Week 5 Assignment [4 Points]

Generative AI

Saarland University – Winter Semester 2024/25

Submission due: 25 November 2024 by 6pm CET

genai-w24-tutors
genai-w24-tutors@mpi-sws.org

1 Reading Assignment

This week's reading assignment comprises of the following:

(R.1) Training Language Models to Follow Instructions with Human Feedback [1].
Paper link: https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

(R.2) Direct Preference Optimization: Your Language Model is Secretly a Reward Model [2].
Paper link: https://proceedings.neurips.cc/paper_files/paper/2023/file/a85b405ed65c6477a4fe8302b5e06ce7-Paper-Conference.pdf.

[OPTIONAL] Below, we are providing additional reading material covering content relevant to the lecture:

- Proximal Policy Optimization Algorithms [3].
  Paper link: https://arxiv.org/pdf/1707.06347
- MaxMin-RLHF: Alignment with Diverse Human Preferences [4].
  Paper link: https://openreview.net/pdf?id=8tzjEMF0Vq

2 Exercise Assignment

This week's exercise assignment aims to give you insights into the Bradley-Terry preference model and the alignment methods RLHF and DPO. These questions should be answered in the PDF submission file – see instructions in Section 4.

2.1 Bradley-Terry Model and Diverse Preferences

Suppose we have a prompt x with three possible responses y_1, y_2 and y_3. Each response is rated by users from four distinct subpopulations with diverse preferences. User ratings from each subpopulation are shown below.

- **Subpopulation 1:** $r_1(x, y_1) = 1, r_1(x, y_2) = 1, r_1(x, y_3) = 2$
- **Subpopulation 2:** $r_2(x, y_1) = 2, r_2(x, y_2) = 1, r_2(x, y_3) = 2$
- **Subpopulation 3:** $r_3(x, y_1) = 3, r_3(x, y_2) = 1, r_3(x, y_3) = 0$
- **Subpopulation 4:** $r_4(x, y_1) = 0, r_4(x, y_2) = 2, r_4(x, y_3) = 3$

For example, users from **Subpopulation 1** equally prefer responses y_1 and y_2 for prompt x, and response y_3 twice as much.

(E.1) Using the **Bradley-Terry model** for pairwise comparisons, construct the preference table for each subpopulation based on the given ratings, i.e., compute the probabilities $Pr(y_1 \succ y_2)$,

$Pr(y_1 \succ y_3)$ and $Pr(y_2 \succ y_3)$ for each of the four subpopulations. In addition, determine which response is most preferred within each subpopulation.

(E.2) **[OPTIONAL]** Now, consider the preferences of the entire population. In order to account for the diverse preferences of Subpopulations 1-4, in this exercise you will use an alignment method inspired by the Egalitarian principle in social choice theory. Consider the **MaxMin** approach where the chosen response y_c **max**imizes the **min**imum rating across all subpopulations, i.e.,

$$c \in \operatorname*{argmax}_i \min_j r_j(x, y_i).$$

Which response(s) achieve the highest minimum score? How does this outcome compare to the preference tables you constructed in (E.1)?

REMARK: The idea behind this exercise follows from [4]. You do not need to have read the paper in order to solve this exercise, but you are encouraged to do so if you are interested in learning more about the problem of aligning with diverse human preferences.

2.2 RLHF vs. DPO

You are fine-tuning a language model $\mathcal{M}_\theta$ with policy π_θ, to align with human preferences. Assume access to a dataset $\mathcal{D}_p = \{x_p^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^N$ of human preferences, where response y_w is preferred to response y_l for prompt x_p. Responses y_w and y_l are generated by a reference language model $\mathcal{M}_{SFT}$, whose policy is denoted by π_{SFT}. Let r_ϕ be a learned reward function that fits the preference data.

