Technical Customer Service Support with RAG Fine Tuned LLaMA 3

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

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.

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, 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.

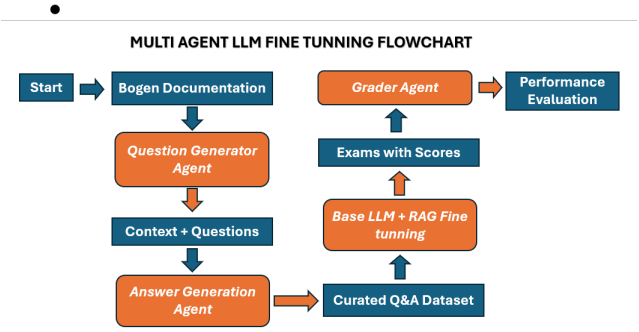

Figure 1: Diagram of the multi-agent system for fine-tuning and evaluating the LLM. The agents streamline the dataset creation, fine-tuning, and evaluation processes.

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.

LLMs in Technical Customer Support

Wulf and Meierhofer (Wulf and Meierhofer 2024) 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.

LLMs for Domain-Specific Chatbots

Kumar and Srivastava (Kumar 2023) 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.

LLMs in Debugging and Troubleshooting

Kang and Chen (Sungmin Kang 2023) 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.

Collaborative Multi-Agent Systems

Zhou and Li (Zhou 2024) 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.

Prompt Fine-Tuning and Hallucination Mitigation

Wang and He (Yacine Majdoub 2023) 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.

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.

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.

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.

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.

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.

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 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.

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.

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.

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.

Results

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.

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 2.

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 3 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.

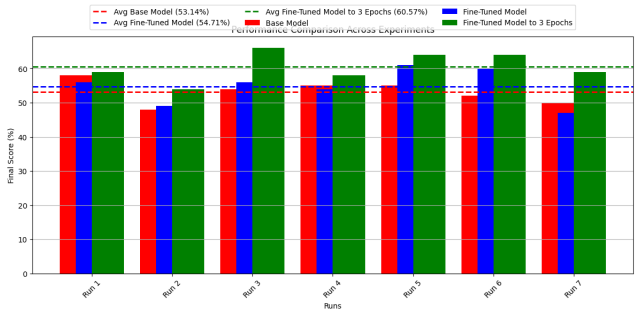

Figure 2: Performance comparison of the Base LLM, Fine-Tuned LLM (1 epoch), and Fine-Tuned LLM (3 epochs) across all runs.

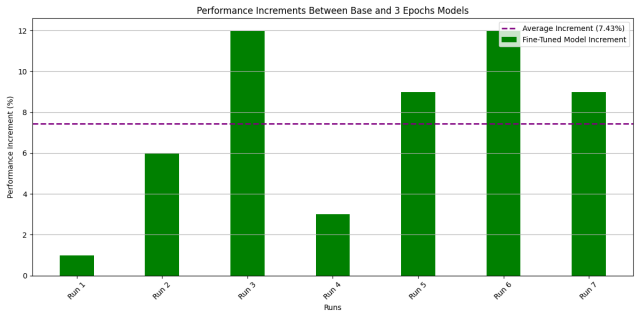

Figure 3: Performance increments between the Base Model and the Fine-Tuned Model (3 epochs) across different runs.

The results underscore the effectiveness of fine-tuning for optimizing model performance in specific application areas. The quantitative evaluation, as visualized in Figures 2 and 3, provides strong evidence of the improvements gained through iterative fine-tuning and highlights the practical advantages of adapting language models to domain-specific requirements.

Qualitative Evaluation Results

To evaluate the effectiveness of domain-specific fine-tuning, we conducted a human-in-the-loop qualitative assessment of responses generated by the fine-tuned LLaMA-3-8B-Instruct model compared to the base LLM. This evaluation was conducted on a real-world dataset of 48 customer queries related to the Bogen E7000 system. Each query consisted of a subject line and a detailed problem description.

Scoring Criteria We used a two-dimensional scoring framework, each on a scale of 1 to 3:

- **Domain Knowledge:** The degree to which the model demonstrated understanding of the Bogen E7000 system, including system components, terminology, device behavior, and operational context.

- **Usefulness:** The clarity, relevance, and actionability of the answer. A high score indicates that the response provided a helpful and implementable solution.

Each model's response was manually scored by a subject matter expert, and the two scores were summed to produce a *Total Score* out of 6. The results showed that the fine-tuned model consistently outperformed the baseline across all metrics. On average, the fine-tuned model achieved a domain knowledge score of **2.67** compared to **1.71** for the baseline, a usefulness score of **2.79** versus **2.04**, and a total score of **5.46** compared to **3.75** for the base model.

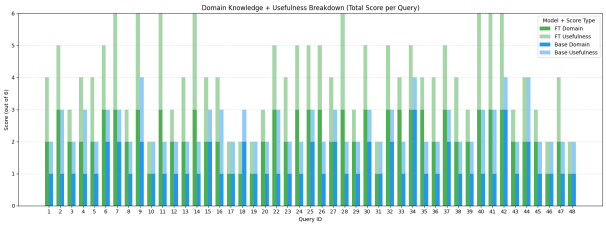

Figure 4: Stacked bar chart comparing total score and domain knowledge across both models.

