Rewards in Language

- Dense human feedback is super expensive (also noisy)
- Whole field of automatic **human proxy** metrics for NLP:
  - Learned: human preference classifiers, BERTScore,
  - Rule-based per task: BLEU, ROUGE, etc.

Classical NLP Metrics

Candidate 1: It is a guide to action which ensures that the military always obeys the commands of the party.

Candidate 2: It is to insure the troops forever hearing the activity guidebook that party direct.

Reference 1: It is a guide to action that ensures that the military will forever heed Party commands.

Reference 2: It is the guiding principle which guarantees the military forces always being under the command of the Party.

Reference 3: It is the practical guide for the army always to heed the directions of the party.

Classical NLP Metrics

Candidate: the the the the the the the.

Reference 1: The cat is on the mat.

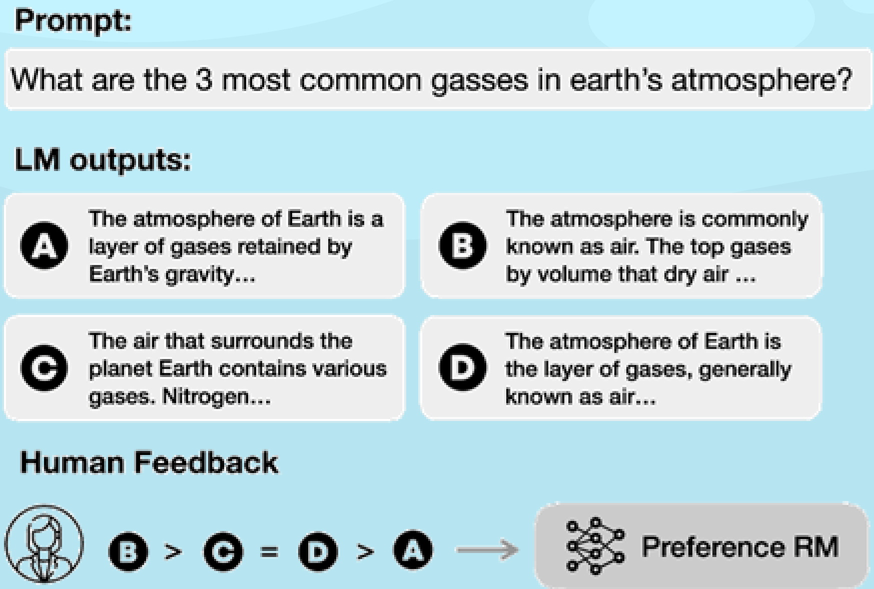
Reference 2: There is a cat on the mat.

Classical NLP Metrics

- Requires lots of bespoke written rules
- Can't scale
- "Every time I fire a linguist, the performance [...] goes up"
  --Frederick Jelinek

Learned Rewards

Step 1: Collect preference feedback



Train a new metric, a reward function: Human judgment proxy.

Trained via (variant of) a ranking loss.

Learned Rewards

What format of feedback do you have?

- +1/-1
- A vs B
- Ranked List
- Natural Language

Learned Rewards

What format of feedback do you have?

- +1/-1 – contextual bandits
- A vs B – Bradley Terry
- Ranked List – Plackett Luce
- Natural Language - ???

A vs B: Bradley Terry

- Pairwise preference models make the Bradley Terry assumption that underlying preference distribution is IID and pairwise prefs are generated with a fn of the form for some real no.s all

$$\mathbb{P}\{i \succ j\} = \frac{\alpha_i}{\alpha_i + \alpha_j}.$$

A vs B vs C vs D… Plackett Luce

- Listwise preference models make the Plackett Luce assumption that underlying preference distribution is IID and pairwise prefs are generated with a fn of the form for some real no.s all

$$\mathbb{P}\{\text{choosing } i \text{ from } S\} = \frac{\alpha_i}{\sum_{j \in S} \alpha_j}.$$

$$\begin{aligned}\mathbb{P}\{i_3 \succ i_1 \succ i_2\} &= \mathbb{P}\{\text{choosing } i_3 \text{ from } \{i_1, i_2, i_3\}\} \\ &\quad \cdot \mathbb{P}\{\text{choosing } i_1 \text{ from } \{i_1, i_2\}\} \\ &\quad \cdot \mathbb{P}\{\text{choosing } i_2 \text{ from } \{i_2\}\} \\ &= \frac{\alpha_{i_3}}{\alpha_{i_1} + \alpha_{i_2} + \alpha_{i_3}} \cdot \frac{\alpha_{i_1}}{\alpha_{i_1} + \alpha_{i_2}} \cdot \frac{\alpha_{i_2}}{\alpha_{i_2}}.\end{aligned}$$

Plackett Luce (contd)

- No existing reward models actually use Plackett Luce (though the concept is very relevant)
- Most take a list of A vs B vs C… and make pairwise preferences then apply Bradley Terry from that
  - Remember that Plackett Luce of list size 2 reduces to Bradley Terry

Why Contrastive Preferences?

