Reinforcement Learning in the Era Of LLMs

Sep. 2023
Hao Sun
University of Cambridge
hs789@cam.ac.uk

Content

- Introduction to RL
- Reinforcement Learning from Human Feedback
- RL + LLMs

Some 'Terms' You May Have Heard About...

- Markov Decision Processes
- States, Observations, Transitions (Dynamics), Actions, Rewards, Discout Factors...
- Model-Based / Model Free
- Value-Based / Policy-Based / Actor-Critic
- On-Policy / Off-Policy
- Online / Offline
- Discrete Control / Continuous Control

But those are not necessary...

- ~~Markov Decision Processes~~
- ~~States, Observations, Transitions (Dynamics), Actions, Rewards, Discout Factors...~~
- ~~Model-Based / Model Free~~
- ~~Value-Based / Policy-Based / Actor-Critic~~
- ~~On-Policy / Off-Policy~~
- ~~Online / Offline~~
- ~~Discrete Control / Continuous Control~~

Essential Ideas

- What is RL?

  An agent learns from trial and error, to maximize a cumulative reward.

- agent: can be human or neural networks. It has a policy (clinical guidelines, public policies)
- trial and error: Online or Offline? Real or (data-drive) Simulator?
- cumulative reward: the *Reward Hypothesis*

Reward Can be Sparse...

- "Win the game", but how?
- Imitation Learning (IL) Can Help.
- 1. Behavior Clone

  Pros: Simple, Does not need further interactions with the environment

  Cons: Compounding error, multi-modality modeling
- 2. GAIL: Generative Adversarial Imitation Learning [Ho et al. 2016]

  Pros: Solves the above cons.

  Cons: Needs more interactions

Ho, Jonathan, and Stefano Ermon. "Generative adversarial imitation learning." Advances in neural information processing systems 29 (2016).

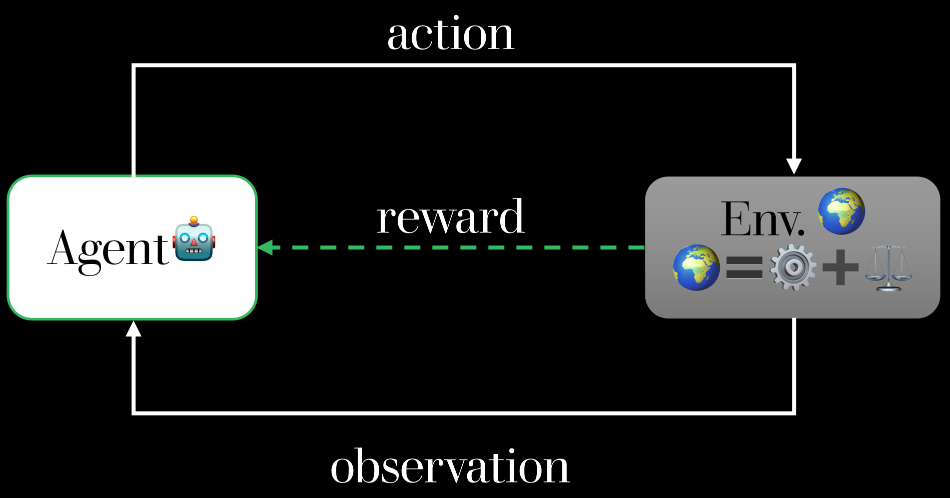
Difference between Inverse RL and IL

- Inverse-RL ≈ Imitation Learning, with an emphasis on reward learning

  Learning from logged trial and error, to find out what cumulative reward is being optimized.

- logs can be either expert decisions or non-expert decisions. Extrapolation.
- trial and error are offline data
- (the estimated) reward can be used as an evaluator of trajectories/policies

