Enhancing Multi-Step Reasoning via Process-Supervised Reinforcement Learning from Human Feedback

Anonymous ACL submission

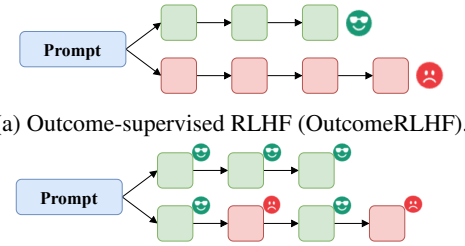
Abstract

Large language models (LLMs) demonstrate significant reasoning capabilities, especially through step-by-step reasoning paths. However, their proficiency in mathematical reasoning remains limited. We generalize the Reinforcement Learning from Human Feedback (RLHF) framework by integrating per-step reward signals in order to enhance LLMs' reasoning abilities. This approach differs from traditional outcome-based models by offering step-wise guidance during learning. Experiments on MATH and PRM800K datasets show that our process-supervised RLHF significantly improves reasoning accuracy over its outcome-based counterpart, marking a notable advancement in LLMs for complex reasoning tasks.

1 Introduction

Recent studies have highlighted the potential of large language models (LLMs) in developing reasoning capabilities, in particular by constructing intermediate reasoning steps (Nye et al., 2021; Wei et al., 2022; Kojima et al., 2022; Lewkowycz et al., 2022). These steps bridge the gap between input and output, effectively breaking down complex reasoning tasks into simpler, more manageable ones.

Despite these advancements, multi-step mathematical reasoning remains a challenging area for LLMs, with frequent logical errors being a primary concern. A particularly persistent problem is the *error propagation* issue, where initial inaccuracies in reasoning steps lead to further errors, resulting in incorrect conclusions (Shen et al., 2021). To mitigate this, researchers proposed "rescoring" models to assess LLM outputs (Cobbe et al., 2021; Uesato et al., 2022; Lightman et al., 2023). These models, which evaluate either the entire reasoning path ("outcome-based") or each reasoning step ("process-based"), have shown promise in enhancing LLM reasoning performance. However, their effectiveness is often constrained by the need to generate a large number of (diverse) candidate outputs, with the quality of these outputs critically dependent on the LLM's reasoning ability. This limitation can lead to suboptimal ranking and underutilization of the scoring models' full potential.



(a) Outcome-supervised RLHF (OutcomeRLHF).

(b) Process-supervised RLHF (ProcessRLHF, *this work*).

Figure 1: OutcomeRLHF vs. ProcessRLHF in multi-step reasoning. The former assesses the final result, while the latter evaluates each step, providing immediate feedback to guide the model towards the correct path.

An alternative approach, which we employ, use these scoring models to guide self-improvement of LLMs during the training phase rather than post-processing. We generalize the widely-used Reinforcement Learning from Human Feedback (RLHF) framework (Christiano et al., 2017; Ziegler et al., 2019; Böhm et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022) by integrating process-based rewards (Fig. 1b), which better fits multi-step reasoning. This generalization marks a notable shift from the traditional RLHF (Ouyang et al., 2022), which operates on outcome-based rewards (Fig. 1a). We make the following contributions:

- We propose ProcessRHLF by integrating process-based supervision.
- We propose two types of process reward models to utilize various stepwise reward signals.
- Experiments on MATH and PRM800K datasets show that processRLHF significantly improves reasoning accuracy over its outcome-based counterpart.

