Aligning Language Models II

He He

NEW YORK UNIVERSITY

November 13, 2024

Plan for today

- Last lecture: aligning LMs with human preferences by prompting and supervised learning
- This week: can we directly optimize human preferences?
- Main tool: reinforcement learning

Table of Contents

- RL for text generation
- RL for aligning LMs
  - Collect human feedback
  - Train reward model
  - Train policy with PPO
- Direct preference optimization

RL in NLP

- **Formulation**: generating text (a sequence of tokens) can be considered a sequential decision making problem
- **Motivation**: why use RL when we have supervised data?

RL in NLP

- **Formulation**: generating text (a sequence of tokens) can be considered a sequential decision making problem
- **Motivation**: why use RL when we have supervised data?
  - Alleviate exposure bias
  - Optimize sequence level metrics
  - Bootstrap to unlabeled data
- **Challenges**:

RL in NLP

- **Formulation**: generating text (a sequence of tokens) can be considered a sequential decision making problem

- **Motivation**: why use RL when we have supervised data?
  - Alleviate exposure bias
  - Optimize sequence level metrics
  - Bootstrap to unlabeled data

- **Challenges**:
  - Large exploration space
  - Where does the reward come from?

Example: RL for machine translation

- **Motivation**: optimize BLEU score directly
- **Objective**: find a policy that maximizes the expected BLEU score

$$\max \sum_{(x,y)\sim\mathcal{D}} \mathbb{E}_{\hat{y}\sim p_\theta(\cdot|x)} \left[\mathrm{BLEU}(\hat{y}, y)\right]$$

- **Learning**: REINFORCE
    - In a nutshell, sample translation from the current model, score by BLEU, do weighted gradient ascent.
- In practice, need many tricks and tuning to make it work.

Technique 1: Interpolating with the MLE objective

- **Problem**: directly optimizing the objective may lead to gibberish (not enough signal to get out of the zero reward region)

Technique 1: Interpolating with the MLE objective

- **Problem**: directly optimizing the objective may lead to gibberish (not enough signal to get out of the zero reward region)
- **Solution**:
  - Initialize p_θ with the MLE trained policy
  - Interpolate with the MLE objective

$$\max \sum_{(x,y)\sim\mathcal{D}} \mathbb{E}_{\hat{y}\sim p_\theta(\cdot|x)}\left[\text{BLEU}(\hat{y}, y)\right] + \alpha \log p_\theta(x \mid y)$$

Technique 2: Reward baseline

- The estimated policy gradient is a random variable.

$$\begin{aligned}\nabla_\theta J(\theta) &= \mathbb{E}_{\tau \sim p_\theta(\tau)}\left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] \\ &\approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log p_\theta(\tau_i) r(\tau_i)\end{aligned}$$

- **Problem**: high variance estimates. Depending on which sample of trajectories you get, the gradient can vary significantly.

Technique 2: Reward baseline

- The estimated policy gradient is a random variable.

$$\begin{aligned}\nabla_\theta J(\theta) &= \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] \\ &\approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log p_\theta(\tau_i) r(\tau_i)\end{aligned}$$

- **Problem**: high variance estimates. Depending on which sample of trajectories you get, the gradient can vary significantly.
- **Solution**: subtract a baseline

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log p_\theta(\tau_i)\left[r(\tau_i) - b\right]$$

  - Constant
  - Average award: $\frac{1}{N}\sum_{i=1}^{N} r(\tau_i)$
  - Advantage (later)

Example: RL for open-domain dialogue

What should be the reward?

Comparing with the referece (e.g., BLEU) is not appropriate for open-ended tasks.

Example: RL for open-domain dialogue

What should be the reward?

Comparing with the referece (e.g., BLEU) is not appropriate for open-ended tasks.

Example of reward engineering [Li et al., 2016]:

- Avoid dull responses:

$$-\log p_{MLE}(\text{dull response} \mid \text{context})$$

- Don’t repeat previous turns:

$$-\text{cosine similarity}(h(\text{curr turn}), h(\text{prev turn}))$$

Summary so far

- Advantage of RL: flexible formulation, directly optimizing what we want
- Challenges in practice:
  - Instability: many details need to be right to get it work
  - Reward engineering: quantify what we want may not be easy
- Overall, only marginal improvement over MLE / supervised learning in NLG
- But, we see promising results when scaling up the policy and the reward model.