Graph: RL and Offline-RL

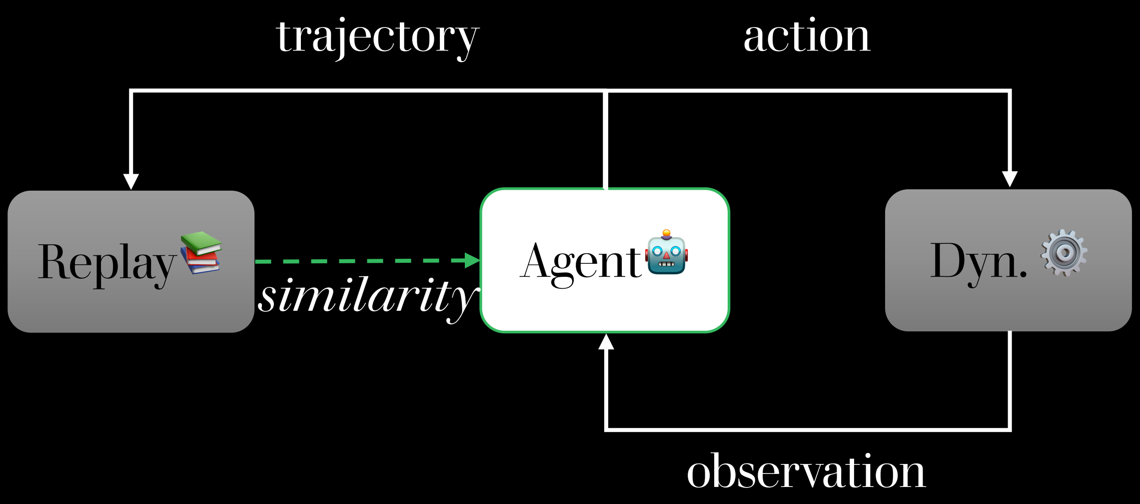


RL

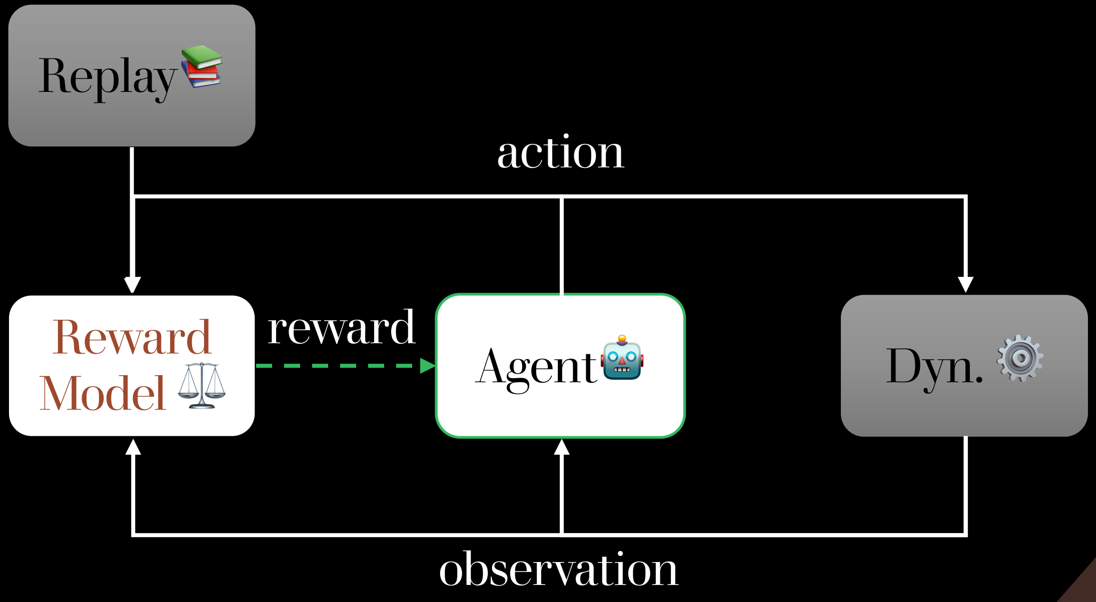
Offline-RL

Env. 🌍 = Dyn.⚙️ + Rew.⚖️

Graph: IL and IRL

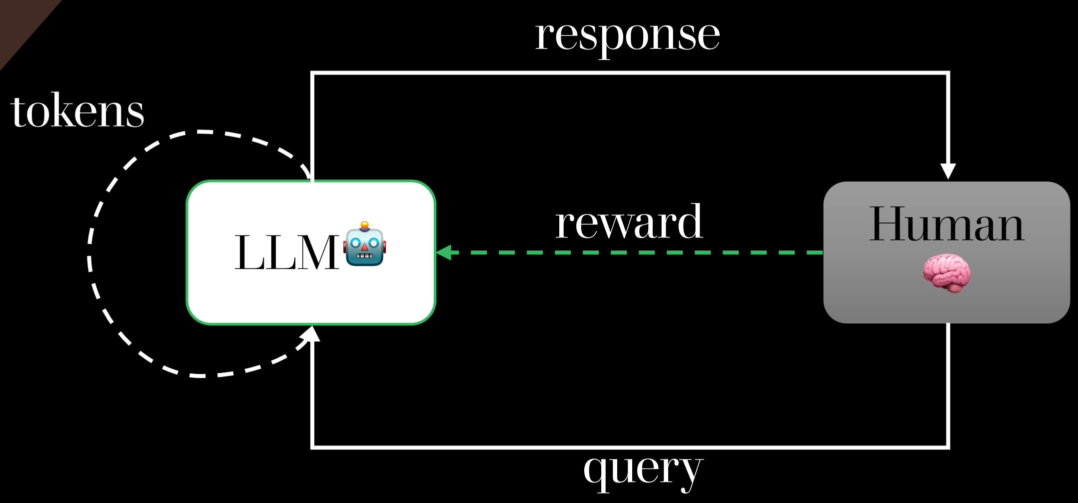


IL



IRL: Potential to be better

LLM Alignment with Human Feedback



Alignment as RL

- Why RL?

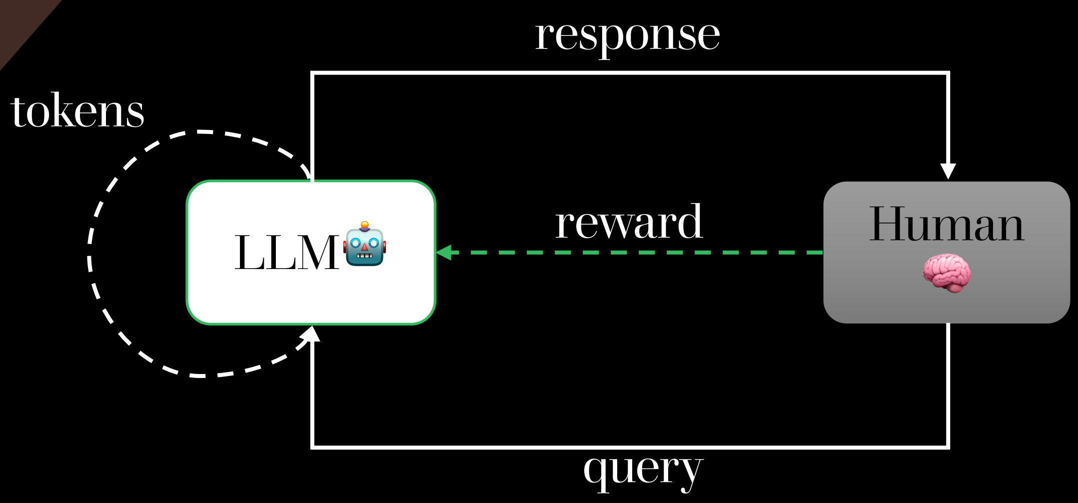
  We can not define the desired objective as a metric function.

  We can not do back-prop through a black-box 🧠

- Why not?

  Too expensive.

LLM Alignment with Human Feedback



Alignment as RL

- Why RL?

  We can not define the desired objective as a metric function.

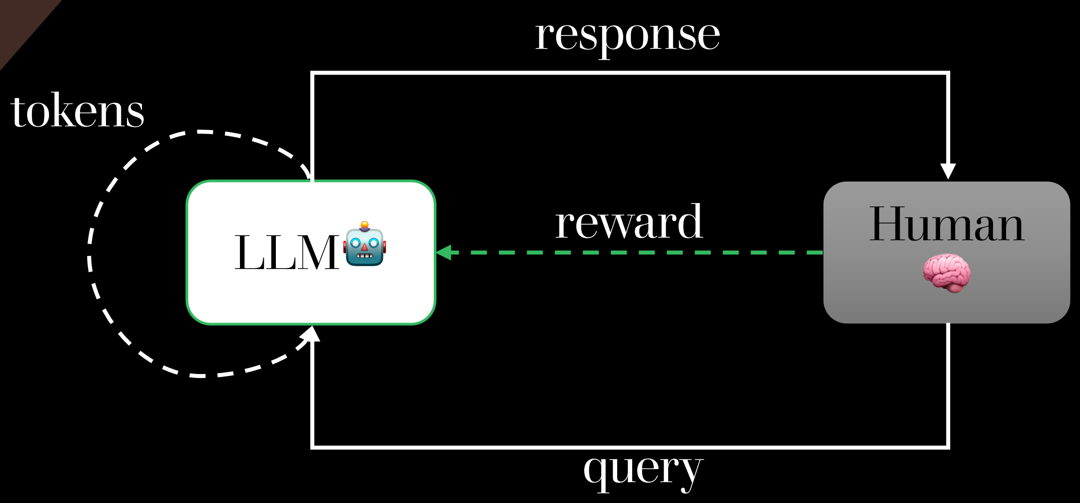
  We can not do back-prop through a black-box 🧠

- Why not?

  ~~*Too expensive.*~~

  OpenAI: unless you have enough volunteers (users)...

LLM Alignment with GPT-4 Feedback?



Alignment as RL

- Why RL?

  We can not define the desired objective as a metric function.

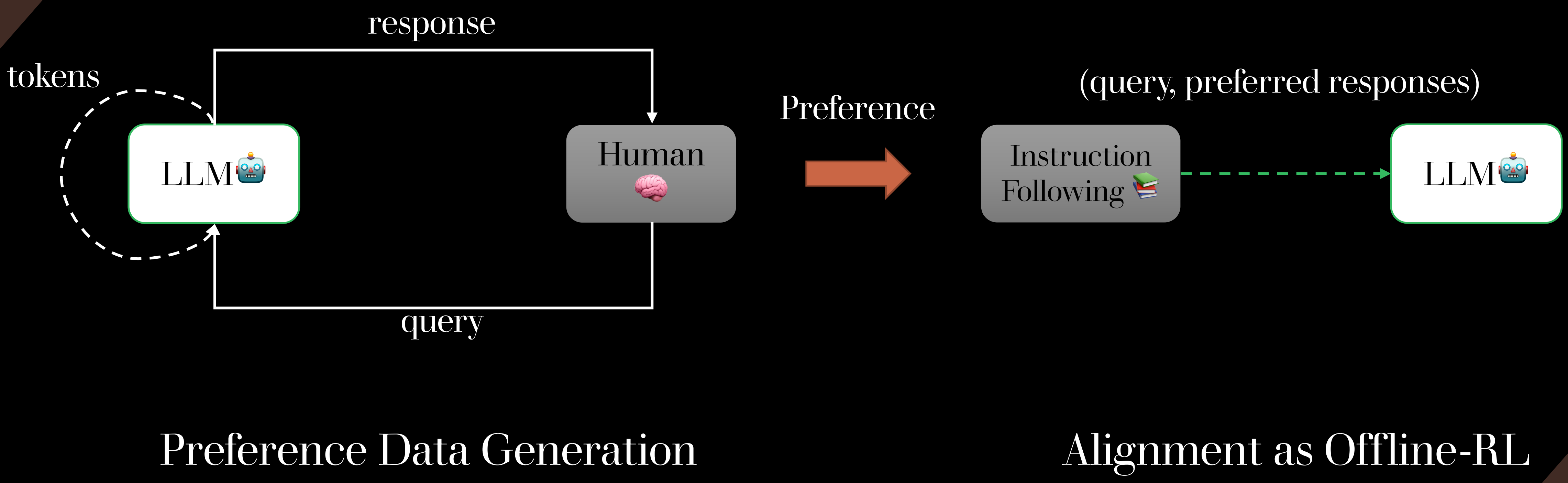
  We can not do back-prop through a black-box 🧠

- Why not?

  ~~Too expensive.~~

  OpenAI: unless you have enough volunteers (users)...

LLM Alignment with Human Feedback Logs



Alignment as Offline-RL: How to Learn?

Alignment as Offline-RL

- We can always use behavior clone.

