Тестовое Alignment

Тестовое задание:

Reinforcement Learning from Human Feedback — фундаментальный метод алаймента языковых моделей. Он появился достаточно давно, но активно используется до сих пор. Его проблема заключается в том, что он очень нестабилен и сложен в реализации. Это связано с тем, что в основе алгоритма используется алгоритм PPO.

В работе **Back to Basics: Revisiting REINFORCE Style Optimization for Learning from Human Feedback in LLMs** (https://arxiv.org/pdf/2402.14740v1) было показано, что более простой метод REINFORCE способен решать некоторые задачи алаймента лучше, чем PPO.

Более новая работа **WARP: On the Benefits of Weight Averaged Rewarded Policies (**https://arxiv.org/pdf/2406.16768) предлагает модификацию REINFORCE с использованием различных техник объединений моделей. Ваша задача разобраться в этой статьей и реализовать предлагаемый в ней метод.

Задание:

- Обязательно прочитайте правила выполнения задания. Не игнорируйте их.
- Возьмите датасет imdb — датасет бинарной классификации сантиментов для отзывов к фильмам;
- Составьте все возможные пары вида (positive comment, negative comment) из train подвыборки. Обучите модель наград на полученном наборе данных (рекомендуется использовать библиотеку trl). В качестве базовой модели рекомендуется взять distilbert-base-cased.
- Реализуйте метод WARP

Algorithm 1 *WARP* for KL-reward Pareto optimal alignment

Input: Weights θ_{sft} pre-trained and supervised fine-tuned
Reward model r, prompt dataset $\mathcal{X}$, optimizer Opt
I iterations with M RL runs each for T training steps
μ *EMA* update rate, η *LITI* update rate

Define $\theta_{\text{init}} \leftarrow \theta_{\text{sft}}$
for iteration i from 1 to I **do**
  for run m from 1 to M **do** ▷ Run in parallel
    Define $\theta^m, \theta^m_{\text{ema}} \leftarrow \theta_{\text{init}}$
    for step t from 1 to T **do**
      Generate completion $\boldsymbol{y} \sim \pi_{\theta^m}(\cdot \mid \boldsymbol{x})$ for $\boldsymbol{x} \in \mathcal{X}$
      Compute $r_\beta(\boldsymbol{y}) \leftarrow r(\boldsymbol{x}, \boldsymbol{y}) - \beta \log \frac{\pi_{\theta^m}(\boldsymbol{y}|\boldsymbol{x})}{\pi_{\theta^m_{\text{ema}}}(\boldsymbol{y}|\boldsymbol{x})}$ ▷ KL regularized reward
      Update $\theta^m \leftarrow \text{Opt}\big(\theta^m, r_\beta(\boldsymbol{y}) \nabla_\theta [\log \pi_{\theta^m}(\boldsymbol{y} \mid \boldsymbol{x})]\big)$ ▷ Policy gradient
      Update $\theta^m_{\text{ema}} \leftarrow (1-\mu) \cdot \theta^m_{\text{ema}} + \mu \cdot \theta^m$ ▷ Equation (*EMA*): update anchor
    end for
  end for
  Define $\theta^i_{\text{slerp}} \leftarrow \text{slerp}\Big(\theta_{\text{init}}, \{\theta^m\}_{m=1}^M, \lambda = \frac{1}{M}\Big)$ ▷ Equation (*SLERP*): merge M weights
  Update $\theta_{\text{init}} \leftarrow (1-\eta) \cdot \theta_{\text{init}} + \eta \cdot \theta^i_{\text{slerp}}$ ▷ Equation (*LITI*): interpolate towards init
end for

Output: KL-reward Pareto front of weights $\{(1-\eta) \cdot \theta_{\text{sft}} + \eta \cdot \theta^I_{\text{slerp}} \mid 0 \le \eta \le 1\}$

(всего 14 строчек 🙂)

- В качестве θ_{sft} возьмите gpt2-imdb;
- В качестве модели r — модель наград, обученную в предыдущем пункте;
- Датасет промптов $\mathcal{X}$ составьте самостоятельно (можно обрезать тексты из train подвыборки imdb датасета, оставив первые 5 - 15 токенов).
- $I = 2$
- $M = 2$
- $T = 100$
- $\mu = 0.01$
- $\lambda = 0.5$
- $\nu = 0.5$
- batch size = 64

    - остальные параметры возьмите из статьи или основываясь на своих представлениях прекрасного
- Составьте датасет из 100 промптов из test подвыборки imdb (обрежьте последовательности, оставив 5 - 20 первых токенов). Сгенерируйте продолжения для этих промптов, используя обученную модель. Замерьте среднюю награду и среднюю KL дивергенцию обученной модели с θ_{sft}. Сделайте такой же замер для генераций sft модели. Получилось ли увеличить среднее значение награды?
- Выберите один из гиперпараметров ($I, M, T, \mu, \lambda, \eta$). Обучите модель с двумя другими значениями выбранного гиперпараметра. Измерьте среднюю награду и KL, отобразите три точки на графике (три значения гиперпараметра), аналогичном графикам из статьи.
- Проанализируйте полученные результаты. Это важный пункт, потому что хочется увидеть не только числа с полученными метриками. Как вы объясняете уведенное поведение?
- Напишите небольшой отчет о проведенных экспериментах. Что получилось? Что нет?
- **Bonus**: любопытно, можно ли улучшить результаты, которые показывает WARP? Придумайте способ улучшить полученные результаты произвольным новым способом (можно попробовать применить хитрости из статьи **Back to Basics**).

Правила:

- **Нет правильного способа решить задачу.** Не стоит беспокоиться, что вы делаете что-то неправильно. Мы хотим увидеть ваши способности к исследованиям, а не какое-то конкретное решение задачи. Возможно, вы придумаете то, о чем мы даже не задумывались изначально – это будет самый высший класс.
- Убедитесь в том, что результатам можно доверять. Исключите вариант случайности, etc.

- Вы можете использовать Google Colab, чтобы получить доступ к бесплатным вычислительным ресурсам.
- Присылайте решение в виде репозитория на github с отчетом по решению и чёткими инструкциями, как запустить ваш код. **Убедитесь, что мы сможем запустить ваше решение по этим инструкциям.**
- Вы можете использовать любые библиотеки и фреймворки, которые вам могут быть необходимы.
- Рекомендуется изучить код библиотеки trl, там есть много полезной логики:
  - https://github.com/huggingface/trl/blob/main/trl/trainer/ppo_trainer.py
  - https://github.com/huggingface/trl/blob/main/trl/trainer/ppov2_trainer.py
- **Сфокусируйтесь на том, чтобы код был чист и понятен**. Если вы считаете, что какая-то его часть может быть непонятна, то добавьте комментарии. Мы очень сильно ценим хорошо написанный код, поэтому **если решение задачи будет оформлено грязно, то мы можем отклонить заявку.**