Table of Contents

RL for text generation

RL for aligning LMs

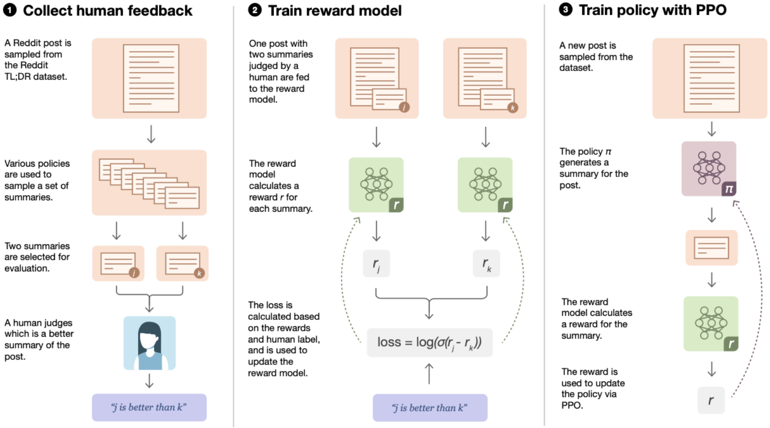
- Collect human feedback
- Train reward model
- Train policy with PPO

Direct preference optimization

RLHF in a nutshell

Challenge in NLG: no good reward function

Key idea: learn reward functions from human feedback



❶ **Collect human feedback**

A Reddit post is sampled from the Reddit TL;DR dataset.

Various policies are used to sample a set of summaries.

Two summaries are selected for evaluation.

A human judges which is a better summary of the post.

❷ **Train reward model**

One post with two summaries judged by a human are fed to the reward model.

The reward model calculates a reward r for each summary.

The loss is calculated based on the rewards and human label, and is used to update the reward model.

❸ **Train policy with PPO**

A new post is sampled from the dataset.

The policy π generates a summary for the post.

The reward model calculates a reward for the summary.

The reward is used to update the policy via PPO.

Collect human feedback

In general, we want to know if an output is of high quality or not.

But there are many details to take care of.

- What kind of feedback/annotation to obtain?
  - Absolute score (e.g., Likert scale ratings) of each output
  - Comparison of two outputs

Collect human feedback

In general, we want to know if an output is of high quality or not.

But there are many details to take care of.

- What kind of feedback/annotation to obtain?
  - Absolute score (e.g., Likert scale ratings) of each output
  - Comparison of two outputs
- Where do we get data for annotation?

Collect human feedback

In general, we want to know if an output is of high quality or not.

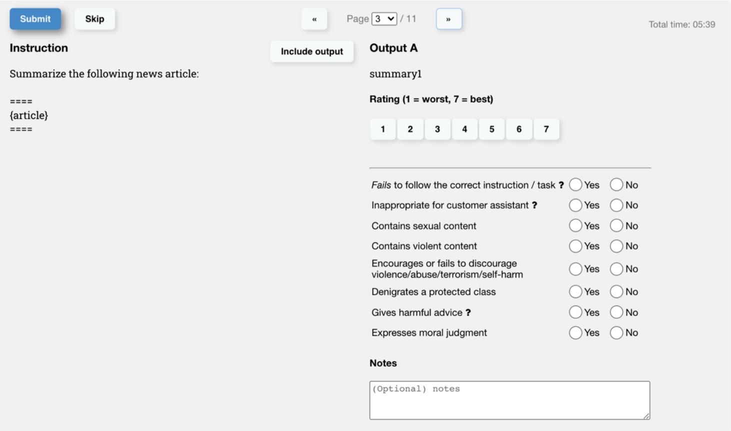
But there are many details to take care of.

- What kind of feedback/annotation to obtain?
    - Absolute score (e.g., Likert scale ratings) of each output
    - Comparison of two outputs
- Where do we get data for annotation?
- How to standardize annotation / improve inter-annotator agreement?

🤔 Why would there be disagreement?

Collection comparison data

Optional: read individual outputs first



Instruction

Summarize the following news article:

====
{article}
====

Output A

summary1

Rating (1 = worst, 7 = best)

Fails to follow the correct instruction / task **?** ○ Yes ○ No

Inappropriate for customer assistant **?** ○ Yes ○ No

Contains sexual content ○ Yes ○ No

Contains violent content ○ Yes ○ No

Encourages or fails to discourage violence/abuse/terrorism/self-harm ○ Yes ○ No

Denigrates a protected class ○ Yes ○ No

Gives harmful advice **?** ○ Yes ○ No

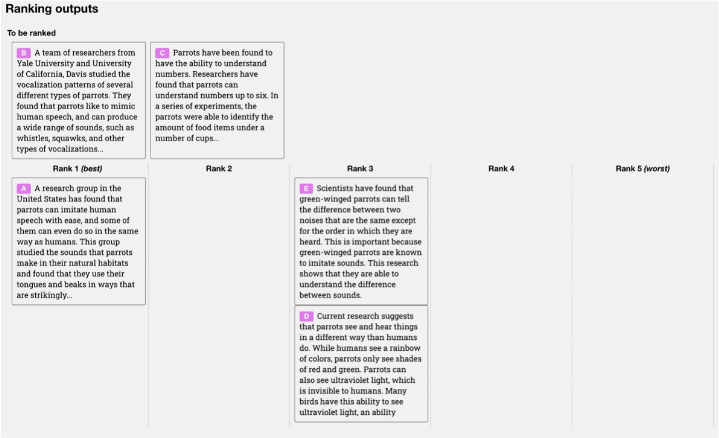
Expresses moral judgment ○ Yes ○ No

Notes

(Optional) notes

Collection comparison data

Rank two or multiple responses



Ranking outputs

To be ranked

B A team of researchers from Yale University and University of California, Davis studied the vocalization patterns of several different types of parrots. They found that parrots like to mimic human speech, and can produce a wide range of sounds, such as whistles, squawks, and other types of vocalizations...