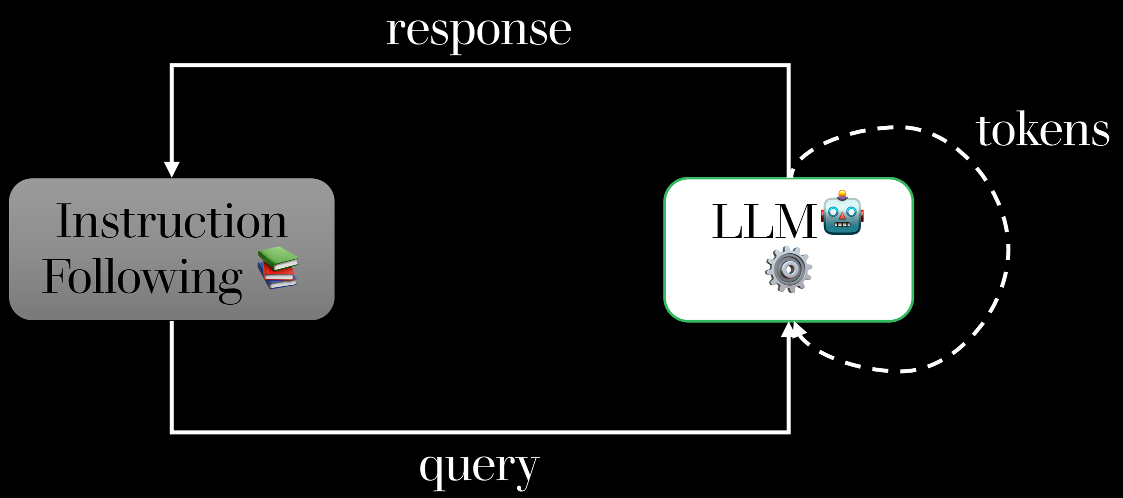
  BC = SFT, supervised-fine-tuning

- It is simple, stable, efficient.
- But language modeling is not 1-step decision.

  Compounding errors

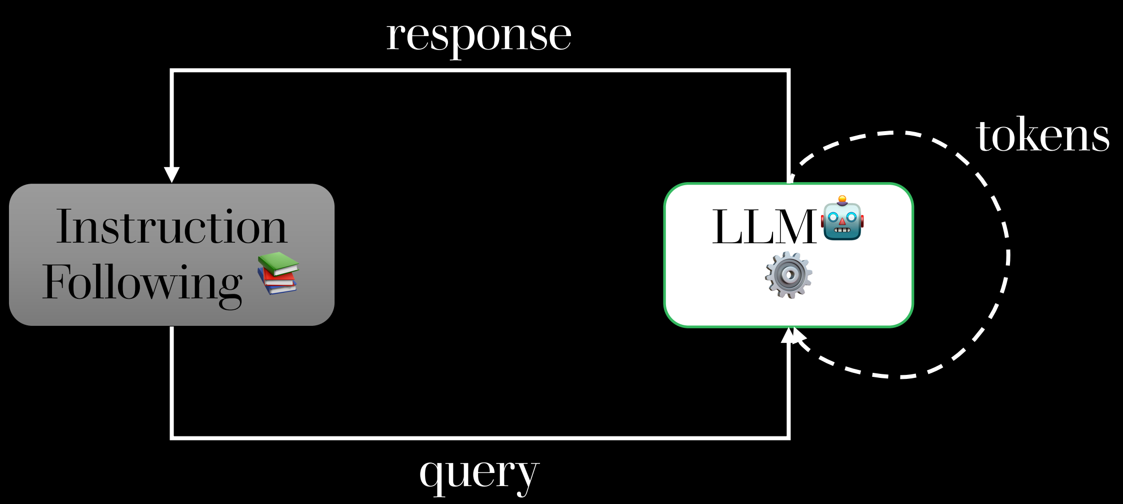
What Makes LLM Alignment Special?

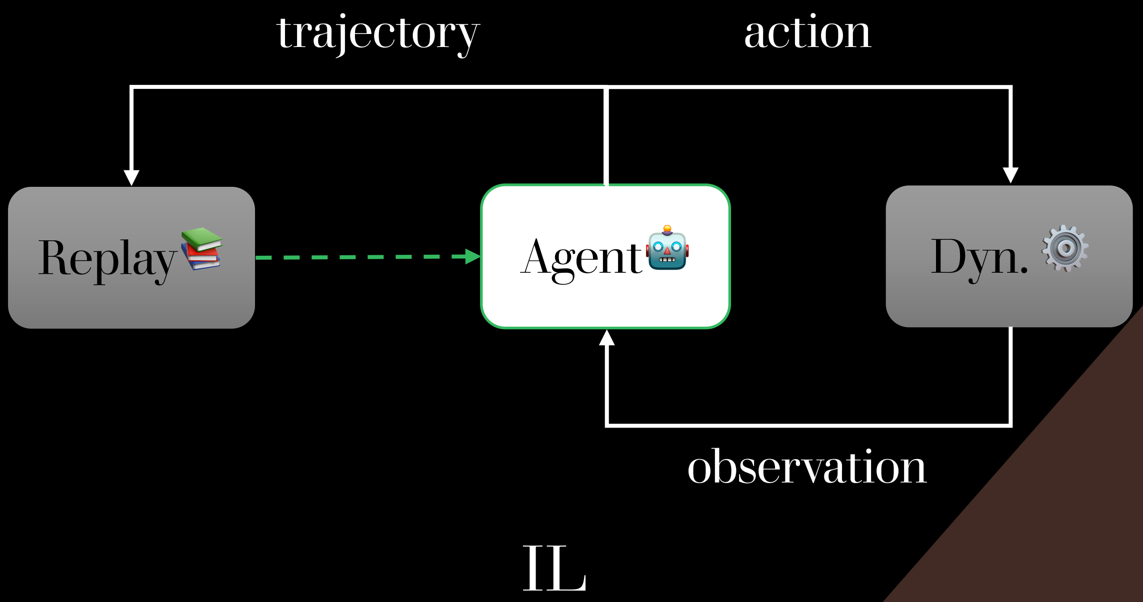
- The transition dynamics is deterministic and known!



What Makes LLM Alignment Special?

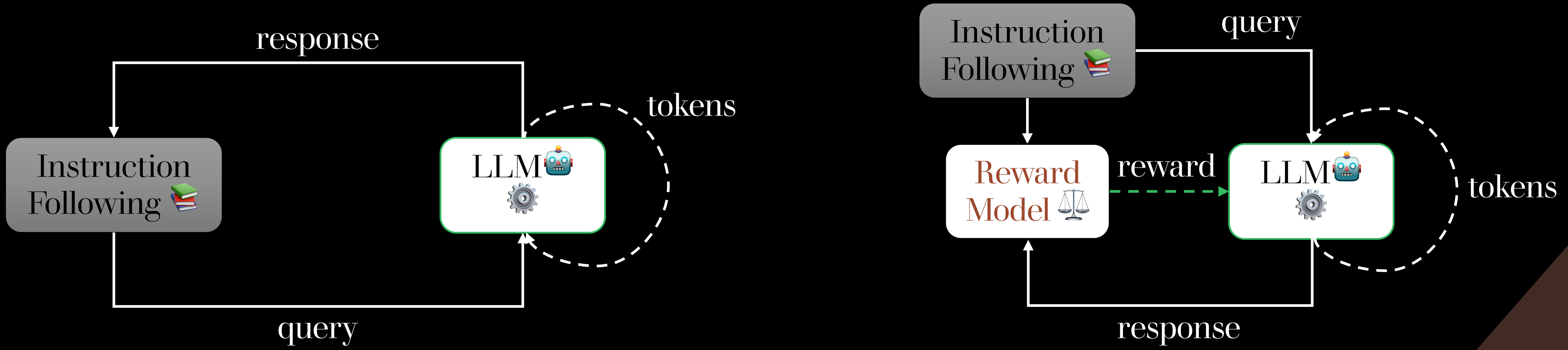
- The transition dynamics is deterministic and known!
- Recall the framework of Imitation Learning.





RLHF: Solving Offline-RL via Online Inverse RL

- Inverse RL: learn the reward model, then optimize the policy.



SFT vs RLHF*: from the RL Perspective

- LLM alignment with logged human feedback (preference) can be interpreted as
  1. *Offline-RL --- solve it with behavior clone --- SFT*
  2. *Imitation Learning --- solve it with IRL --- RLHF*

- We can always do both:
  RLHF using SFT as a warm-start

- Potential Alternatives? Someone would try GAIL...

