

# ASSIGNMENT 3: TRAIN YOUR OWN LLMs

**Siyu Li, Juhao Liang, Xidong Wang, Benyou Wang\***

The teaching team of Large Language Models, the Chinese University of Hong Kong, Shenzhen  
Shenzhen, China

{siyulil, juhaoliang1, xidongwang1}@link.cuhk.edu.cn, wangbenyou@cuhk.edu.cn

## 1 INTRODUCTION

Since the introduction of OpenAI’s ChatGPT series, the field of large-scale language models (LLMs) has gained significant attention. Within this realm of research, there remain numerous unexplored avenues related to the fine-tuning of these models. We encourage you to leverage this assignment to delve into topics that intrigue you within the scope of LLM research.

In our previous lectures, we have introduced the common recipe of training your own LLM from scratch. This assignment builds upon that knowledge and focuses on fine-tuning an existing pre-trained LLM and evaluating its performance on a downstream task. You are encouraged to exercise creativity and adapt the training process to engage in interesting explorations, especially during the instruction fine-tuning process.

In addition to supervised fine-tuning (SFT), a recent and widely used method for improving model alignment is **Reinforcement Learning from Human Feedback (RLHF)**. RLHF enhances LLMs by incorporating human preference signals rather than relying solely on written instructions. It usually consists of three stages: (1) SFT to obtain a base model, (2) training a reward model from human preference comparisons, and (3) optimizing the model with reinforcement learning (e.g., PPO) to maximize predicted reward. This process enables models like ChatGPT to better align with human values and conversational quality. You can optionally explore simplified variants such as Direct Preference Optimization (DPO) to understand preference-based fine-tuning in practice.

## 2 TASK DESCRIPTION

### OVERVIEW

1. Select a downstream task and collect the relevant training data.
2. Fine-tune a pre-trained large language model for the chosen downstream task.(SFT or RLHF)
3. Evaluate the performance of the trained model using a suitable metric, comparing it with baseline models.
4. Complete an experimental report along with the implementation code for the assignment.

In this assignment, you are expected to submit a report and accompanying code, and the specific guidelines for submission are detailed in Section 5.

### 2.1 DOWNSTREAM TASKS

#### 2.1.1 DEFAULT TASKS

You can pick a research topic concerning the fine-tuning of LLMs above six training tasks. The given tasks are across two languages and three domains, as presented in Table 1. While we provide predefined research topics for inspiration, you also have the freedom to choose your own research topic if you have a specific interest.

---

\*Benyou Wang is the instructor.

| Language | Domain    | Dataset Description                  | Huggingface Dataset Link  |
|----------|-----------|--------------------------------------|---------------------------|
| English  | Law       | Legal instructions dataset           | Lawyer-Instruct           |
|          | Medical   | Medical QA dataset                   | MedQuad-MedicalQnADataset |
|          | Financial | Financial QA dataset                 | finance-alpaca            |
| Chinese  | Law       | Chinese legal dataset                | DISC-Law-SFT              |
|          | Medical   | Chinese medical dialogue dataset     | Huatuo26M-Lite            |
|          | Financial | Chinese financial evaluation dataset | FinGPT-fineval            |

Table 1: Examples of language Model training tasks.

### 2.1.2 RLHF TASKS

In addition to the default and customized supervised fine-tuning (SFT) tasks, students are encouraged to explore alignment-oriented datasets designed for **Reinforcement Learning from Human Feedback (RLHF)** or its simplified variant **Direct Preference Optimization (DPO)**. These datasets contain pairs of responses labeled by human or AI preference (chosen vs. rejected), which can be directly used to train or evaluate preference-based models.

Recommended datasets include:

- **UltraFeedback-Binarized** (trl-lib/ultrafeedback\_binarized): A high-quality English preference dataset widely used for reward modeling and DPO fine-tuning.
- **DPO-En-Zh-20k** (llamafactory/DPO-En-Zh-20k): A bilingual (EnglishChinese) preference dataset for direct preference optimization experiments.

You may use these datasets to train a reward model, perform DPO-based fine-tuning, or evaluate preference accuracy on a held-out test split. Such tasks emphasize aligning model responses with human or AI preference signals rather than simple instruction-following accuracy, making them ideal for exploring the effectiveness of RLHF-style alignment training.

### 2.1.3 CUSTOMIZED TASKS

You could select one of the above, or any customized one upon to the approval of a TA or the instructor.