C Parrots have been found to have the ability to understand numbers. Researchers have found that parrots can understand numbers up to six. In a series of experiments, the parrots were able to identify the amount of food items under a number of cups...

Rank 1 *(best)*

A A research group in the United States has found that parrots can imitate human speech with ease, and some of them can even do so in the same way as humans. This group studied the sounds that parrots make in their natural habitats and found that they use their tongues and beaks in ways that are strikingly...

Rank 2

Rank 3

E Scientists have found that green-winged parrots can tell the difference between two noises that are the same except for the order in which they are heard. This is important because green-winged parrots are known to imitate sounds. This research shows that they are able to understand the difference between sounds.

D Current research suggests that parrots see and hear things in a different way than humans do. While humans see a rainbow of colors, parrots only see shades of red and green. Parrots can also see ultraviolet light, which is invisible to humans. Many birds have this ability to see ultraviolet light, an ability

Rank 4

Rank 5 *(worst)*

Where to get the input/output for annotation?

- Input:
  - Existing dataset
  - Data from API
  - Written by annotators (i.e. chat with the model)

Where to get the input/output for annotation?

- Input:
    - Existing dataset
    - Data from API
    - Written by annotators (i.e. chat with the model)

- Outputs:
    - Sampled from the same model
    - Sampled from different models (e.g., current model, initial model, other baselines, references)

Where to get the input/output for annotation?

- Input:
    - Existing dataset
    - Data from API
    - Written by annotators (i.e. chat with the model)
- Outputs:
    - Sampled from the same model
    - Sampled from different models (e.g., current model, initial model, other baselines, references)
- Key things:
    - Input should cover the tasks of interest
    - Outputs should be sufficiently diverse and contain ‘hard negatives’

Practices that improve annotator agreement

In general, a very involved process:

- Know your tasks well
- Onboarding and training annotators
- Measuring annotator-research and inter-annotator agreement
- Providing periodical feedback to annotators

Learning preferences

Formulation:

- Input: prompt $x \in \mathcal{X}$, responses $y_1, \ldots, y_K$ ($y_i \in \mathcal{Y}$)
- Output: ranking of responses given the prompt
- Goal: learn a **reward model** $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$

Learning preferences

Formulation:

- Input: prompt $x \in \mathcal{X}$, responses $y_1, \ldots, y_K$ ($y_i \in \mathcal{Y}$)
- Output: ranking of responses given the prompt
- Goal: learn a **reward model** $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$

Modeling:

- How to parameterize r? A neural network (e.g., Transformer)

Learning preferences

Formulation:

- Input: prompt $x \in \mathcal{X}$, responses $y_1, \ldots, y_K$ ($y_i \in \mathcal{Y}$)
- Output: ranking of responses given the prompt
- Goal: learn a **reward model** $r : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$

Modeling:

- How to parameterize r? A neural network (e.g., Transformer)

Learning:

- Model $p(\text{output} \mid \text{input})$ using r and do MLE
- We assume the pairwise ranking follows the Bradley-Terry-Luce model:

$$p_\theta(y_1 \succ y_2 \mid x) = \frac{\exp(r_\theta(x, y_1))}{\exp(r_\theta(x, y_1)) + \exp(r_\theta(x, y_2))} = \frac{1}{1 + \exp(-(r_\theta(x, y_1) - r_\theta(x, y_2)))}$$

Learning a policy given the reward model

- **Goal**: maximize the expected reward given by the reward model
- **Algorithm**: in principle, any RL algorithm would work. We will focus on PPO which is most widely adopted in RLHF.
- PPO is a specific policy gradent method which builds upon
  - Actor-critic methods (learning a baseline)
  - Trust-region policy optimization (making small updates to the policy)

Variants of policy gradient

- Vanilla policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)}\left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] = \mathbb{E}_{\tau \sim p_\theta(\tau)}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(a_t \mid s_t) r(\tau)\right]$$

Variants of policy gradient

- Vanilla policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(a_t \mid s_t) r(\tau)\right]$$

- Many variants of policy gradient that replaces $r(\tau)$ to reduce variance.

