

CS 690: Human-Centric Machine Learning

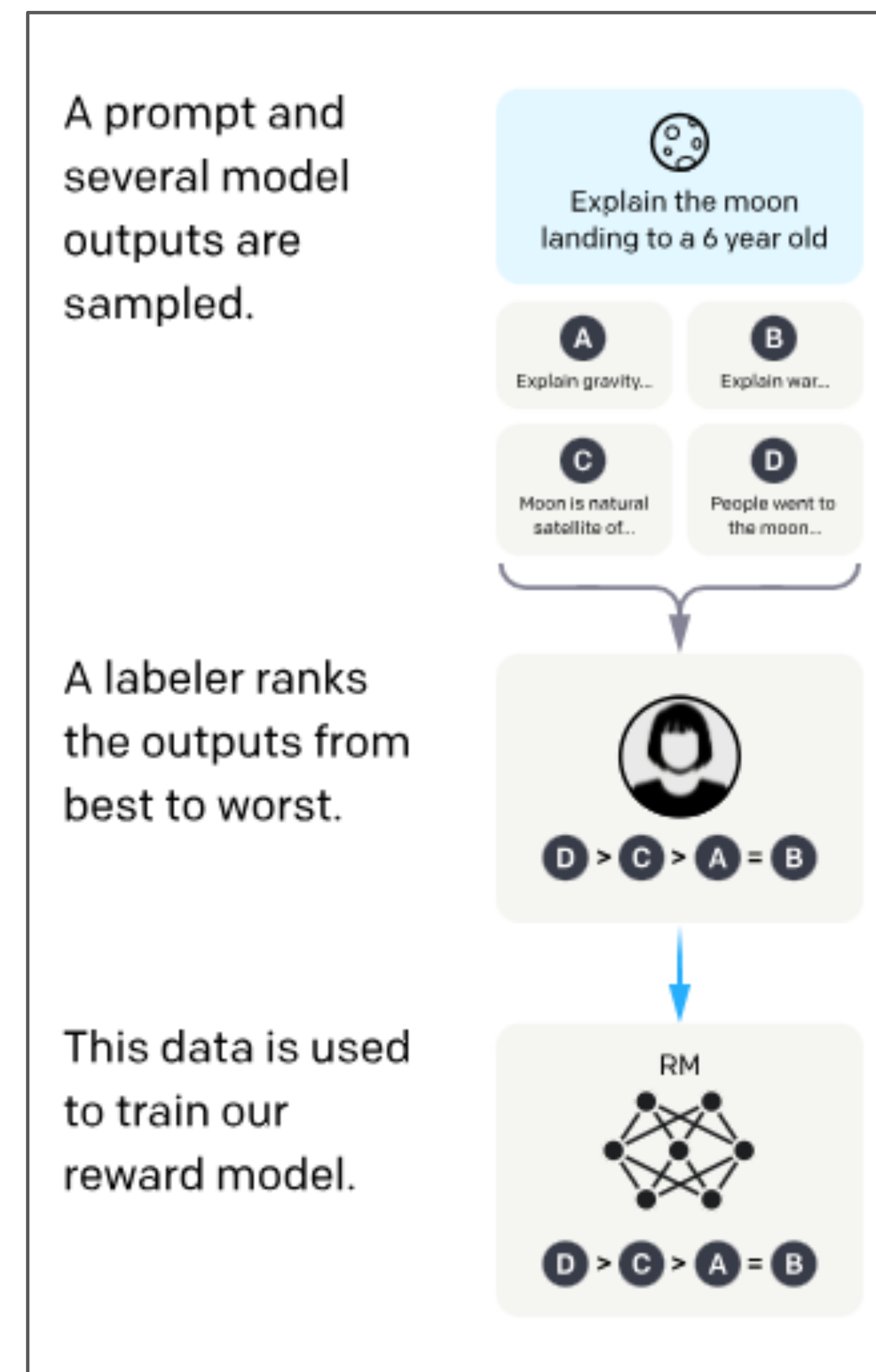
Prof. Scott Niekum

Fine-grained RLHF

Are binary labels enough?

Challenges with standard RLHF binary preference data collection process:

- Holistic and therefore conveys limited information
- Difficult for human annotators to compare overall model responses when outputs contain a mix of undesired behaviour
- No information about which aspects of the output influenced the human preference
- Different people may agree on all fine grained aspects, but disagree on weighting and therefore overall rating!



<https://openai.com/research/instruction-following>

Fine-grained RLHF

The proposed framework enables learning fine-grained rewards in two respects:

1. **Density**, providing a reward after every segment (e.g., a sentence) is generated
2. Incorporating **multiple reward models** associated with different feedback types (e.g., factual incorrectness, irrelevance, and information incompleteness).

It also allows combining reward models with different weights and thus control the model training process towards a customized combination of desired behaviors

Prompt:

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

LM output:

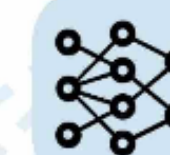
The atmosphere of Earth is a layer of gases retained by Earth's gravity. The most common gas, by dry air volume, is nitrogen. The second most is oxygen. The third most is carbon dioxide.

Fine-Grained Human Feedback

Irrelevant / Redundant

Unverifiable / Untruthful

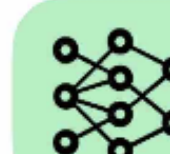
Missing The third most is Argon.