---

* OpenAI's SFT is based on a separated high-quality response written by human.

RLHF

- Underlying assumptions:
  1. *Learning a reward model is statistically easier than directly learning aligned LLMs.*
  2. *There are some higher-level metrics that can not be captured by token-level distances.*

- Two steps
  1. *Reward Learning (Response Evaluation)*
  2. *LLM Optimization (Response Optimization)*

Step 1. Reward Model Learning

- Ranking is better than scoring, because the latter is noisier.

- LM with different sizes are used:

  OpenAI: 6B RM for 175B LM

  DeepMind: 75B RM for 75B LM

- Core Idea: The RM should be able to **understand** responses.

RM Implementation:

- Basic component: ψ: pre-trained LM with last layer replaced by linear.

  Ranking Loss *[Stiennon et al. 2020]*

$$\mathcal{L}_R(\psi) = -\log \sigma(r_\psi(x, y_+) - r_\psi(x, y_-))$$

  Alternatively *[Askell et al 2021]*

$$\mathcal{L}_R(\psi) = \log(1 + \exp(r_\psi(x, y_-) - r_\psi(x, y_+)))$$

- Imitation Loss [Askell et al 2021]

$$\mathcal{L}_{IL}(\psi') = -\log P_{\psi'}(y_+|x) = -\sum_{t}^{\text{<EOS>}} \log P_{\psi'}(y_+^{(t)}|y_+^{(<t)}, x)$$

- RM Learned: $\mathcal{R}_\psi$

$$r_\psi = \arg\min_{\psi} \lambda \mathcal{L}_R(\psi) + \beta \mathcal{L}_{IL}(\psi')$$

Stiennon, Nisan, et al. "Learning to summarize with human feedback." Advances in Neural Information Processing Systems 33 (2020): 3008-3021.
Askell, Amanda, et al. "A general language assistant as a laboratory for alignment." arXiv preprint arXiv:2112.00861 (2021).

RM Implementation:

- Regularizer: KL-div

$$\mathcal{R}_{KL}(\pi_\phi^{RL}) = \mathrm{KL}\left(\pi_\phi^{RL}(y|x) \middle| \pi_{\phi_0}^{SFT}(y|x)\right)$$

- Total Reward:

$$\mathcal{R}_{total} = r_\psi - \eta \mathcal{R}_{KL}$$

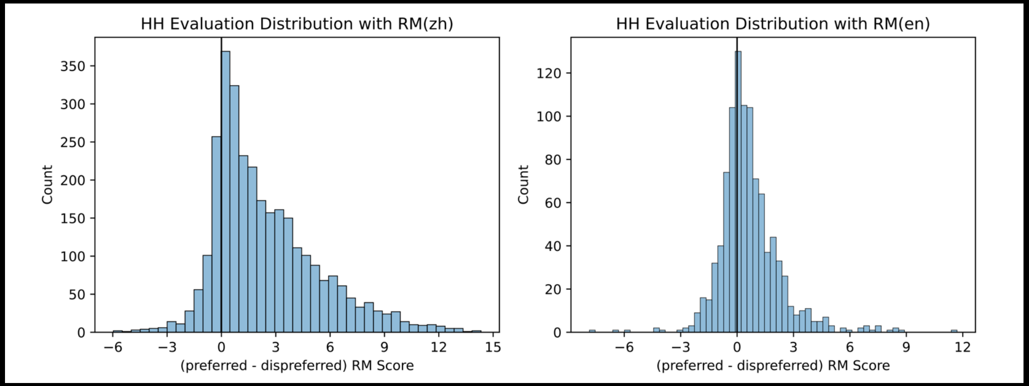
- Intuitive Interpretation by maximizing this reward:

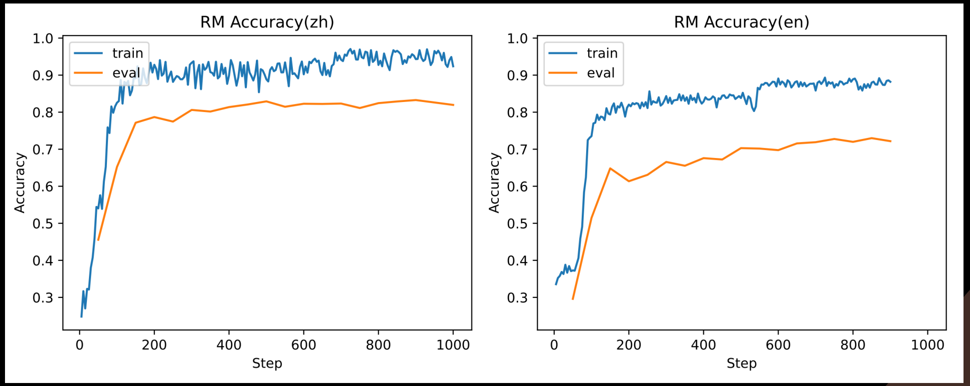
Maximize the sentence-level reward, while minimize the changes made to the SFT (base) model.

Stiennon, Nisan, et al. "Learning to summarize with human feedback." Advances in Neural Information Processing Systems 33 (2020): 3008-3021.
Askell, Amanda, et al. "A general language assistant as a laboratory for alignment." arXiv preprint arXiv:2112.00861 (2021).

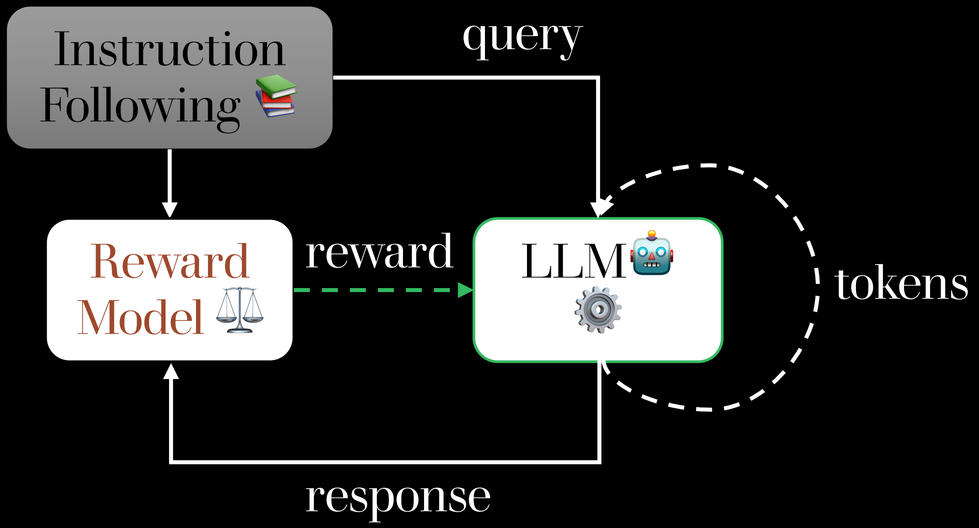
Training Details

- Model: LLaMA-7B / OpenChineseLLaMA
- Dataset (en): HH-RLHF 118k helpful + 42k harmless as train, 7.5k as val., 1k as test.
- Dataset (zh): annotated 31k helpful + 8k harmless, 30k train, 6k val., 3k as test.
- Results:





Step 2. Learning with RM



- RL Algorithms

  - PPO: ([Secrets of RLHF in Large Language Models](#))

  - ILQL

  Challenges:

  - multiple LLMs required. e.g., reference model, actor, critic, reward model.

  - not stable, hard to train, sensitive to hyper-params & seeds.

- Evolution Strategies can be a scalable alternative [Salimans et al. 2017]

  - RAFT

  - RRHF

  Sample a batch, and select the best using RM

Salimans, Tim, et al. "Evolution strategies as a scalable alternative to reinforcement learning." arXiv preprint arXiv:1703.03864 (2017).