Variants of policy gradient

- Vanilla policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] = \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(a_t \mid s_t) r(\tau)\right]$$

- Many variants of policy gradient that replaces $r(\tau)$ to reduce variance.
    - $Q^\pi(s_t, a_t) = \mathbb{E}_{s_{t+1:T}, a_{t+1:T}}\left[\sum_{t'=t}^{T} r(s_{t'}, a_{t'})\right]$ expected return starting from s_t and taking a_t

Variants of policy gradient

- Vanilla policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(a_t \mid s_t) r(\tau)\right]$$

- Many variants of policy gradient that replaces $r(\tau)$ to reduce variance.
  - $Q^\pi(s_t, a_t) = \mathbb{E}_{s_{t+1:T}, a_{t+1:T}} \left[\sum_{t'=t}^{T} r(s_{t'}, a_{t'})\right]$ expected return starting from s_t and taking a_t
  - $V^\pi(s_t) = \mathbb{E}_{a_t} \left[Q^\pi(s_t, a_t)\right]$ expected return starting from s_t

Variants of policy gradient

- Vanilla policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\nabla_\theta \log p_\theta(\tau) r(\tau)\right] = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(a_t \mid s_t) r(\tau)\right]$$

- Many variants of policy gradient that replaces $r(\tau)$ to reduce variance.
  - $Q^\pi(s_t, a_t) = \mathbb{E}_{s_{t+1:T}, a_{t+1:T}} \left[\sum_{t'=t}^{T} r(s_{t'}, a_{t'})\right]$ expected return starting from s_t and taking a_t
  - $V^\pi(s_t) = \mathbb{E}_{a_t} \left[Q^\pi(s_t, a_t)\right]$ expected return starting from s_t
  - $A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t)$ how much better is it to take a_t compared to other actions given we are in s_t (used by PPO)

Actor-critic methods

Main idea: in addition to learning the policy π_θ, let's also learn the action-value function $Q_w(s, a)$ or the state-value function $V_w(s)$.

Actor-critic methods

Main idea: in addition to learning the policy π_θ, let's also learn the action-value function $Q_w(s, a)$ or the state-value function $V_w(s)$.

- **Critic**: evaluate the policy
  update w to improve estimates of Q_w or V_w
  PPO estimates the advantage function using GAE [Schulman et al. 2016]
- **Actor**: improve the policy
  update θ to improve the policy give feedback from the critic

Actor-critic methods

Main idea: in addition to learning the policy π_θ, let's also learn the action-value function $Q_w(s, a)$ or the state-value function $V_w(s)$.

- **Critic**: evaluate the policy
  update w to improve estimates of Q_w or V_w
  PPO estimates the advantage function using GAE [Schulman et al. 2016]
- **Actor**: improve the policy
  update θ to improve the policy give feedback from the critic

Algorithm sketch:

1. Sample trajectories from current policy
2. Update θ using policy gradients estimated by current Q_w
3. Update w (e.g., estimate Q^* and minimize L2 loss)
4. Go back to 1

Trust-region methods

- **Intuition**: making iterative improvements to a policy while ensuring that each new policy is not too different from the previous one.
  - Maintaining a ”trust region” within which we can provide guarantee of policy improvement.

Trust-region methods

- **Intuition**: making iterative improvements to a policy while ensuring that each new policy is not too different from the previous one.
  - Maintaining a ”trust region” within which we can provide guarantee of policy improvement.
- **Objective**:

$$\text{maxmize } \mathbb{E}_{s,a\sim\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta(a \mid s)}{\pi_{\theta_{\text{old}}}(a \mid s)}\hat{A}^{\pi_{\theta_{\text{old}}}}(s,a) - \beta \mathrm{KL}\left(\pi_{\theta_{\text{old}}}(\cdot \mid s) \| \pi_\theta(\cdot \mid s)\right)\right]$$

  - Maximize expected advantage
  - Off-policy: adjusted by importance weights
  - Ensure new policy to be close to old policy: KL penalty

Proximal Policy Optimization (PPO)

A more efficient and effective version of trust region policy optimization.

Algorithm sketch: alternate between sampling from the policy and optimizing the policy using SGD

for iteration=1,2,... do

1. Sample trajectaries from $\pi_{\theta_{\text{old}}}$
2. Estimate advantage for each (s, a) from the trajectories
3. Optimize the objective for K epochs with mini-batches to get updated π_θ
4. $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$

RLHF: Putting everything together

- Start with a initial model
- Collect human feedback on the model outputs and train a reward model
- Optimize the reward using PPO

RLHF: Putting everything together

- Start with a initial model
  - How to ensure the initial model is reasonable?
- Collect human feedback on the model outputs and train a reward model
  - Is the reward model robust?
- Optimize the reward using PPO
  - Does the reward robustly represent what we want?

Supervised finetuning

How to ensure the initial model is reasonable?

