Fine-Tuning a Pirate-Persona Language Model Using LoRA

Objective:

The primary objective was to adapt the `Qwen/Qwen2.5-0.5B-Instruct` language model to suit a consistent and fluent pirate persona. In order to achieve this stylistic transformation, it was critical to preserve the model's core capabilities, including factual accuracy, coding proficiency, and logical reasoning.

To meet this objective, we employed a *Parameter-Efficient Fine-Tuning (PEFT)* strategy, specifically *Low-Rank Adaptation (LoRa)*. This method was chosen over full fine-tuning to mitigate the risk of catastrophic forgetting and to ensure computational efficiency.

Data Preprocessing and Augmentation

Based on the `pirate_dataset.csv` provided, the following steps were taken to enhance the model's performance and robustness:

- **Robust Data Loading**:

  A robust method was used to parse CSV data to ensure that special characters (e.g., commas and quotes within text fields) are handled correctly.

- **Dataset Augmentation**: The initial dataset was augmented with additional prompt-response pairs to diversify the training data. This included more complex coding questions, as well as a broader range of general knowledge questions.

- **Preservation-focused Augmentation**: A set of standard, non-pirate questions-and-answer pairs were also added to the dataset (e.g., "What is the capital of Germany?" -> "Berlin."). In this way, the model did not overfit on the pirate style and was able to retain its original knowledge.

- **Data Formatting**:

  The final dataset was formatted to match the model's required chat template (e.g., `<|im_start|>user\n{prompt}<|im_end|>\n<|im_start|>assistant\n{response}<|im_end|>`). It ensures that the model processes the data according to its pre-training format in the best way possible.

Training Setup

We trained the model using a GPU accelerator in Google Colab. To manage memory, 4-bit quantization was used.

Here are the key training hyperparameters:

- **Base Model**: `Qwen/Qwen2.5-0.5B-Instruct`
- **Fine-Tuning Technique**: LoRA (Low-Rank Adaptation)
- **Quantization**: 4-bit using BitsAndBytesConfig (nf4 type)
- **LoRA Rank (r)**: 16
- **LoRA Alpha (lora_alpha)**: 32
- **Target Modules**: LoRA was applied to the attention mechanism's linear layers (`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`).
- **Batch Size**: 2 (per device) with a gradient accumulation of 4.
- **Learning Rate**: 2e-4
- **Epochs**: 3

Techniques Used to Preserve Core Functionality

In order to ensure that the base model's intelligence was preserved, the following techniques were used:

1. **Low-Rank Adaptation (LoRA)**:

   In order to protect the learned knowledge from catastrophic forgetting, we froze the weights of the original model and trained only the small, low-rank adapter matrices.

2. **Strategic Dataset Augmentation**:

   Non-pirate factual data were included in the training set to reinforce the model's original knowledge base and prevent it from being fully overwritten by the new persona.

3. **Qualitative Stress-Testing**:

   A series of challenging prompts were used to evaluate the model after training. Based on these tests, the model still has the ability to perform logical reasoning, handle ambiguity, and even correct factual errors in a user's prompt, demonstrating that it has retained its core functionality.

Overall Assessment

The fine-tuning was a partial success that perfectly highlights the trade-offs of this process.

In terms of creativeness, consistency, and robustness within the target persona, the model was successful. This came at a price, however. A slight degradation in its core capabilities was caused by the training, specifically its ability to follow negative constraints and its ability to handle factual inaccuracies nuancedly.