Direct Preference Optimization: Your Language Model is Secretly a Reward Model

Denis Tarasov

ETH Zurich
Seminar in Deep Neural Networks
27.02.2024
Warning

This presentation contains some sensitive content which is required for motivation of the problem.
We want to build good generative AI systems

- To easily acquire information
- To increase our productivity
- To make our work easier
- To help us with advices
But we also need them to “behave well”
But we also need them to “behave well”
But we also need them to “behave well”

We need our models to be well-aligned

Data quality:
Alignment for Language Models

We want our Language Models (LMs) to provide useful but safe responses.

**Sensitive question:** Why are prisons full of Black and Brown people?

**Harmful:** Because they all are criminals.

**Useless:** Sorry, I cannot respond to this content.

**Good:** That is a very serious problem. Research has shown that Black and Brown people, especially men, are disproportionately incarcerated compared to white people in the United States due to systemic racial biases throughout the criminal justice system.
Alignment for Language Models

When generating code we want to get the best solutions.

**Programming questions:** How do I find maximum element in the Python list \( arr \)?

**Sub-optimal:**
```python
sorted_arr = sorted(arr)
maximum = sorted_arr[-1]
```

**Overcomplicated:**
```python
maximum = arr[0]
for a in arr:
    maximum = max(maximum, a)
```

**Perfect:**
```python
maximum = max(arr)
```
Training autoregressive LMs

Input text → Tokens

Language Model

t0 → t1 → t2 → t3

Tokens
Training Large LMs

Huge internet dump → Pre-trained LM → Fine-tuned LM → Task-specific dataset

Self-supervised training

\[
\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i
\]

Supervised training

Keep only best examples
or
Modify loss to, e.g. decrease likelihood of bad examples

Simplest options for alignment
In-context learning

Zero-shot
The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:
2  cheese => ..........................................................
```

Few-shot
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1  Translate English to French:
2  sea otter => loutre de mer
3  peppermint => menthe poivrée
4  plush giraffe => girafe peluche
5  cheese => ..........................................................
```

One-shot
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1  Translate English to French:
2  sea otter => loutre de mer
3  cheese => ..........................................................
```

Examples from “Language Models are Few-Shot Learners” (Brown et al., 2020)
Reinforcement Learning for LMs

Objective is to maximize:

\[
E_{\tau \sim p_\pi(\tau)} \left[ \sum_{t=0}^{H} \gamma^t r(s_t, a_t) \right]
\]

Reinforcement Learning from Human Feedback

Step 1
Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.

Step 2
Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Step 3
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

Schema from Ouyang et al. 2022
Why is RLHF good?

- It is possible to optimize any numerical objective with Reinforcement Learning.
- We can train for multiple objectives at the same time with it, e.g. being useful but not toxic.
- RLHF has provided us with the most powerful LMs, e.g. ChatGPT, Claude, Gemini, Copilot.
Problems with RLHF

Step 1
Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

Reward model is a function approximation.

This data is used to fine-tune GPT-3 with supervised learning.

Step 2
A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.

Step 3
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.

RL is hard to train

Schema from Ouyang et al. 2022
Direct Policy Optimization goal

Figure from "Direct Preference Optimization: Your Language Model is Secretly a Reward Model" (Rafailov et al. 2023)
Bradley-Terry preference model for RLHF

Bradley-Terry:

\[ p^*(y_1 \succ y_2 \mid x) = \frac{\exp(r^*(x, y_1))}{\exp(r^*(x, y_1)) + \exp(r^*(x, y_2))} \]

Reward model objective:

\[ \mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[ \log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l)) \right] \]

RLHF policy objective

\[ \max_{\pi_\theta} \mathbb{E}_{x \sim D, y \sim \pi_\theta(y | x)} [r_\phi(x, y)] - \beta \mathbb{D}_{KL} [\pi_\theta(y | x) \parallel \pi_{\text{ref}}(y | x)] \]

Maximizing the reward

Forcing policy to have distribution similar to SFT model
DPO derivation

**Optimal solution for RL objective**

\[
\pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y \mid x) \exp \left( \frac{1}{\beta} r(x, y) \right)
\]

\[
Z(x) = \sum_y \pi_{\text{ref}}(y \mid x) \exp \left( \frac{1}{\beta} r(x, y) \right)
\]

Intractable to compute
DPO derivation

Optimal solution for RL objective

\[ \pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{ref}(y \mid x) \exp\left( \frac{1}{\beta} r(x, y) \right) \]

\[ r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{ref}(y \mid x)} + \beta \log Z(x) \]

Bradley-Terry model

\[ p^*(y_1 \succ y_2 \mid x) = \frac{\exp(\Delta^*(x, y_1))}{\exp(\Delta^*(x, y_1)) + \exp(\Delta^*(x, y_2))} \]

\[ p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left( \beta \log \frac{\pi^*(y_2 \mid x)}{\pi_{ref}(y_2 \mid x)} - \beta \log \frac{\pi^*(y_1 \mid x)}{\pi_{ref}(y_1 \mid x)} \right)} \]

\[ \mathcal{L}_{DPO}(\pi_\theta; \pi_{ref}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[ \log \sigma \left( \beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{ref}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{ref}(y_l \mid x)} \right) \right] \]
DPO gradient

\[ L_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l) \sim \mathcal{D}} \left[ \log \sigma \left( \beta \log \frac{\pi_\theta(y_w | x)}{\pi_{\text{ref}}(y_w | x)} - \beta \log \frac{\pi_\theta(y_l | x)}{\pi_{\text{ref}}(y_l | x)} \right) \right] \]

\[ \nabla_\theta L_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\beta \mathbb{E}_{(x,y_w,y_l) \sim \mathcal{D}} \left[ \sigma(\hat{r}_\theta(x,y_l) - \hat{r}_\theta(x,y_w)) \left[ \nabla_\theta \log \pi(y_w | x) - \nabla_\theta \log \pi(y_l | x) \right] \right] \]

higher weight when reward estimate is wrong

increase likelihood of \( y_w \)

decrease likelihood of \( y_l \)
Theoretical analysis of DPO

**Def:** reward functions $r(x, y)$ and $r'(x, y)$ are equivalent iff $r(x, y) - r'(x, y) = f(x)$ for some function $f$

**Lm 1:** Two equivalent reward functions induce the same preference distribution.

**Lm 2:** Two equivalent reward functions induce the same optimal RL policy.

**Theorem:** Under some assumptions, all reward classes consistent with Bradley-Terry models can be represented with the reparametrization $r(x, y) = \beta \log \frac{\pi(y|x)}{\pi_{ref}(y|x)}$ for some model $\pi(y \mid x)$ and a given reference model $\pi_{ref}(y \mid x)$
Experimental results. Sampling temperature sensitivity
### Human preferences against RLHF

<table>
<thead>
<tr>
<th></th>
<th>DPO</th>
<th>SFT</th>
<th>PPO-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>N respondents</td>
<td>272</td>
<td>122</td>
<td>199</td>
</tr>
<tr>
<td>GPT-4 (C) win %</td>
<td>54</td>
<td>32</td>
<td>12</td>
</tr>
<tr>
<td>Human win %</td>
<td>58</td>
<td>43</td>
<td>17</td>
</tr>
</tbody>
</table>
Conclusion

- **DPO is an elegant approach** which rewrites RLHF objective for the preference optimization into supervised learning objective.
- **DPO helps to get rid of 3 major RLHF problems**: explicit reward model training, LM output sampling during training and RL pipeline. So it is easier much easier to run.
- However, this approach is **tested only for “small” model sizes**, proposed only for Bradley-Terry preference model and applied only to NLP problem.