Problem $\mathbf{x}$ **and Solution** $\mathbf{y} = (\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \mathbf{y}^{(3)}, \mathbf{y}^{(4)}, \mathbf{y}^{(5)}, \mathbf{y}^{(6)})$
$\mathbf{x}$: How many positive whole-number divisors does 196 have?
$\mathbf{y}^{(1)}$: First prime factorize $196 = 2^2 \cdot 7^2$.
$\mathbf{y}^{(2)}$: The prime factorization of any divisor of 196 cannot include any primes other than 2 and 7.
$\mathbf{y}^{(3)}$: We can choose either 0, 1, or 2 as the exponent of 2 in the prime factorization of a divisor of 196.
$\mathbf{y}^{(4)}$: Similarly, we may choose 0, 1, or 2 as the exponent of 7.
$\mathbf{y}^{(5)}$: In total, there are $3 \times 3 = 9$ possibilities for the prime factorization of a divisor of 196.
$\mathbf{y}^{(6)}$: Distinct prime factorizations correspond to distinct integers, so there are $\boxed{9}$ divisors of 196.

Table 1: Illustration of Multi-Step Reasoning in the MATH Dataset. The model is trained to demarcate the final answer with a box for clear evaluation. The content within the final boxed area is extracted as the model's predicted solution for the problem.

2 Problem Formalization

In multi-step reasoning tasks, we define a reasoning problem as a sequence of tokens $\mathbf{x} = (x_1, x_2, \ldots, x_m)$, where each x_i denotes the i-th element or token of the input problem. The primary objective for the model in this context is to process the given input $\mathbf{x}$ and generate a coherent and logically structured sequence of reasoning steps. This sequence is denoted as $\mathbf{y} = (\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(T)})$, where T represents the total number of reasoning steps. Each individual reasoning step, $\mathbf{y}^{(t)}, t = 1, 2, \ldots, T$, is composed of a series of tokens $y_1^{(t)}, y_2^{(t)}, \ldots, y_{n_t}^{(t)}$, where $y_i^{(t)}$ corresponds to the i-th token within the t-th step, and n_t denotes the number of tokens in this step. The final solution or conclusion of the reasoning task is encapsulated in the last step $\mathbf{y}^{(T)}$.

Typically, a pre-trained language model (LM), p_θ, is used to predict the reasoning steps $\mathbf{y}$ from a given problem $\mathbf{x}$, aiming to learn the conditional probability $p_\theta(\mathbf{y} \mid \mathbf{x})$, with parameters θ. The optimization involves Supervised Fine-Tuning (SFT) on a dataset $\mathcal{D}_{\mathbf{xy}}$ of problem-reasoning pairs, by maximizing their log-likelihood:

$$\max_\theta \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_{\mathbf{xy}}} \log p_\theta(\mathbf{y} \mid \mathbf{x}). \tag{1}$$

3 RL for Multi-Step Reasoning

Multi-step reasoning tasks can be alternatively conceptualized within a Reinforcement Learning (RL) framework, where the task is viewed as a sequence of decision-making steps. In this formulation, the reasoning process is represented as a trajectory $\tau = \left\{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots, (s^{(T)}, a^{(T)})\right\}$, where each state-action pair corresponds to a step in the reasoning process. The state $s^{(t)}$ at step t encapsulates the problem $\mathbf{x}$ and all preceding reasoning steps $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(t-1)}$, thus $s^{(1)} = \mathbf{x}$, and $s^{(t)} = (\mathbf{x}, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(t-1)})$ for $t > 1$. The action $a^{(t)}$ at each step is the generation of the current reasoning step $\mathbf{y}^{(t)}$. For notational convenience, we introduce $\mathbf{y}^{<t}$ to represent all preceding reasoning steps up to but not including step t (i.e., $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(t-1)}$), and $\mathbf{y}^{\leq t}$ to denote all steps up to and including step t (i.e., $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(t)}$).

After the SFT phase (Eq. 1), the RL is employed to further enhance the model's performance from utilizing learned reward models.