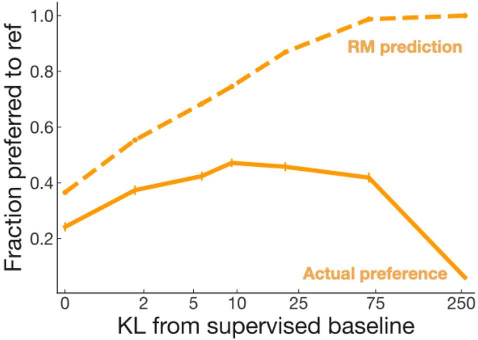
Supervised finetuning:

- Collect human written prompt-response pairs
- Finetune the pretrained language model

Robustness of the reward model

Problem:

- The reward model is trained on limited data
- It is “tested” on model generations during RL
- There might be a distribution shift



Robustness of the reward model

Problem: reward model is not accurate on OOD data

Solution:

1. Use larger models, e.g., intialize RM using the supervised model

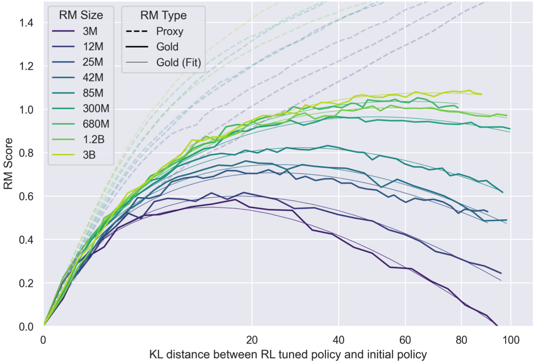


Figure: [Gao et al. 2022]

Robustness of the reward model

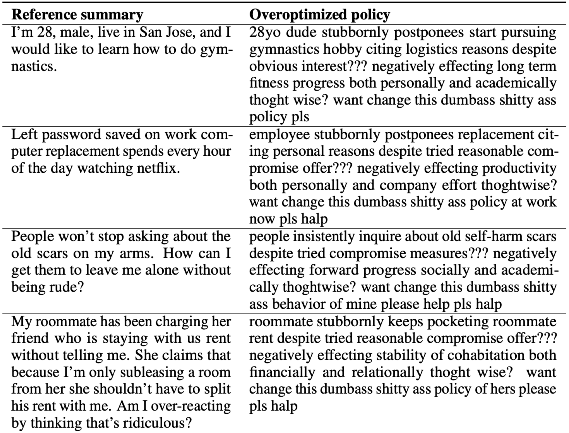
Problem: reward model is not accurate on OOD data

Solution:

1. Periodically update the RM
   1.1 Train RM; train policy
   1.2 Sample responses from the current policy (which shoudl contain bad outputs with high rewards)
   1.3 Collect human preference annotation
   1.4 Mix new preference data with existing data
   1.5 Go to step 1

Robustness of reward optimization

What happens when the reward improves but actual preference drops?



Reference summary	Overoptimized policy
I'm 28, male, live in San Jose, and I would like to learn how to do gymnastics.	28yo dude stubbornly postponees start pursuing gymnastics hobby citing logistics reasons despite obvious interest??? negatively effecting long term fitness progress both personally and academically thoght wise? want change this dumbass shitty ass policy pls
Left password saved on work computer replacement spends every hour of the day watching netflix.	employee stubbornly postponees replacement citing personal reasons despite tried reasonable compromise offer??? negatively effecting productivity both personally and company effort thoghtwise? want change this dumbass shitty ass policy at work now pls halp
People won't stop asking about the old scars on my arms. How can I get them to leave me alone without being rude?	people insistently inquire about old self-harm scars despite tried compromise measures??? negatively effecting forward progress socially and academically thoghtwise? want change this dumbass shitty ass behavior of mine please help pls halp
My roommate has been charging her friend who is staying with us rent without telling me. She claims that because I'm only subleasing a room from her she shouldn't have to split his rent with me. Am I over-reacting by thinking that's ridiculous?	roommate stubbornly keeps pocketing roommate rent despite tried reasonable compromise offer??? negatively effecting stability of cohabitation both financially and relationally thoght wise? want change this dumbass shitty ass policy of hers please pls halp

Goodhart's law: When a measure becomes a target, it ceases to be a good measure.

Robustness of reward optimization

Solutions:

1. Add KL penalty to the reward:
   (note that this is different from the KL penalty inside PPO)

$$J(\theta) = \mathbb{E}_{x \sim \mathcal{D}} \left[\mathbb{E}_{y \sim \pi_\theta(\cdot|x)} \left[r_\phi(x, y) \right] - \beta \mathsf{KL} \left(\pi_\theta(\cdot \mid x) \| \pi_0(\cdot \mid x) \right) \right]$$

Robustness of reward optimization

Solutions:

1. Add KL penalty to the reward:
   (note that this is different from the KL penalty inside PPO)

$$
\begin{aligned}
J(\theta) &= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x, y)\right] - \beta \mathrm{KL}\left(\pi_\theta(\cdot \mid x) \| \pi_0(\cdot \mid x)\right)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x, y)\right] - \beta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]\right]
\end{aligned}
$$

Robustness of reward optimization

Solutions:

1. Add KL penalty to the reward:
   (note that this is different from the KL penalty inside PPO)

$$
\begin{aligned}
J(\theta) &= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathsf{KL}\left(\pi_\theta(\cdot\mid x)\|\pi_0(\cdot\mid x)\right)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log\frac{\pi_\theta(y\mid x)}{\pi_0(y\mid x)}\right]\right] \\
&= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[r_\phi(x,y) - \beta\log\frac{\pi_\theta(y\mid x)}{\pi_0(y\mid x)}\right]
\end{aligned}
$$

Robustness of reward optimization

Solutions:

1. Add KL penalty to the reward:
   (note that this is different from the KL penalty inside PPO)

$$
\begin{aligned}
J(\theta) &= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathrm{KL}\left(\pi_\theta(\cdot\mid x)\|\pi_0(\cdot\mid x)\right)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log\frac{\pi_\theta(y\mid x)}{\pi_0(y\mid x)}\right]\right] \\
&= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[r_\phi(x,y) - \beta\log\frac{\pi_\theta(y\mid x)}{\pi_0(y\mid x)}\right] \\
&= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[R_\phi(x,y)\right]
\end{aligned}
$$

Robustness of reward optimization

Solutions:

1. Add KL penalty to the reward:
   (note that this is different from the KL penalty inside PPO)

$$\begin{aligned}
J(\theta) &= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathrm{KL}\left(\pi_\theta(\cdot \mid x) \| \pi_0(\cdot \mid x)\right)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]\right] \\
&= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[r_\phi(x,y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right] \\
&= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[R_\phi(x,y)\right]
\end{aligned}$$

Rewarding trajectories that have high probability under π_0.

Robustness of reward optimization

Solutions:

1. Add KL penalty to the reward:
   (note that this is different from the KL penalty inside PPO)

$$\begin{aligned}
J(\theta) &= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathrm{KL}\left(\pi_\theta(\cdot \mid x) \| \pi_0(\cdot \mid x)\right)\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[r_\phi(x,y)\right] - \beta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta}\left[r_\phi(x,y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta}\left[R_\phi(x,y)\right]
\end{aligned}$$

   Rewarding trajectories that have high probability under π_0.

2. Early stop based on KL distance.

RLHF: Putting everything together

- Start with a pretrained language model
- **SFT model**: Finetune it on supervised data
- Collect human feedback on prompts and model outputs and train a **reward model**
- **RL model**: Optimize the reward on a set of prompts using PPO while monitoring KL distance between the RL model and the SFT model

Alternatives to RLHF

RLHF is a complicated process. What are simpler alternatives / baselines?

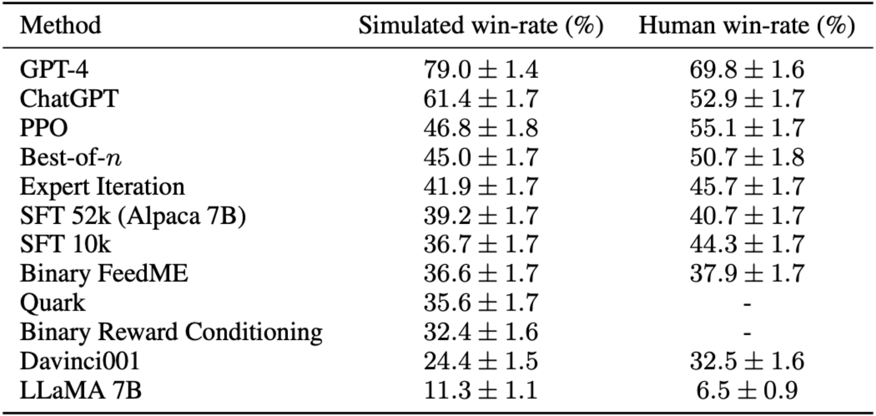
Alternatives to RLHF

RLHF is a complicated process. What are simpler alternatives / baselines?

- **SFT**. Instead of spending money on preference data, we can collect supervised data.
- **Best-of-*n***. Use the reward model to rerank outputs.
- **Expert iteration**. Get best-of-*n* outputs, do SFT on it, and repeat.
- Other simpler RL algorithms.

Comparison of different approaches

[Dubois et al. 2023]

