CMPSC 442: Homework 6

Fine-Tuning a Language Model with Group Relative Policy Optimization (GRPO)

Due Date: Apr 28th

Learning Objectives

This assignment is designed to help you build hands-on experience with fine-tuning language models using Group Relative Policy Optimization (GRPO). You will apply concepts from reinforcement learning, parameter-efficient tuning (LoRA), and summarization tasks in NLP. By the end of this assignment, you should be able to:

- Describe the purpose and benefits of GRPO in language model fine-tuning.

- Implement LoRA to reduce the number of trainable parameters.

- Create and apply a reward function for reinforcement learning.

- Generate and interpret output from a fine-tuned model.

Submission Instructions

Please submit the following files to Gradescope:
1. homework6_<psu_id>.py — Your code with all completed sections on **gradescope.**
2. homework6_log.txt — A log of your training process and generated outputs **on canvas**
3. evaluation_results.json — the script is provided **on canvas**

Replace <psu_id> with your actual Penn State user ID.

Please Note: this homework will not be graded by gradescope

Section 1: Environment Setup

Why: Installing the correct packages ensures you can run and train models without errors. GRPO and LoRA require specific versions of Hugging Face libraries and reinforcement learning utilities.

Task: It's recommended to use a *conda* environment for this setup

Refer for [wandb](): which you can observe the training processing

```
pip install datasets==3.2.0 transformers==4.47.1 trl==0.14.0 \
peft==0.14.0 accelerate==1.2.1 wandb==0.19.7
```

Section 2: Load Model and Tokenizer

Why: The model we use is *'SmolLM-135M-Instruct'*, a compact instruction-following language model. Understanding the loading and device allocation steps will help you use larger models later in research or production.

Task: Load the model and tokenizer using Hugging Face Transformers. Use CPU if CUDA is not available. ***Print out which device is being used and include this in your log file.***

Section 3: Load the Dataset

Why: The dataset *'mlabonne/smoltldr'* provides short prompts and summaries. Using real-world data gives you practical insight into NLP generation tasks.

Task: Load the dataset using Hugging Face Datasets. ***Print three example prompts and their summaries.***

Section 4: Add LoRA

Why: LoRA (Low-Rank Adaptation) reduces the number of trainable parameters. This makes fine-tuning feasible even on modest hardware.

Task: Apply LoRA to your model using the provided configuration. ***Print the number of trainable parameters in your log file.***

Section 5: Define a Reward Function

Why: In GRPO, we train a model based on rewards instead of matching exact outputs. You'll design a reward function that evaluates generated output by its length.

Task: Implement the provided reward function that favors summaries around 50 tokens.

Section 6: Configure GRPO

Why: GRPO uses reinforcement learning to optimize model behavior. Configuring it properly ensures stable and meaningful training.

Task: Change the GRPOConfig class to set your training arguments. Increase epochs and see how this changes training time and reward evolution.

Section 7: Train the Model

Why: Training the model lets it learn to generate better outputs according to your reward function.

Task: Run your GRPOTrainer with your settings. ***Record the reward progression in your log.***

Section 8: Generate and Evaluate Output

Why: The final evaluation shows how well your model learned. You will test generation on unseen prompts and analyze whether the outputs match your goals.

Task: Generate text for all prompts from the test set. The results will be save in the evaluation_results.json file

Grading Rubric

This assignment will be graded based on two components:

1. **Structural Completion (50 points)**
   Gradescope will automatically check whether your code runs and the required functions are implemented. If all structural requirements are met, you will receive **50 base points**.
2. **Model Performance (50 points)**
   The remaining 50 points will be awarded based on your model's performance, measured by the **ROUGE-1 score**, a common metric for evaluating the overlap between generated summaries and reference summaries.

   Your ROUGE-based score will be computed using the formula:

   $$\textbf{Score} = \textbf{Min}\left((0.84 + \textbf{ROUGE-1}) \times 50 \text{ , } 50\right)$$

   This means you need a **ROUGE-1 score of at least 0.16** to receive full credit for this portion. If your ROUGE-1 is lower, the score will be scaled proportionally.

ROUGE-1 measures unigram (single word) overlap between your generated summary and the reference. A higher score indicates better content alignment

P.S. If you encounter any issues during this homework, please contact Zhuoyang Zou via Canvas email or visit her during office hours.