectives on troubleshooting activities, and actionable suggestions in situations where subject-matter experts are not readily available makes it a valuable asset. By serving as a reliable support system, the model can significantly enhance the efficiency and confidence of customer service teams. The fine tuned model is available for download from Hugging Face at [2]

Despite these promising outcomes, areas for improvement remain. The dataset preparation process requires significant manpower to ensure that context and Q&A pairs are concise and informative. Automating this process further while maintaining high-quality outputs could greatly enhance scalability. Additionally, refining the pipeline settings, such as token limits, temperature, and top_p values, may optimize model performance further, particularly in qualitative tasks.

In summary, the study validates the effectiveness of RAG fine-tuning for improving LLM performance in domain-specific tasks. Future work will focus on the following:

- Increasing the number of training epochs to improve loss metrics and model accuracy further.
- Streamlining and scaling the dataset creation process to generate high-quality Q&A pairs more efficiently.
- Addressing document parsing challenges to ensure cleaner and more relevant context inputs.
- Fine-tuning pipeline settings to enhance both qualitative and quantitative outputs.

These steps aim to build on the current successes and address existing challenges, paving the way for deploying robust, domain-optimized LLMs for technical customer support in real-world scenarios.

References

[1] P. Dwivedi A. Kumar, V. Srivastava. 2023. Large-Language-Models (LLM)-Based AI Chatbots: Architecture, In-Depth Analysis and Their Performance Evaluation. In *AI in Technical Systems*. Springer, 220–230. https://doi.org/10.1007/978-3-031-53085-2_20

[2] Jose Della Sala. 2024. Llama-3-8B-Instruct-E7000-Final-3Epochs-RAG. https://huggingface.co/josedellasala/Llama-3-8B-Instruct-E7000-Final-3Epochs-RAG. Accessed: 2024-12-03.

[3] Jose Dela Sala. 2024. llm-nyquist. https://github.com/josedella/llm-nyquist.git. Accessed: 2024-12-04.

[4] Shin Yoo Jian-Guang Lou Sungmin Kang, Bei Chen. 2023. Explainable Automated Debugging via Large Language Model-driven Scientific Debugging. *arXiv preprint arXiv:2304.02195* (2023). https://arxiv.org/abs/2304.02195

[5] Jochen Wulf and Juerg Meierhofer. 2024. Utilizing Large Language Models for Automating Technical Customer Support. *arXiv preprint arXiv:2406.01407* (2024). https://arxiv.org/abs/2406.01407

[6] Eya Ben Charrada Yacine Majdoub. 2023. Debugging with Open-Source Large Language Models: An Evaluation. *arXiv preprint arXiv:2409.03031* (2023). https://arxiv.org/abs/2409.03031

[7] G. et al Zhou, X. Li. 2024. D-Bot: Database Diagnosis System using Large Language Models. *arXiv preprint arXiv:2312.01454* (2024). https://arxiv.org/abs/2312.01454