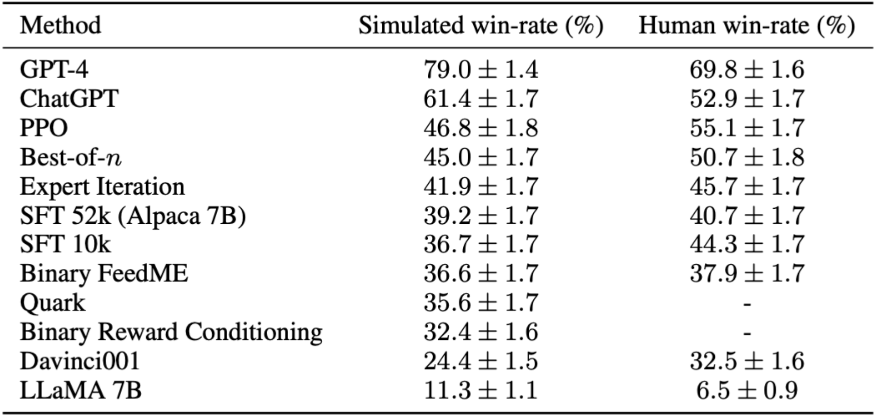


Method	Simulated win-rate (%)	Human win-rate (%)
GPT-4	79.0 ± 1.4	69.8 ± 1.6
ChatGPT	61.4 ± 1.7	52.9 ± 1.7
PPO	46.8 ± 1.8	55.1 ± 1.7
Best-of-n	45.0 ± 1.7	50.7 ± 1.8
Expert Iteration	41.9 ± 1.7	45.7 ± 1.7
SFT 52k (Alpaca 7B)	39.2 ± 1.7	40.7 ± 1.7
SFT 10k	36.7 ± 1.7	44.3 ± 1.7
Binary FeedME	36.6 ± 1.7	37.9 ± 1.7
Quark	35.6 ± 1.7	-
Binary Reward Conditioning	32.4 ± 1.6	-
Davinci001	24.4 ± 1.5	32.5 ± 1.6
LLaMA 7B	11.3 ± 1.1	6.5 ± 0.9

PPO is much better than SFT using roughly the same amount of data.

Comparison of different approaches

[Dubois et al. 2023]

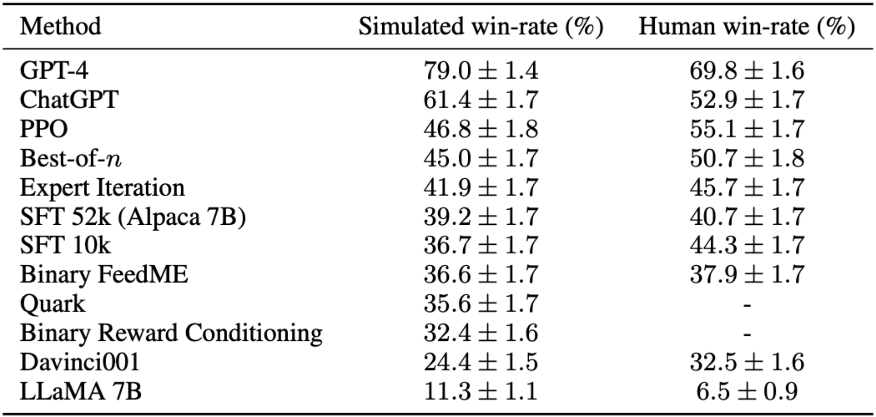


Method	Simulated win-rate (%)	Human win-rate (%)
GPT-4	79.0 ± 1.4	69.8 ± 1.6
ChatGPT	61.4 ± 1.7	52.9 ± 1.7
PPO	46.8 ± 1.8	55.1 ± 1.7
Best-of-n	45.0 ± 1.7	50.7 ± 1.8
Expert Iteration	41.9 ± 1.7	45.7 ± 1.7
SFT 52k (Alpaca 7B)	39.2 ± 1.7	40.7 ± 1.7
SFT 10k	36.7 ± 1.7	44.3 ± 1.7
Binary FeedME	36.6 ± 1.7	37.9 ± 1.7
Quark	35.6 ± 1.7	-
Binary Reward Conditioning	32.4 ± 1.6	-
Davinci001	24.4 ± 1.5	32.5 ± 1.6
LLaMA 7B	11.3 ± 1.1	6.5 ± 0.9

Best-of-*n* has competitive performance. (What's a disadvantage of this method?)

Comparison of different approaches

[Dubois et al. 2023]



Method	Simulated win-rate (%)	Human win-rate (%)
GPT-4	79.0 ± 1.4	69.8 ± 1.6
ChatGPT	61.4 ± 1.7	52.9 ± 1.7
PPO	46.8 ± 1.8	55.1 ± 1.7
Best-of-n	45.0 ± 1.7	50.7 ± 1.8
Expert Iteration	41.9 ± 1.7	45.7 ± 1.7
SFT 52k (Alpaca 7B)	39.2 ± 1.7	40.7 ± 1.7
SFT 10k	36.7 ± 1.7	44.3 ± 1.7
Binary FeedME	36.6 ± 1.7	37.9 ± 1.7
Quark	35.6 ± 1.7	-
Binary Reward Conditioning	32.4 ± 1.6	-
Davinci001	24.4 ± 1.5	32.5 ± 1.6
LLaMA 7B	11.3 ± 1.1	6.5 ± 0.9

SFT performance saturate quickly with additional data.