Relevance RM

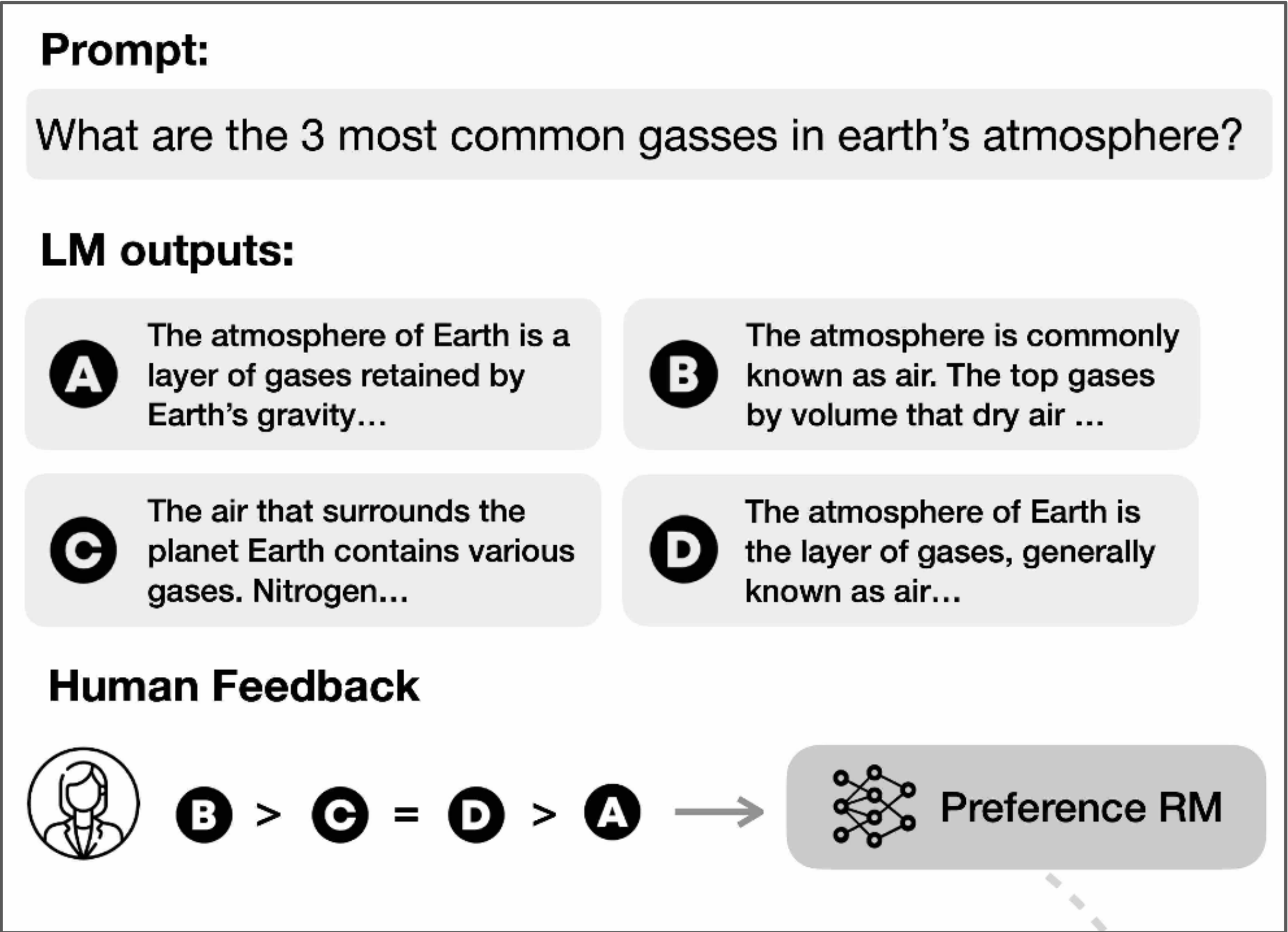


Factuality RM

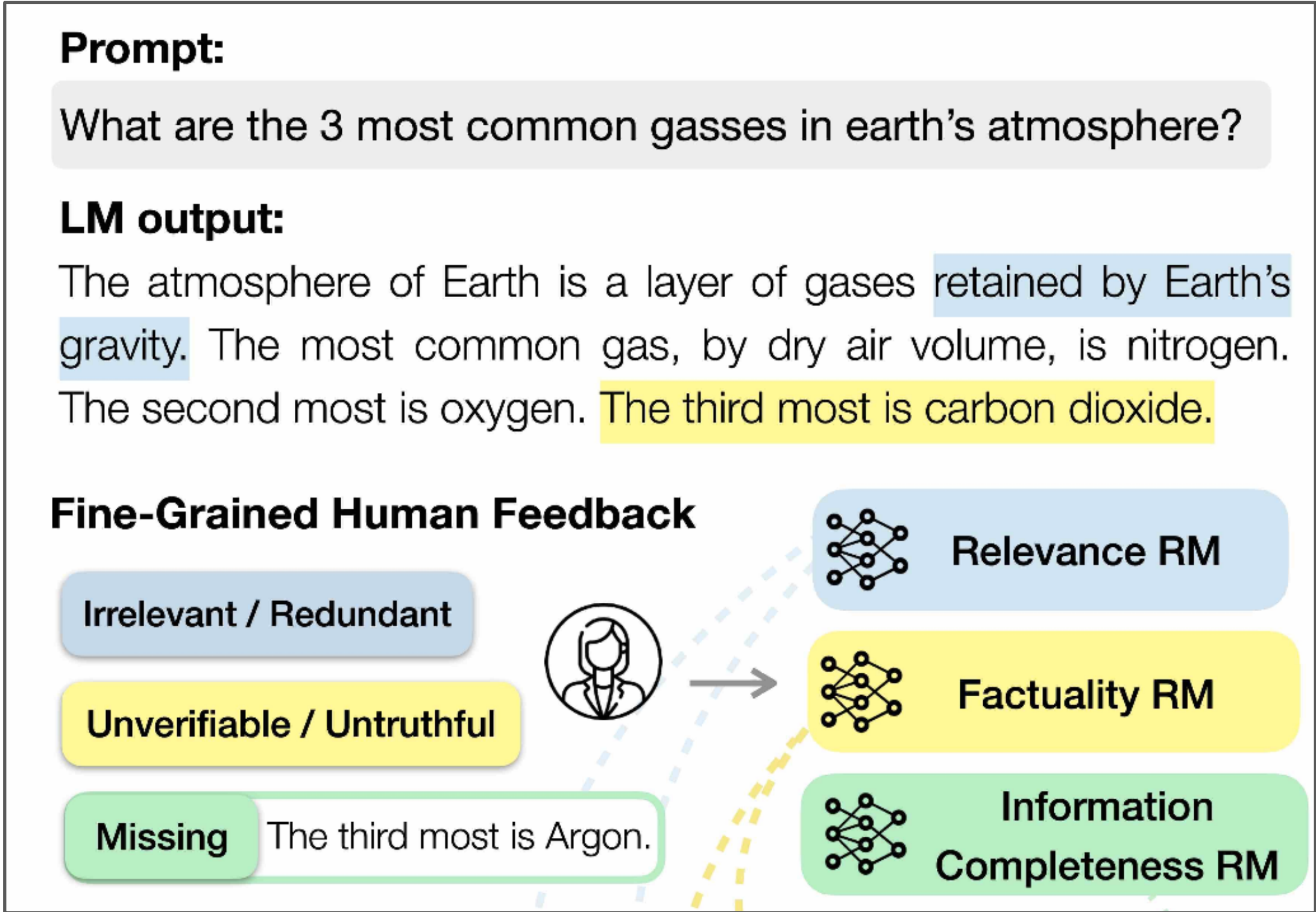


Information Completeness RM

RLHF vs Fine-grained RLHF



Single reward model



3 different reward models

Fine-grained RLHF

The fine-grained R computes rewards on distinct categories of undesired behaviors

Let $k \rightarrow$ number of error categories and each error categories denoted by C_k

Examples

$C1$: irrelevance, repetition, or incoherence

$C2$: incorrect or unverifiable facts

$C3$: incomplete information

R computes rewards densely: over subsequences of the generated output

The output y is segment into L_k segments corresponding to the density level of the reward

$$y = (y_1^k, y_2^k, \dots, y_{L_k}^k)$$

Each segment y_j^k ends at timestep T_j^k

The reward model

The combined reward function for each token a_t

$$r_t = \sum_{k=1}^K \sum_{j=1}^{L_k} \left(\mathbb{1}(t = T_j^k) w_k R_{\phi_k}(x, y, j) \right) - \beta \log \frac{P_{\theta}(a_t | s_t)}{P_{\theta_{\text{init}}}(a_t | s_t)}$$

K fine-grained
reward models
for different error
categories

weight assigned
to the different
reward models

$R_{\phi_k}(x, y, j)$
is the reward
output for each
segment

KL penalty term to
maintain fluency of
the output

Task 1 - Detoxification

- Only behaviour studied in this task is toxicity
- A dense sentence-level fine-grained paradigm is compared to the holistic RLHF reward
- Conducted on 'REALTOXICITYPROMPTS' dataset
- **Holistic reward**
 - PERSPECTIVE API generates a toxicity score between 0 & 1 for entire output
 - $R = 1 - \text{PERSPECTIVE}(y)$
- **Fine-grained reward**
 - PERSPECTIVE API is queried for each sentence
 - $R(y_j) = \text{PERSPECTIVE}(y_1, \dots, y_{j-1}) - \text{PERSPECTIVE}(y_1, \dots, y_j)$