3.1 Baseline: Outcome-Supervised RLHF

In the OutcomeRLHF (Ouyang et al., 2022), there is an Outcome Reward Model (ORM) R_ϕ, trained on a human preference dataset $\mathcal{D}_{\mathbf{xy}_\pm}$ consisting of tuples $(\mathbf{x}, \mathbf{y}_+, \mathbf{y}_-)$. Here, $\mathbf{x}$ represents a reasoning problem, $\mathbf{y}_+$ is the preferred model output, and $\mathbf{y}_-$ is the non-preferred output, as adjudged by human evaluators. The ORM's role is to assign a singular, comprehensive score to each reasoning trajectory. The training objective of the ORM is defined as:

$$\max_\phi \sum_{(\mathbf{x},\mathbf{y}_+,\mathbf{y}_-) \in \mathcal{D}_{\mathbf{xy}_\pm}} \log \sigma\Big(R_\phi(\mathbf{x}, \mathbf{y}_+) - R_\phi(\mathbf{x}, \mathbf{y}_-)\Big)$$

where σ is the sigmoid function, and $R_\phi(\mathbf{x}, \mathbf{y})$ represents the ORM's output score for a given problem $\mathbf{x}$ and its reasoning steps $\mathbf{y}$, parameterized by ϕ.

With a collection of unlabeled reasoning problems $\mathcal{D}_{\mathbf{x}}$, we optimize the model p_θ to maximize the expected reward across all potential trajectories derived from the model, with the objective

$$\max_\theta \sum_{\mathbf{x} \in \mathcal{D}_{\mathbf{x}}} \mathbb{E}_{\mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{x})} \left[R_\phi(\mathbf{x}, \mathbf{y})\right]. \tag{2}$$

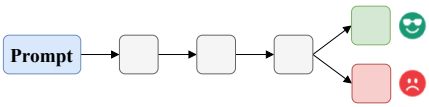
This is typically done via policy optimization algorithms, such as Proximal Policy Optimization (Schulman et al., 2017).

3.2 Our Process-Supervised RLHF

To enhance the supervisory signal during RL, we propose employing a Process Reward Model (PRM) R_ϕ, which evaluates the immediate outcomes of each action, providing step-by-step feedback. The PRM assigns a reward for each state-action pair $(s^{(t)}, a^{(t)})$ as $R_\phi(\mathbf{x}, \mathbf{y}^{\leq t})$. Given a set of unlabeled reasoning questions $\mathcal{D}_{\mathbf{x}}$, we optimize model p_θ to maximize the expectation of accumulated rewards of all possible trajectories:

$$\max_\theta \sum_{\mathbf{x} \in \mathcal{D}_{\mathbf{x}}} \mathbb{E}_{\mathbf{y} \sim p_\theta(\mathbf{y}|\mathbf{x})} \left[\sum_t R_\phi(\mathbf{x}, \mathbf{y}^{\leq t})\right] \tag{3}$$

(a) PRM via Step-wise Binary Classification (PRM-step).



(b) PRM via Alternative Steps Comparison (PRM-alt).

Figure 2: Proposed methods for constructing Process Reward Models (PRMs). (a) **PRM-step**: each step of the reasoning process is annotated with a binary label to indicate its correctness. The PRM is trained to predict these labels. (b) **PRM-alt**: training samples are constructed from reasoning sequences with a shared prefix, followed by divergent alternative steps. The PRM is trained to rank these alternatives correctly.

where $\mathbf{y}^{\leq t} = (\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(t)})$, and for time step t, the model will get an instance reward $R_\phi(\mathbf{x}, \mathbf{y}^{\leq t})$. This objective is also optimized using policy optimization algorithms. We refer this new framework as ProcessRLHF.

The effectiveness of ProcessRLHF on multi-step reasoning tasks greatly depends on the design and implementation of reward models. Below we explore two methods for constructing such models.

