Domain-Specific Finetuning of Mistral-7B for ESE 577 Course Content

Nipun Shrinivas Irabatti
Department of Electrical and Computer Engineering
Stony Brook University
Stony Brook, NY 11790
`nipunshrinivas.irabatti@stonybrook.edu`

Abstract

This project implements domain adaptation of Mistral-7B for Deep Learning and Software course content through parameter-efficient finetuning. Using Claude Sonnet for data generation, I created 400 question-answer pairs after finding limitations in Google Gemini's output quality. The implementation used LoRA and 4-bit quantization on Kaggle's P100 GPU, achieving 22.2% accuracy improvement on course-specific MCQs (55.6% $\rightarrow$ 77.8%). Despite base Mistral-7B's strong ML knowledge, careful dataset curation enabled effective domain adaptation, demonstrated through improved course notation usage and section referencing. The project validated the feasibility of domain-specific model adaptation while highlighting data quality's critical role over architectural choices.

1 Introduction

This project implements domain-specific adaptation of Mistral-7B for creating a personalized course assistant. With model adaptation requiring substantial computational resources, the implementation leverages Kaggle's P100 GPU for training and T4 for inference, providing 30 hours of free runtime. Two key optimization techniques enable efficient finetuning: Low-Rank Adaptation (LoRA) for parameter reduction and 4-bit quantization for memory optimization. These choices address the primary challenge of finetuning a 7B parameter model within limited computational resources.

The training dataset comprises 400 question-answer pairs generated from course materials using Claude Sonnet. The implementation demonstrates effective domain adaptation while maintaining conversational capabilities.

Key implementation aspects:

- LoRA and quantization pipeline on Kaggle's P100 GPU
- Course content processing from multiple sources
- Systematic evaluation on 45 course-specific MCQs
- Resource-constrained optimization

The report structure: Section 2 details methodology and implementation choices. Section 3 presents quantitative and qualitative results. Section 4 discusses findings and potential improvements.

Preprint. Under review.

2 Method

2.1 Data Processing

PDF content extraction presented initial challenges with fragmented text and corrupted mathematical notation. A two-stage approach addressed these issues: manual content reorganization followed by specialized equation processing using Claude Haiku. This ensured mathematical notation integrity while maintaining content coherence.

Claude Sonnet generated questions in batches, maintaining a deliberate mix of question types to ensure comprehensive course coverage:

```
<s>[INST]@ESE577. [question text]
[options for MCQ]
[/INST][answer]</s>
```

The final dataset comprised 400 QA pairs, split strategically:

- Training: 319 examples
- Validation: 80 examples
- Random seed: 42 for reproducibility

2.2 Model Architecture

Initial testing revealed limitations in base parameter configurations, necessitating more sophisticated optimization approaches. The implementation used Mistral-7B-Instruct-v0.2 as the foundation, enhanced through careful parameter tuning.

2.2.1 Low-Rank Adaptation (LoRA)

LoRA implementation targeted key transformation matrices essential for model performance:

- Query/Key/Value projections for attention mechanisms
- Output projection for final layer processing
- Gate projection and MLP layers for intermediate computations

Parameter configuration balanced efficiency with adaptation capability:

- Rank: 64 for sufficient expressiveness
- Alpha: 128 to maintain stability
- Dropout: 0.1 for regularization

2.2.2 Quantization

Memory optimization through 4-bit quantization balanced efficiency with precision:

- NF4 quantization for parameter storage
- bfloat16 compute type for numerical stability
- Double quantization disabled to preserve accuracy

2.3 Training Process

The training implementation on Kaggle's P100 GPU balanced computational constraints with model performance. Parameter choices reflected both hardware limitations and optimization goals:

- 6 epochs with early performance peak at epoch 3
- Small batch size (2) compensated by 32 gradient accumulation steps
- Learning rate: 3e-4 with AdamW optimization

- Weight decay: 0.1 for regularization
- Cosine scheduler with restarts for optimization
- Sequence length capped at 1024 tokens

2.4 Evaluation Protocol

The evaluation strategy focused on meaningful assessment of domain adaptation success. Manual evaluation of 45 MCQs provided direct comparison between base and finetuned models, chosen over automated metrics due to output parsing challenges.