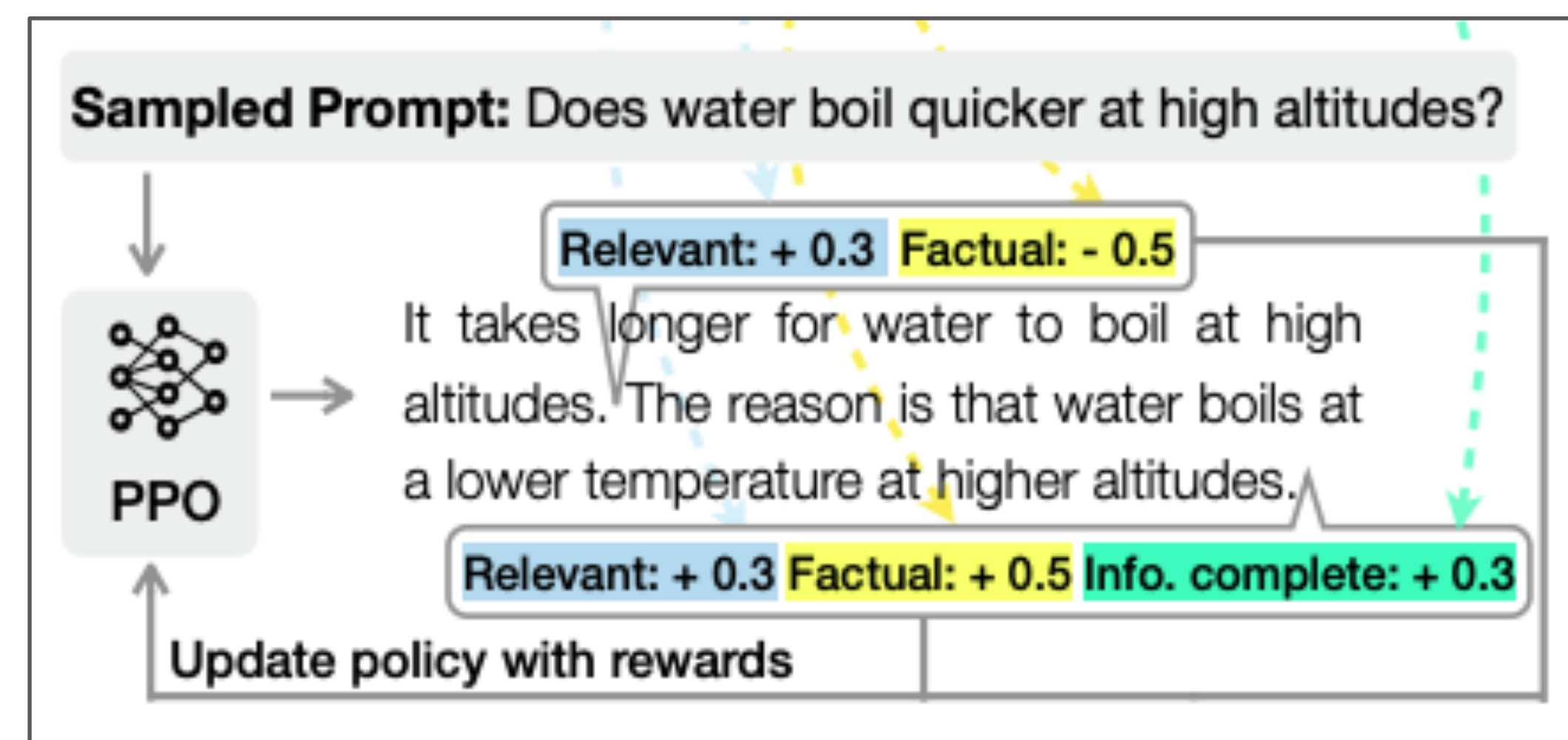
Detoxification - Results

- Models compared:
 - Holistic RLHF
 - GeDI: generative discriminator model to steer responses in desired direction
 - DEXPERTS: combines a pre-trained LM with “expert”/”anti-expert” LM
- Fine grained RL achieved a lower toxicity and perplexity score while keeping diversity similar

	Toxicity avg max (↓)	Fluency PPL (↓)	Diversity dist-2 (↑) dist-3 (↑)	
GPT-2	0.192	9.58	0.947	0.931
Controlled Generation				
GeDi	0.154	24.78	0.938	0.938
DEXPERTS	0.136	22.83	0.932	0.922
Hol. RLHF	0.130	11.75	0.943	0.926
F.G. RLHF	0.081	9.77	0.949	0.932

Task 2 - Long-form QA

- QA-FEEDBACK dataset is collected for answering ambiguous factoid questions
- Given a question q and set of passages $P = \{p_1, \dots, p_{|P|}\}$, generate a long form response y
- T5-large is trained on 1k samples from the above dataset: This model is called the **SFT**
- Outputs are sampled from the **SFT** to collect fine-grained human feedback on three error categories at three density levels: sub-sentence, sentence and whole sequence
- Annotators mark the span of text associated with each identified error type
- Also collect pairwise-preference comparison data



Long-form QA - Reward models

- C1: irrelevance, repetition, and incoherence (rel.); The reward model has the density level of sub-sentences; i.e., returns a score for each sub-sentence. If the sub-sentence is irrelevant, repetitive, or incoherent, the reward is -1; otherwise, the reward is +1.
- C2: incorrect or unverifiable facts (fact.); The reward model has the density level of sentences; i.e., returns a score for each sentence. If the sentence has any factual error, the reward is -1; otherwise, the reward is +1.
- C3: incomplete information (comp.); The reward model checks if the response is complete and covers all the information in the reference passages that are related to the question. This reward model gives one reward for the whole response.

Long-form QA - Results

- The proposed model is compared to the initial T5 SFT model, RLHF with holistic preference-based reward and also a fully supervised T5 model
- Fine-grained RLHF performs better than Preference RLHF on all error types
- Overall, RLHF is more effective in removing factual errors compared to SFT