You are given a datapoint $\{x_p, y_w, y_l\} \sim \mathcal{D}_p$, with the following values:

- $\pi_\theta(y_w|x_p) = 0.2$; $\pi_\theta(y_l|x_p) = 0.3$;
- $\pi_{SFT}(y_w|x_p) = 0.2$; $\pi_{SFT}(y_l|x_p) = 0.4$;
- $r_\phi(x_p, y_w) = 0.5$; $r_\phi(x_p, y_l) = 0.8$.

For a temperature parameter $\beta = 2$, answer the following questions:

(E.3) Calculate the loss used to update the learned reward model in the RLHF method (see Reward Model objective in slide #35) and the DPO loss for fine-tuning π_θ (see DPO objective in slide #41) w.r.t. the given datapoint. Below we also provide these two loss functions for completeness.

$$\mathcal{L}_R(r_\phi) = -\mathbb{E}_{(x_p,y_w,y_l)\sim\mathcal{D}_p}\Big[\log\sigma\big(r_\phi(x_p,y_w) - r_\phi(x_p,y_l)\big)\Big]$$

$$\mathcal{L}_{DPO}(\pi_\theta;\pi_{SFT}) = -\mathbb{E}_{(x_p,y_w,y_l)\sim\mathcal{D}_p}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x_p)}{\pi_{SFT}(y_w|x_p)} - \beta\log\frac{\pi_\theta(y_l|x_p)}{\pi_{SFT}(y_l|x_p)}\right)\right]$$

(E.4) Describe the expected changes in policy π_θ when applying either RLHF or DPO methods. No numerical computations are needed for this part.

(E.5) Analyze the effect of varying the temperature parameter β on the DPO loss. Consider the case where $\pi_{SFT}(y_l|x_p) = 0.2$ and everything else remains the same. How does this change the effect of β on the DPO loss? What conclusions can you draw from this analysis?

3 Implementation Based Assignment

In this week's implementation assignment, you will compare the performance of preference-tuned language models vs. supervised fine-tuned language models on two different settings.

3.1 Preference Tuning for Shakespeare Completions

In these questions, the goal of preference tuning is to guide the model in generating completions for prefixes from Shakespeare's works that align better with certain preferences. We will compare the

performance of the preference-tuned Phi-1.5 and the supervised fine-tuned Phi-1.5 (which we also used in Week 4 assignment). Given the constraints on free computation resources available on Google Colab, we have already preference-tuned Phi-1.5 model, using a server with $8 \times$ H100 Nvidia GPUs, based on certain preferences for completions of prefixes from Shakespeare's works, and uploaded it to Hugging Face. We have provided the following files in `week5_implementation.zip`:

- `llm_completion.py`: An implementation that automatically downloads and makes use of an LLM from Hugging Face (https://huggingface.co/) in order to generate completions for given prefixes. Note that the model gets downloaded to the Colab environment, not to your local machine. The script reads an input file containing text prefixes, generates completions for each prefix, and outputs its results in a new file. The script is capable of using GPU resources from Colab when available – please first read and follow Section 5.2 for more details on how to use GPU resources and how to install all the required packages for this week's assignment. Note that this file is the same as provided in Week 4 assignment and only the model names have changed.
- `input_shakespeare_prefixes.txt`: This file contains a set of text prefixes from Shakespeare's works, which the LLM will use as prompts for generating completions. Note that this file is the same as provided in Week 4 assignment.
- `pref_training_completion.py`: This is the training script used to do the preference tuning[1] of Phi-1.5 model to perform Shakespeare completions that align better with certain preferences.
- `pref_data_completion.jsonl`: This file contains preference data used to do the preference tuning of the Phi-1.5 model.
- `pref_data_completion_20samples.txt`: This file shows 20 samples of preferences used for tuning the Phi-1.5 model to perform Shakespeare completions in alignment with these preferences.

Note that first you will need to install dependencies for the scripts to run, as described in Section 5.2. Next, as part of this assignment, you will execute the provided code by setting different variables in the `__main__` function without implementing any new functionality. More concretely, in the `__main__` function of `llm_completion.py`, the script will read the input file `input_shakespeare_prefixes.txt`, generate completions using the LLM, and write the generated completions to `output_shakespeare_completions_{MODEL_NAME}.txt`. Once the code is executed, you can review the generated output.