An Empirical Study on PPO in RLHF [Zheng et al. 2023]

- Stable training of RLHF is still a puzzle
- Policy Constraints is important
- PPO-Max: empirical tricks
- Summary: stabilize training

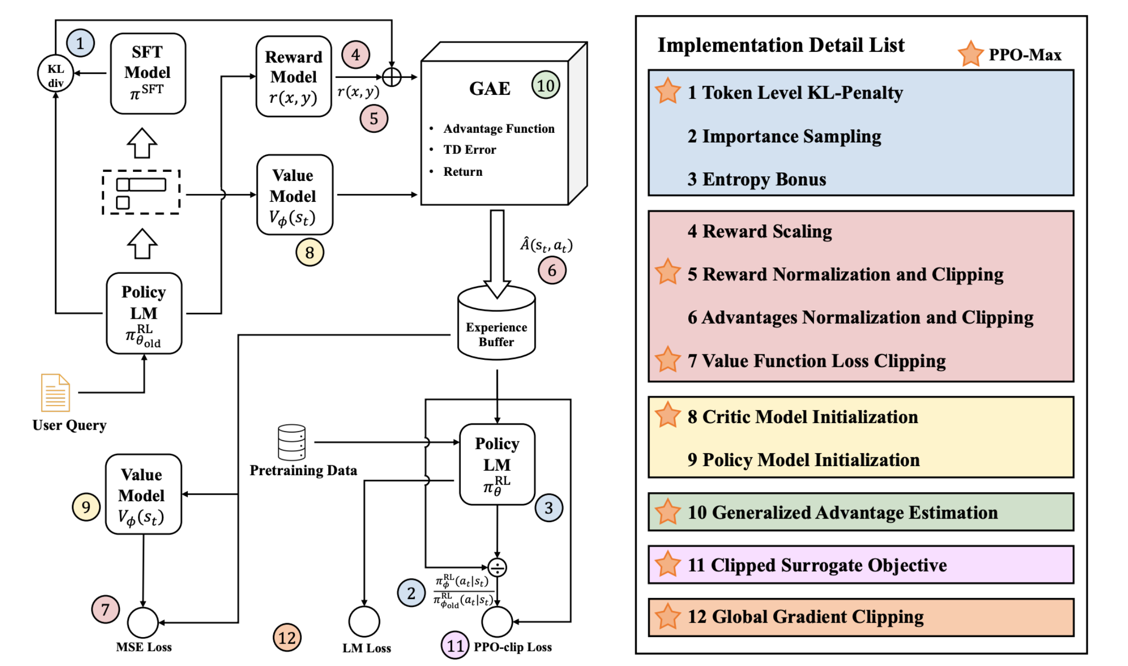


Figure 5: **Left** shows an equivalent structure to the RLHF framework in Figure 1. **Right** shows an implementation detail list for PPO. The number with circle indicates where this strategy is used in the PPO training. The pentagram indicates the method used by PPO-max.

Zheng, Rui, et al. "Secrets of RLHF in Large Language Models Part I: PPO." arXiv preprint arXiv:2307.04964 (2023).

More Details

- 4 models need to be loaded

  Reference model and policy model are init by 7B SFT model

  Use reward model to init the critic.

- 8 x A100 GPU + 1TB RAM + 128CPU
- Debug:

  Monitoring Metrics

  Smaller Datasets

- PPO-Max performs like...

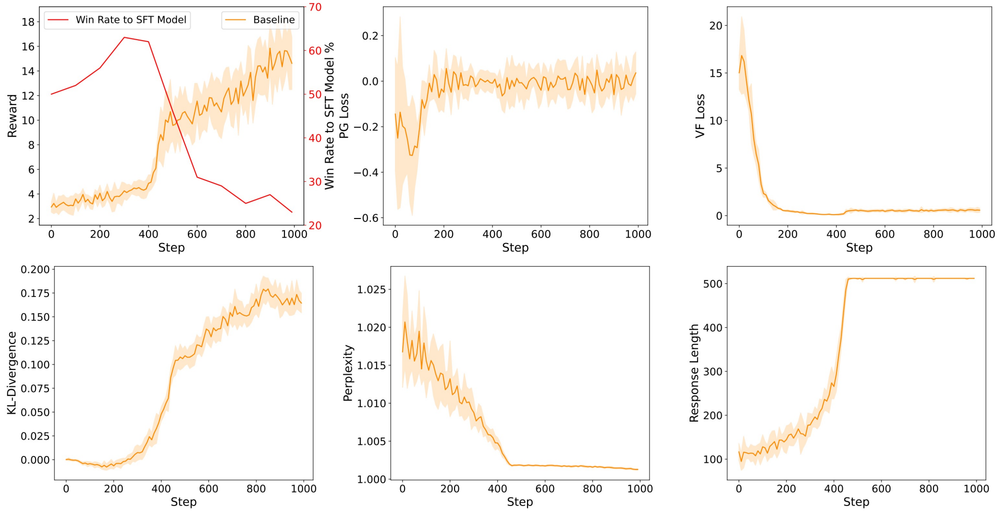


Figure 4: **(Top)** We show the response reward and training loss under vanilla PPO implementation. The red line in the first sub-figure shows the win rate of policy model response compared to SFT model response. **(Bottom)** Informative metrics for the collapse problem in PPO training, we observe significant variation in these metrics when there was a misalign between the human evaluation results and reward scores.

More Details

- 4 models need to be loaded

Reference model and policy model are init by 7B SFT model

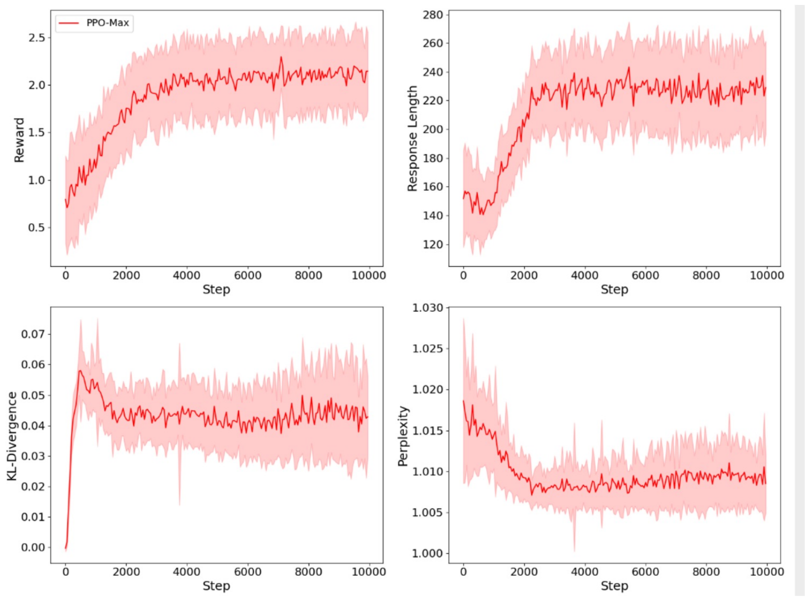


Figure 9: 10K steps training dynamics of PPO-max. PPO-max ensures long-term stable policy optimization for the model.

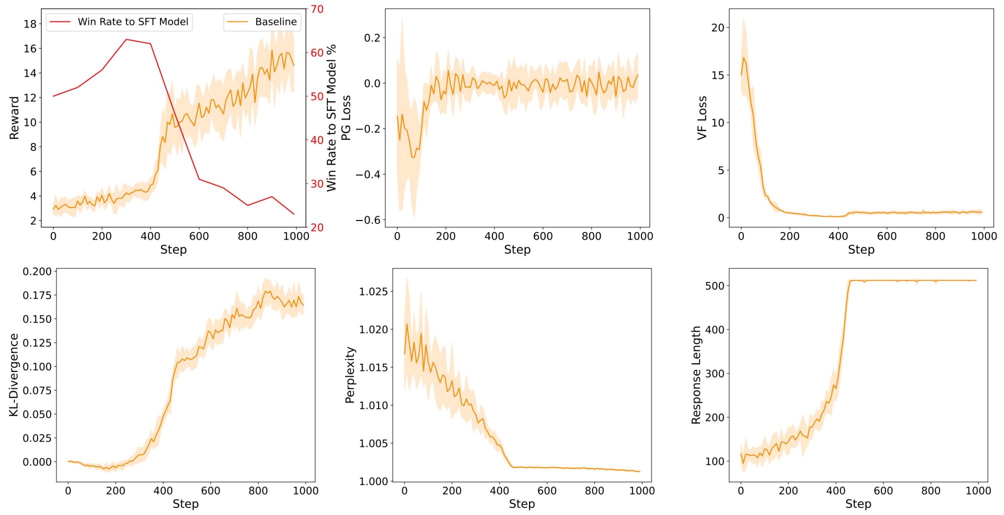


Figure 4: **(Top)** We show the response reward and training loss under vanilla PPO implementation. The red line in the first sub-figure shows the win rate of policy model response compared to SFT model response. **(Bottom)** Informative metrics for the collapse problem in PPO training, we observe significant variation in these metrics when there was a misalign between the human evaluation results and reward scores.