Learning Process Reward Model via Step-wise Binary Classification. In the first method, human preferences are provided as step-wise annotations (Fig. 2a). The dataset $\mathcal{D}_{\mathbf{xyl}}$ includes tuples $(\mathbf{x}, \mathbf{y}, \mathbf{l})$, where alongside a problem $\mathbf{x}$ and a reasoning trajectory $\mathbf{y}$, there is a sequence of binary labels $\mathbf{l} = (l_1, l_2, \ldots, l_T)$. These labels indicate the correctness of each step, with $l_t = 1$ signifying correctness. The reward model R_ϕ is trained to classify these labels accurately, maximizing the negative binary cross-entropy (or equivalently, minimizing the binary cross-entropy):

$$\max_\phi \sum_{(\mathbf{x},\mathbf{y},\mathbf{l}) \in \mathcal{D}_{\mathbf{xyl}}} \sum_t \Big[l_t \log \sigma \left(R_\phi(\mathbf{x}, \mathbf{y}^{\leq t}) \right) + (1 - l_t) \log \big(1 - \sigma \left(R_\phi(\mathbf{x}, \mathbf{y}^{\leq t}) \right) \big) \Big]$$

Here, $R_\phi(\mathbf{x}, \mathbf{y}^{\leq t})$ calculates the reward for the t-th step given the problem $\mathbf{x}$ and the reasoning trajectory up to that step $\mathbf{y}^{\leq t}$, with ϕ representing the model parameters. We refer the reward model learned by this approach as **PRM-step**.

Learning Process Reward Model via Alternative Step Comparison. The second method leverages comparisons of alternative reasoning steps for the same preceding steps (Fig. 2b). The dataset $\mathcal{D}_{\mathbf{xy}^{\leq}_{\pm}}$ contains tuples $(\mathbf{x}, \mathbf{y}_+^{\leq t}, \mathbf{y}_-^{\leq t})$. Here, $\mathbf{x}$ is the problem statement, while $\mathbf{y}_+^{\leq t}$ and $\mathbf{y}_-^{\leq t}$ are two reasoning trajectories with the same prefix up to step $t-1$, but diverging at step t (i.e., $\mathbf{y}_+^{<t} = \mathbf{y}_-^{<t}$ and $\mathbf{y}_+^{(t)} \neq \mathbf{y}_-^{(t)}$). We train the model R_ϕ to favor the preferred reasoning path $\mathbf{y}_+^{\leq t}$ over the non-preferred $\mathbf{y}^{\leq t}{}_-$ by maximizing:

$$\max_\phi \sum_{(\mathbf{x},\mathbf{y}_+^{\leq t},\mathbf{y}_-^{\leq t}) \in \mathcal{D}_{\mathbf{xy}^{\leq}_{\pm}}} \log \sigma \Big(R_\phi(\mathbf{x}, \mathbf{y}_+^{\leq t}) - R_\phi(\mathbf{x}, \mathbf{y}_-^{\leq t}) \Big)$$

In this formulation, $R_\phi(\mathbf{x}, \mathbf{y}^{\leq t})$ computes the reward for the t-th step based on the problem $\mathbf{x}$ and the trajectory up to that step $\mathbf{y}^{\leq t}$, with ϕ as the parameter set. We refer the reward model learned by this method as **PRM-alt**.

4 Experiments and Results

4.1 Datasets

Our experiments employ two datasets. The primary dataset is the MATH dataset (Hendrycks et al., 2021), complemented by the PRM800K dataset (Lightman et al., 2023), which provides step-level human preference annotations on solutions derived from the MATH dataset:

MATH. This dataset contains 12,500 high school math competition problems, each with a detailed step-by-step solution. See Tab. 1 for an example problem with reasoning steps.

PRM800K. This dataset contains around 800,000 step-level correctness labels for 75,000 model-generated solutions to 12,000 problems from the MATH dataset.

