Question 3

LLM Fine-tuning Challenge:

Team Name: **DataMavericks**

Table of Contents

1 INTRODUCTION

This report documents the methodology, technical decisions, and implementation of fine-tuning the Qwen 2.5 3B model using Gradient Policy Optimization (GRPO) and integrating it into a Retrieval-Augmented Generation (RAG) system. The objective was to create a specialized system capable of answering questions based on technical AI research papers, blogs, and related documents. This work adheres to the requirements outlined in the hackathon challenge document.

2 PROBLEM STATEMENT

This project addresses the challenge of creating a specialized question-answering system capable of understanding and responding to inquiries related to technical AI research papers, blogs, and associated documentation. The core objective is to leverage and adapt a pre-trained large language model (LLM), specifically Qwen 2.5 3B, to enhance its reasoning capabilities and integrate it into a Retrieval-Augmented Generation (RAG) system. This system will then be deployed in a resource-efficient manner to ensure practicality.

To achieve this overarching goal, the following specific requirements were addressed:

- **Fine-tuning Qwen 2.5 3B to Improve Reasoning**: The initial requirement involved adapting the Qwen 2.5 3B model, either the base or instruct version, to improve its ability to reason when answering questions. Strong reasoning is essential for understanding complex technical topics and extracting relevant information to formulate accurate responses. This fine-tuning process aimed to better align the model's outputs with the expected responses for technical inquiries, significantly enhancing its capacity to provide informed answers.
- **Dataset Management**: To facilitate effective training and evaluation, the challenge included an element of dataset creation/management. Whether generating a synthetic dataset or working with existing data (as we did, using the QASA dataset), the task required transforming this dataset into a usable format for fine-tuning. High quality datasets are paramount to training a usefule model. The dataset was therefore to be processed into JSON format to allow for use with the GRPO process.
- **Quantization for Efficient Deployment**: Given the potential resource constraints of deployment environments, the challenge mandated quantizing the fine-tuned model to a 4-bit .gguf format. Quantization reduces the model's size and computational requirements, enabling efficient deployment on devices with limited memory and processing power without sacrificing too much accuracy. This step is crucial for ensuring the system can be deployed practically in real-world scenarios.
- **Implementation of Retrieval-Augmented Generation (RAG) System**: Central to the project was the creation of a RAG system. This architecture combines the power of information retrieval with the generative capabilities of LLMs. The RAG system retrieves relevant context from a corpus of documents (technical AI research papers, blogs, etc.) and provides this context to the LLM, enabling it to generate more accurate and informed responses. This approach addresses the limitations of LLMs in terms of up-to-date knowledge and factual accuracy by grounding the model's responses in external sources.
- **Documentation and Evaluation**: The challenge emphasized the importance of thorough documentation of all technical decisions made throughout the project. This includes detailing the methodology, the reasoning behind technology choices, and the implementation process. Furthermore, the system's performance was to be rigorously evaluated using appropriate metrics to demonstrate its effectiveness in answering questions based on technical AI documents. This documentation ensures reproducibility and provides valuable insights for future improvements.

3 METHODOLOGY

3.1 Model Selection

- **Qwen 2.5 3B-Instruct**: We opted for the Qwen 2.5 3B-Instruct model over the base model due to its inherent instruction-following capabilities. This characteristic is crucial for question-answering tasks, as it enables the model to better understand and respond to prompts phrased as questions. The 'Instruct' version is pre-trained to align with human instructions, thereby reducing the need for extensive instruction tuning.

- **ColBERTv2**: For the dense passage retrieval component of the RAG system, we selected ColBERTv2 as our embedding model. ColBERTv2's architecture allows efficient similarity search, particularly on GPU, which is essential for rapid retrieval of relevant context from our document corpus. Unlike traditional dense retrieval methods, ColBERTv2 computes fine-grained contextual embeddings for each token in the document and the query, enabling more precise matching.

3.2 Dataset Creation

Instead of generating a synthetic dataset, we leveraged an existing dataset from the QASA (Question Answering with Strategic Answer Selection) repository(https://github.com/lgresearch/QASA/tree/main).

This dataset provides a structured collection of question-answer-reasoning triplets, making it suitable for fine-tuning the Qwen 2.5 3B-Instruct model for improved reasoning capabilities.

