Unlocking Token Rewards via Training-Free Reward Attribution

Sitong Wu[1] Haoru Tan[3,✉] Bin Xia[1] Xichen Zhang[2] Jingyao Li[1] Shaofeng Zhang[4]
Xiaojuan Qi[3] Bei Yu[1] Jiaya Jia[1,2,✉]

[1]The Chinese University of Hong Kong [2]The Hong Kong University of Science and Technology
[3]The University of Hong Kong [4]University of Science and Technology of China

Abstract

In this paper, we propose an extremely efficient, training-free method to extract token-level reward signals directly from an existing deep reward model. Our core idea is to attribute the overall process reward to individual tokens by estimating each token's influence. This influence is defined as the change in the final macroscopic reward (e.g., the process reward) when a token is replaced with a semantically null token. Naively calculating this influence is computationally infeasible, requiring N forward passes through the PRM for an N-token sequence. We overcome this bottleneck by proposing a highly efficient gradient-based estimator. Specifically, we use a first-order Taylor approximation, which simplifies the influence calculation to the inner product of the difference between the token embedding and the null token embedding, and the gradient of the reward with respect to the token embedding. This requires only a single forward and backward pass. The resulting token-level rewards enable standard RL algorithms to perform precise credit assignment without requiring additional reward model training. Experiments on challenging reasoning benchmarks demonstrate that our method substantially improves policy optimization efficiency and enhances the generalization of LLM reasoning capabilities. Our P2T outperforms the outcome reward by +4.9% on MathVista for Qwen2.5-VL-7B-Instruct, and +11.5% on AIME24 for Qwen2.5-Math-7B. The code is available at: https://github.com/JIA-Lab-research/P2T.

1. Introduction

Reinforcement Fine-Tuning (RFT) has become a cornerstone paradigm for advancing the reasoning capabilities of Large Models [9, 14, 17, 39, 45]. The two most prevalent forms of supervision signals are the Outcome Reward (based on the final answer's correctness) and the Process Reward. The latter typically requires training a dedicated Process Reward Model (PRM) to evaluate the quality of individual steps within a reasoning trajectory. However, both outcome and process rewards ultimately share the limitation of coarse-grained supervision. Even PRM step-level scores remain ambiguous across the hundreds or thousands of tokens in a single step.

In parallel, recent work attempts to decompose coarse rewards into fine-grained, token-level supervision. Existing methods fall into two categories: The first involves training a dedicated token-level reward model, often relying on weakly- or unsupervised methods [8, 36] to generate synthetic labels. However, this approach incurs substantial training costs while yielding reward models of uncertain reliability due to the high noise and low fidelity of the pseudo-labels. The second avenue employs cheap, heuristic proxy metrics, such as token entropy [1], as substitutes for true rewards. While computationally inexpensive, these proxies fundamentally lack semantic alignment with the actual token quality; they serve as mere heuristics only loosely correlated with a token's contribution to the final reward, rather than principled reward estimators. Consequently, current token-level supervision is caught in a dilemma: either incur heavy costs for uncertain models or compromise optimization fidelity with inaccurate proxy signals. Crucially, training a truly fine-grained, token-level reward model is infeasible due to the prohibitive expense of acquiring large-scale human annotations. A critical open question emerges: *How can we devise a method for token-level reward estimation that is simultaneously cost-effective, principled, and robust?*

In this paper, we propose a principled, efficient, and robust solution, termed process-to-token **(P2T)** reward attribution. P2T directly decomposes the reward from any differentiable coarse-grained reward model (like a PRM) into accurate token-level rewards. Our core idea is to attribute the influence of each token on the coarse-grained reward. Specifically, a token's influence is defined as the change in

✉ Corresponding author

the overall process reward if that token were replaced with a non-informative "null" token. However, a naive implementation of this approach would require N separate PRM inferences (one per token) for an N-token sequence, rendering it computationally expensive. To achieve efficient computation, we propose an approximation solver. This estimator allows us to obtain the token-level influence score for all tokens in the entire process simultaneously with only a single forward and backward pass. Conclusively, the contribution of any token to the whole process is approximated using a first-order Taylor expansion: the inner product between (a) the gradient of the process reward with respect to the token's embedding, and (b) the difference between the null token embedding and the token's original embedding. Subsequently, we normalize the calculated token influences and use them as weights to redistribute the coarse-grained process reward, yielding the final token-level reward for each token.