## 2.2 CODE IMPLEMENTATION

Here is an example of how Qwen is trained to adapt for a Chinese medical task:

[https://colab.research.google.com/drive/1h3jm17xUGafXsisw8UAU7dwQ\\_g9t3c\\_K?usp=sharing](https://colab.research.google.com/drive/1h3jm17xUGafXsisw8UAU7dwQ_g9t3c_K?usp=sharing)

Here is an example of how to DPO QWEN:

<https://colab.research.google.com/drive/1zzbNse317ZOKyI5EnPbd50fA-srELFW?usp=sharing#scrollTo=V5NzRZPXLNyy> What you should pay attention is like:

1. **Data Engineering:** The quality of your training data significantly impacts your model's performance. Consider exploring different data engineering techniques to generate high-quality training data. You can use various methods, including leveraging tools like ChatGPT, Phoenix, or other accessible LLMs.
2. **Backbone Model Selection:** Choosing a suitable backbone model can provide a robust foundation for your training process. While the provided notebook suggests a baseline approach using a 4B QWEN3 model, we encourage you to exercise your creativity, experiment with a variety of models, and compare their performance in the context of specific tasks to gain valuable insights.
3. **Training Techniques:** While the provided baseline notebook utilizes the QLoRA technique for training the 7B model, you have the freedom to explore alternative training methodologies beyond QLoRA. Conducting experiments with various approaches can yield valuable insights and potentially improve the performance of your model.

### 2.3 EVALUATION

Note that the popular benchmark like multiple-choice questions (MMLU) cannot effectively assess the instruction-following ability of LLMs, which is of the core of this assignment. We recommend using ChatGPT as the evaluator. We offer 20 test questions for each task mentioned in Tab. 1, and you could utilize ChatGPT to generate baseline answers and your independently trained LLM to produce enhanced answers. Subsequently, employ ChatGPT to compare the quality of the answers. We have provided a template for your reference.

In addition to qualitative comparison via ChatGPT judgment, models fine-tuned with Reinforcement Learning from Human Feedback (RLHF) should also be quantitatively evaluated on the corresponding test split of the training dataset. Specifically, for RLHF-style datasets that contain human preference pairs (i.e., chosen vs. rejected), the evaluation can be conducted by computing the reward score or classification accuracy of the trained reward model or policy model on the held-out test set. Use the trained reward model or policy to predict whether the model prefers the chosen over the rejected response, and report the **preference accuracy** or **average reward score**.

In the final report, both SFT and RLHF experiments should include the corresponding **training and validation loss curves**, along with a short discussion on convergence behavior and generalization performance.

## 3 PRACTICE PLATFORM

You have the flexibility to conduct your experiments on a platform of your choice. Regardless of your selection, the final submission for this assignment should be in the form of a python code.

- **Local:** If you have access to local GPU resources available, you can choose to run the experiment locally.
- **Colab<sup>1</sup>:** You can run your experiments on Google Colab, which provides a T4 GPU for free computing.

## 4 SOME USEFUL LINKS

### 4.1 PRE-TRAINED MODELS

You could use one of many of the following models:

- Llama3-{8b, 70b} <https://huggingface.co/meta-llama>,
- Phoenix-7b <https://github.com/FreedomIntelligence/LLMZoo>,
- QWen3-{4B, 30B} <https://github.com/QwenLM/Qwen3>,
- or any open-source models...

### 4.2 OPTIONAL TRAINING TRICKS

You can use other parameter-efficient fine-tuning methods to be able to fine-tune large models on limited GPU resources, here are some useful open source github repositories that may help:

- PEFT <https://github.com/huggingface/peft>
- OpenDelta <https://github.com/thunlp/OpenDelta>
- QLoRA <https://github.com/artidoro/qlora>

---

<sup>1</sup>[https://colab.research.google.com/drive/1lo\\_ZDiJ4eevZR3nqaVLX7vBjHLCmYVOP?usp=sharing](https://colab.research.google.com/drive/1lo_ZDiJ4eevZR3nqaVLX7vBjHLCmYVOP?usp=sharing)

#### 4.3 TRAINING FRAMEWORKS

Using notebooks for long-term model training is inconvenient in some cases. You can also try the following alternative model training frameworks:

- LLMZoo <https://github.com/FreedomIntelligence/LLMZoo>
- stanford\_alpaca [https://github.com/tatsu-lab/stanford\\_alpaca](https://github.com/tatsu-lab/stanford_alpaca)
- LLaMA-Factory <https://github.com/hiyouga/LLaMA-Factory>
- FastChat <https://github.com/lm-sys/FastChat>
- DeepSpeed-Chat <https://github.com/microsoft/DeepSpeedExamples/tree/master/applications/DeepSpeed-Chat>

#### 4.4 RLHF FRAMEWORKS

Training RLHF models requires an additional reward function or a separately trained reward model. For quick implementation and experimentation, the following open-source frameworks provide well-structured pipelines for RLHF or DPO training:

- **OpenRLHF:** <https://github.com/OpenRLHF/OpenRLHF>
- **TRL (Hugging Face):** <https://github.com/huggingface/trl>
- **LLaMA Factory:** <https://github.com/hiyouga/LLaMA-Factory>

### 5 SUBMISSION FORMAT

In your submission, you are expected to provide a experiment report that with the following section

- **Research Topic:** Clearly state the research topic you have chosen to investigate.
- **Experiment design:** Describe the design of your experiments, including the methodology and approach you employed.
- **Code implementation:** Present your code implementation, demonstrating how you translated your ideas into code.
- **Result analysis:** Collect and present the results obtained from your experiments, along with an analysis of the insights gained.
- **Conclusion:** Conclude your report by summarizing the key findings and outcomes of your research.