- Humans can't always articulate why they prefer something
- Comparison to something else instead of raw score grounds things
- Idea is to learn implicit preferences through data

Why not Contrastive Preferences?

- Humans aren't transitive, may have prefs: A > B, B > C, C > A
- Harder to debug reward models of implicit human preferences, can't know why reward hacking is occurring
- Bradley Terry / Plackett Luce originally created for sports team rankings, assume that each A vs B vs C sample is IID and are single point values
    - Preferences are for language!! There is token level compositionality, you can like parts of a response but dislike others

“Verifiable” Rewards

- Will get into details later but just think of it as rewards with ~0 error for actual task you’re trying to get them to do

Problem 0:
Reward Hacking

Great rewards/metric scores, but spirit of task is unsolved

- "When measure becomes target, it ceases to be a good measure"

Example Reward: Positive Sentiment Score

I want you to make as positive a movie review as possible for me no matter how negatively it starts

Ok, I can do that. How should I start this review?

"I loved the book but really hated the movie"

Amaze brilliant great yay 10/10 -IGN

Reward hacked

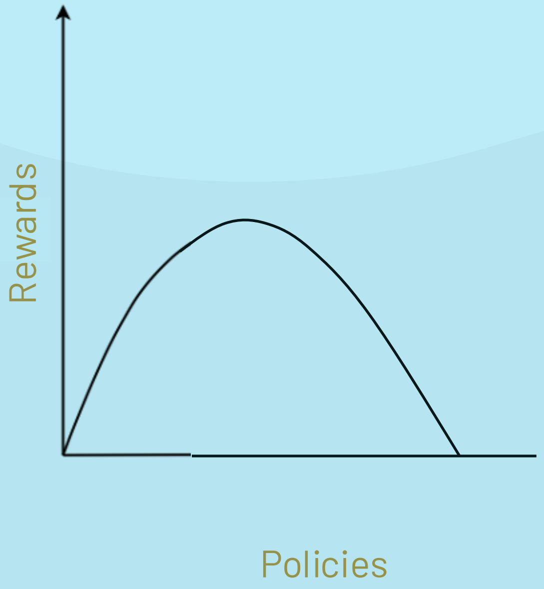
At first anyway, but I warmed slowly as I watched. Here, I'll tell you why …

IDEAL

Problem 0: Reward Hacking
The (Partial) Fix

- Optimize for this reward function

$$\mathbb{E}_\pi[\sum_{t=0}^{K} \gamma^t R(\boldsymbol{s}_t, a_t)]$$



Ziegler et al. *Fine-tuning Language Models from Human Preferences.* Preprint. 2020.

Problem 0: Reward Hacking
The (Partial) Fix

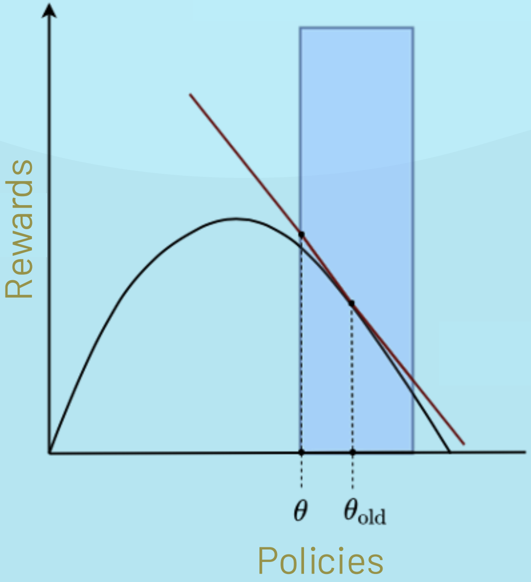
- KL Divergence from LM creates "Trust Region of Relevant Natural Language"

$$\mathbb{E}_\pi\Big[\sum_{t=0}^{K} \gamma^t R(\boldsymbol{s}_t, a_t)\Big] - \alpha \mathbb{E}_\pi\big[\mathrm{KL}(\pi_\theta || \pi_0)\big]$$

Current Policy Original Policy

Long Term Expected Task Rewards Naturalness Penalty

Reason 2 why we need Pre-training+SFT. The outputs of the initial model need to already be somewhat reasonable to put us in the right (approx.) trust region.