Our method offers three key advantages: (1) **Training-free**. It builds on the powerful PRM while eliminating the significant cost of training dedicated token reward models. (2) **High fidelity**. The resulting token rewards are attributed based on reliable PRM outputs, significantly outperforming proxy metrics in both interpretability and accuracy. (3) **Scalable efficiency**. The gradient-based approximation enables the efficiency of our attribution computation. Extensive experiments demonstrate that our P2T significantly improves policy optimization stability and final reasoning accuracy across various benchmarks, outperforming both proxy-based and trainable token reward baselines. For example, P2T outperforms the outcome reward by +4.9% and +5.6% on MathVista for Qwen2.5-VL-7B-Instruct and LLaVA-CoT-11B, respectively. For text-only LLMs, P2T still exhibits strong advantages of +11.5% on AIME24 over the outcome reward for Qwen2.5-Math-7B and +14.2% on MinervaMath for DeepSeek-R1-Distill-Qwen-1.5B. In addition, P2T can accelerate training, achieving convergence approximately 4$\times$ faster than the outcome reward counterpart. Our contributions are threefold:

- We propose a novel and principled algorithm to convert process-level rewards into token-level rewards in a completely training-free manner, effectively bridging the gap between coarse and fine-grained supervision for reinforcement learning.
- We propose a highly efficient approximation for token contribution attribution, reducing the complexity from $\mathcal{O}(N)$ to $\mathcal{O}(1)$, ensuring its practical application to large-scale RL training.
- We empirically demonstrate that using the derived token-level rewards for RL training leads to significant improvements in the reasoning performance of LLMs compared to both sparse rewarding methods and other fine-grained rewarding methods.

2. Related Work

Reinforcement fine-tuning (RFT) [43, 45] can substantially enhance the performance of large models [4, 9, 14, 39, 48], including their reasoning and generalization abilities [7]. The RFT paradigm generally relies on two key components. The first is the reinforcement learning algorithm, which uses these rewards to compute advantages and drive policy updates, progressively refining the model's parameters to maximize expected performance under the given reward criteria. The second is the reward function or model, responsible for assessing the quality and alignment of the model's outputs and providing the scalar feedback that guides learning. Next, we introduce the specific designs of the reward functions.

Outcome Reward. Initially, a series of works starting with DeepSeek-R1 [14] leveraged the simplest form of reward construction: a binary outcome reward that solely determines whether the model's final output is correct. This outcome reward paradigm has proven highly effective on numerous reasoning tasks, such as code and mathematical reasoning [9, 19, 39, 47, 57, 59]. Recent work [23, 38, 52, 66] involves using powerful Large Language Models (LLMs) as reward models. These LLMs are instructed to score the output of the model under training based on grammar, correctness relative to the truth, and preference. This score is then used as the reward. The advantage of this technique is its versatility, making it suitable for a wide range of unstructured tasks. Furthermore, several open-source reward models [30, 58], such as DeepSeek-GRM [30], are available and can be employed for reward modeling across a broad range of applications, including writing and mathematical proof. However, outcome rewards suffer from a severe limitation: reward sparsity. The vast majority of tokens and intermediate reasoning steps generated by the model do not receive customized reward signals. This often leads to training instability, where, for instance, incorrect early steps in an otherwise correct reasoning trajectory might still be inadvertently reinforced [49].

Process Reward. Due to the shortcomings of outcome rewards, research into process supervision [24, 26, 29, 31, 33, 37, 40, 44, 49, 51, 62], which provides a more fine-grained reward signal than the outcome alone, has gained significant attention. Training a Process Reward Model (PRM) requires extensive annotation of fine-grained data. OpenAI [27] utilized human labor to annotate a massive volume of step-level preference data specifically in the mathematics domain, which could then be used to train mathematics-focused PRMs. Math-Shepherd [51] proposed an automated pipeline for efficiently annotating step-wise reward data. Specifically, it generates multiple rollouts starting from an arbitrary existing step, calculates the accuracy of these rollouts, and uses this accuracy value as an effective

proxy for the reward signal of that particular step. Some open-source works [45], such as qwen2.5-math-PRM [65], also provide options that can serve as PRMs for more general scenarios. Some recent works [10, 28, 35, 53, 63] have also explored constructing PRMs for multimodal settings. VisualPRM [53], for instance, compiles 400K reward samples for vision-language PRM training and shows promising results in both multimodal and text-only reward tasks. Similarly, URSA [35] collects over one million vision-language reward samples to train an 8B PRM, substantially boosting the capabilities of math-oriented VLLMs. Although process rewards are more fine-grained than outcome rewards, they remain relatively coarse, as each step still encompasses many tokens [60].

Token Reward. Constructing token-level rewards and appropriately assigning them to individual tokens can significantly improve both training effectiveness and efficiency [5, 8, 12, 20, 21, 36, 56]. PRIME [8] proposes using the difference in log probabilities between the reward model and the reference model as the reward for each token. SPRO [12] assigns each token a reward based on the difference in log probabilities between the policy model and a reference model, without requiring any additional reward model. OREAL [36] trains a token-level reward model by reusing the policy network backbone to assign scalar scores to each token, ensuring that the average token score matches the trajectory's final binary reward, and uses these scores to reweight the policy loss during training. CAPO [56] converts step-level rewards into token-level rewards by assigning each token the reward of its corresponding step, using a pre-existing LLM to identify incorrect reasoning steps and applying a penalty to tokens within those steps, without requiring a dedicated token-level reward model. TVM [21] trains a token-level reward model, built on the LLM architecture with an added scalar head, to predict each token's probability of leading to a correct final answer, using sampled reasoning paths and minimizing the MSE between predicted scores and the true probabilities. T-SPMO [20] assigns each token a reward equal to its expected incremental contribution given the preceding prefix, computed as the difference between the average rewards of responses with the prefix and those with the prefix followed by the token, relying solely on binary outcome rewards without any additional model.