We further constructed the following *derived datasets* for different training phases and models:

- **SFT set $\mathcal{D}_{\mathbf{xy}}$ & RLHF set $\mathcal{D}_{\mathbf{x}}$**: We split the MATH dataset into 12,000 training and 500 test problems, in line with Lightman et al. (2023). $\mathcal{D}_{\mathbf{xy}}$ contains 10,000 problems with their solutions for training, and $\mathcal{D}_{\mathbf{x}}$ contains the remaining 2,000 problems for evaluation.
- **ORM set $\mathcal{D}_{\mathbf{xy}_\pm}$** contains 200K pairs of pairwise comparison data from the PRM800K dataset for training the outcome reward model.
- **PRM-step set $\mathcal{D}_{\mathbf{xyl}}$** contains 200K step-level correctness annotations from PRM800K for training the process reward model via step-wise binary classification.
- **PRM-alt set $\mathcal{D}_{\mathbf{xy}^{\leq}_{\pm}}$** contains 200K annotated comparisons of alternative steps from PRM800K for training process reward model via alternative step comparison.

4.2 Evaluation Metrics

We use two primary metrics, Preference Accuracy and Exact Match Accuracy. Preference Accuracy is the proportion of correctly predicted instances within the test data, which assesses the performance of different reward models (RMs). The criteria for "correct prediction", however, varies according to the type of RM being evaluated:

- **ORM:** For a comparison pair $(\mathbf{x}, \mathbf{y}_+, \mathbf{y}_-)$, the prediction is correct if the model scores higher on the preferred solution over the non-preferred one: $R_\phi(\mathbf{x}, \mathbf{y}_+) > R_\phi(\mathbf{x}, \mathbf{y}_-)$.
- **PRM-step:** For a step annotation $(\mathbf{x}, \mathbf{y}^{\leq t}, l_t)$, the prediction is correct if the score aligns with the step label: $R_\phi(\mathbf{x}, \mathbf{y}^{\leq t})(l_t - 0.5) > 0$.
- **PRM-alt:** For comparison pair $(\mathbf{x}, \mathbf{y}_+^{\leq t}, \mathbf{y}_-^{\leq t})$, the prediction is correct if the model scores higher for the preferred steps over non-preferred ones: $R_\phi(\mathbf{x}, \mathbf{y}_+^{\leq t}) > R_\phi(\mathbf{x}, \mathbf{y}_-^{\leq t})$.

On the other hand, to evaluate the learned reasoning model p_θ, we test it on the test set of the MATH dataset using the Exact Match Accuracy, which compares the normalized predicted answer with the correct answer (Hendrycks et al., 2021).

4.3 Model and Experimental Setups

In our experiments, we utilized the Llama2-70B model (Touvron et al., 2023) as both the initial agent p_θ and the reward models. For reward models, their final linear prediction layer randomly initialized prior to training. The agent model was initially fine-tuned on the SFT dataset $\mathcal{D}_{\mathbf{xy}}$ to establish our SFT baseline model. Subsequently, separate reward models were trained on the ORM, PRM-step, and PRM-alt training datasets to derive the respective reward models. These models were then utilized to further tune the SFT baseline model on the RLHF dataset $\mathcal{D}_{\mathbf{x}}$ for evaluation.

In RL training, we use the outcome-supervised RL objective (Eq. 2) for an ORM, and the process-supervised RL objective (Eq. 3) for a PRM.

4.4 Results

Reward Models Tab. 2 compares Preference Accuracy between three reward models. Despite the different training conditions and objectives of these models, making a direct comparison provides insights into the training complexity of reward models. PRMs are generally easier to train than ORMs, as they assess individual steps rather than the entire process, impacting their training complexity.

Reward Model	Preference Accuracy
ORM	61.0
PRM-step (ours)	73.5
PRM-alt (ours)	79.4

Table 2: Preference Accuracies (%) of reward models.

Model	EM
SFT	20.4
SFT + OutcomeRLHF w/ ORM	22.4
SFT + ProcessRLHF w/ PRM-step	23.6
SFT + ProcessRLHF w/ PRM-alt	23.8

Table 3: Exact Match (%) across models, comparing SFT, OutcomeRLHF, and our proposed ProcessRLHF.