Ziegler et al. *Fine-tuning Language Models from Human Preferences.* Preprint. 2020.

Problem 0: Reward Hacking
The (Partial) Fix

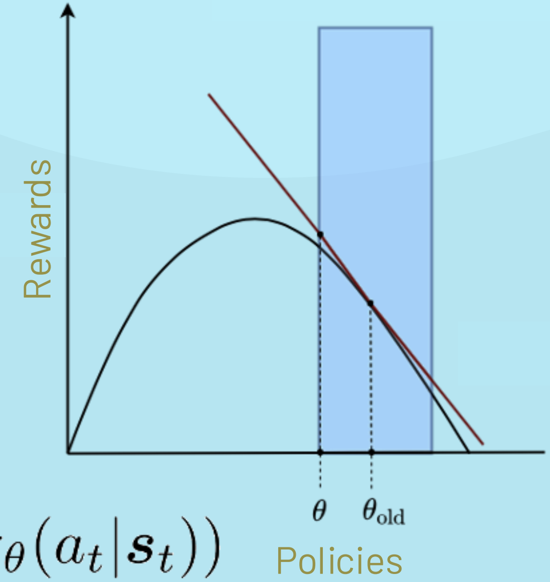
- KL Divergence from LM creates "Trust Region of Relevant Natural Language"

$$\mathbb{E}_\pi\left[\sum_{t=0}^{K}\gamma^t R(\boldsymbol{s}_t, a_t)\right] - \alpha\mathbb{E}_\pi[\mathrm{KL}(\pi_\theta || \pi_0)]$$

Long Term Expected Task Rewards

Current Policy   Original Policy

Naturalness Penalty



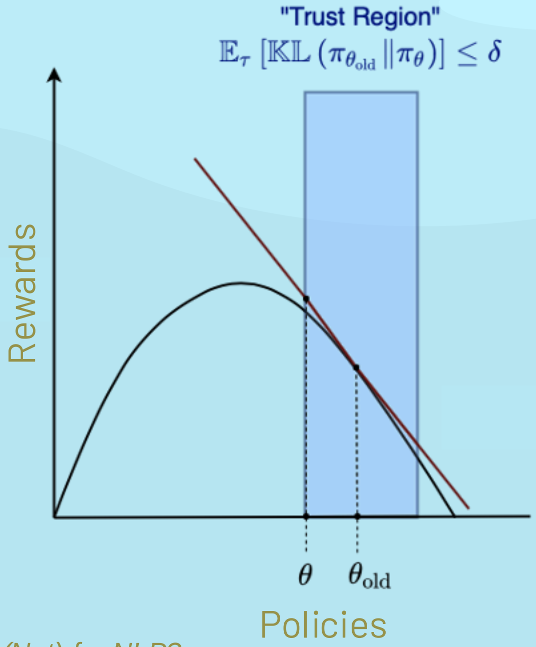
$$\mathrm{KL}\left(\pi_\theta(a_t|\boldsymbol{s}_t)||\pi_0(a_t|\boldsymbol{s}_t)\right) = \left(\log\pi_0(a_t|\boldsymbol{s}_t) - \log\pi_\theta(a_t|\boldsymbol{s}_t)\right)$$

Ziegler et al. *Fine-tuning Language Models from Human Preferences.* Preprint. 2020.

Why does *this* work?

