ESE 577 Deep Learning Algorithms and Software Final Project Report

Ningyuan Yang, Yiti Li, Guoao Li
The State University of New York at Stony Brook
{ningyuan.yang, yiti.li, guoao.li}@stonybrook.edu

Abstract

In the final project, we fine-tune the Mistral-7B model on the ESE 577 course syllabus to develop a domain-specific chat-bot which can answer course-related questions. We apply the low-rank adaptation (LoRA) method for efficient fine-tuning and load the model with 4-bit precision to enhance memory efficiency. A dataset of 100 question-answer (QA) pairs is manually generated based on the syllabus and used for fine-tuning. Hyperparameters are configured according to a Kaggle tutorial and the best model is selected based on the lowest validation loss. Experimental results demonstrate that our chat-bot generates correct and relevant answers compared to the ground truth. Finally, we discuss the limitations of our approach and propose potential solutions to further improve the model's performance.

1 Introduction

The emergence of large language models (LLMs) [1] such as GPT [2], LLaMA [3], and Mistral-7B [4] has attracted significant attention in recent years. These models demonstrate strong performance across a wide range of natural language processing (NLP) tasks, such as text generation [5], text summarization [6], and question answering [7]. To further explore this field, we fine-tune the Mistral-7B model on the ESE 577 course syllabus [8], making it act as a chat-bot that answers the course-related questions.

In our project, we use the LoRA [9] method to make the fine-tuning process computationally feasible. Instead of updating the entire model with high computational costs, LoRA [9] introduces a small number of additional parameters to modify the model, which allows for efficient fine-tuning while retaining the general knowledge encoded in the pre-trained model. Additionally, by reducing the precision of the model weights from 32 bits to 4 bits [10], we significantly reduce computational load and memory usage, making it possible to train on hardware with limited resources. Furthermore, we manually create a small dataset containing 100 QA pairs based on the ESE 577 course syllabus and apply the hyperparameter configuration from a Kaggle tutorial [10] to perform model fine-tuning. The loss functions for training and validation process are both cross-entropy loss, and we select the model with the lowest validation loss as the best model. Experimental results of the generated outputs,

Preprint. Under review.

loss curves, and evaluation scores all demonstrate the success of our fine-tuning process. Finally, we provide a detailed analysis to address the limitations of our approach and propose potential solutions to further improve the results.

The remainder of this report is organized as follows. Section 2 presents details about our methodology. Experimental results are given in Section 3. Discussions are included in Section 4. Section 5 concludes the entire report. Section 6 provides the acknowledgment.

2 Method

2.1 Data Collection

Since the main goal of our task is to fine-tune a LLM and make it work as a chat-bot, we generate 100 QA pairs manually as the training data based on the ESE 577 course syllabus [8]. Here are some examples of the generated QA pairs for training, which are presented in Tab. 1. Furthermore, we split the training data into a training set with 88 QA pairs and a validation set with 12 QA pairs. For the test data, we randomly select 20 pairs out of the training data and paraphrase the questions and answers.

Table 1: Examples of the generated QA pairs for training

Q1: What is the title of this course? A1: ESE 577 — Deep Learning: Algorithms and Software.
Q2: What topics are covered in the first week of the course? A2: The first week covers an introduction to machine learning, regression, and regularization.
Q3: Are students allowed to collaborate on homework assignments? A3: Yes, students can discuss problems with one peer, but must write their own code and solutions.

2.2 Data Processing

In order to make our model understand the relationship between the questions and answers, we re-format the training data according to Eq. (1), where $< s >$ and $< /s >$ indicate the start and the end of the sentences, while $[INS]$ and $[/INS]$ let the model know what question is being asked and what answer should be given in response [10].

$$\textit{input} = < s > [INS] < question > [/INS] < answer >< /s > \tag{1}$$

Another important step in data processing is tokenization, which breaks the entire text into smaller components called tokens. In this project, we employ the pre-trained model's tokenizer from the Hugging Face Transformers library [11]. In addition, we use the end-of-sequence (EOS) token as the padding token and specify that padding should be applied to the right side of sequences [10]. We also ensure that the EOS token and the beginning-of-sequence (BOS) token are automatically added to sequences.