RAFT: Reward rAnked Fine Tuning

- Best-of-N: 1. sample N for each query; 2. select the best-of-N; 3. supervised update
- Much less hyper-params.

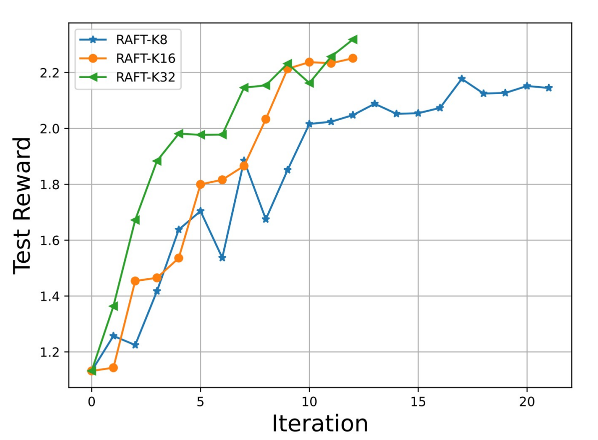
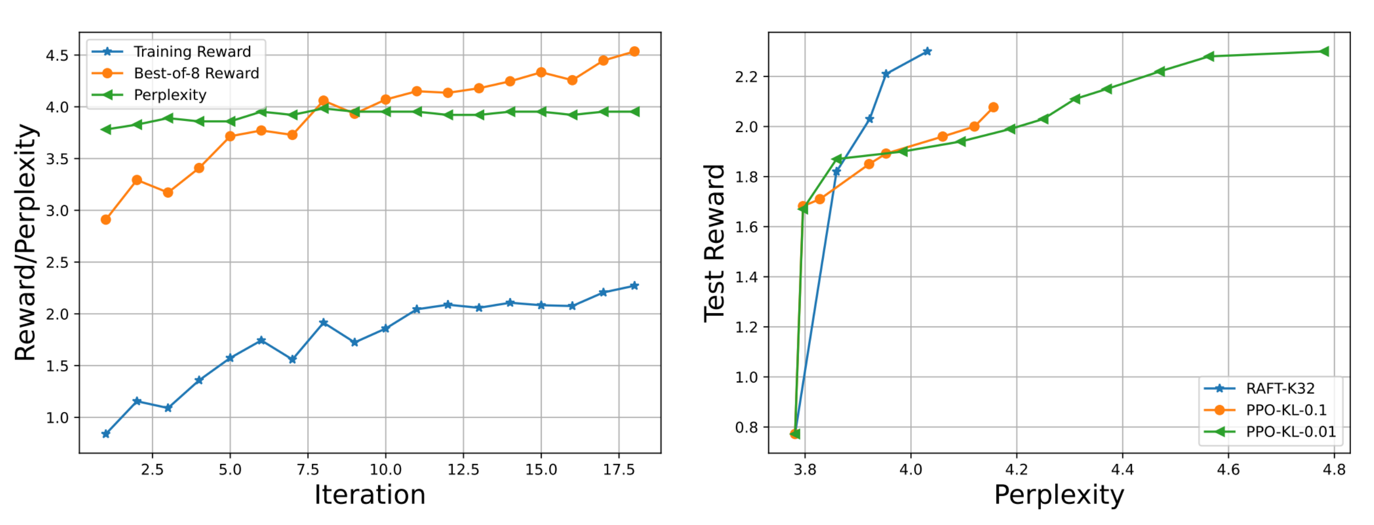


Figure 2: The test reward w.r.t. the iteration under different $K \in \{8, 16, 32\}$.



Resources

- GitHub Repo on RLHF: https://github.com/opendilab/awesome-RLHF
- Secrets of RLHF in Large Language Models https://github.com/OpenLMLab/MOSS-RLHF
- RAFT Official Implementation: https://github.com/OptimalScale/LMFlow
- TRL/ TRLx by huggingface: https://github.com/huggingface/trl
- RL4LMs: https://github.com/allenai/RL4LMs

Instruction Following by Prompting

- Prompt engineering is an effective approach in eliciting the abilities of LLMs
- In-context Learning/Fine-tuning

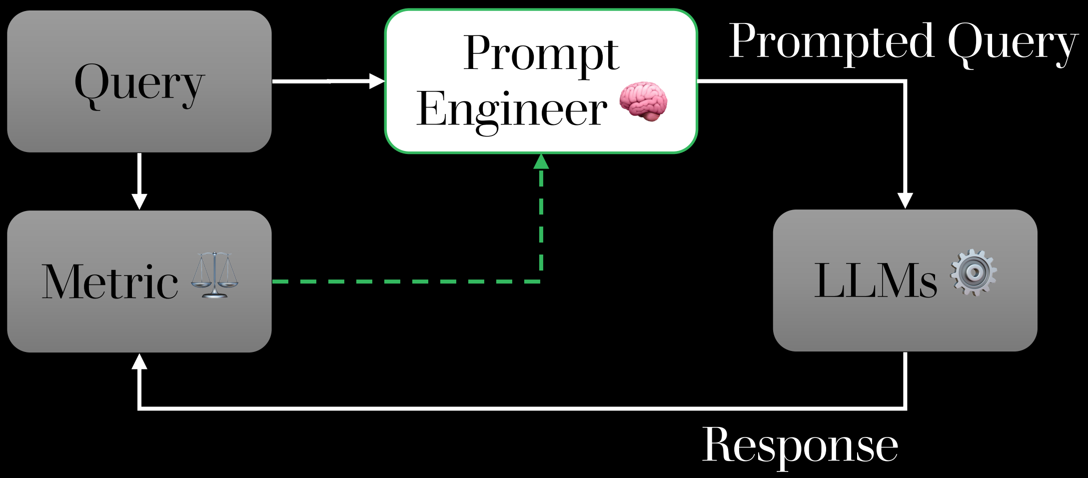
  Few-Shot Prompting + In-Context Learning

  Zero-Shot Prompting
- Examples:

  CoT: let's think step by step...

  OPRO*: take a deep breath ...*
- How to design? Previous approaches: learning from trial and error.

Instruction Following by Prompting



Prompt Engineers are doing RL

- Make it automatic?

  RL Agent 🤖 as Prompt Engineer

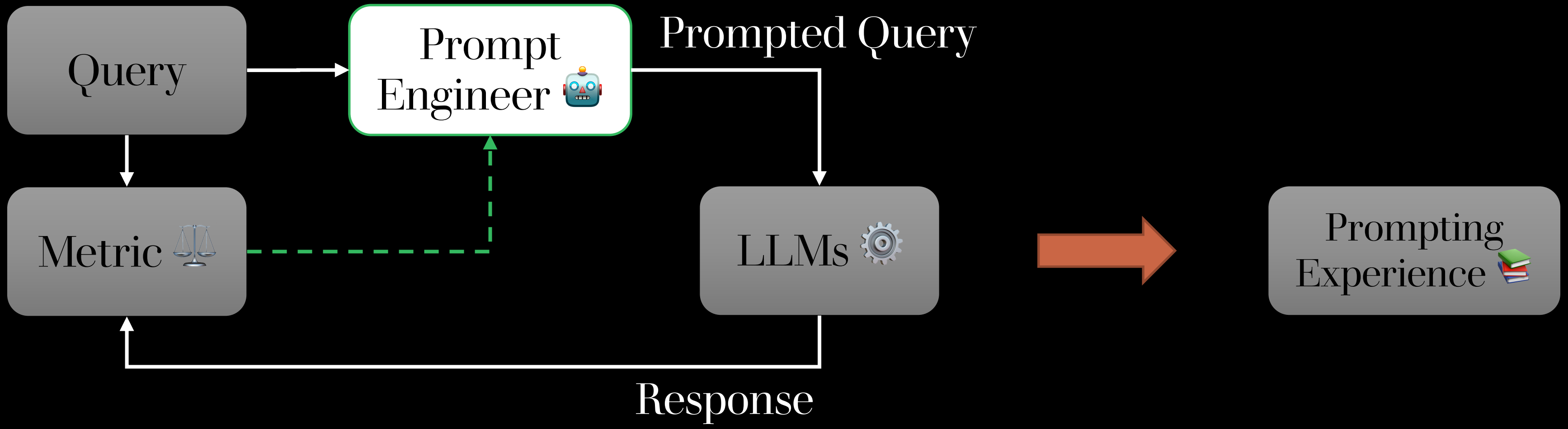
- Challenges:

  Too expensive to explore

  The action space is too large

Prompter Alignment with LLM Feedback

- We are aligning prompter using feedback from LLMs.
- Inspired by the great success of RLHF, can we do Imitation Learning?



