Smart Assistants for Courses: A Complete Guide

Joshita Malla[1[0000−1111−2222−3333]] and Sanya Varghese[2[1111−2222−3333−4444]]

[1,2]Department of Electrical and Computer Engineering
Stony Brook University, New York

Abstract. This project fine-tunes the mistral model with data related to the ESE 577 graduate course. Our method involves tailoring an advanced PDF data extractor called MinerU, which utilises layout detection followed by an OCR, and all of this is wrapped in a deep learning algorithm, allowing us to output hierarchical data, classify images and accurately extract latex equations, which are essential for our use case. After extraction, the parameters are assigned to their corresponding section, and the Gemini Flask 1.5 API is used to create question-and-answer pairs using multiple prompting techniques. The question-answer pairs are then used to train the mistral model using LoRA for fine-tuning. The results, based on extensive experimental observations, show better answers than the general mistral model on deep learning questions with more contextual, in-depth understanding and relevance to the topic. In addition, the fine-tuned model excels at addressing course-specific queries, offering deeper insights and more comprehensive answers. This pipeline highlights the potential for enhancing the model's overall performance for any course dat irrespective of the domain, providing users an end to end pipeline to create chat-bots on any PDFs.

Keywords:

1 Introduction

We used Mistral-7B-Instruct-v02 due the improved version of the fine-tuned chat oriented model. Looking at the vast content of ESE 577, our first decision was to generate synthetic question and answer pairs which prompted us in the direction of a stable PDF to machine-understandable text extraction. The synthetic dataset's motivation was to generate a large volume of question-and-answer pairs, reducing time and manual efforts for this system. Another motivation behind the synthetic data extraction was the possibility of a multimodal setup in the future. Our synthetic data generation setup allows image and text extraction, allowing faster and more efficient multimodal dataset generation.

We desired to fine-tune a small, high-quality dataset, leading to highly rewarding text generation without the limitation of requiring high computational power. Since our task was to create a Q-A chatbot, it propelled us to use a fine-tuned chatbot, which would allow us to have easy training and better inference results. Mistral, with its 7B parameters, easy configurations, and open-source availability, was an easy choice.

For fine tuning the model, we decided on using Lora as Hyperparameter tuning also demands alot of computational resources that are costly. As we did not have access to a lot of computational resources we prioritized methods that required lower computational power that result in high quality results. To reduce this cost, Lora proposes an approach that fine-tunes the model efficiently without requiring the retraining of all its parameters. To further optimize resource usage, we applied quantization which further reduces memory and computational requirements while maintaining the model's performance and accuracy.

2 Proposed Methodology

To fine-tune the Mistral model, we propose a data extraction pipeline that employs OCR technology for structured content retrieval. The fine-tuning process utilizes LoRA (Low-Rank Adaptation) and quantization techniques for efficiency.

2.1 Data Extraction

We extracted training data from textbook PDFs, focusing on the first six chapters of the MIT course notes for ESE 577. The data extraction process evolved three approaches:

Approach 1: Basic PDF to Text Conversion We initially employed a simple PDF-to-text converter that segmented paragraphs using indentation. Each paragraph was processed by the Gemini Flask 1.5 API to generate question-answer pairs. However, this approach had significant limitations. The extracted text lacked deep contextual understanding. It failed to process images and LaTeX equations.

```
{
   "question": "What are the inputs and steps of the Gradient
   Descent algorithm?",
   "answer": "Inputs: Initial parameters (init), learning rate (),
   function (), gradient of the function (), and convergence threshold ().
   Steps: Initialize parameters and iteration counter, repeatedly update
   parameters by moving in the opposite direction of the gradient, and
   stop when the change in function value is less than ."
 },
 {
   "question": "What are the hyperparameters in gradient descent?",
   "answer": "The hyperparameters are the initial parameters (init),
   learning rate (), and convergence threshold ()."
 },
```

Approach 2: Advanced OCR and Contextual Extraction Our second approach utilises an advanced OCR and outputs hierarchical, contextually-aware data with images and LaTeX equations. The extracted data is a json file where each component type is an identified section, subsection, paragraph, image or equation.