- KL penalty creates a approximation of “trust region” of general natural language

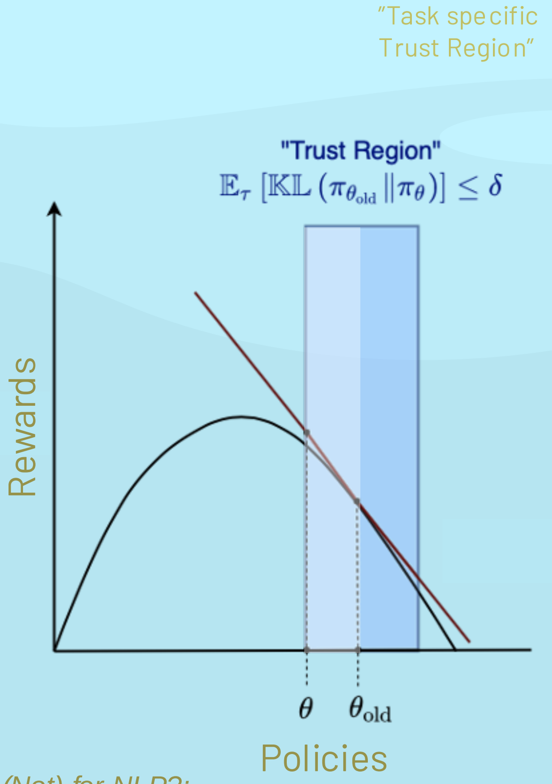
- Masking policy creates ”task specific trust region” = language specific to current domain



Ramamurthy*, Ammanabrolu*, Brantley, Hessel, Sifa, Bauckhage, Hajishirzi, Choi. *Is RL (Not) for NLP?: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization.* ICLR 2023.

Why does *this* work?

- RL algorithm "searches" in region for exact point to optimize rewards



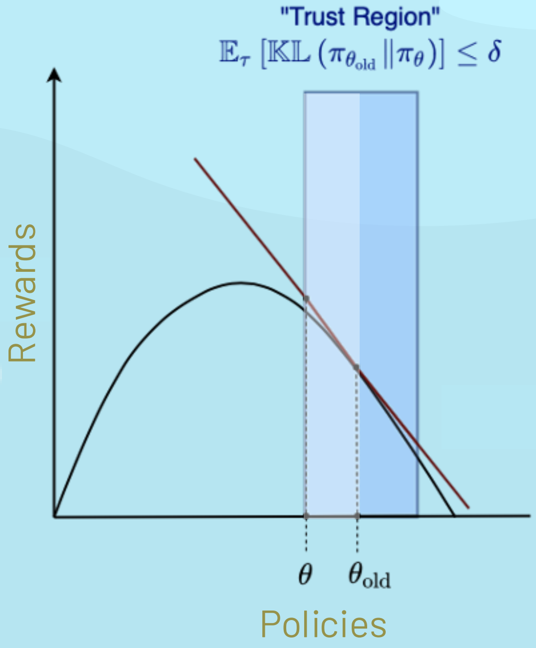
Ramamurthy*, Ammanabrolu*, Brantley, Hessel, Sifa, Bauckhage, Hajishirzi, Choi. *Is RL (Not) for NLP?: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization.* ICLR 2023.

Why does *this* work?

"Task specific
Trust Region"

- RL algorithm "searches" in region for exact point to optimize rewards

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D},y\sim\pi(y|x)}\left[r(x,y)\right] - \beta\mathbb{D}_{\mathrm{KL}}(\pi(\cdot \mid x)||\pi_{\mathrm{ref}}(\cdot \mid x))$$

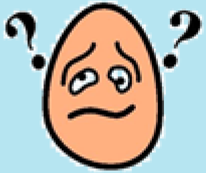


Problem 1: Challenging overall quality comparison

Hard to compare LM outputs with *a mixture of diverse undesired behaviors*

Unreliable human feedback

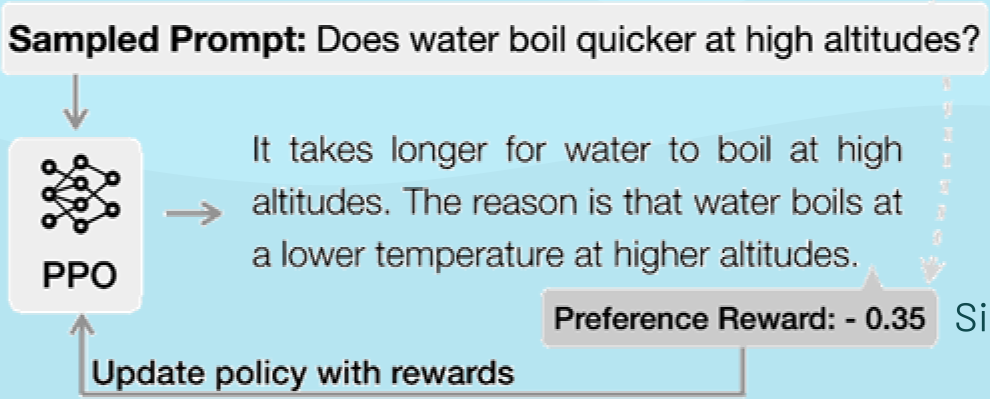
Output A:
Sentence 1: Factual 👍 but not informative 👎
Sentence 2: ...
...