(I.1) In the `__main__` function of `llm_completion.py`, set the `MODEL_NAME` to `course-genai-w24/week4-phi-1.5-sft-shakespeare`[2] in order to use the supervised fine-tuned Phi-1.5 as the LLM, then run the script. The completions will be written to `output_shakespeare_completions_week4-phi-1.5-sft-shakespeare.txt`. This `.txt` file should be included as part of the ZIP submission file – see instructions in Section 4.

(I.2) In the `__main__` function of `llm_completion.py`, set the `MODEL_NAME` to `course-genai-w24/week5-phi-1.5-pref-shakespeare`[3] in order to use the preference-tuned Phi-1.5 as the LLM, then run the script. The completions will be written to `output_shakespeare_completions_week5-phi-1.5-pref-shakespeare.txt`. This `.txt` file should be included as part of the ZIP submission file – see instructions in Section 4.

(I.3) Provide a qualitative comparison of the generated completions from the preference-tuned Phi-1.5 model and the supervised fine-tuned Phi-1.5 model, focusing on how they differ in terms of alignment with the provided sample preferences. This should be answered in the PDF submission file – see instructions in Section 4.

- For each evaluation sample (i.e., a prefix), do a pairwise comparison of the three completions generated by both models (e.g., compare `COMPLETION 1-A` from the preference-tuned model with `COMPLETION 1-A` from the supervised fine-tuned model, `COMPLETION`

[1]The preference tuning process is done using odds ratio preference optimization (ORPO) [5]. Paper link: https://aclanthology.org/2024.emnlp-main.626.pdf.

[2]This is the same model as the one used in Week 4 assignment. More details at https://huggingface.co/course-genai-w24/week4-phi-1.5-sft-shakespeare.

[3]This model was preference tuned starting from the `course-genai-w24/week4-phi-1.5-sft-shakespeare` model. More details about this model are at https://huggingface.co/course-genai-w24/week5-phi-1.5-pref-shakespeare.

1-B with COMPLETION 1-B, and COMPLETION 1-C with COMPLETION 1-C). For these pairwise comparisons, which model produces the completion that better aligns with the sample preferences? Report the number of times the preference-tuned Phi-1.5 model wins / loses / ties for these three comparisons for each evaluation sample (i.e., a prefix).

- Then, provide an overall qualitative comparison between the two models across all the evaluation samples (i.e., all prefixes). You can answer this question in a few sentences and no detailed analysis is required.

3.2 Preference Tuning for Text Summarization

In these questions, the goal of preference tuning is to guide the model to generate summaries that are *more comprehensive and provide more context* for a given Reddit post (domain what was considered in the Week 4 assignment). We will compare the performance of the preference-tuned and the supervised fine-tuned versions of the GPT-J model[4]. We are going to use existing models: `CarperAI/openai_summarize_tldr_sft`[5], which is the supervised fine-tuned GPT-J, and `CarperAI/openai_summarize_tldr_ppo`[6], which is the preference-tuned GPT-J.[7] Given the constraints on free computation resources available on Google Colab, we have already done the inference for the evaluation posts with both models using a server with 2 $\times$ A100 Nvidia GPUs. We have provided the following files in `week5_implementation.zip`:

- `llm_summarization.py`: An implementation capable of downloading and using models to generate summarizations for given posts – currently, it outputs a warning telling the user that the script will not run in Colab due to resource limitations, and that the output is already provided.
- `input_reddit_posts.txt`: This file contains 5 Reddit posts from TL;DR validation dataset. These posts will be used to evaluate the summarization skills of the models. Note that this file is the same as provided in Week 4 assignment.
- `output_reddit_summaries_week5-sft-tldr.txt` This file contains the output from `CarperAI/openai_summarize_tldr_sft`, the supervised fine-tuned GPT-J model.
- `output_reddit_summaries_week5-ppo-tldr.txt` This file contains the output from `CarperAI/openai_summarize_tldr_ppo`, the preference-tuned GPT-J model.