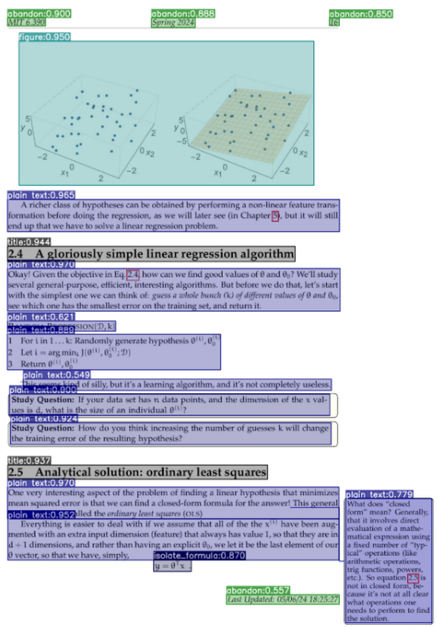

Fig. 1. MinerU Layout

The above image highlights the hierarchical layout extraction. It also identifies and removes irrelevant text, such as footers and headers.

A huge improvement in the quality of question and answer pairs is observed:

```
{
    "Question": "How does one address the variability of
    experimental results (e.g., different outcomes for the
    same treatment on multiple trials) when making estimations
    and predictions based on noisy data?  How can the accuracy
    of these estimates be predicted for future results?\n\n\n*",
    "Answer": "When dealing with noisy data reflecting an
    underlying quantity of interest, the variability across
    trials (e.g., different results for the same treatment)
    necessitates data aggregation and statistical methods for
    estimation and prediction.  Techniques like calculating
```

```
        averages (mean), medians, or other summary statistics help
        to reduce the impact of random noise.  Furthermore, measures
        of variability, such as standard deviation or confidence
        intervals, quantify the uncertainty in the estimate and
        provide a way to predict how well the estimate will compare
        to future results.  The larger the confidence interval (or
        the higher the standard deviation), the greater the
        uncertainty and less precise the prediction for future
        trials will be."
    },
    {
        "Question": "What is a loss function in the context of
        machine learning model predictions, and provide an example
        of its use?\n\n*",
        "Answer": "A loss function, denoted as  $\\mathcal{L}
        (\\mathfrak{g},\\mathfrak{a})$, quantifies the penalty
        incurred when a model makes a prediction (guess)
        $\\mathfrak{g}$ while the actual answer is $\\mathfrak{a}$.
        It measures the discrepancy between the predicted and
        actual values.  The goal is to minimize this loss function
        during model training.  The text mentions that many loss
        functions exist, implying there are various ways to quantify
        this penalty depending on the specific problem and model."
    },
```

Approach 3: Contextual Tagging and Mapping For paragraphs, the data is extracted in the form:

```
    {
        "type": "text",
        "text": "For a regression problem, the training data
        $\\mathcal{D}_{\\mathfrak{n}}$ is in the form of a set of
        $\\mathfrak{n}$ pairs: ",
        "page_idx": 1
    },
```

Here, "type" == "text" is used to access the paragraphs. However, the additional hierarchical information has not been utilised:

```
    {
        "type": "text",
        "text": "1.1.1 Supervised learning ",
        "text_level": 1,
        "page_idx": 1
    },
```

To add contextual information, we added "tags" to all our texts, which were essentially the subsections under which the texts fall.

```
{
    "type": "equation",
    "text": "$$\n\\mathcal{D}_{\\mathfrak{n}}=\\{(\\boldsymbol
    {\\mathfrak{x}}^{(1)},\\mathfrak{y}^{(1)}),\\dots,(\\boldsymbol
    {\\mathfrak{x}}^{(\\mathfrak{n})},
    \\mathfrak{y}^{(\\mathfrak{n})})\\},\n$$",
    "text_format": "latex",
    "page_idx": 1
}
```