Output B:
Sentence 1: Informative 👍 but unverifiable 👎
Sentence 2: ...
...

Ramamurthy*, Ammanabrolu*, Brantley, Hessel, Sifa, Bauckhage, Hajishirzi, Choi. *Is RL (Not) for NLP?: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization.* ICLR 2023.

Problem 2: Sparse, unreliable rewards for training



Single, noisy holistic reward for the full output

Unreliable RL training

Ramamurthy*, Ammanabrolu*, Brantley, Hessel, Sifa, Bauckhage, Hajishirzi, Choi. *Is RL (Not) for NLP?: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization.* ICLR 2023.

Fine-grained feedback is more explicit and reliable!

Prompt:

What are the 3 most common gasses in earth's atmosphere?

LM output:

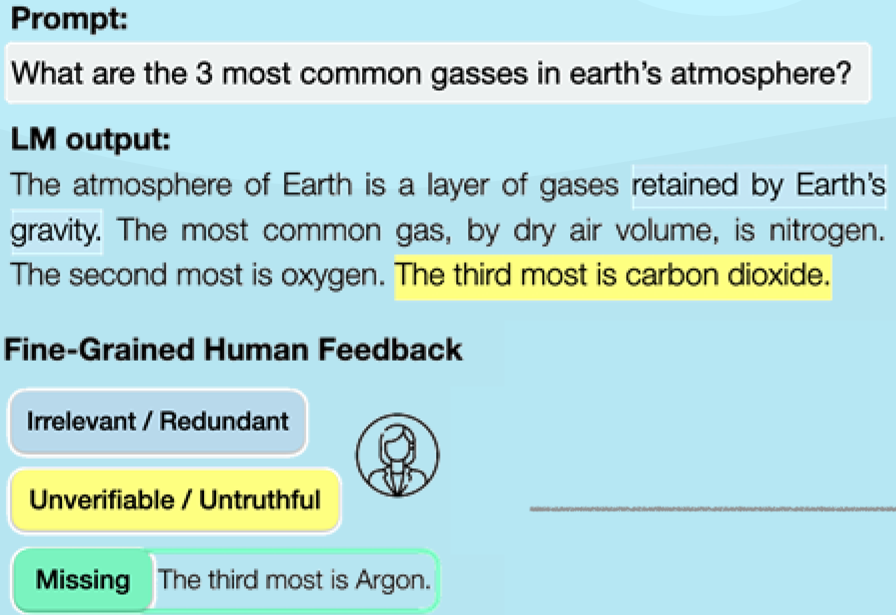
The atmosphere of Earth is a layer of gases retained by Earth's gravity. The most common gas, by dry air volume, is nitrogen. The second most is oxygen. The third most is carbon dioxide.

Localizing feedback / reward

Fine-Grained Human Feedback

Wu, Hu, Shi, Dziri, Suhr, Ammanabrolu, Smith, Ostendorf, Hajishirzi. *Fine-Grained Human Feedback Gives Better Rewards for Language Model Training.* NeurIPS 2023.

Fine-grained feedback is more explicit and reliable!