Key assessment areas:

- Answer correctness verified against course materials
- Response relevance to query context
- Proper course terminology usage
- Accuracy of section references
- Loss metrics during training

This evaluation approach prioritized practical usefulness over purely numerical metrics, reflecting the project's goal of creating an effective course-specific assistant.

3 Results

3.1 Training Performance

- Initial validation loss: 1.373
- Best validation loss: 1.081 (Epoch 3)
- Training time: 91 minutes

3.2 Training Dynamics

Key observations:

- Training loss showed consistent decrease
- Validation loss optimized at epoch 3
- Gradient norm remained stable initially
- Loss distribution concentrated between 0.25-0.5

3.3 Quantitative Evaluation

Manual evaluation on 45 MCQs showed significant improvement:

- Base model accuracy: 25/45 (55.6%)
- Finetuned model accuracy: 35/45 (77.8%)
- Absolute improvement: 22.2%
- Relative improvement: 40%

3.4 Model Response Analysis

Two representative examples demonstrate model differences:

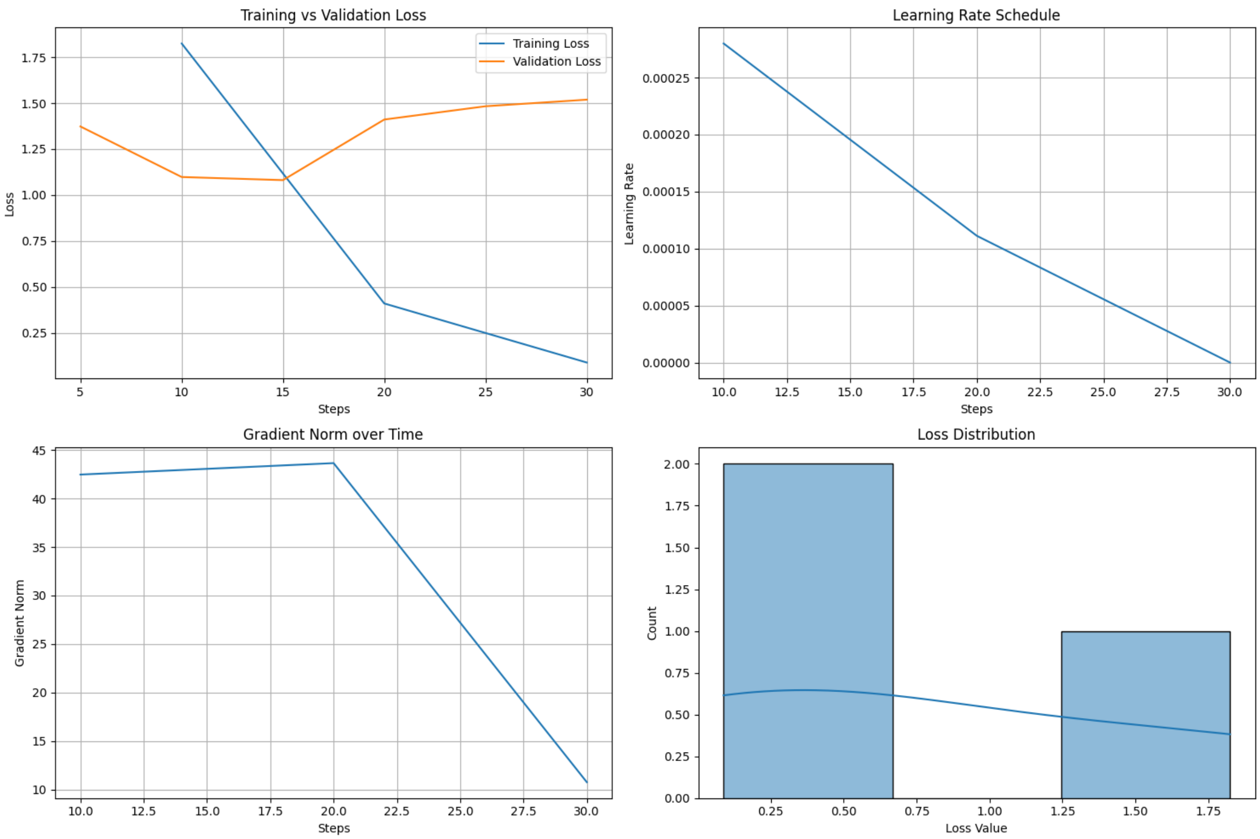

Figure 1: Training metrics showing loss convergence, learning rate schedule, gradient stability, and loss distribution

