Exercise 2: Best Subset for DPO Training

Warning: Since this excerise involves aligning LLMs to be less harmful, data may include examples of offensive language or other harmful content.

1 Introduction

Creating chat-capable large language models (LLMs) that are aligned with human values is typically achieved through two major steps: Supervised Fine-Tuning (SFT), where the model learns to follow instructions and respond to human prompts, and Reinforcement Learning from Human Feedback (RLHF), where the model is further optimized for human preferences. As discussed in the lecture, RLHF can be implemented with reinforcement learning methods like Proximal Policy Optimization (PPO) or more direct approaches, such as Direct Preference Optimization (DPO). Unlike PPO, DPO bypasses the reinforcement learning stage and instead optimizes the model directly on human preference data using a closed-form objective.

In this assignment, you will implement a method to select the best subset of instructions for DPO training.

2 Problem Statement

You are provided with a dataset containing 5,000 prompts from Anthropic's hh-rlhf dataset[1] (returned from `prepare_dpo_data` in **data.py**). Each prompt includes a *chosen* (helpful/less harmful) and a *rejected* (less helpful/more harmful) response, alongside two models:

- A DPO-aligned gpt2 model[2], which has undergone both SFT and DPO steps on the full dataset.
- A baseline gpt2 model[3], which has only undergone the SFT step.

Your task is to optimize the performance metric `benchmark_performance`, defined as:

$$\texttt{refusal\_rate(harmful\_data)} - \texttt{refusal\_rate(harmless\_data)}$$

[1] `https://huggingface.co/datasets/Anthropic/hh-rlhf`

[2] `ttps://huggingface.co/qwenzoo/utn-llm-assign2-gpt2-DPO`

[3] `https://huggingface.co/qwenzoo/utn-llm-assign2-gpt2-SFT`

Here, the refusal rate is computed by detecting predefined text refusal patterns in model responses. The harmful and harmless datasets each contain 20 synthetic prompts (see `data.py`).

The goal is to select a minimal subset of the full dataset that maximizes this performance metric. In other words, you must identify the smallest possible set of training instructions that achieves a `benchmark_performance` value close to that of the DPO-aligned model. For reference, the DPO-aligned model obtains a score of 0.4, while the SFT model achieves 0.1 on the public benchmark dataset (see `benchmark.ipynb`).

For evaluation, a different but related set of harmful and harmless data will be used to test the generalizability of your solution.

We recommend running this project on Google Colab.

3 Project Structure

- **data.py**: This file includes the harmful datasert `harmful_prompts`, harmless dataset `harmless_prompts`, refusal prefixes `_test_prefixes`, the performance metric function `benchmark_performance`, `prepare_dpo_data` that returns the training and validation datasets for DPO training, `chat_template` that takes a prompt and a response and returns the expected input of the model.
- **dpo.py**: This file includes the training procedure for DPO. This includes the following functions: `train_dpo_hh_rlhf` which downloads the full dataset, filter it and starts training; `train_dpo` which takes a batch or a dataloader and train on it.
- **sft.py**: This is file includes the function used to train the SFT step. No need to change this file.
- **benchmark.ipynb**: A notebook to showcase loading the models and running the performance benchmark function.

4 Deliverables

The following files should be submitted along with the rest of the codebase.

- **main.ipynb**: This notebook should contain your method implementation along with explantions.
- **report.pdf**: This file should include your motivation and detailed explantation of your method.
- **subset.csv**: This file should include the samples that your method chose from the full dataset. Each row should include the original columns *chosen*, *rejected*, *prompt* along with an index *idx* indicating the original index of the sample in the full dataset.