**Template** Template for the report can be found at

[https://www.overleaf.com/read/fkshyngħħrrgħk#901680](https://www.overleaf.com/read/fkshyngħrrgħk#901680).

### 6 GRADING CRITERIA

The grading for this assignment is based on the following criteria:

- **[5 marks]** Code Quality: Evaluation of the quality of your code.
- **[15 marks]** Report Quality: Among these 15 marks, 5 are allocated for the clarity of your report, 5 for the quality of your experimental results (completeness, experimental rigor, and technical innovation), and 5 for your experimental analysis, which includes case studies, sensitivity analysis of individual experiment components, ablation studies, and any other interesting discussions.
- **[1 mark Bonus]** This bonus mark is awarded if you tackle a significant task not listed in Section 2. We encourage innovative and meaningful endeavors, but the significance is subject to evaluation.

- **[1 mark Bonus]** An additional bonus mark is awarded for exceptionally clean and clear code. This bonus may not be granted if you attach your code as supplementary materials. Code clarity is crucial for this bonus.

## 7 LAST TIP

Please do not submit too loooooog reports. Be concise and say something that matters.

An example prompt for ChatGPT evaluation

**Prompt:**

We would like to request your feedback on the two AI assistants in response to the user question displayed above.

Please evaluate the helpfulness, relevance, accuracy, level of details of their responses. You should tell me whether Assistant 1 is ‘better than’, ‘worse than’, or ‘equal to’ Assistant 2.

Please first compare their responses and analyze which one is more in line with the given requirements.

In the last line, please output a single line containing only a single label selecting from ‘Assistant 1 is better than Assistant 2’, ‘Assistant 1 is worse than Assistant 2’, and ‘Assistant 1 is equal to Assistant 2’, avoiding any potential bias and ensuring that the order in which the responses were presented does not affect your judgment.

**Question:** How’s the weather today?

**Assistant1:** I’m unable to provide real-time information as my knowledge was last updated in January 2022, and I don’t have access to current data. To get the current weather for your location, I recommend using a weather website or app, asking a virtual assistant like Siri or Alexa, or checking a trusted news source.

**Assistant2:** I’m sorry, but I don’t have access to real-time information, including current weather conditions. To find out the current weather in your location, I recommend checking a weather website, using a weather app, or asking a voice-activated device like Amazon Alexa or Google Assistant. They can provide you with up-to-date weather information for your area.

**Expected Output:**

Both Assistant 1 and Assistant 2 provide very similar responses to the user’s question. They both inform the user that they cannot provide real-time weather information due to their lack of access to current data. Additionally, both assistants offer alternative suggestions for finding current weather information, such as using weather websites or apps and asking voice-activated devices like Siri, Alexa, or Google Assistant.

In this case, there is no significant difference in the helpfulness, relevance, accuracy, or level of detail in their responses. Both responses are equally suitable and align with the user’s request for information about the current weather.

Therefore, the judgment is: **Assistant 1 is equal to Assistant 2.**

Figure 1: An example prompt. Pay attention to design a prompt that produce a well-formatted answer for result extraction (like *better*, *worse* or *equal*).

## ACKNOWLEDGMENT

Please acknowledge this course if you publish any materials based on this assignment.

## A CHATGPT EVALUATION

Figure A shows a prompt example using Chatgpt for evaluation. You can also refer to the implementation in assignment 2 for model evaluation.

## B SELF-INSTRUCT

## REFERENCES

### An example prompt for Self-instruct

**Prompt:**

You are asked to come up with a set of 20 diverse task instructions. These task instructions will be given to a GPT model and we will evaluate the GPT model for completing the instructions.

Here are the requirements:

1. Try not to repeat the verb for each instruction to maximize diversity.
2. The language used for the instruction also should be diverse. For example, you should combine questions with imperative instructions.
3. The type of instructions should be diverse. The list should include diverse types of tasks like open-ended generation, classification, editing, etc.
2. A GPT language model should be able to complete the instruction. For example, do not ask the assistant to create any visual or audio output. For another example, do not ask the assistant to wake you up at 5pm or set a reminder because it cannot perform any action.
3. The instructions should be in English.
4. The instructions should be 1 to 2 sentences long. Either an imperative sentence or a question is permitted.
5. You should generate an appropriate input to the instruction. The input field should contain a specific example provided for the instruction. It should involve realistic data and should not contain simple placeholders. The input should provide substantial content to make the instruction challenging but should ideally not exceed 100 words.
6. Not all instructions require input. For example, when a instruction asks about some general information, "what is the highest peak in the world", it is not necessary to provide a specific context. In this case, we simply put "`|noinput|`" in the input field.
7. The output should be an appropriate response to the instruction and the input. Make sure the output is less than 100 words.

List of 20 tasks:

Figure 2: Example of a self-instruct prompt. Pay attention to designing tips that match your data requirements.