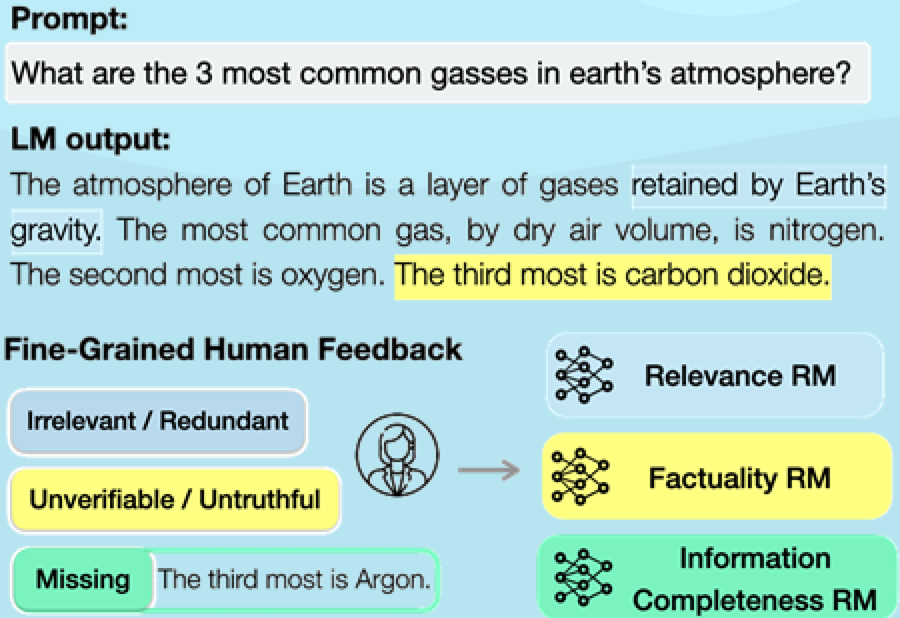
Localizing feedback / reward

Categorizing feedback / reward

Wu, Hu, Shi, Dziri, Suhr, Ammanabrolu, Smith, Ostendorf, Hajishirzi. *Fine-Grained Human Feedback Gives Better Rewards for Language Model Training.* NeurIPS 2023.

Fine-grained RLHF

Step 1: Collect fine-grained feedback and train reward models

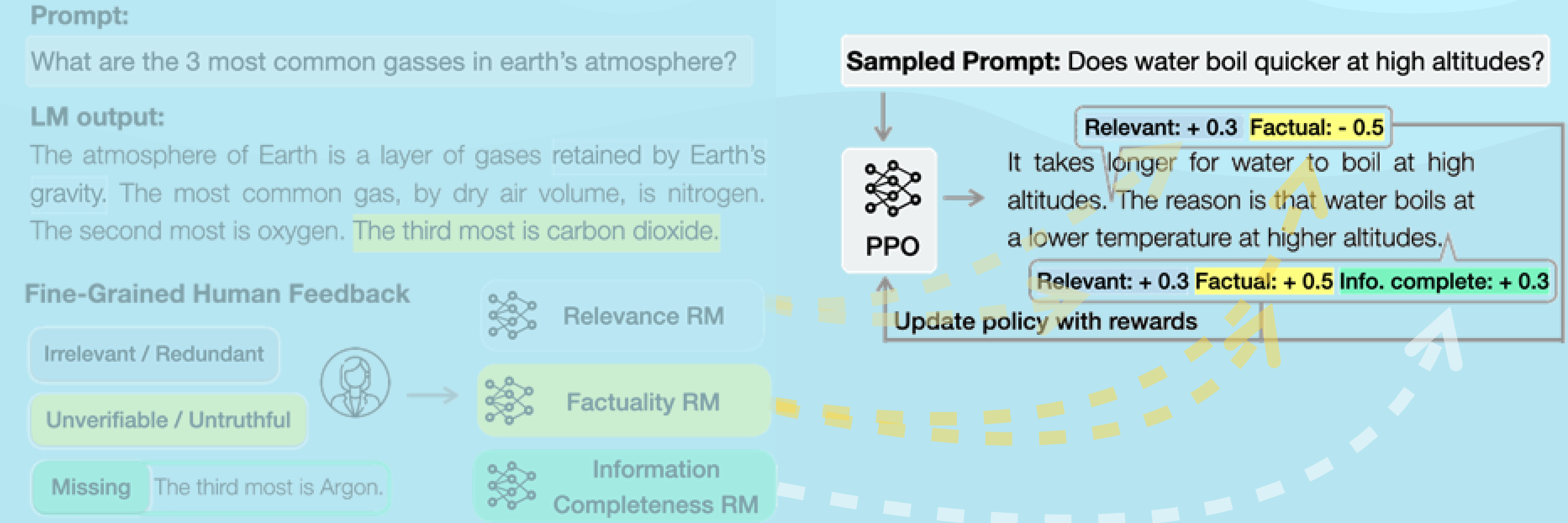


Wu, Hu, Shi, Dziri, Suhr, Ammanabrolu, Smith, Ostendorf, Hajishirzi. *Fine-Grained Human Feedback Gives Better Rewards for Language Model Training.* NeurIPS 2023.