- **QASA Dataset Overview**: The QASA dataset comprises questions, corresponding answers, and a 'context' column representing the reasoning behind the answer. We adapted this dataset to align with our fine-tuning objectives.
- **Dataset Acquisition**: The dataset was downloaded directly from the provided QASA repository.

Data Transformation: The raw QASA dataset underwent the following transformations to be compatible with the GRPO fine-tuning process and to better suit the instruction-following paradigm of Qwen 2.5 3B-Instruct:

We utilized 500 entries from the QASA dataset to prepare a specialized training dataset in a structured format. The dataset was designed to focus on *Question-Answer-Reasoning (QAR)* pairs, where each entry includes a question, an answer, and the reasoning behind the answer. Below is the detailed explanation of how the dataset was restructured:

Dataset Preparation Process

1. Original QASA Format:

"0": {

"paper_id": "paper_1",

"title": "Is Reinforcement Learning (Not) for Natural Language Processing?: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization",

"question_id": 1,

"question": "How do these automated metrics for human preferences differ and what factors do they consider when predicting human preferences?",

"question_section": "Introduction",

"question_trigger_sentence": "Automated metrics offer a promising compromise: learned models of human preference like BERTScore (Zhang et al., 2019), BLEURT (Sellam et al., 2020), summarization preferences (Wu et al., 2021) have significantly improved correlation with human judgment compared to earlier metrics (BLEU, METEOR, etc.), and are cheap to evaluate.",

"question_type": "Deep/complex question",

"evidential_info": [

{

"context": "The ultimate aim of language technology is to interact with humans. However, most language models are trained without direct signals of human preference, with supervised target strings serving as (a sometimes crude) proxy. One option to incorporate user feedback is via human-in-the-loop, i.e., a user would be expected to provide feedback for each sample online as the model trains, but this degree of dense supervision is often prohibitive and inefficient. Automated metrics offer a promising compromise: learned models of human preference like BERTScore (Zhang et al., 2019), BLEURT (Sellam et al., 2020), summarization preferences (Wu et al., 2021) have significantly improved correlation with human judgment compared to earlier metrics (BLEU, METEOR, etc.), and are cheap to evaluate. But — these functions are usually not per-token differentiable: like humans, metrics can only offer quality estimates for full generations. Reinforcement Learning (RL) offers a natural path forward for optimizing non-differentiable, scalar objectives for LM-based generation when it is cast as a sequential decision-making problem. However, Goodhart's Law paraphrases: When a measure becomes a target, it ceases to be a good measure. looms: particularly in the case of imperfect metrics that use neural networks, it is easy to find nonsense samples that achieve high-quality estimates. Recent works have shown promising results in aligning LMs to human preferences via RL by constraining preference-based rewards to incorporate notions of fluency (Wu et al., 2021; Ouyang et al., 2022) but progress in this line of work is heavily hindered by a lack of open-source benchmarks and algorithmic implementations—resulting in perception that RL is a challenging paradigm for NLP (Choshen et al., 2020; Kreutzer et al., 2021).",

"rationale": "The automated metrics mentioned in this introductory paragraph are BERTScore (Zhang et al., 2019), BLEURT (Sellam et al., 2020), and Ouyang et al (2022)."

}

],

"composition": "The automated metrics that are mentioned while discussing related work are BERTScore (Zhang et al., 2019), BLEURT (Sellam et al., 2020), and Ouyang et al (2022). More information on these automated metrics, including the differences between them, can probably be gleaned by reading these cited works. The current paper does not contain any additional information about these related automated metrics.",

"arxiv_id": "2210.01241",

"s2orc_url": "https://www.semanticscholar.org/paper/912a39c2e0e4a35747531669cfa952d2c5627729",

"arxiv_url": "https://arxiv.org/abs/2210.01241"

},

2. Conversion to QAR Format:

To prepare the data for training, 500 entries were transformed into a concise format where:

- The Question field was directly taken from the QASA question column.
- The Answer field combined both the reasoning (from the context column) and the actual answer (from the composition column). These were separated by #### to clearly distinguish between reasoning and answer components.

The final format for each entry was:

Q: [QASA 'question' column]

A: [QASA 'context' column - i.e., the reasoning] #### [QASA ‘composition’ column]

3. Purpose of Restructuring:

This format ensures that models trained on this data learn not only to provide accurate answers but also to include reasoning as part of their response.