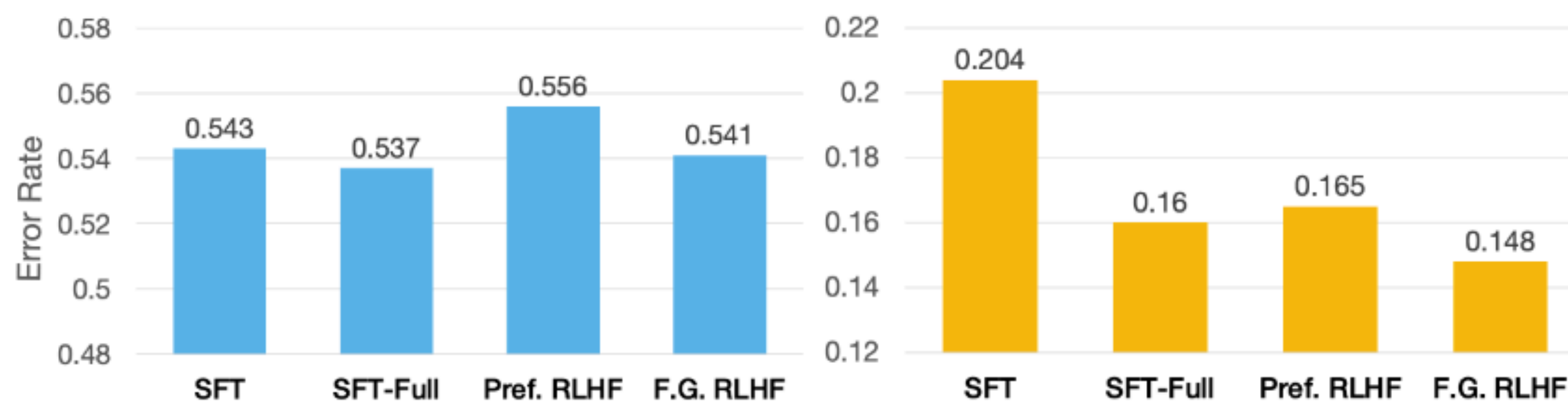
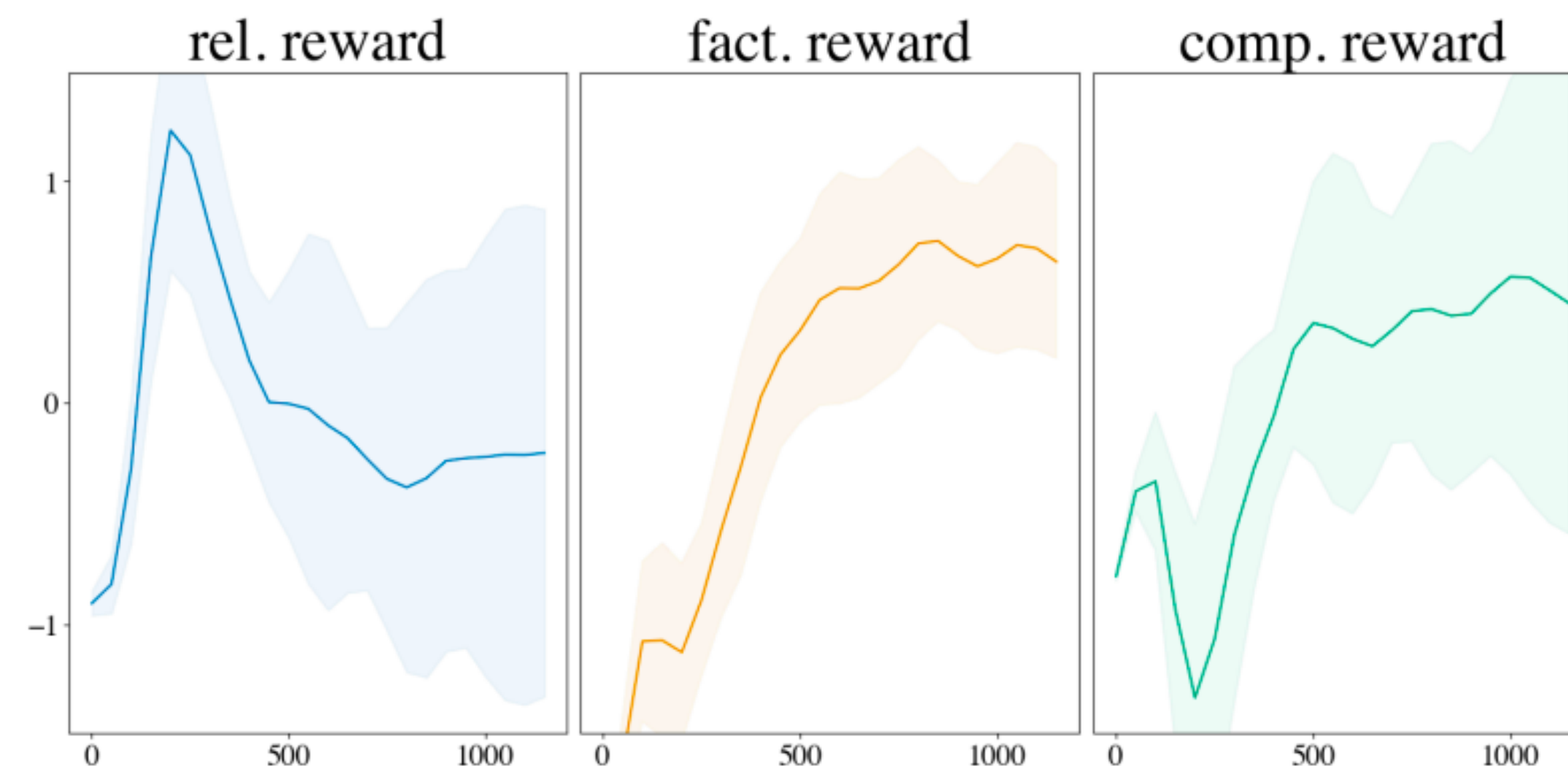


Figure 3: Human evaluation on *rel.* (left) and *fact.* (right) error, measured by % of sub-sentences that contain the error type (↓).