As part of this assignment, you will simply have to review and compare the generated summaries that are given in the following `.txt` files: `output_reddit_summaries_week5-sft-tldr.txt` and `output_reddit_summaries_week5-ppo-tldr.txt`.

(I.4) Provide a qualitative comparison of the generated summaries from the preference-tuned GPT-J model and the supervised fine-tuned GPT-J model, focusing on how they differ in terms of alignment with the goal of producing summaries that are more comprehensive and provide more context. This should be answered in the PDF submission file – see instructions in Section 4.

- For each evaluation sample (i.e., a post), do a pairwise comparison of the three summaries generated by both models (e.g., compare SUMMARY 1-A from the preference-tuned model with SUMMARY 1-A from the supervised fine-tuned model, SUMMARY 1-B with SUMMARY 1-B, and SUMMARY 1-C with SUMMARY 1-C). For these pairwise comparisons, which model produces the summary that better aligns with the desired characteristics? Report the number of times the preference-tuned GPT-J model wins / loses / ties for these three comparisons for each evaluation sample (i.e., a post).
- Then, provide an overall qualitative comparison between the two models across all the evaluation samples (i.e., all posts). You can answer this question in a few sentences and no detailed analysis is required.

[4]This model is not related to the GPT family of models by OpenAI. However, it is also a GPT-2-like language model. You can read more details about this model at https://huggingface.co/EleutherAI/gpt-j-6b.

[5]Details about this model are at https://huggingface.co/CarperAI/openai_summarize_tldr_sft.

[6]Details about this model are at https://huggingface.co/CarperAI/openai_summarize_tldr_ppo.

[7]More information about the training processes at https://wandb.ai/carperai/summarize_RLHF/reports/Implementing-RLHF-Learning-to-Summarize-with-trlX--VmlldzozMzAwODM2

4 Submission Instructions

Please submit the following two files:

- **<Lastname>_<Matriculation>_week5.pdf**. This PDF file will report answers to the following questions: E.1, E.3, E.4, E.5, I.3, and I.4. If you would like, you can also report answer to the optional question E.2. The PDF file should be generated in LaTeX using NeurIPS 2024 style files; see further formatting instructions in Section 5.1. The PDF file size should **not exceed** 1**mb**. Please use the following naming convention: Replace <Lastname> and <Matriculation> with your respective Lastname and Matriculation number (e.g., Singla_1234567_week5.pdf).
- **<Lastname>_<Matriculation>_week5.zip**. This ZIP file will report answers to the following questions: I.1 and I.2. Importantly, the ZIP file should unzip to a folder named **<Lastname>_<Matriculation>_week5**. Inside this folder, include two `.txt` files containing the generated completions. The ZIP file size should **not exceed** 1**mb**. Replace <Lastname> and <Matriculation> with your respective Lastname and Matriculation number (e.g., Singla_1234567_week5.zip).

5 Technical Instructions

5.1 Instructions for Preparing PDFs

Refer to Week 1 assignment.

5.2 Instructions for Implementation

Refer to Week 4 assignment.

References

[1] Long Ouyang et al. Training Language Models to Follow Instructions with Human Feedback. In *NeurIPS*, 2022.

[2] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *NeurIPS*, 2023.

[3] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347, 2017.

[4] Souradip Chakraborty, Jiahao Qiu, Hui Yuan, Alec Koppel, Dinesh Manocha, Furong Huang, Amrit Bedi, and Mengdi Wang. MaxMin-RLHF: Alignment with Diverse Human Preferences. In *ICML*, 2024.

[5] Jiwoo Hong, Noah Lee, and James Thorne. ORPO: Monolithic Preference Optimization without Reference Model. In *EMNLP*, 2024.

Based on template from `https://media.neurips.cc/Conferences/NeurIPS2024/Styles.zip`