Evaluate Existing Prompts as Offline Dataset

- For the same query, prompt engineers have tried different prompts

  e.g., on the GSM8K dataset, CoT, APE, ToT prompts are evaluated

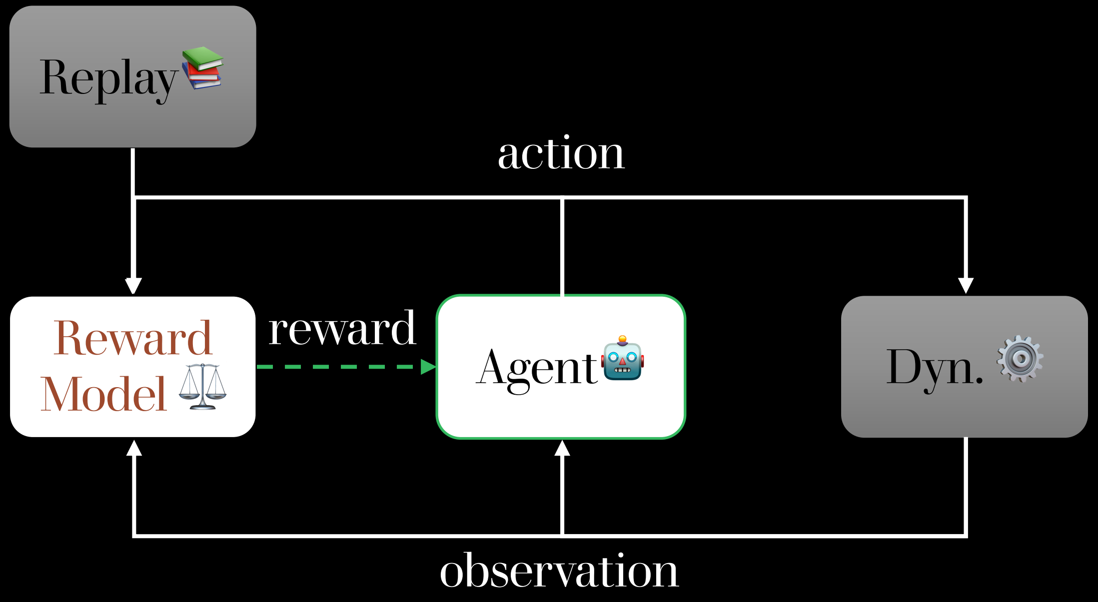
  consider there are N queries with golden answers, and M prompting strategies.

  (querie, prompt, response, correctness of answer)

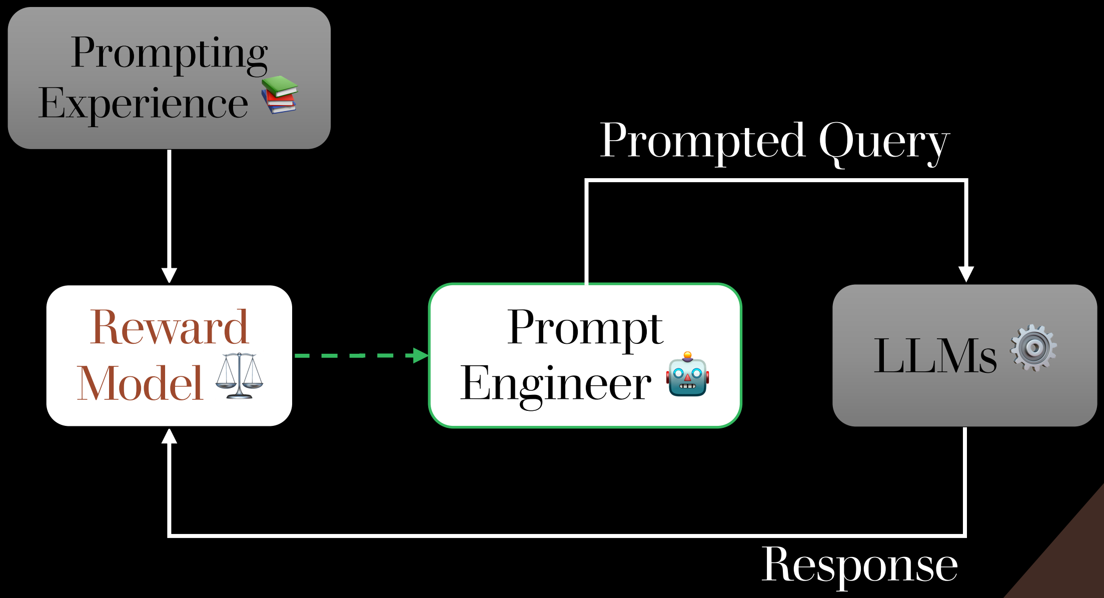
$$\left(q_i, p_j, a_{ij}, r_{ij}\right)_{i\in[N], j\in[M]}$$

How to learn?