Fine-grained RLHF

Step 1: Collect fine-grained feedback and train reward models

Step 2: Refine the policy LM against the reward models using RL



Wu, Hu, Shi, Dziri, Suhr, Ammanabrolu, Smith, Ostendorf, Hajishirzi. *Fine-Grained Human Feedback Gives Better Rewards for Language Model Training.* NeurIPS 2023.

Fine-grained reward assignment during RL

Multiple reward models associated with different feedback types

Provide dense reward for every LM output segment

$$r_t = \sum_{k=1}^{K} \sum_{j=1}^{L_k} \left(\underline{\mathbb{1}(t = T_j^k)}\, w_k\, \underline{R_{\phi_k}(x, y, j)} \right)$$

Assign at the end of each segment

Each reward model outputs a reward for every segment in LM output

A vs B: Bradley Terry

- Pairwise preference models make the Bradley Terry assumption that underlying preference distribution is IID and pairwise prefs are generated with a fn of the form for some real no.s all

$$\mathbb{P}\{i \succ j\} = \frac{\alpha_i}{\alpha_i + \alpha_j}.$$

$$\mathcal{D} = \{x^i, y_w^i, y_l^i\}$$

Prompt

Preferred response

Dispreferred response

Reward assigned to **preferred** and **dispreferred** responses

$$p(y_w \succ y_l \mid x) = \sigma(r(x, y_w) - r(x, y_l))$$

Slide credits to Rafael Rafailov, Archit Sharma, Eric Mitchell. Stanford.

DPO

Bradley Terry Reward Loss

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

Rewrite rewards in terms of policy. “Closed form”

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

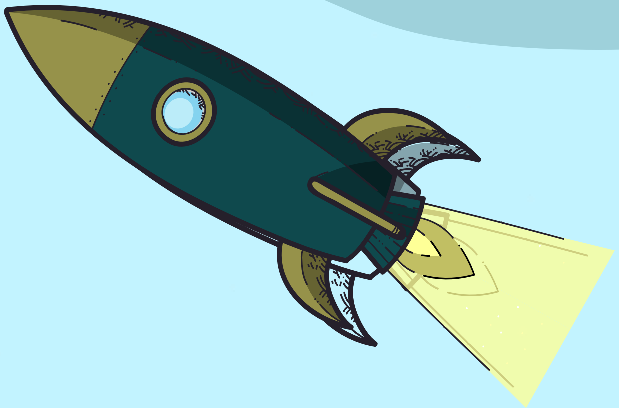
Put it all together

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

Slide credits to Rafael Rafailov, Archit Sharma, Eric Mitchell. Stanford.

Cans of Worms Time

Things I hear a lot I don't want to hear from y'all so I'm preempting it by opening the cans first



DPO or RLHF?

Incorrect question. DPO is also RL!

It is just **Offline** RL while PPO is **Online** RL

Offline RL: You have a large dataset of <data, reward/preference> pairs and need to learn policy from that.

Online RL: You have a reward function you can query while actively generating. Much closer to learning from “realtime” feedback

PPO vs DPO

PPO (any online RL)

Pros:
- Can optimize for arbitrary forms of feedback and metrics
- Theoretically much higher perf due to exploration + personalized learning

Cons:
- Many Eng challenges*

DPO

Pros:
- Easy to implement
- Can recover a reward from trained policy

Cons:
- No exploration (personalized learning)
- Cannot use any type of feedback except for BT/PL
- Easy to overfit to noisy offline dataset

*つ ◕_◕ ༽つ PLS GIB ENG SUPPORT つ ◕_◕ ༽つ つ ◕_◕ ༽つ WE'RE DYING PLS SEND HELP つ ◕_◕ ༽つ

Big remaining (reward) problems

- Human preference distributions are long-tailed, averaging them into one RM is not ideal. What now? (multi-objective RL)
- Humans are bad at expressing their own preferences. Can we elicit them? (yes)
- How to improve sample efficiency of human feedback learning?
- How to chase changing preferences through time?

RLHF is only for "AI Safety"

- No <3
- RLHF is: improving reasoning paths, calibrating confidences, etc etc. It is not the lobotomy algorithm.
- "AI Safety" is mostly just legal coverage, should be defined by users at inference (esp in enterprise usecases)
- "Harmlessness" training directly reduces "helpfulness" – very difficult multi objective optimization