Reasoning Models Tab. 3 lists the Exact Match (EM) Accuracy of various models. Despite the lower Preference Accuracy of the ORM compared to the PRMs, as noted in the previous section, integrating ORM with outcome-supervised RLHF still enhances the SFT baseline. Moreover, our process-supervised RLHF approach, when combined with PRMs, achieves an even greater improvement over the SFT baseline. Interestingly, both types of PRMs (step-wise supervision, "PRM-step", and alternative step comparison, "PRM-alt") demonstrate similar levels of enhancement over SFT.

5 Related Work

Recent advancements in large language models (LLMs) have made significant strides in multi-step reasoning tasks. The recent "chain of thought" prompting has transformed LLMs from simple question-and-answer systems to sophisticated reasoning tools capable of complex problem-solving (Nye et al., 2021; Wei et al., 2022). This approach has been further enhanced by techniques that encourage structured responses through zero-shot reasoning (Kojima et al., 2022) and by integrating self-consistency mechanisms for better performance across reasoning benchmarks (Wang et al., 2023). Additionally, specialized finetuning has improved LLMs' capabilities in mathematical reasoning (Lewkowycz et al., 2022), highlighting their growing sophistication and versatility.

6 Conclusions and Future Work

We generalize RLHF to integrate stepwise reward signals during training. This ProcessRLHF approach showed improved accuracy over traditional OutcomeRLHF on MATH and PRM800K datasets, indicating its potential in complex reasoning tasks. In the future, we plan to test this method across various datasets to confirm and possibly boost its effectiveness in diverse domains.

7 Limitations

Requirement for Stepwise Annotation. Unlike traditional RLHF, which requires only a single annotation per sample, our approach requires detailed step-level annotations for each sample. This could potentially increase both the cost and time required to obtain sufficient supervisory data, making the process more resource-intensive compared to traditional methods.

Comparison Fairness. Ensuring fairness in comparison presents a challenge, particularly in controlling the extent of supervision each method receives. There's no straightforward metric to measure the exact amount of supervision absorbed by each approach (i.e., ORM, PRM-step, PRM-alt). In our experiments, we controlled the number of annotations across all methods to be roughly the same. However, developing a method to accurately quantify the supervision each approach benefits from remains an open area for further research.

Evaluation Thoroughness. For an ideal assessment, our methods should be tested across a broader range of models and datasets. Unfortunately, due to computational resource constraints, our evaluation was limited to the specific settings outlined in this paper. Expanding the scope of our testing to include additional models and datasets would significantly enhance the robustness of our findings.

Ethics Statement

We believe that the work presented in this study does not exacerbate existing biases within the datasets and pretrained large language models utilized. Consequently, we anticipate no ethical issues arising from our research.

References

Florian Böhm, Yang Gao, Christian M. Meyer, Ori Shapira, Ido Dagan, and Iryna Gurevych. 2019. Better rewards yield better summaries: Learning to summarise without references. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3108–3118. Association for Computational Linguistics.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. 2018. Reward learning from human preferences and demonstrations in atari. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8022–8034.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *NeurIPS*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *CoRR*, abs/2305.20050.

Maxwell I. Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2021. Show your work: Scratchpads for intermediate computation with language models. *CoRR*, abs/2112.00114.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2269–2279. Association for Computational Linguistics.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize from human feedback. *CoRR*, abs/2009.01325.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Jonathan Uesato, Nate Kushman, Ramana Kumar, H. Francis Song, Noah Y. Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *CoRR*, abs/2211.14275.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Zhewei Yao, Reza Yazdani Aminabadi, Olatunji Ruwase, Samyam Rajbhandari, Xiaoxia Wu, Ammar Ahmad Awan, Jeff Rasley, Minjia Zhang, Conglong Li, Connor Holmes, Zhongzhu Zhou, Michael Wyatt, Molly Smith, Lev Kurilenko, Heyang Qin, Masahiro Tanaka, Shuai Che, Shuaiwen Leon Song, and Yuxiong He. 2023. Deepspeed-chat: Easy, fast and affordable RLHF training of chatgpt-like models at all scales. *CoRR*, abs/2308.01320.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593.