LM customization




- With multiple rewards, adjusting weights leads to different LM behaviours
- *Increasing w_1* : When the weights for the ‘rel.’ model are increased → shorter responses, sometimes incomplete and factually incorrect
- Some objectives can be clashing (like relevance and information completeness)







Reward vs Training steps




Process-based supervision

The denominator of a fraction is 7 less than 3 times the numerator. If the fraction is equivalent to $2/5$, what is the numerator of the fraction? (Answer:)

   Let's call the numerator x .

   So the denominator is $3x-7$.

   We know that $x/(3x-7) = 2/5$.

   So $5x = 2(3x-7)$.

   $5x = 6x - 14$.

   So $x = 7$.

Figure 1: A screenshot of the interface used to collect feedback for each step in a solution.

Domain and dataset

- Domain: MATH dataset
- They release a human dataset, PRM800K, which contains 800K step-level labels across 75K solutions to 12K MATH problems

MATH Dataset (Ours)

Problem: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?

Solution: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2} = 6$ results). The total number of distinct pairs of marbles Tom can choose is $1 + 6 = \boxed{7}$.

Problem: If $\sum_{n=0}^{\infty} \cos^{2n} \theta = 5$, what is $\cos 2\theta$?

Solution: This geometric series is $1 + \cos^2 \theta + \cos^4 \theta + \dots = \frac{1}{1 - \cos^2 \theta} = 5$. Hence,

$$\cos^2 \theta = \frac{4}{5}. \text{ Then } \cos 2\theta = 2 \cos^2 \theta - 1 = \boxed{\frac{3}{5}}.$$

Problem: The equation $x^2 + 2x = i$ has two complex solutions. Determine the product of their real parts.

Solution: Complete the square by adding 1 to each side.

Then $(x + 1)^2 = 1 + i = e^{\frac{i\pi}{4}} \sqrt{2}$, so $x + 1 = \pm e^{\frac{i\pi}{8}} \sqrt[4]{2}$.

The desired product is then

$$\begin{aligned} & (-1 + \cos(\frac{\pi}{8}) \sqrt[4]{2}) (-1 - \cos(\frac{\pi}{8}) \sqrt[4]{2}) = \\ & 1 - \cos^2(\frac{\pi}{8}) \sqrt{2} = 1 - \frac{(1 + \cos(\frac{\pi}{4}))}{2} \sqrt{2} = \boxed{\frac{1 - \sqrt{2}}{2}}. \end{aligned}$$

Summary of results

1. We show that process supervision can train much more reliable reward models than outcome supervision. We use our state-of-the-art PRM to solve 78.2% of problems from a representative subset of the MATH test set.
2. We show that a large reward model can reliably approximate human supervision for smaller reward models, and that it can be used to efficiently conduct large-scale data collection ablations.
3. We show that active learning leads to a $2.6\times$ improvement in the data efficiency of process supervision.
4. We release our full process supervision dataset, PRM800K, to promote related research.

Results

Let

$$x^8 + 3x^4 - 4 = p_1(x)p_2(x) \cdots p_k(x),$$

where each non-constant polynomial $p_i(x)$ is monic with integer coefficients, and cannot be factored further over the integers. Compute $p_1(1) + p_2(1) + \cdots + p_k(1)$.

I notice that the given polynomial has even degree and only even powers of x , so I can try to make a substitution to simplify it.

Let $y = x^4$, then the polynomial becomes $y^2 + 3y - 4$, which is a quadratic equation.

I can factor this quadratic equation as $(y + 4)(y - 1)$, so the original polynomial is $(x^4 + 4)(x^4 - 1)$.

Now I need to factor each of these factors further, if possible, over the integers.

For the first factor, $x^4 + 4$, I recall the Sophie Germain identity, which says that $a^4 + 4b^4 = (a^2 + 2b^2 + 2ab)(a^2 + 2b^2 - 2ab)$.

Applying this identity with $a = x$ and $b = 1$, I get $x^4 + 4 = (x^2 + 2x + 2)(x^2 - 2x + 2)$.

For the second factor, $x^4 - 1$, I can use the difference of squares formula, which says that $a^2 - b^2 = (a + b)(a - b)$.

Applying this formula with $a = x^2$ and $b = 1$, I get $x^4 - 1 = (x^2 + 1)(x^2 - 1)$.

Then I can apply the difference of squares formula again with $a = x$ and $b = 1$, to get $x^2 - 1 = (x + 1)(x - 1)$.

So the original polynomial is $(x^4 + 4)(x^4 - 1) = (x^2 + 2x + 2)(x^2 - 2x + 2)(x^2 + 1)(x + 1)(x - 1)$.

These are all the irreducible factors over the integers, so $p_1(x) = x^2 + 2x + 2$, $p_2(x) = x^2 - 2x + 2$, $p_3(x) = x^2 + 1$, $p_4(x) = x + 1$, and $p_5(x) = x - 1$.

To find the sum of their values at $x = 1$, I just need to plug in $x = 1$ into each factor and add them up.

I get $p_1(1) = 1^2 + 2 \cdot 1 + 2 = 5$, $p_2(1) = 1^2 - 2 \cdot 1 + 2 = 1$, $p_3(1) = 1^2 + 1 = 2$, $p_4(1) = 1 + 1 = 2$, and $p_5(1) = 1 - 1 = 0$.

So the final answer is $5 + 1 + 2 + 2 + 0 = 10$.

Answer: 10

Let

$$x^8 + 3x^4 - 4 = p_1(x)p_2(x) \cdots p_k(x),$$

where each non-constant polynomial $p_i(x)$ is monic with integer coefficients, and cannot be factored further over the integers. Compute $p_1(1) + p_2(1) + \cdots + p_k(1)$.