3. Method

In this section, we describe our approach to converting the more macroscopic rewards provided by a Reward Model into token-level rewards in an efficient, training-free manner. The key idea is token attribution, that is, the marginal effect of each token on the overall reward. Then we will show how to estimate this using first-order gradient information, thereby enabling reinforcement fine-tuning with token-level credit assignment.

3.1. Definition of Token Attribution Score

The output of the policy model is treated as a sequence of tokens $Y = (y_1, y_2, \ldots, y_N)$, where each token y_i has an associated embedding vector $\mathbf{e}_i \in \mathbb{R}^d$. These vectors form the full sequence embedding $E = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N]$. A differentiable reward model $R(\cdot)$ then computes a scalar reward $R(E) \in \mathbb{R}$ for the entire sequence. We define the **Token Attribution Score** $\mathcal{I}_i$ for a token y_i as the precise difference in the total reward when its embedding $\mathbf{e}_i$ is replaced by a dedicated *null token* embedding $e_\varnothing$, without changing the sequence length:

$$\mathcal{I}_i = R(E) - R(E_{i \leftarrow \varnothing}), \tag{1}$$

where $E_{i \leftarrow \varnothing}$ denote the modified sequence embedding where the i-th token's embedding $\mathbf{e}_i$ is substituted by $e_\varnothing$, $R(E)$ is the original reward, and $R(E_{i \leftarrow \varnothing})$ is the new reward after null-token replacement. This score provides a direct, localized measure of the marginal importance of token y_i to the final sequence reward. A positive score ($\mathcal{I} > 0$) indicates that y_i was a positive contributor to the sequence's high reward, as replacing it caused the reward to drop. Conversely, a negative score ($\mathcal{I} < 0$) suggests y_i was a detrimental contributor that lowered the overall reward, and a score near zero signifies that y_i has a negligible marginal effect.

Choice of NULL Tokens. We select the Padding Token (*e.g.*, [PAD]) from the model's existing vocabulary to serve as the dedicated null token, $e_\varnothing$, for calculating the Token Attribution Score $\mathcal{I}_i$. This choice is highly motivated by the padding token's inherent design within standard Transformer architectures: it is explicitly engineered to lack practical semantic meaning and is used solely to maintain fixed input lengths without contributing content, perfectly aligning with the requirement that $e_\varnothing$ represents a null input. Furthermore, in many implementations, attention mechanisms are configured to explicitly mask out the padding token, minimizing its interaction with other tokens in the sequence embedding E. By substituting the i-th token's informative embedding $\mathbf{e}_i$ with the padding token's embedding $e_\varnothing$, we leverage the model's built-in mechanism to treat that position as semantically inert, allowing the resulting attribution score, $\mathcal{I}_i = R(E) - R(E_{i \leftarrow \varnothing})$, to precisely quantify the reward loss incurred when the original token's contribution is replaced by a known non-contributory placeholder.

3.2. Efficient Approximation

While conceptually simple and direct, this substitution method is computationally intensive, requiring N forward passes through the reward model to compute all token

scores for a sequence of length N, which motivates the search for efficient gradient-based approximations.

We approximate it via a first-order Taylor expansion around $\mathbf{e}_i$:

$$\mathcal{I}_i^{\text{Grad}} \approx \nabla_{\mathbf{e}_i} R(E)^\top (\mathbf{e}_i - \mathbf{e}_\varnothing). \tag{2}$$

Here, $\nabla_{\mathbf{e}_i} R(E)$ is the gradient of the reward with respect to the embedding of token i. This approximation can be computed efficiently in a single backward pass, since gradients with respect to all token embeddings are obtained simultaneously. A positive value indicates that token y_i contributes positively to the reward, while a negative value suggests the opposite.

3.3. Implementation Details

Next, we will introduce some application details: how to calculate our P2T token reward in Sec.3.3.1, and how to integrate it with popular RL algorithms in Sec.3.3.2.

3.3.1. P2T Token Reward Calculation

The attribution score $\mathcal{I}_i$, derived from the aforementioned vanilla gradient-based approximation or the integrated gradient approximation, provides a fine-grained token influence score, but these scores are not directly used as rewards. Instead, we normalize the calculated token influences and use them as weights to redistribute the coarse-grained process reward R, yielding the final token-level reward R_i for each token y_i:

$$R_i^{\text{P2T}} = R + \omega \cdot R \cdot \frac{\exp(\mathcal{I}_i)}{\sum_{j=1}^{N} \exp(\mathcal{I}_j)} \tag{3}$$

This normalization and redistribution strategy guarantees that the sum of the token-level rewards exactly equals the original sequence reward. This also ensures that the reward R is fairly distributed according to the measured marginal contribution of each token, allowing the policy to accurately learn which specific actions (token generation) led to the high or low coarse-grained reward.