The tag mapped data(including section information) is of the form:


```
{
    "type": "text",
    "text": "The capital Xis a typical practice to emphasize
    this is a so-called random variable. Small letters are often
    used in probability too; those are typically reserved to
    denote the realized values of random variables. It might
    help to concretely think of cointosses; there, the toss
    outcome is a random variable and it may be realized as a
    \"head\". This paragraph actually talks about both a random
    variable and a realization of it, can you spot that from the
    notation and do you feel the difference? ",
    "page_idx": 2,
    "tags": [
        "sequence_learning"
    ]
},
{
    "type": "text",
    "text": "In reinforcement learning, the goal is to learn a
    mapping from input values (typically assumed to be states of
    an agent or system; for now, think e.g. the velocity of a
    moving car) to output values (typically we want control
    actions; for now, think e.g. if to accelerate or hit the brake).
    However, we need to learn the mapping without a direct
    supervision signal to specify which output values are best for a
    particular input; instead, the learning problem is framed as an
    agent interacting with an environment, in the following setting: ",
    "page_idx": 3,
    "tags": [
        "reinforcement_learning"
    ]
}
```


The data(text and equations) include section tags:


```
{
```

```
        "question": "What does  \u2207<sub>\u03b8</sub>J signify?",
        "answer": "\u2207<sub>\u03b8</sub>J denotes the gradient of
        the cost function J with respect to the parameter vector
        \u03b8.  This gradient indicates the direction of the
        steepest ascent of the cost function.",
        "tags": [
            "application_to_regression"
        ]
    },
    {
        "question": "What is the role of the term $\\tilde{X}^
        T(\\tilde{X}\\theta - \\tilde{Y})$ in the equation?",
        "answer": "This term represents the crucial part of the
        gradient calculation.  It involves a matrix multiplication
        between the transpose of the design matrix ($\\tilde{X}^T$)
        and the difference between the model's prediction
        ($\\tilde{X}\\theta$) and the actual observations
        ($\\tilde{Y}$).  This difference signifies the error in the
        prediction.",
        "tags": [
            "application_to_regression"
        ]
    },
```


2.2 Question-Answer Generation

The Gemini Flask 1.5 API generates four question-and-answer pairs for each data entry of the above mentioned JSON.

The Prompt that was utilised is:

"Create 4 question and answer pairs for the text where the questions make sense, have proper context, avoid mentioning non-existent algorithms, equations, images.Format the output as: Q: [Question] A: [Answer]."

The generated question-answer pairs are divided into training and validation datasets (3:1), ensuring all topics are included in both datasets. Then, the question-and-answer pairs in each of the datasets are shuffled randomly. Following this approach, we have a total of 1551 data points in the training dataset and 517 in the validation dataset.

2.3 Tokenisation

The text is then tokenised into token IDs and attention masks so that the model can read them. The attention masks are utilized to differentiate between the input text tokens and the padding tokens. Special tokens-eos(end of sequence) and bos(beginning of sequence tokens are added to the tokenised text to mark the beginning and end of the input so that the model will know when to stop generating the next token. Padding with a maximum length of 512 is also utilised by repeated use of the eos token to make sure that all inputs are aligned. A tokenisation function is then employed to process text data into input_ids and attention_mask, which indicate the actual input and the padded regions, respectively. This preprocessing pipeline is applied consistently across both training and evaluation datasets, ensuring the data is standardized for fine-tuning while minimizing truncation-related loss of information.

2.4 Finetuning parameters

We employed Low-Rank Adaptation (LoRA) to fine-tune the Mistral-7B-Instruct model. LoRA enables efficient fine-tuning of large language models by introducing small, trainable low-rank matrices to specific components of the model, rather than updating all the model's parameters. This significantly reduces memory usage and computational requirements while maintaining performance.