This problem is about factoring a polynomial over the integers, which can be a tricky task.

One possible strategy is to look for rational roots of the polynomial, using the rational root theorem.

However, that would involve checking a lot of possible factors of -4 , and there is no guarantee that the polynomial has any rational roots at all.

Another possible strategy is to use some algebraic tricks to simplify the polynomial or reduce its degree.

I notice that the polynomial has only even powers of x , so I can try to substitute $y = x^2$ and see what happens.

Then the polynomial becomes $y^4 + 3y^2 - 4$, which is a quadratic equation in y^2 .

I can try to factor this quadratic equation by looking for two numbers that multiply to -4 and add to 3 .

Those numbers are 4 and -1 , so I can write $y^4 + 3y^2 - 4 = (y^2 + 4)(y^2 - 1)$.

Now I can substitute back $x^2 = y$ and get $x^8 + 3x^4 - 4 = (x^4 + 4)(x^4 - 1)$.

This looks promising, because I have reduced the degree of the factors from 8 to 4 .

I can try to factor each of these factors further, using the difference of squares formula.

I get $x^8 + 3x^4 - 4 = (x^2 + 2)(x^2 - 2)(x^2 + 1)(x^2 - 1)$.

I can apply the difference of squares formula again to the last factor and get $x^8 + 3x^4 - 4 = (x^2 + 2)(x^2 - 2)(x^2 + 1)(x + 1)(x - 1)$.

Now I have factored the polynomial completely into monic linear and quadratic factors with integer coefficients.

These are the $p_i(x)$'s that the problem is asking for.

To find the sum of their values at $x = 1$, I just need to plug in $x = 1$ into each factor and add them up.

I get $p_1(1) + p_2(1) + \cdots + p_k(1) = (1^2 + 2)(1^2 - 2)(1^2 + 1)(1 + 1)(1 - 1)$.

Simplifying, I get $p_1(1) + p_2(1) + \cdots + p_k(1) = (3)(-1)(2)(2)(0)$.

Multiplying, I get $p_1(1) + p_2(1) + \cdots + p_k(1) = 0$.

Answer: 0

Figure 2: Two solutions to the same problem, graded by the PRM. The solution on the left is correct while the solution on the right is incorrect. A green background indicates a high PRM score, and a red background indicates a low score. The PRM correctly identifies the mistake in the incorrect solution.