It is worth noting that we do not entirely discard the coarse-grained sequence Reward (R) in this framework. In the calculation of our final token-level reward R_i, we introduce the macro-level R as a baseline (the first term in the equation), and the estimated token-level influence scores, after normalization and exponentiation, are reweighted by a factor ω and added to this baseline to yield the final token reward R_i. The coarse-grained sequence reward R serves as a baseline, providing a stable, minimum reward signal to every token. This is crucial because the attribution scores $\mathcal{I}_i$, often derived from approximate gradient calculations, can be inherently noisy or imprecise. Introducing R as the baseline limits the disturbance that attribution errors can introduce to the final reward, preventing the policy from fluctuating too wildly or being dominated by a few noisy token rewards during the learning phase.

3.3.2. GRPO with P2T Token Reward

The core step in applying token-level rewards to standard reinforcement learning algorithms is correctly calculating the advantage function at the token level. Taking GRPO [45] as an example, it is designed to be value-function-free and estimates the advantage by comparing the reward of a generated response to the average reward of other responses in the same group.

$$\hat{A}_n = \frac{R_n^{\text{out}} - \text{mean}(\{R^{\text{out}}\})}{\text{std}(\{R^{\text{out}}\})}, \tag{4}$$

where R_n^{out} is the outcome reward of n-th response. For example, for math problems, it can be the correctness indicator of the final answer (1 for correct and -1 for wrong). When incorporating our P2T token-level rewards, the advantage will be

$$\widetilde{A}_{n,i} = \hat{A}_n + \alpha \cdot R_{n,i}^{\text{P2T}}, \tag{5}$$

where i is the token index, and $R_{n,i}^{\text{P2T}}$ denotes the P2T reward for i-th token in n-th response. α is a weight to balance the contribution of the original outcome-derived advantage and our token-level guidance. By default, we set $\alpha = 0.1$ for short CoT models (such as Qwen2.5 series) and $\alpha = 1.0$ for long CoT models (such as DeepSeek-R1-Distill models).

4. Experiments

4.1. Settings

Baseline Models. We conduct RL training on various multimodal and text-only models, encompassing a diverse range of models from the Qwen and LLaMA series with varying sizes (from 0.6B to 7B). For multimodal models, we utilize Qwen2.5-VL-7B-Instruct [4] for its popularity and strong performance. For text-only models, we consider both base models after pre-training (Qwen2.5-Math [58]), and instruction-tuned models from post-training with different reasoning paradigms: CoT-style model LLaMA3.2-Instruct [11]), LongCoT-style models (DeepSeek-R1-Distill-Qwen [14], DeepScaleR-1.5B-Preview [34]), and hybrid-style models (Qwen3 [59]).

Training Dataset. For multimodal LLMs, we employ a mixture of two data sources: VerMulti-65K [41] and Vision-R1-10K [18], ensuring a coverage of both general and math multimodal tasks. For text-only LLMs, the training data is specialized by different models and tasks. By default, we use the MATH [16] dataset for mathematical training and the LeetCodeDataset [55] for code generation tasks. For recent models, like DeepScaleR-1.5B-Preview [34] and Qwen3 [59], we utilize the more challenging DeepScaleR-40K [34] as training data accordingly. For the Eurus-2-7B-SFT [8] model, we adopt the Eurus-2-RL-Data [8] as its training corpus, following prior works [8, 12] for a fair comparison.

Table 1. Comparison of LLMs trained on math reasoning dataset using GRPO with outcome reward and our P2T token reward. The advantages of the P2T token reward over the outcome reward are highlighted in red. For Qwen3 model, the results are reported in the format: performance in "thinking mode" / "non-thinking mode".

Model	AIME24	AIME25	AMC23	MATH-500	MinervaMath	Avg
Qwen2.5-Math-7B	13.2	9.8	40.0	53.6	17.2	26.8
+ GRPO (outcome reward)	28.8	14.1	67.3	81.8	36.1	45.6
+ GRPO (P2T token reward)	**40.3** (+11.5)	**20.0** (+5.9)	**72.8** (+5.5)	**84.5** (+2.7)	**41.9** (+5.8)	**51.9** (+6.3)
LLaMA3.2-3B-Instruct	3.3	0.2	22.5	48.0	16.5	18.1
+ GRPO (outcome reward)	13.3	1.5	27.5	56.8	20.2	23.9
+ GRPO (P2T token reward)	**21.7** (+8.4)	**10.8** (+9.3)	**46.5** (+19.0)	**62.6** (+5.8)	**27.9** (+7.7)	**33.9** (+10.0)
DeepSeek-R1-Distill-Qwen-1.5B	28.9	22.8	62.9	83.9	26.5	45.0
+ GRPO (outcome reward)	30.0	23.6	67.5	84.3	27.2	46.5
+ GRPO (P2T token reward)	**37.8** (+7.8)	**28.6** (+5.0)	**76.5** (+9.0)	**88.2** (+3.9)	**41.4** (+14.2)	**54.5** (+8.0)
Qwen3-1.7B	46.7 / 13.4	35.4 / 8.8	85.3 / 50.5	91.9 / 73.0	49.2 / 32.6	61.7 / 35.7
+ GRPO (outcome reward)	53.7 / 23.5	42.5 / 19.5	88.7 / 63.1	92.8 / 83.5	50.7 / 40.6	65.7 / 46.0
+ GRPO (P2T token reward)	**59.0 / 31.9** (+5.3) / (+8.4)	**47.0 / 30.1** (+4.5) / (+10.6)	**90.9 / 70.4** (+2.2) / (+7.3)	**93.4 / 86.9** (+0.6) / (+3.4)	**52.0 / 44.2** (+1.3) / (+3.6)	**68.5 / 52.7** (+2.8) / (+6.7)

Table 2. Comparison of different dense rewards for RL. All the experiments use the Eurus-2-7B-SFT [8] as the baseline model, which is finetuned from Qwen2.5-Math-7B-Base on math and code datasets. The training dataset for RL is Eurus-2-RL-Data [8].