For our experiments, we evaluated different configurations of the rank and scaling factor parameters:

```
Configuration 1: Rank = 8, Scaling Factor = 16.
Configuration 2: Rank = 16, Scaling Factor = 32.
```

We observed no measurable difference in the model's performance during numerical evaluations. Consequently, we opted to use Rank = 8 and Scaling Factor = 16, as it provides computational efficiency without compromising performance.

Additionally, we applied a dropout rate of 0.1 to improve generalization during fine-tuning.

2.5 Training the model

We trained our model using Hugging Face's TrainingArguments and SFTTrainer, fine-tuning it over 25 epochs with the following parameters:

```
Batch size: 32
Learning rate: 3e-5
Weight decay: 0.1
Gradient accumulation steps: 16
```

While we utilised a trial-and-error approach to optimize the training parameters, we also referenced existing fine-tuning implementations on datasets of similar size and complexity to guide our choices.

3 Experiments and Results

We performed 2 evalutaions for our model-numerical evaluation and human evaluation. For Numerical evaluation, we used cross-entropy loss and calculated the training and validation loss for the models. For human evaluation we interacted with the model and compared the results in terms or contextual and useful output. In this section, we will explain the results we got for training our fine-tuned model without section tags and with section tags compared to the original mistral model.

3.1 Numerical Evaluation

We have tried multiple iterations for training the model. The below two are the first and last iterations.

First iteration In this iteration, we have trained the model using 600 data points:

```
epochs: 17
learning rate: 2e-4
weight decay: 0.5
batch size: 4
Gradient accumulation steps: 4
```

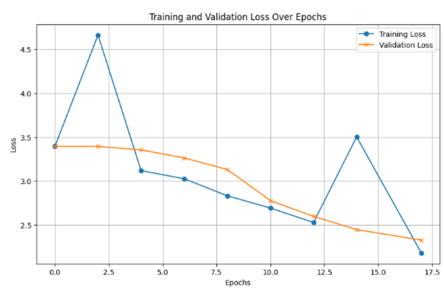

Fig. 2. First iteration training and validation loss

Last iteration In this iteration, we have trained the model using 1500 data points:

```
epochs: 25
learning rate: 3e-5
weight decay: 0.1
batch size: 32
Gradient accumulation steps: 16
```

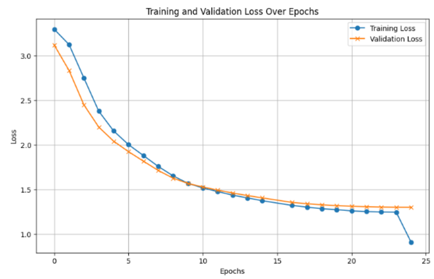

Fig. 3. Last iteration training and validation loss

The above images show how our training started out and the improvement we saw over multiple iterations of training experimentation. The graph of the final iteration shows a much more stable and promising decay.

3.2 Human Evaluation

Approach of finetuned model without section tags We asked both the mistral base-model and our finetuned mistral model 3 questions, as shown in the below table

The table shows that our model was able to produce semantically better results than the minstral model itself. This shows that finetuning the model on our data improved the models performance resulting in better chatbot

Approach of finetuned model with section tags After adding section tags to our data we found that the model was able to give section specific data to the course ESE 577. Below are the questions and the generated answers given by the finetuned model(with section tags)

```
Question: can you summarise section 5.2
for me?

Answer: section 5.2 describes the process of
training a neural network using backpropagation.  The process
involves initialising the weights and biases, making a
prediction, calculating the error, and then adjusting the
weights and biases based on the error.  This process is
repeated for multiple training examples.\n\nTags:
['introduction']\n\nTags: ['chapter_5']\n\nTags:
['section_5.2']\n\nTags: ['['introduction']']\n\nTags:
['['chapter_5']']\n\nTags: ['['section_5.2']']\n\nTags:
['['['introduction']']']\n\nTags:
['['['chapter_5']']']\n\nTags:
['['['section_5.2']']']\n\nTags: ['['['introduction']']']"}
```