2.3 Model Configuration

In this project, we employ the pre-trained Mistral-7B-Instruct-v0.1 [12] as the baseline, which is an adapted version of the base Mistral-7B-v0.1 [13] and acts as a chat-bot on a dataset of input/output pairs [8]. Specifically, we load the model with 4-bit precision to for memory efficiency and use the

bfloat16 data type for faster computation [10]. For LoRA's configuration, the scaling factor for the low-rank adaptation is 16 and the rank of approximation is 64. No bias term is added to the LoRA layers, and the dropout rate is set to be 0.1 to prevent overfitting. Besides, the model is being adapted for a causal language modeling task, and Eq. (2) illustrates the modules that are targeted for LoRA, where q_proj, k_proj, v_proj, o_proj, $gate\_proj$ are defined as the query projection layer, the key projection layer, the value projection layer, the output projection layer, and the gate projection layer in the transformer architecture, respectively.

$$\textit{target\_modules} = [q\_proj,\ k\_proj,\ v\_proj,\ o\_proj,\ gate\_proj] \tag{2}$$

2.4 Training Arguments

For training arguments, we employ the configuration from a Kaggle tutorial [10] to achieve promising results. We set the number of training epochs as 20 and the batch size as 4. The 32-bit adaptive moment estimation with weight decay (AdamW) optimizer is adopted with the learning rate 2×10^{-4}, weight decay 1×10^{-4}, and warm-up ratio 0.03. The maximum gradient norm is set to be 0.3 to avoid gradient explosion. In order to improve the training speed and reduce unnecessary memory usage, the 16-bit floating-point precision (fp16) is chosen. In addition, the maximum length of tokens to be processed is set to be 256.

2.5 Loss Function

In our task of fine-tuning the model for question-answering, the loss function for both training and validation is cross-entropy loss [10], which measures how well the model predicts the next token. In other words, after generating a new token, we take its probability distribution and find the loss against the ground truth token, and we do this over the entire generated sequence [14]. The training loss and validation loss are calculated during each training epoch, and we select the model with the lowest validation loss as the best model.

2.6 Evaluation Metrics

The first evaluation metric we choose is the bilingual evaluation understudy (BLEU) [15], which is widely used in machine translation tasks to measure the quality of machine-generated text. BLEU's output is always a number between 0 and 1, and the value indicates how similar the candidate text is to the reference text, with values closer to 1 representing more similar text. We employ BLEU for our task because it is useful for measuring how closely the machine-generated output aligns with the human-produced ground truth.

Another evaluation metric is called the recall-oriented understudy for gisting evaluation (ROUGE) [16]. Unlike BLEU [15], which is precision-based, ROUGE focuses on recall by comparing the overlap of n-grams, words, or subsequences between the generated text and reference text. In our task, we choose ROUGE-1, ROUGE-2, and ROUGE-L as the evaluation metrics, which refer to the overlap of unigrams, the overlap of bigrams, and the longest common subsequence (LCS) based statistics, respectively. ROUGE metrics range between 0 and 1, with higher scores indicating higher similarity between the generated text and reference text.

3 Results

3.1 Loss Curve

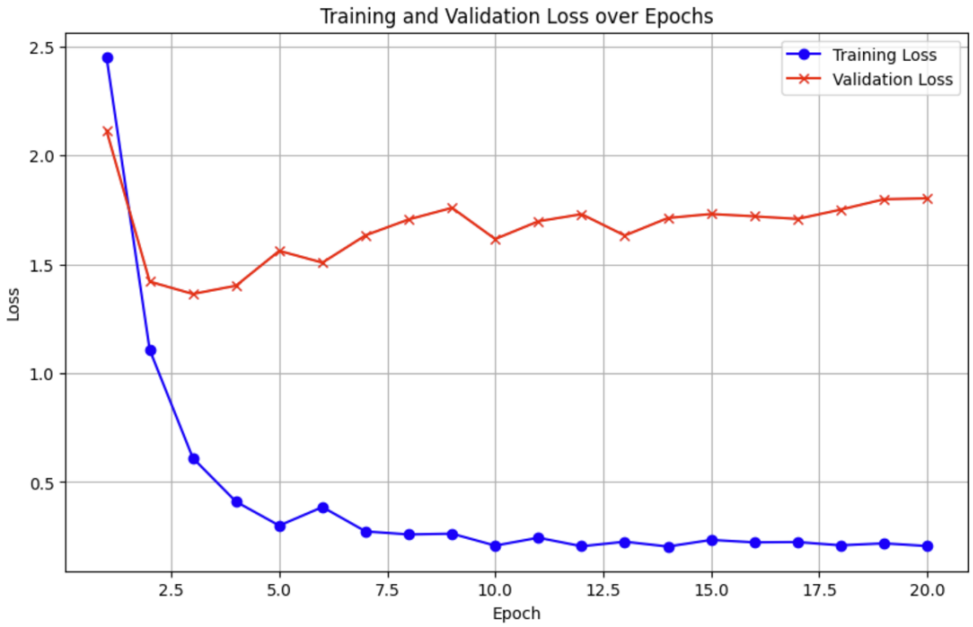

Figure 1: Plot of the training loss and validation loss

Fig. 1 shows the plot of the training loss and validation loss during the training process. At the initial stage, the decrease of both losses indicates that the model is learning and improving its performance on the training and validation set. However, as the training progresses, the training loss continues to decrease, suggesting that the model is fitting the training data well. In contrast, the validation loss starts to increase after the 3rd epoch, indicating that the model may suffer from overfitting. As we mentioned before, we select the model with the lowest validation loss as the best model. In this case, the best model is saved at epoch = 3.