Model	AMC	MATH	OlympiadBench	K12	CodeForces	CodeContests	Avg
Eurus-2-7B-SFT	21.1	48.1	13.9	40.0	5.2	11.3	22.8
+ GRPO (outcome reward)	23.6	51.8	20.5	48.6	28.6	28.1	33.5
+ PRIME	31.2	52.7	25.4	54.6	26.4	25.9	36.0
+ SPRO	31.9	53.6	28.2	55.0	29.4	32.1	38.4
+ GRPO (P2T token reward)	**33.8** (+1.9)	**55.4** (+1.8)	**30.6** (+2.4)	**56.1** (+1.1)	**32.3** (+2.9)	**35.8** (+3.7)	**40.7** (+2.3)

Evaluation. All the multimodal LLMs are evaluated on three multimodal math benchmarks (MathVista [32], MathVerse [64], and MathVision [50]) and two general multimodal benchmarks (MMMU [61] and MMStar [6]). For text-only LLMs trained on math data, the evaluation is conducted on seven commonly-used challenging math benchmarks, including AIME24 [2], AIME25 [2], AMC23 [3], MATH-500 [16], MinervaMath [22], OlympiadBench [15], and GaoKao2023en [13]. Following [8, 12], for the experiments on Eurus-2-7B-SFT [8] model trained on a combination of math and code data, we perform evaluation on five benchmarks, including AMC23, MATH, OlympiadBench, CodeForces [42], and CodeContests [25], For each test data, we generate 8 responses with the temperature of 0.6 and topp of 0.95. The averaged `pass@1` accuracy is reported as metric.

Training Settings. The multimodal models are trained for 1 epoch with a global batch size of 128. The text-only models are trained for 500 steps with a global batch size of 256. We use a constant learning rate of 1×10^{-6} and weight decay of 0.01. For each training data, we generate 8 responses with temperature of 1.0 and topp of 1.0. The KL coefficient is set to 0.001.

Implementation Details in P2T. Our Process-to-Token (P2T) method relies on a Process Reward Model (PRM) to provide the process-level reward for each reasoning step. The choice of the PRM is critical and tailored to the domain. In our experiments, we use VisualPRM [54] to provide the process reward for multimodal models, due to the strength of VisualPRM in handling a diverse range of multimodal tasks, including both general multimodal understanding and domain-specific tasks such as multimodal mathematics reasoning. For text-only models, we obtain the process reward from ReasonFlux-PRM [67] as it offers the crucial advantage of supporting and evaluating LongCoT reasoning trajectories, which is not well-supported in earlier PRMs. For the hyperparameter ω in Eq. (3), we set $\omega = 0.6$ by default. For α in Eq. (5), we set $\alpha = 0.1$ for short CoT models and $\alpha = 1.0$ for long CoT models.

4.2. Main Results

4.2.1. Comparison with Outcome Reward RL

To verify the effectiveness of fine-grained supervision with our P2T token-level reward, we compare it against the standard outcome reward. For fair comparison, all experiments are trained with the same policy optimization algorithm, GRPO [46], except for the reward scheme.

Multimodal LLMs. As shown in Figure 1, our P2T token reward demonstrates robust and clear advantages in the complex multimodal tasks over the outcome reward counterpart. For Qwen2.5-VL-7B-Instruct model, P2T achieves an improvement of +6.9% on MathVista, while outcome re-

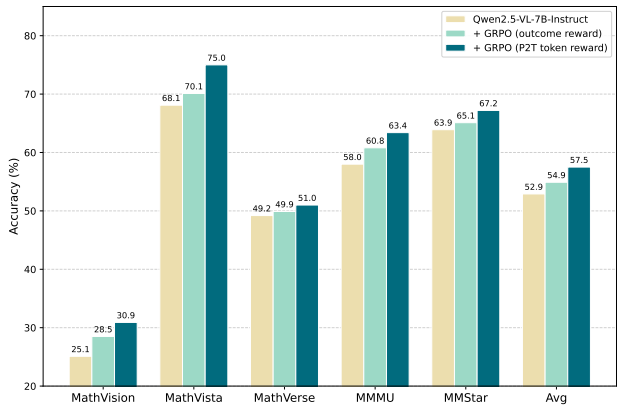

Figure 1. Comparison of MLLMs trained using GRPO with outcome reward and our P2T token reward.

ward only brings +2.0% gain. The key insight here lies in the difficulty of multimodal credit assignment. A final incorrect answer could stem from a visual perception error, a textual reasoning flaw, or a failure to correctly ground the text in the visual data. A coarse outcome reward struggles to distinguish these failure modes, providing a noisy signal. While, our P2T reward excels in this scenario by precisely identifying the source of failure. This strongly suggests that P2T is effectively identifying and correcting subtle errors in the visual-textual reasoning chain that the outcome reward fails to directly reveal. Crucially, P2T consistently outperforms outcome reward on every single benchmark for both models, demonstrating its robustness in multimodal settings.