Results

	ORM	PRM	Majority Voting
% Solved (Best-of-1860)	72.4	78.2	69.6

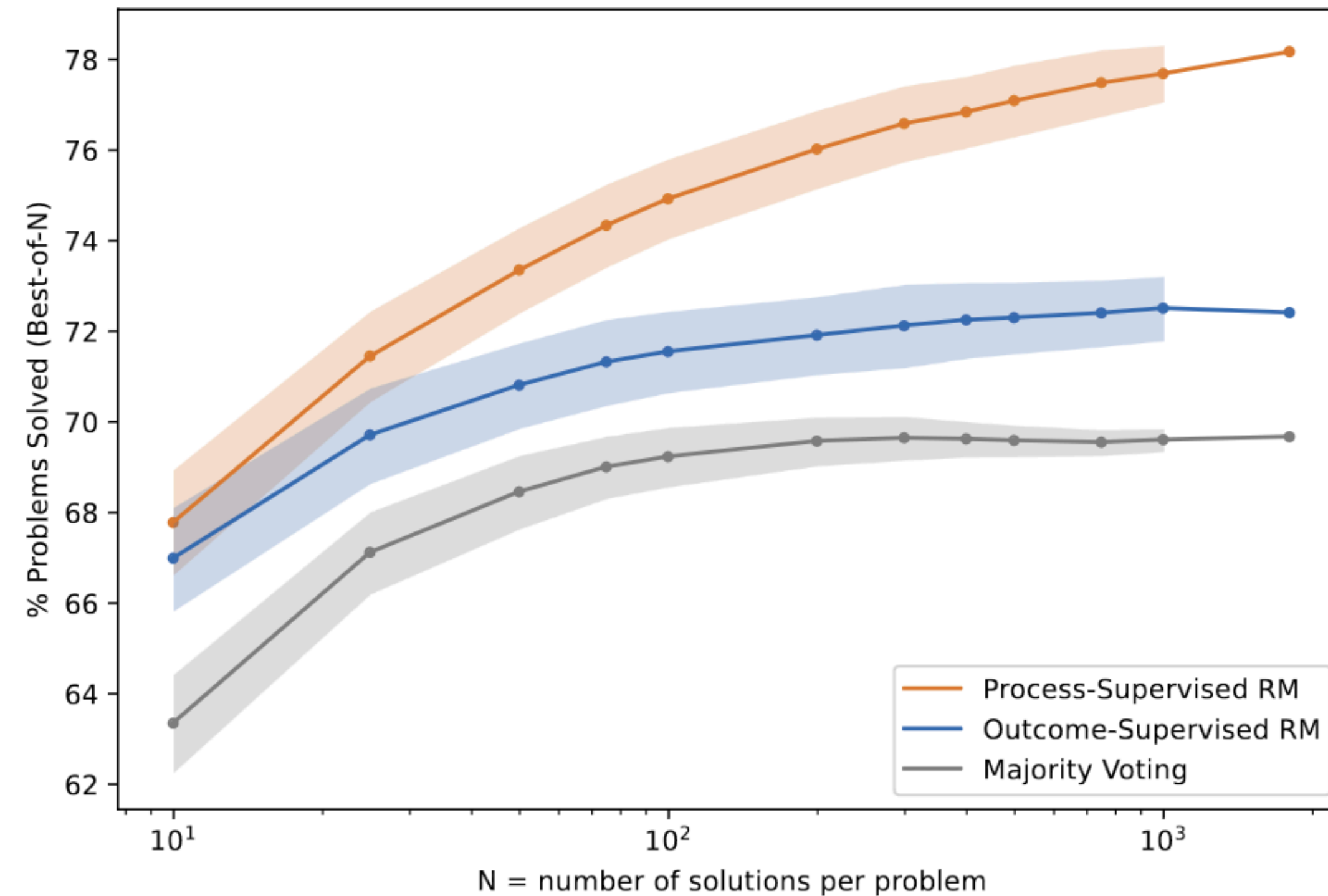
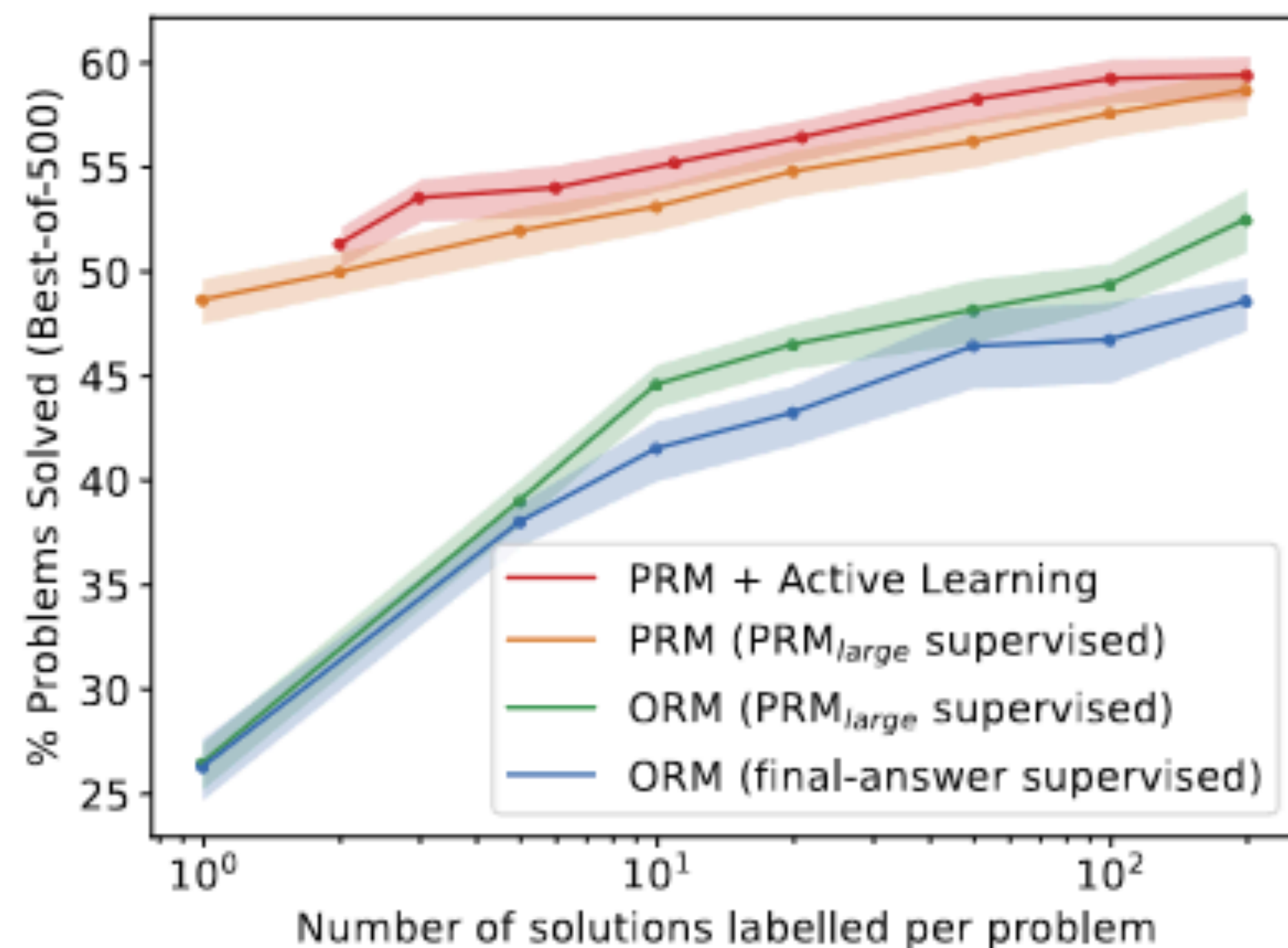


Figure 3: A comparison of outcome-supervised and process-supervised reward models, evaluated by their ability to search over many test solutions. Majority voting is shown as a strong baseline. For $N \leq 1000$, we visualize the variance across many subsamples of the 1860 solutions we generated in total per problem.

Results



(a) Four series of reward models trained using different data collection strategies, compared across training sets of varying sizes.

Hypothesized benefits of process supervision

- Better credit assignment
- More human-interpretable: makes LLM “think like” a human
- Inherently safer: directly optimizes for reasoning rather than a proxy (e.g. getting the right answer for the wrong reason)
- Better empirical sample efficiency/performance

Moving forward with fine-grained feedback