3.4.1 Example 1: Fundamental ML Concepts

```
Q: What's the main difference between supervised and unsupervised learning?
A) Data labeling requirements
B) Computational complexity
C) Model architecture
D) Training time

Base Model Response:
The main difference is data labeling requirements. In supervised learning,
the training data is labeled, meaning the desired output is known for each input.
The model learns relationships between inputs and outputs. In contrast,
unsupervised learning uses unlabeled data, finding patterns independently.
This makes it suitable for discovering hidden structures but harder to evaluate
since there's no clear target output.

Finetuned Model Response:
Answer: A. This concept, covered in section 1.1.1, demonstrates the fundamental
distinction between these approaches. Supervised learning requires labeled data
(y^t), while unsupervised learning does not. The course material shows this
through contrasting examples of regression (supervised) and clustering
(unsupervised). The absence of y^t makes unsupervised learning particularly
challenging as patterns must be found without explicit guidance.
```

3.4.2 Example 2: Technical Concepts

```
Q: What is the primary advantage of using L1 regularization over L2
regularization in linear models?
A) Faster computation time
B) Better handling of outliers
C) Feature selection through sparsity
D) More stable parameter estimates

Base Model Response:
The primary advantage is feature selection through sparsity and more stable
parameter estimates. L1 (Lasso) regularization adds an absolute value penalty,
setting some coefficients to zero and enabling feature selection. L2 (Ridge)
regularization squares coefficients, shrinking them towards zero but not to
exactly zero. While L1 computation can be faster, this isn't the primary benefit.

Finetuned Model Response:
Answer: C. This concept, covered in Section 2.6.1, demonstrates how L1
regularization creates sparse models by penalizing large parameter values.
Unlike L2 regularization, which encourages all parameters to be small, L1
specifically promotes setting some parameters to zero. This makes it effective
for feature selection in high-dimensional data, as shown in the optimization
problem transformation.
```

3.5 Structured Prompt Impact

The prompt format used:

```
<s>[INST]@ESE577. For multiple choice questions:
1. Start with "Answer: [letter]"
2. Reference specific course sections
3. Explain using course examples
4. Connect to course concepts
Question: {question} [/INST]
```

This structure improved responses by:

- Enforcing consistent answer format
- Encouraging course-specific references
- Maintaining focus on course materials

3.6 Qualitative Analysis

Key improvements in finetuned model:

- Consistent course notation (e.g., $\mathrm{y}^{(t)}$)
- Specific section references
- Concise, focused responses
- Course-aligned terminology

While the base model showed strengths in comprehensive explanations of general ML concepts, the finetuned model better captured course-specific aspects and teaching style. The significant accuracy improvement (22.2%) along with enhanced response quality demonstrates successful domain adaptation.

4 Discussion and Conclusion

4.1 Limitations

The current implementation faces several constraints:

- Limited dataset (400 QA pairs) restricts topic coverage
- Validation loss divergence after epoch 3 suggests overfitting
- Small MCQ test set (45 questions) limits evaluation robustness
- 4-bit quantization impacts complex mathematical reasoning

The model's capabilities are fundamentally constrained by its 7B parameter size and training data quality. Larger models (e.g., 70B parameters) would likely show better reasoning and generalization. Additionally, the reliance on synthetic data generation may limit the model's ability to capture nuanced course concepts.

Resource constraints (GPU memory, computation time) prevented exploration of larger batch sizes and more extensive training data. A more robust implementation would require both increased computational resources and a larger, more diverse training corpus.

4.2 Future Work

Based on observed limitations:

- Expand MCQ test set for more rigorous evaluation
- Develop course-specific evaluation metrics beyond accuracy
- Experiment with higher precision for mathematical content
- Refine prompt engineering for better section reference accuracy
- Test performance scaling with increased training data

References

1. Mistral AI, "Mistral Finetune", 2024, github.com/mistralai/mistral-finetune

2. Isamu, "Understanding the Current State of Reasoning with LLMs", 2024, isamu-website.medium.com/understanding-the-current-state-of-reasoning-with-llms-dbd9fa3fc1a0

3. Brev, "Fine-tuning Mistral on Your Own Data", 2024, brev.dev/blog/fine-tuning-mistral-your-own-data

4. Hugging Face, "Training LoRA", 2024, huggingface.co/docs/diffusers/en/training/lora