Text-only LLMs. For base models without explicit instruction tuning (*e.g.*, Qwen2.5-Math-7B in Table 1), we first note that RL fine-tuning with both outcome reward and P2T token reward achieves significant performance gains over the pre-trained baseline (*e.g.*, more than +20 points for the average pass@1 accuracy of seven math benchmarks) However, our P2T token reward provides a notably larger improvement. On Qwen2.5-Math-7B, while outcome reward lifts the average score from 26.8% to 45.6%, our P2T reward pushes it further to 51.9% (with +6.3% gain compared to only outcome reward). This is a crucial finding: it demonstrates that fine-grained supervision is not a crutch only for smaller and weaker models. As models become more capable, their errors often become more subtle and deeply embedded in the reasoning process. A coarse outcome reward may fail to identify these nuanced mistakes, while our P2T method can precisely pinpoint and correct them. This suggests that the more powerful the model, the more it may benefit from the precise credit assignment that P2T provides.

CoT-instruct LLMs. CoT instruction-tuned models have already undergone extensive post-training and possess capabilities far exceeding the base models. We observe that the gains from outcome reward are often minimal on these already capable models. For example, in Table 1, our P2T achieves a +15.8% averaged accuracy improvement for LLaMA3.2-3B-Instruct model on math benchmarks, surpassing outcome reward by +10.0%.

This highlights a key insight: CoT-instruction models frequently produce a structural reasoning chain that is mostly correct but fails due to a single subtle error (*e.g.*, a miscalculation or a flawed logical step). A coarse outcome reward punishes the entire correct sequence for this one error, providing a sparse and noisy learning signal. Our P2T reward, however, excels in this scenario. It performs surgical credit assignment, identifying the specific flawed tokens responsible for the failure and penalizing them, while simultaneously reinforcing the correct portions of the reasoning path. This ability to precisely correct nuanced errors is why P2T continues to deliver strong improvements even on highly optimized models where outcome-based RL has hit a performance ceiling.

LongCoT-instruct LLMs. LongCoT-instruct LLMs conduct a deeper, deliberative thinking process before generating the final solution, thus their reasoning trajectory is characterized by a long-form pattern. This poses a more serious challenge for outcome-based RL. When a reward is only provided at the end of a very long sequence, the model cannot effectively learn which steps in the long chain were erroneous. As shown in Table 1, the outcome reward struggles to improve DeepSeek-R1-Distill-Qwen-1.5B, achieving a minimal average gain of only +1.5% (45.0% vs. 46.5%). In contrast, our P2T reward, which provides precise, fine-grained supervision at every token, boosts the average accuracy from 45.0% to 54.5%, which is +8.0% better than the outcome reward. As the reasoning chain length increases, the utility of outcome reward diminishes, while the necessity and effectiveness of our P2T token reward become even more pronounced. Notably, P2T's advantage is not uniformly distributed on each benchmark; instead, it is more significant on difficult benchmarks. For example, on MinervaMath, the outcome reward saw a marginal gain of just +0.7%, while our P2T reward achieves a substantial +14.9% improvement. This verifies P2T's ability to identify and fix specific, deep-seated flaws in the long-form reasoning process that a coarse-grained reward signal completely overlooks.

Hybrid-mode LLMs. Unlike the CoT or LongCoT instruct models, hybrid-mode models are designed to operate in two distinct paradigms based on the input prompt: a "thinking mode" which generates an explicit Chain-of-Thought (CoT) reasoning path, and a "no-thinking mode" which provides a direct CoT trajectory. Consequently, we report performance for both modes in Table 1 for the Qwen3 series, which is the representative open-source LLM with hybrid thinking

Table 3. Ablation Study for key designs in our P2T reward calculation. The experimental setup is identical to the Qwen2.5-Math-7B, except for (e), which follows the Qwen2.5-VL-7B-Instruct setup.

(a)

Null Token	Pad Token	EOS Token	Zero Embedding	Mean of all tokens in vocab
Pass@1	53.8	50.3	51.4	52.5

(b)

ω **in Eq.(5)**	0.25	0.5	0.75	1.0
Pass@1	53.5	53.8	53.4	53.2

(c)

m **in Eq.(4)**	2	3	4	5
Pass@1	5	53.8	53.4	53.0

(d)

PRM (text-only)	ReasonFlux PRM-7B	Qwen2.5-Math PRM-7B	Skywork PRM-7B
Pass@1	53.8	52.4	50.0

(e)

PRM (multimodal)	Visual PRM	MM PRM
Pass@1	57.5	55.3

(f)

Reward	Outcome	+ process-reward from PRM	+ token-reward from P2T
Pass@1	48.0	50.6	53.8

(g)

Approx Method	Vanilla in Eq.(2)	Integrated in Eq.(4)
Pass@1	52.6	53.8

mode. The results show that our P2T token reward provides consistent and significant improvements in both modes. For instance, on the Qwen3-1.7B model, P2T achieves a +2.2% average gain in "thinking mode" and an even larger +5.6% average gain in "no-thinking mode" compared to the outcome reward. This demonstrates the versatility of our P2T token reward.