Inverse RL

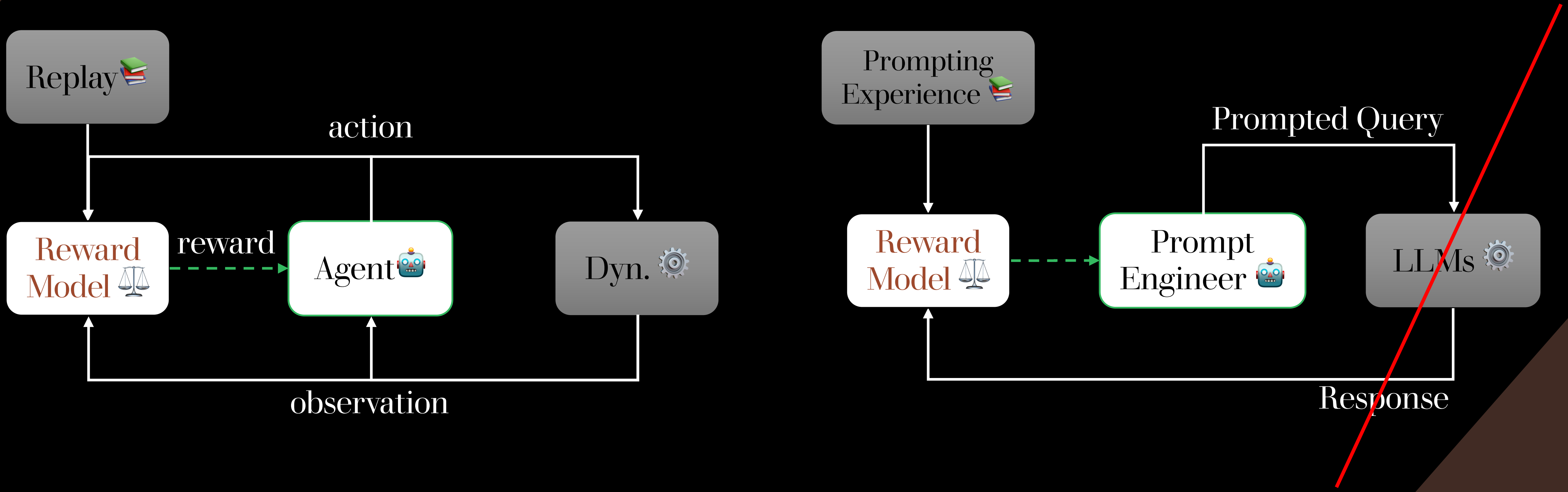


IRL



Prompting as IRL

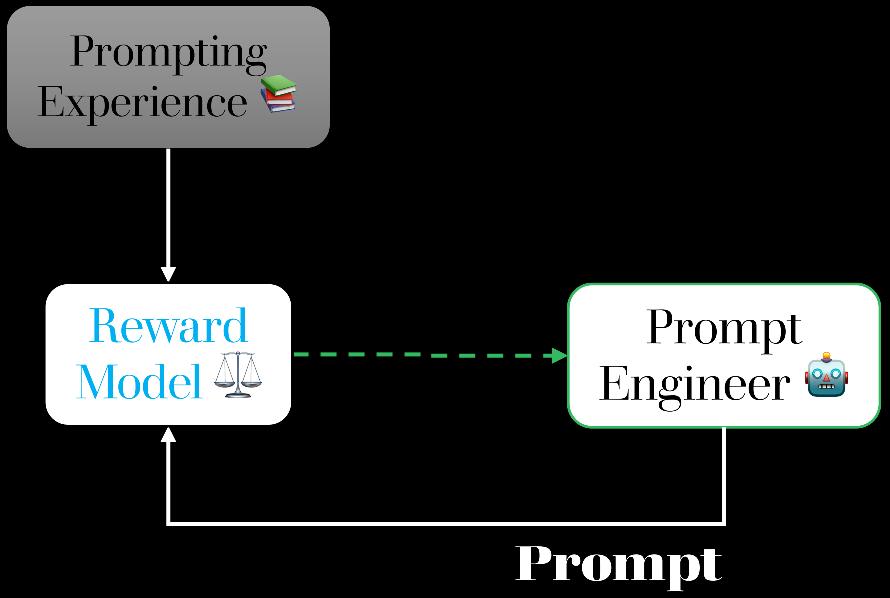
Inverse RL

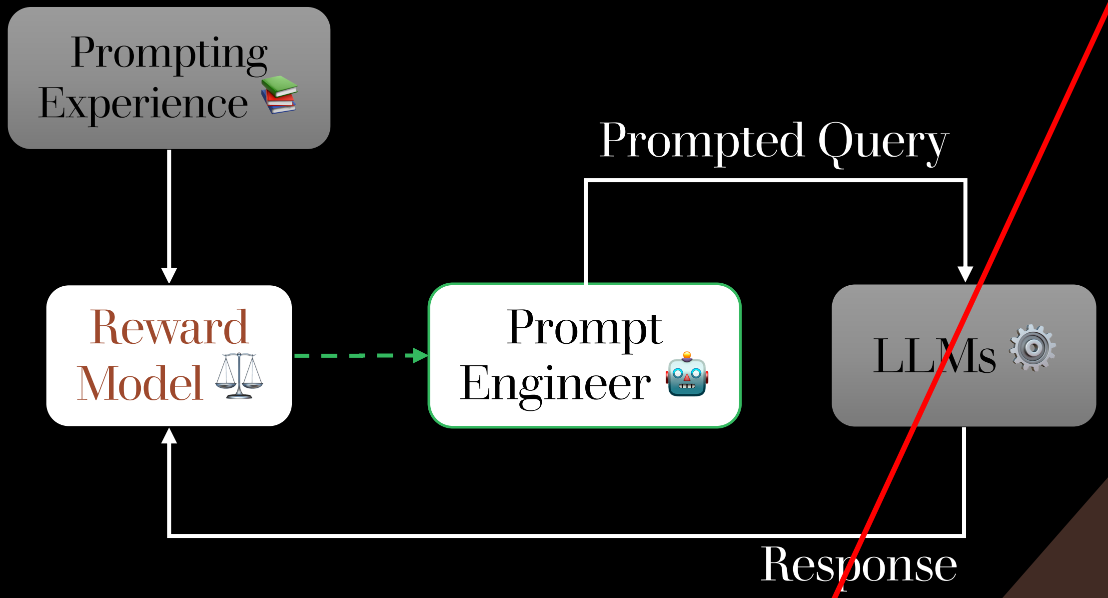


IRL

Prompting as IRL

Offline Inverse RL





Prompting as Offline IRL

~~Prompting as IRL~~

Reward Model: $q_i, p_j \mapsto r_{ij}$; Reward Model: $q_i, a_{ij} \mapsto r_{ij}$

Prompt-OIRL

- Offline Prompt Evaluation and Optimization with IRL [Sun et al. 2023]
- Two Steps:
  1. *Reward Model Learning --- for Prompt Evaluation*

  2. *Prompt Optimization --- with the learned reward model*

Sun, Hao et al. "Offline Prompt Evaluation and Optimization with Inverse Reinforcement Learning." arXiv preprint arXiv:2309.06553 (2023).

Reward Model Learning

- $\hat{r}$ is learned by supervised learning:

$$\mathcal{L}_{CE}(\hat{r}) = -\mathbb{E}_{ij}\left[r_{ij}\log\sigma\left(\hat{r}(q_i,p_j)\right) + (1-r_{ij})\log\left(1-\sigma\left(\hat{r}(q_i,p_j)\right)\right)\right]$$

- $\hat{r}$ is different from the original evaluation metric function in that

  $\hat{r} = \hat{r}(q_i, p_j)$ *does not require access to the LLMs, yet* $r = r(q, llm(p+q))$

  $\hat{r}$ *can do evaluation, but* r *can not (estimate whether the answer is correct in test time)*

- *q and p in experiments are represented by their embeddings.*

Prompt Optimization

- With $\hat{r}$, prompt optimization can be executed without LLMs:

$$p_i^*(q_i) = \operatorname*{argmax}_p \hat{r}(q_i, p)$$

- The optimized prompt is query-dependant
- How to instantiate this argmax?
  1. *Reinforcement learning: train a prompting LM*
  2. *Sample N and select the best*

Prompt Optimization

- With $\hat{r}$, prompt optimization can be executed without LLMs:

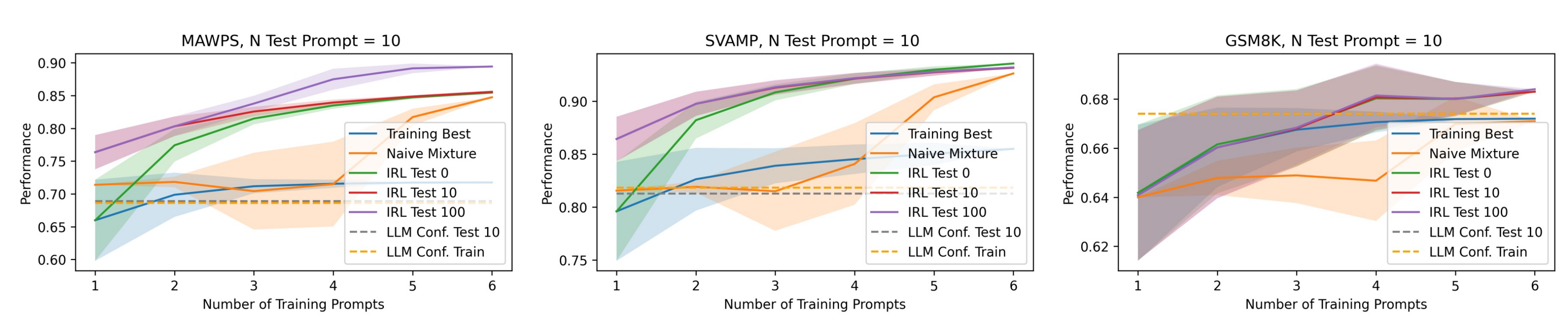
$$p_i^*(q_i) = \underset{p}{\operatorname{argmax}}\, \hat{r}(q_i, p)$$

- The optimized prompt is query-dependant
- How to instantiate this argmax?

1. *Reinforcement learning: train a prompting LM*
2. *Sample N and select the best*

Results

- Experiments on Arithmetic Reasoning Datasets (GSM8K, SVAMP, MAWPS)
- TakeAways:
  1. *Prompt-OIRL further improve the ability of LLMs in inference.*
  2. *It is extremely cheap to train and deploy Prompt-OIRL.*



Summary

- RL is learning from trial and errors to maximize a cumulative reward.
- Define reward function can be easy, but exploration of RL is hard.
- With expert demonstrations, IL can improve learning efficiency.
- Behavior Clone is the simplest IL, but it suffers from compounding errors.
- IRL first learns a RM, and then use the learned RM to optimize policy.
- SFT is behavior clone, RLHF is online IRL.
- Given an RM, there are multiple approaches to optimize LLMs to align with human.
- Prompt optimization can be formulated as an (extremely hard) RL problem.
- Using Offline-IRL, prompt optimization can be much easier.
- Prompt-OIRL is able to effectivly and efficiently perform offline promt evaluation and optimization.