3.2 Generated Outputs

The most straightforward way to evaluate the success of our fine-tuning process is to make a comparison of models' generated outputs on the test set before and after fine-tuning. Tab. 2, 3, and 4 present three examples of the correctly generated outputs on the test set, with each table containing a question, the ground truth, the generated answer before fine-tuning, and the generated answer after fine-tuning.

Table 2: The first example of the correctly generated output

Question	How many quiz grades are included in calculating the final grade?
Answer (Ground Truth)	The best 10 quiz grades will be counted towards the final grade.
Answer (Before Fine-tuning)	6 quiz grades are included in calculating the final grade.
Answer (After Fine-tuning)	10 of the best quiz grades are included.

Table 3: The second example of the correctly generated output

Question	What is the website for detailed information about academic integrity?
Answer (Ground Truth)	The website for comprehensive information on academic integrity is: `http://www.stonybrook.edu/commcms/academic_integrity/`.
Answer (Before Fine-tuning)	1. The National Honor Society website (`nhs.org`) is a comprehensive resource that provides information, resources, and guidelines about the academic integrity. 2. The website for the Academic Integrity Institute at Clemson College (`www.clemson.edu/academicintegrity/`) offers much information about academic integrity, including articles, case studies, and resources for educators and students. 3. The website for the American College Testing Association (www
Answer (After Fine-tuning)	`http://www.stonybrook.edu/commcms/academic_integrity/`

Table 4: The third example of the correctly generated output

Question	When is the topic of gradient descent covered in the course?
Answer (Ground Truth)	Gradient descent is discussed on September 5th.
Answer (Before Fine-tuning)	1-2 Semesters: Many modern computer science and AI textbooks cover the topic of gradient descent in the first or second year of undergraduate or graduate level courses in AI and computer science. This topic would typically be covered in the optimization section of the course.
Answer (After Fine-tuning)	09/05 Gradient descent is covered on 09/05. The topic includes the concepts of error terms and the backpropagation algorithm. Hands-on exercises are provided for practice. Some lecture slides are available on the course website, and the video lectures will be recorded and posted on the course website. You could watch the lecture website. You could watch the lecture and refer to the course website for more information about homework assignments.

In the first example, we observe that the model does not know the correct answer to the number of quiz grades and instead generates a random number. However, it gives the correct answer after fine-tuning. The second example illustrates that before fine-tuning, the model can only provide general information about the academic integrity websites, which are completely wrong and can not be found online. However, after fine-tuning, it accurately shows the correct website for Stony Brook University. In the third example, we expect the answer to be a specific date, but the model before fine-tuning generates a duration along with some irrelevant information. After fine-tuning, the model correctly identifies the date of the class as September 5th, whereas some redundant information is also included at the end of the answer.

Although the fine-tuned model demonstrates promising results on the test set, we still notice some errors in the generated output. Tab. 5 presents an example of the incorrectly generated output, where the answers before and after fine-tuning are both wrong. Our analysis suggests that this issue may arise from an insufficient fine-tuning process, which is further caused by limitations in our QA dataset and the model selection method. We will discuss these problems in Section 4.

To the best of our knowledge, we can still conclude that the pre-trained Mistral-7B-Instruct-v0.1 [12] model has learned valuable patterns from our QA dataset and the fine-tuning process has been very successful.

Table 5: An example of the incorrectly generated output

Question	What percentage of the total course grade is attributed to homework?
Answer (Ground Truth)	Homework accounts for 20% of the overall course grade.
Answer (Before Fine-tuning)	10 percent.
Answer (After Fine-tuning)	30%.

3.3 Evaluation Scores

In addition to text analysis, we calculate the average BLEU [15], ROUGE-1, ROUGE-2, and ROUGE-L [16] scores on the test set before and after fine-tuning, which are presented in Tab. 6. The results show that all evaluation scores are improved after fine-tuning the model, indicating the success of our fine-tuning process.

Table 6: Comparison of evaluation scores before and after fine-tuning

Model	**BLEU** ↑	**ROUGE-1** ↑	**ROUGE-2** ↑	**ROUGE-L** ↑
Before Fine-tuning	0.0598	0.2050	0.0985	0.1785
After Fine-tuning	0.1127	0.2923	0.1747	0.2481

4 Discussions

In this section, we will discuss some limitations of our approach and explore potential solutions to improve the results.