4.2.2. Comparison with Existing Token-level Reward

We compare our P2T token reward with other existing token-level reward mechanisms, including: (1) PRIME [8] that introduces an auxiliary network that is co-optimized during RL training to provide token-level credit; (2) SPRO [12] that calculates a token-level reward by using the log-probability difference between the policy model and reference model.

For a fair comparison, all methods are applied to the same Eurus-2-7B-SFT model, using the Eurus-2-RL-Data for training, following the setup in [8]. As shown in Table 2, our P2T approach achieves the best performance among these dense reward methods, surpassing the SPRO by +2.3% and PRIME by an even larger margin of +4.7%. Compared with PRIME, our P2T is training-free during the RL stage, which does not require co-optimizing an auxiliary network, making the fine-tuning process more stable, efficient, and straightforward. The advantage over SPRO lies in the source and interpretability of the reward signal. SPRO uses a heuristic proxy statistic as the token-level reward, while somehow reasonable, but does not explicitly evaluate the correctness of the reasoning step itself. In contrast, our P2T reward is derived from a well-trained Process Reward Model (PRM) that was explicitly optimized to judge the quality of reasoning steps, enabling a more reasonable and interpretable credit attribution.

4.2.3. Breaking the Bottleneck

Interestingly, we discover that P2T can serve as a secondary optimization step to unlock further performance. As shown in Figure 2, models that are fully converged using outcome reward exhibited a significant performance boost after being fine-tuned with our P2T reward on the same training data, for example, +12.1% on MinervaMath and +6.8% on AIME24 for DeepScaleR-1.5B-Preview.

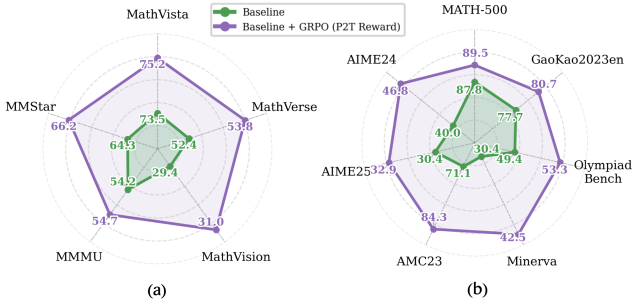

Figure 2. Effect of breaking the outcome-based RL convergence bottleneck. We further finetune the (a) Vision-R1-7B [18] and (b) DeepScaleR-1.5B-Preview [34] model using GRPO with our P2T reward. Note that we use the same training dataset used in their outcome-based RL process (*i.e.*, Vision-R1-10K and DeepScaleR-40K, respectively).

4.2.4. Training Efficiency

Our P2T can boost the convergence speed by approximately 4× compared with the outcome reward under RL with GRPO algorithm.

4.3. Ablation Study

We ablate the key design of our method, including the effect of hyperparameter ω, the choice of different PRMs, and the independent effect of PRM score and our token attribution credit assignment strategy. Full results are shown in Table 3.

5. Conclusion

In this paper, we present a highly efficient, training-free method to extract token-level reward signals from existing deep reward models. By defining a token's influence as its impact on the final process reward, we use a gradient-based, first-order Taylor approximation to compute token-level rewards with just a single forward and backward pass. This enables precise credit assignment for standard RL algorithms without additional reward model training. Experiments on challenging reasoning benchmarks show that our method significantly improves policy optimization efficiency and LLM generalization.

6. Acknowledgements

This work was supported in part by the Research Grants Council under the Areas of Excellence scheme grant AoE/E-601/22-R. The work has been supported by Hong Kong Research Grant Council-General Research Fund Scheme (Grant No. 17202422, 17212923, 17215025), Theme-based Research (Grant No. T45-701/22-R), and Strategic Topics Grant (Grant No. STG3/E-605/25-N). Part of the described research work is conducted in the JC STEM Lab of Robotics for Soft Materials funded by The Hong Kong Jockey Club Charities Trust.

References

[1] Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv preprint arXiv:2505.15134*, 2025. 1

[2] AIME. American invitational mathematics examination, 2024. 5

[3] AMC. American mathematics competitions, 2023. 5

[4] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 2, 4

[5] Meng Cao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Scar: Shapley credit assignment for more efficient rlhf. *arXiv preprint arXiv:2505.20417*, 2025. 3

[6] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087, 2024. 5

[7] Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025. 2

[8] Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025. 1, 3, 4, 5, 7

[9] Google DeepMind. Gemini 2.5 flash, 2025. 1, 2

[10] Lingxiao Du, Fanqing Meng, Zongkai Liu, Zhixiang Zhou, Ping Luo, Qiaosheng Zhang, and Wenqi Shao. Mm-prm: Enhancing multimodal mathematical reasoning with scalable step-level supervision, 2025. 3