As shown in Figure 4, the fine-tuned model demonstrated consistently stronger performance in both total score and domain knowledge across almost all of the 48 evaluated queries. Each bar represents the combined score for *domain knowledge* and *usefulness*, with the lower section of each bar specifically highlighting domain understanding.

The chart makes clear that in nearly every case, the fine-tuned model not only achieved a higher total score, but also contributed a significantly larger portion of that score from accurate, system-specific knowledge. Only a couple of responses received equal scores from both models, and in no case did the base model outperform the fine-tuned version. This visual summary reinforces the conclusion that fine-tuning on Bogen E7000 documentation and ticket data leads to meaningful gains in response quality and utility.

Response Behavior and Insights The fine-tuned model demonstrated clear improvements in domain-specific language understanding and problem-solving ability:

- **Entity Recognition and System Concepts:** The fine-tuned model consistently recognized Bogen-specific entities such as VoIP Speaker Stations, DCS, and the GA10PV, and it demonstrated an understanding of call types, extensions, and how these integrate with the web UI. In contrast, the base model frequently defaulted to generic advice related to networking or electrical troubleshooting.
- **Practical and Context-Aware Recommendations:** In responses to deployment planning questions (e.g., Query 4), the fine-tuned model recommended appropriate device configurations based on context (e.g., classroom versus administrative use), while the base model lacked this nuance.
- **Actionable System Diagnosis:** For device error conditions such as flashing indicators or call failures (e.g., Query 8), the fine-tuned model suggested precise steps such as checking SIP registration, server logs, or configuration routines. These responses mirrored common internal support practices and were noticeably more helpful than the baseline model's vague responses.
- **Reference to Documentation:** The fine-tuned model often cited relevant sections of the configuration manual or the web UI navigation path when applicable (e.g., Query 9). This added credibility and reinforced the model's value as a support tool.

Limitations While results are promising, the evaluation was limited by the nature of the dataset: these were live customer queries, which were sometimes informal, incomplete, or ambiguous. Moreover, some issues described by users were edge cases or system bugs not documented in training manuals, especially those introduced by recent feature updates. These factors inherently constrained the model's ability to generate perfect responses in all scenarios.

Conclusion Overall, the evaluation supports the conclusion that fine-tuning significantly enhanced the model's domain-specific capabilities. The model's ability to understand internal system components, generate configuration steps, and reference tools like the web UI reflects its practical utility for customer support. For complete code, data, and evaluation scripts, refer to the project's GitHub repository: (Sala 2024).

Discussion and Next Steps

This work confirms the effectiveness of fine-tuning large language models (LLMs) using Retrieval-Augmented Generation (RAG) for domain-specific technical support. The three-epoch fine-tuned LLaMA-3-8B model consistently outperformed the baseline, achieving a total average score of 5.46 out of 6 versus 3.75 in a human evaluation of 48 real-world customer queries.

A simple two-criteria rubric—*domain knowledge* and *usefulness*—was used to assess response quality. The fine-tuned model demonstrated strong familiarity with Bogen E7000 concepts, including device types, extensions, and web UI references. In contrast, the base model often defaulted to vague, general troubleshooting advice. These findings indicate that fine-tuning successfully internalized product-specific knowledge and improved the practical value of responses.

Looking ahead, future improvements include extending fine-tuning beyond three epochs, refining document parsing, and involving human experts in dataset creation to replace synthetic question generation. Additionally, incorporating internal design documents or configuration code could further enhance performance, though this raises intellectual property concerns. If access is granted, training with an open-source model in a secure local environment may be viable.

These directions support the broader goal of deploying domain-optimized LLMs to assist technical support teams more effectively and reliably in real-world settings.

References

[Kumar 2023] Kumar, V. Srivastava, P. D. A. 2023. Large-language-models (llm)-based ai chatbots: Architecture, in-depth analysis and their performance evaluation. In *AI in Technical Systems*. Springer. 220–230.

[Sala 2024] Sala, J. D. 2024. llm-nyquist. `https://github.com/josedella/llm-nyquist.git`. Accessed: 2024-12-04.

[Sungmin Kang 2023] Sungmin Kang, Bei Chen, S. Y. J.-G. L. 2023. Explainable automated debugging via large language model-driven scientific debugging. *arXiv preprint arXiv:2304.02195*.

[Wulf and Meierhofer 2024] Wulf, J., and Meierhofer, J. 2024. Utilizing large language models for automating technical customer support. *arXiv preprint arXiv:2406.01407*.

[Yacine Majdoub 2023] Yacine Majdoub, E. B. C. 2023. Debugging with open-source large language models: An evaluation. *arXiv preprint arXiv:2409.03031*.

[Zhou 2024] Zhou, X. Li, G. e. a. 2024. D-bot: Database diagnosis system using large language models. *arXiv preprint arXiv:2312.01454*.