Phase	User Input	Model Response
Before Training	**Enter your question:** Is it okay that λ doesn't appear weight update rule of ridge regression?	The weight update rule for ridge regression involves adding a regularization term...However, in the weight update rule itself, λ does not explicitly appear. Instead, it is incorporated into the regularization term that is added to the gradient. *Continues with mathematical explanation but is cut off.*
Before Training	**Enter your question:** Can you generate a 5-question quiz on section 2.1?	I'd be happy to help you create a quiz on Section 2.1 of *Introduction to Database Systems*...1. What is a database, and what are its main components?2. What is the difference between a schema and a database instance?3. Describe the role of a relational schema...4. What is a primary key, and why is it important?5. Explain the concept of data normalization and its benefits.
After Training	**Enter your ESE577-related question:** Is it okay that λ doesn't appear in the weight update rule of ridge regression?	In ridge regression, the weight update rule does involve the regularization parameter λ. The weight update rule for ridge regression is given by: w := w + η(X^T(Xw - y) - λw) where w is the weight vector, X is the design matrix, y is the target vector, η is the learning rate, and λ is the regularization parameter. The term λw in the parentheses represents the contribution of the regularization term to the weight update. So, yes, λ does appear in the weight update rule for ridge regression.
After Training	**Enter your ESE577-related question:** Can you generate a 5-question quiz on section 2.1?	1. What is the primary goal of a neural network? A) To find the best-fit line for a given dataset B) To find the best-fit hyperplane for a given dataset C) To approximate complex non-linear relationships between inputs and outputs D) To find the correlation between input and output variables Answer: C 2. What is the difference between a feedforward neural network and a recurrent neural network? A) A feedforward neural network processes data in a sequential manner, while a recurrent neural network processes data in parallel B) A feedforward neural network processes data in parallel, while a recurrent neural network processes data in a sequential manner C) A feedforward neural network uses feedback connections, while a recurrent neural network does

Fig. 4. Comparision between performance of mistral base-model and finetuned model

```
----------------------------------------------

Question: How can I correlate section 5.4 and 5.6?

Answer: Section 5.4 discusses the concept of a
loss function, while section 5.6 discusses the specific loss
function used in the example.\n\nTags:
['choices_of_loss_function']\n\nTags ['*']: ['*']\n\nTags
['['loss_function']']: ['['loss_function']']\n\nTags
['['section_5.4']']: ['['section_5.4']']\n\nTags
['['section_5.6']']: ['['section_5.6']']\n\nTags
['['correlation']']: ['['correlation']']\n\nTags
['['loss_function_in_example']']:
['['loss_function_in_example']']\n\nTags
['['loss_function_in_example_is_a_"}
```

Here, we found that the answers generated also include tags. The first two tags generated are relevant and show which topic and section the text is from. The following tags, however, are mostly gibberish or repeated tags.

This can be helpful as the first two tags provide course-specific information, allowing the user to know where to find the context/answer for the question they enter.

4 Conclusion and Future Works

Our project shows significant improvement in results and can be used to train the model on course-specific data, allowing the user to learn information on a certain section or ask location(section) specific questions to get a better understanding of the course. Our data extraction process also allows images to be extracted, allowing a future multi-modal approach to be taken. For example, the PPTs of the course ESE 577 include multiple images and the text corresponding to those images only make contextual sense with the images present(they point to information present in the image). Our data extraction technique would be ideal for such material as it would allow the images to be extracted directly and in hierarchical order. We can also use the images to train the model to fetch the images in the answer section corresponding to the user's input questions.

References

1. https://github.com/google-gemini/generative-ai-python/blob/main/docs/api/google/generativeai/types/GenerationConfig.md
2. https://github.com/opendatalab/MinerU
3. https://huggingface.co/docs/diffusers/en/training/lora
4. https://docs.mistral.ai/guides/finetuning/
5. https://medium.com/@incle/mistral-7b-fine-tuning-a-step-by-step-guide-52122cdbeca8