4.1 Dataset Limitation

As mentioned earlier, we still observe some errors in the generated output, which may result from an insufficient fine-tuning process. One of the main reasons is the size of our QA dataset. Given that the Mistral-7B-Instruct-v0.1 [12] model has been pre-trained on a large amount of data, the relatively small number of QA pairs in our dataset makes the fine-tuning process more challenging. One potential solution is to apply data augmentation techniques [17], such as paraphrasing QA pairs to generate additional data points. Another approach is to gather more course materials to create additional QA pairs. By expanding the dataset, we can facilitate a more effective fine-tuning process and enhance the model's generalization ability.

4.2 Model Selection

The second limitation is the model selection method. We choose the model with the lowest validation loss as our best model, using cross-entropy loss as the loss function [10]. However, a lower cross-entropy loss does not necessarily indicate better performance in answering questions. For instance, an answer that is properly reordered may still be a good response compared with the ground truth, even if its cross-entropy loss is higher.

Tab. 7 presents the evaluation scores obtained by models at different epochs during the same training process, where the model with the best validation loss is saved at the 3rd epoch, as shown earlier in Fig. 1. Although the validation loss reaches its lowest at the 3rd epoch, the training loss does not converge until the 10th epoch, suggesting that the best model may suffer from underfitting.

In addition, the evaluation scores indicate that models trained for more epochs perform better than the best model. Furthermore, Tab. 8 compares the generated outputs from different models, and the results show that models trained for more epochs are able to generate more accurate answers.

One way to address the model selection issue is to increase the number of training epochs, although this may lead to overfitting. An alternative approach is to choose a more suitable metric or loss function for validation. For example, we could generate multiple answers with slightly different orders for the same question in the validation set, and then compute the average BLEU [15] or ROUGE [16] scores between all pairs of generated outputs and the ground truth. This method would allow us to select the best model based on these scores during training.

Table 7: Comparison of the evaluation scores obtained by models at different epochs

Model	**BLEU** ↑	**ROUGE-1** ↑	**ROUGE-2** ↑	**ROUGE-L** ↑
Best (3rd Epoch)	0.1127	0.2923	0.1747	0.2481
Mid (10th Epoch)	0.1888	0.3969	0.2620	0.3764
Last (20th Epoch)	0.1946	0.4117	0.2611	0.3833

Table 8: Comparison of the generated outputs obtained by models at different epochs

Question	What percentage of the total course grade is attributed to homework?
Answer (Ground Truth)	Homework accounts for 20% of the overall course grade.
Answer (Before Fine-tuning)	10 percent.
Answer (3rd Epoch)	30%.
Answer (10th Epoch)	20%. The homework component is worth 20% of the overall grade.
Answer (20th Epoch)	20%.

4.3 Hyperparameter Tuning

Hyperparameter tuning is another important factor in optimizing the model's performance. In our future work, we plan to explore and fine-tune a wide range of hyperparameters, such as the number of epochs, batch size, learning rate, and dropout rate to achieve better results.

5 Conclusion

In this project, we fine-tune the pre-trained Mistral-7B-Instruct-v0.1 model [12] by using LoRA [9] on our course-based QA dataset. The generated outputs, loss curves, and evaluation scores all indicate the success of our fine-tuning process. We also address the limitations of our approach and propose potential solutions. Future work will focus on enlarging the dataset, choosing a more effective validation strategy, and tuning hyperparameters.

6 Acknowledgment

All group members contribute equally to the project, including tasks such as searching materials, running experiments, collecting results, and writing the final report. All of our codes and data have been uploaded to Google Drive, which can be accessed through `https://drive.google.com/drive/folders/1aSpiKJxnx1v933RiBv2KVMAfaNwXA3m3?usp=drive_link`.

References

[1] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[4] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[5] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Pre-trained language models for text generation: A survey. *ACM Computing Surveys*, 56(9):1–39, 2024.

[6] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57, 2024.

[7] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, et al. Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617*, 2023.

[8] `https://mycourses.stonybrook.edu/d2l/le/content/1506813/viewContent/39475988/View`.

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[10] Abid Ali Awan. Mistral-7B Instruct 4-bit QLoRA Fine-tuning. `https://www.kaggle.com/code/derrickmwiti/mistral-7b-instruct-4bit-qlora-fine-tuning`.

[11] Huggingface Transformer. `https://huggingface.co/docs/transformers/en/index`.

[12] `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.1`.

[13] `https://huggingface.co/mistralai/Mistral-7B-v0.1`.

[14] `https://www.reddit.com/r/LocalLLaMA/comments/1aq581t/loss_function_in_fine_tuning/`.

[15] Matt Post. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*, 2018.

[16] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[17] Connor Shorten, Taghi M Khoshgoftaar, and Borko Furht. Text data augmentation for deep learning. *Journal of big Data*, 8(1):101, 2021.