InterviewMate: Domain-Specific Interview Coaching with Fine-Tuning Falcon

1 Introduction

The capacity to fine-tune large language models (LLMs) for specific use domains is a valuable commodity, now made possible by the advent of generative AI. This document introduces Interview-Mate, an LLM that has been fine-tuned for helping users prepare for technical interviews for the domain of data engineering. The purpose of this project is not merely to improve response correctness and usability for interview environments but also comprehend the limits of low-resource fine-tuning with LoRA on consumer hardware.

2 Methodology and Approach

2.1 Dataset Preparation

We constructed a new dataset of approximately 150 question-answer pairs chosen from real-world data engineering interview questions. The dataset was gathered in JSON format with a joint prompt-answer format stored in the 'text' field. The pre-processing pipeline consisted of removing duplicates, phrasing normalizing, cutting long samples, tokenizing, and padded using Hugging Face's tokenizer API.

The data were split into three sets:

- Training set (80%): Model learning
- Validation set (10%): Hyperparameter adjustment and overfitting avoidance
- Test set (10%): Model assessment

This tidy and well-balanced structure served to ensure model stability in training and inference.

2.2 Model Selection

The selected pre-trained model was tiiuae/falcon-rw-1b, chosen for a trade-off between quality and support for CPU-only environments. We ensured compatibility by resizing the tokenizer to insert a padding token when absent. This model enabled efficient low-mem and low-compute fine-tuning.

2.3 Fine-Tuning Setup

We used the PEFT library to do Low-Rank Adaptation (LoRA) of the Falcon model.

Key config:

- $r = 8$, $\alpha = 16$
- Target modules: query key value
- LoRA dropout: 0.05

The model was trained using Hugging Face Trainer API with the following settings:

- Epochs: 5
- Batch size: 1 (using gradient accumulation)
- Learning rate: $2e^{-4}$
- CPU-only run with log and checkpointing

The trained LoRA adapter was saved in the left model directory.

2.4 Hyperparameter Tuning

For optimal performance, three configs were executed:

A. Learning rate $2e^{-4}$, 3 epochs

B. Learning rate $2e^{-4}$, 5 epochs (best selected)

C. Learning rate $3e^{-4}$, 5 epochs

Each run was measured by the ROUGE-L score and qualitative analysis.

3 Results and Analysis

3.1 Metrics of Evaluation

We used ROUGE-L to measure similarity between reference responses and generated responses:

- Baseline Falcon: ROUGE-L ~ 0.32
- Fine-tuned InterviewMate: ROUGE-L ~ 0.56

3.2 Qualitative Comparison

The fine-tuned model showed:

- Enhanced coherence with technical terminology
- More rational organization of responses
- Improved control of open-ended or multi-part questions

On the other hand, the base model produced generic or vague responses. This shows the importance of domain-specific data.

4 Error Analysis

4.1 Issues Found

There were mostly observed errors in:

- Open behavior questions
- Abstract or theoretical problem-solving scenarios

Example:

Tell me about a time when you fixed a data pipeline failure.

The model produced generic responses without experience-based nuance or depth.

4.2 Root Causes

- Lacking behavior training examples
- Over-reliance on technical terminologies
- Repeated prompting without variance

4.3 Proposed Improvements

- Augment training data with synthetic behavior QA
- Include more diverse phrasings and settings
- Implement early stopping on validation loss

5 Inference Pipeline

We included a simple Python interface for inference. The model is initialized as follows:

```
from transformers import AutoTokenizer, pipeline
from peft import PeftModel
model = PeftModel.from_pretrained(.)
tokenizer = AutoTokenizer.from_pretrained(.)
pipeline = pipeline("text-generation", model=model, tokenizer=tokenizer)
```

This enables interactive user input and CPU inference with average latency of ~ 5 seconds.

6 Lessons Learned

- LoRA performs extremely well in fine-tuning in sparse environments
- The phrasing of the prompt has a large impact on response quality in models
- Incremental experimentation optimizes hyperparameters effectively
- Small datasets can also provide robust domain-specific outputs

7 Limitations

- Weak generalization to out-of-domain prompts
- Limited scope in behavioral responses in terms of creativity
- Evaluation is not diverse with respect to metrics beyond ROUGE

Future releases can employ BLEU, METEOR, and qualitative human evaluation.

8 Conclusion

InterviewMate demonstrates the power of fine-tuning large models with LoRA for narrow use cases like interview practice. From relatively humble compute and data, we saw actual response quality, accuracy, and fluency improvements. Future work will add behavioral capability, generalization, and UI deployment.

9 References

- Hugging Face Transformers: https://huggingface.co/docs/transformers/
- PEFT: https://github.com/huggingface/peft
- Falcon Model Card: https://huggingface.co/tiiuae/falcon-rw-1b
- ROUGE Metric: https://huggingface.co/metrics/rouge