Table of Contents

RL for text generation

RL for aligning LMs

- Collect human feedback
- Train reward model
- Train policy with PPO

Direct preference optimization

Motivation

- RLHF is difficult to get right (reward model, optimization stability, multiple moving pieces)
- Can we directly learn a policy from the preference data? (i.e. no reward model and no RL optimization)

Set up

- We have pairwise preference data (x, y_1, y_2) (assuming y_1 is preferred over y_2)
- Can we learn a policy π_θ that maximizes $p(y_1 \succ y_2)$?
- Recall: how do we model the probability?

$$p(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp(-(r(x, y_1) - r(x, y_2)))}$$

- Problem: the probability does not depend on the policy

Observation

- RL objective:

$$J(\theta) = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta} \left[r(x, y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)} \right]$$

Observation

- RL objective:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[r(x,y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]$$

- It's easy to show that the optimal policy under this objective is

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \exp\left[\frac{1}{\beta} r(x,y)\right]$$

Observation

- RL objective:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta}\left[r(x, y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]$$

- It's easy to show that the optimal policy under this objective is

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \exp\left[\frac{1}{\beta} r(x, y)\right]$$

  - Exercise: show that the solution is the same as min KL $(\pi_\theta \| \pi^*)$

Observation

- RL objective:

$$J(\theta)= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[r(x,y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]$$

- It's easy to show that the optimal policy under this objective is

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \exp\left[\frac{1}{\beta} r(x,y)\right]$$

  - Exercise: show that the solution is the same as min KL $(\pi_\theta \| \pi^*)$
  - Why don't we directly use this optimal policy?

Observation

- RL objective:

$$J(\theta) = \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta}\left[r(x, y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]$$

- It's easy to show that the optimal policy under this objective is

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \exp\left[\frac{1}{\beta} r(x, y)\right]$$

  - Exercise: show that the solution is the same as min KL $(\pi_\theta \| \pi^*)$
  - Why don't we directly use this optimal policy?

- This allows us to relate the reward and the policy

Observation

- RL objective:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta}\left[r(x,y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)}\right]$$

- It's easy to show that the optimal policy under this objective is

$$\pi^*(y \mid x) = \frac{1}{Z(x)} \exp\left[\frac{1}{\beta} r(x,y)\right]$$

  - Exercise: show that the solution is the same as min KL $(\pi_\theta \| \pi^*)$
  - Why don't we directly use this optimal policy?

- This allows us to relate the reward and the policy

- Therefore we can represent the reward using the policy in the objective

$$r^*(x,y) = \beta \log \frac{\pi^*(y \mid x)}{\pi_0(y \mid x)} + \beta \log Z(x)$$

New objective

- MLE objective on the preference dataset:

$$\min -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \log p_\theta(y_1 \succ y_2 \mid x) = -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \left[\frac{1}{1+\exp(-(r_\theta(x,y_1) - r_\theta(x,y_2)))}\right]$$

New objective

- MLE objective on the preference dataset:

$$\min -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \log p_\theta(y_1 \succ y_2 \mid x) = -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \left[\frac{1}{1+\exp(-(r_\theta(x,y_1) - r_\theta(x,y_2)))}\right]$$

- Representing $r_\theta(x,y)$ using $\pi_\theta(y \mid x)$

$$r_\theta(x,y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)} + \beta \log Z(x)$$

Note that the objective only depends on the difference between the two rewards

New objective

- MLE objective on the preference dataset:

$$\min -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \log p_\theta(y_1 \succ y_2 \mid x) = -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \left[\frac{1}{1+\exp(-(r_\theta(x,y_1) - r_\theta(x,y_2)))} \right]$$

- Representing $r_\theta(x,y)$ using $\pi_\theta(y \mid x)$

$$r_\theta(x,y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_0(y \mid x)} + \beta \log Z(x)$$

  Note that the objective only depends on the difference between the two rewards

- we get the DPO objective (σ is the logistic function)

$$\min -\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}} \log \sigma \left(\beta \log \frac{\pi_\theta(y_1 \mid x)}{\pi_0(y_1 \mid x)} - \beta \log \frac{\pi_\theta(y_2 \mid x)}{\pi_0(y_2 \mid x)} \right)$$

What does DPO do?

$$\nabla_\theta \mathcal{L}_{\mathrm{DPO}}(\theta) = -\,\beta\,\mathbb{E}_{(x,y_1,y_2)\sim\mathcal{D}}\left[\hat{p}_\theta(y_2 \succ y_1)\left(\nabla_\theta \log \pi(y_1 \mid x) - \nabla_\theta \log \pi(y_2 \mid x)\right)\right],$$

where

$$\hat{p}_\theta(y_2 \succ y_1) = \sigma\left(\beta \log \frac{\pi_\theta(y_2 \mid x)}{\pi_0(y_2 \mid x)} - \beta \log \frac{\pi_\theta(y_1 \mid x)}{\pi_0(y_1 \mid x)}\right)$$

- Increases the likelihood of the preferred response and decreases the likelihood of dispreferred response
- Large weight on the update if prediction is wrong

Summary

- RL had limited improvement over supervised learning in NLG on small models.
- Scaling helps boost performance of RL: large base model + large reward model
- But RL is still a complicated process in practice, and there are research towards simplifying the process (e.g., DPO).
- Key challenge:
  - Reward hacking / over-optimization
  - Unreliable human annotation