By separating reasoning and answers with ####, we make it easier for downstream tasks to parse and utilize both components effectively.

3.3 Fine-Tuning Process

1. Training Setup:
   - Fine-tuning was conducted using GRPO to enhance reasoning capabilities.
   - Training was performed on a GPU environment (NVIDIA T4) using Unsloth library

2. Hyperparameter Optimization:
   - Learning rate: 5e-6
   - Batch size: 64
   - Lora_rank : 64
   - Max_seq_len : 1024
   - Epochs: 1
   - Max_steps : 200

3. Quantization:
   - The fine-tuned model was quantized into a 4-bit .gguf format using UnsLoTH, reducing memory usage while maintaining performance.

3.4 Retrieval-Augmented Generation (RAG) System

1. PDF Processing:
   - Used PyMuPDF (fitz) to extract text from uploaded PDFs.
   - Texts were split into chunks for indexing using RecursiveCharacterTextSplitter.

2. Embedding Storage:
   - Each text chunk was embedded using ColBERTv2's average pooling mechanism.
   - Embeddings were stored as tensors for similarity computation during retrieval.

3. Retrieval Mechanism:
   - User queries were encoded into embeddings using ColBERTv2's tokenizer.
   - Cosine similarity scores between query embeddings and stored embeddings determined the most relevant chunks.

4. Answer Generation:
   - Retrieved context chunks were fed into Qwen 2.5 along with user questions.
   - A system prompt guided response formatting to ensure clarity in reasoning and answers.

3.5 User Interface

A Gradio-based UI was developed for seamless interaction:

- PDF upload functionality for document processing.
- Continuous chat interface for querying and response generation.
- Loading indicators provided real-time feedback during processing.

4 KEY INNOVATIONS

1. Fine-tuning Qwen 2.5 with GRPO significantly enhanced its reasoning capabilities over the base model.
2. Efficient RAG implementation leveraging ColBERTv2 embeddings ensured high retrieval accuracy.
3. Quantization enabled deployment in low-resource environments without compromising performance.

5 CHALLENGES AND SOLUTIONS

Challenges

1. Memory Constraints During Fine-Tuning:

   Fine-tuning large datasets often leads to out-of-memory (OOM) errors, especially on GPUs with limited VRAM.

   Managing memory efficiently becomes critical for tasks requiring large context sizes or high precision.

2. GPU Runtime Limitations:

   Kaggle provides GPU runtimes for up to 9 hours, while Colab Pro+ extends this to 24 hours. However, these runtimes are often insufficient for large-scale fine-tuning tasks3.

   Running GGUF models from llama.cpp has inconsistent GPU utilization. For instance:

   It works well in Kaggle environments but defaults to CPU in Colab due to lack of proper GPU offload support

3. Quantization Challenges:

   Maintaining model performance after quantization is difficult, as smaller models may lose reasoning capabilities. Experimenting with techniques like UnsLoTH has shown promise in preserving reasoning.

4. Hallucinations in Smaller Models:

   Smaller models tend to hallucinate more frequently, generating incorrect or non-existent information. This is a common issue when reasoning capabilities are limited.

Solutions

1. Memory Management:

   Use parameter-efficient fine-tuning methods like LoRA or QLoRA to reduce memory usage during training by up to 43%[7].

   Implement techniques such as DeepSpeed's model parallelism and tensor parallelism to distribute memory usage across multiple GPUs.

2. Addressing GPU Runtime Limitations:

   Optimize workflows by leveraging Kaggle for GPU-intensive tasks where runtime limits are less restrictive.

   For Colab environments, ensure CUDA is properly installed and configured.
   Compile llama.cpp with GPU BLAS support using the GGML_CUDA flag.

3. Improving Quantization Techniques:

   Experiment with advanced quantization methods like Unsloth to minimize performance degradation while maintaining reasoning capabilities

6 CONCLUSION

This project successfully fine-tuned Qwen 2.5 with GRPO, implemented a robust RAG system, and delivered a user-friendly interface that met all hackathon requirements. The innovations in reasoning enhancement, retrieval accuracy, and quantization efficiency make this solution highly competitive for addressing AI research QA challenges.