[11] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024. 4

[12] Wu Fei, Hao Kong, Shuxian Liang, Yang Lin, Yibo Yang, Jing Tang, Lei Chen, and Xiansheng Hua. Self-guided process reward optimization with redefined step-wise advantage for process reinforcement learning. *arXiv e-prints*, pages arXiv–2507, 2025. 3, 4, 5, 7

[13] Chinese GaoKao. Gaokao2023en, 2024. 5

[14] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 1, 2, 4

[15] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024. 5

[16] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021. 4, 5

[17] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025. 1

[18] Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025. 4, 7

[19] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024. 2

[20] Alan Lee and Harry Tong. Token-efficient rl for llm reasoning. *arXiv preprint arXiv:2504.20834*, 2025. 3

[21] Jung Hyun Lee, June Yong Yang, Byeongho Heo, Dongyoon Han, Kyungsu Kim, Eunho Yang, and Kang Min Yoo. Token-supervised value models for enhancing mathematical problem-solving capabilities of large language models. *arXiv preprint arXiv:2407.12863*, 2024. 3

[22] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35: 3843–3857, 2022. 5

[23] Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, Hai Zhao, and Pengfei Liu. Generative judge for evaluating alignment, 2023. 2

[24] Wendi Li and Yixuan Li. Process reward model with q-value rankings, 2025. 2

[25] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624): 1092–1097, 2022. 5

[26] Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. Making large language models better reasoners with step-aware verifier, 2023. 2

[27] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. 2

[28] Wei Liu, Junlong Li, Xiwen Zhang, Fan Zhou, Yu Cheng, and Junxian He. Diving into self-evolving training for multimodal reasoning, 2025. 3

[29] Yuliang Liu, Junjie Lu, Zhaoling Chen, Chaofeng Qu, Jason Klein Liu, Chonghan Liu, Zefan Cai, Yunhui Xia, Li Zhao, Jiang Bian, Chuheng Zhang, Wei Shen, and Zhouhan Lin. Adaptivestep: Automatically dividing reasoning step through model confidence, 2025. 2

[30] Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. Inference-time scaling for generalist reward modeling, 2025. 2

[31] Jianqiao Lu, Zhiyang Dou, Hongru Wang, Zeyu Cao, Jianbo Dai, Yingjia Wan, and Zhijiang Guo. Autopsv: Automated process-supervised verifier, 2024. 2

[32] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023. 5

[33] Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision, 2024. 2

[34] Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025. Notion Blog. 4, 7

[35] Ruilin Luo, Zhuofan Zheng, Yifan Wang, Xinzhe Ni, Zicheng Lin, Songtao Jiang, Yiyao Yu, Chufan Shi, Lei Wang, Ruihang Chu, Jin Zeng, and Yujiu Yang. Unlocking multimodal mathematical reasoning via process reward model, 2025. 3

[36] Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, Haian Huang, et al. Exploring the limit of outcome reward for learning mathematical reasoning. *arXiv preprint arXiv:2502.06781*, 2025. 1, 3

[37] Qianli Ma, Haotian Zhou, Tingkai Liu, Jianbo Yuan, Pengfei Liu, Yang You, and Hongxia Yang. Let's reward step by step: Step-level reward model as the navigators for reasoning, 2023. 2

[38] Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz, and Jan Leike. Llm critics help catch llm bugs, 2024. 2

[39] OpenAI. Learning to reason with llms, 2024. 1, 2

[40] Sarah Pan, Vladislav Lialin, Sherin Muckatira, and Anna Rumshisky. Let's reinforce step by step, 2023. 2

[41] Yingzhe Peng, Gongrui Zhang, Miaosen Zhang, Zhiyuan You, Jie Liu, Qipeng Zhu, Kai Yang, Xingzhong Xu, Xin Geng, and Xu Yang. Lmm-r1: Empowering 3b lmms with strong reasoning abilities through two-stage rule-based rl. *arXiv preprint arXiv:2503.07536*, 2025. 4

[42] Shanghaoran Quan, Jiaxi Yang, Bowen Yu, Bo Zheng, Dayiheng Liu, An Yang, Xuancheng Ren, Bofei Gao, Yibo Miao, Yunlong Feng, et al. Codeelo: Benchmarking competition-level code generation of llms with human-comparable elo ratings. *arXiv preprint arXiv:2501.01257*, 2025. 5

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[44] Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning, 2024. 2

[45] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 1, 2, 3, 4

[46] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 5

[47] Open-R1 Team. Openr1-math-220k, 2025. 2

[48] Qwen Team. Qwen3-vl: Sharper vision, deeper thought, broader action, 2025. 2

[49] Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback, 2022. 2

[50] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024. 5

[51] Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024. 2

[52] Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. Self-taught evaluators, 2024. 2

[53] Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, Lewei Lu, Haodong Duan, Yu Qiao, Jifeng Dai, and Wenhai Wang. Visualprm: An effective process reward model for multimodal reasoning, 2025. 3

[54] Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, et al. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*, 2025. 5

[55] Yunhui Xia, Wei Shen, Yan Wang, Jason Klein Liu, Huifeng Sun, Siyue Wu, Jian Hu, and Xiaolong Xu. Leetcodedataset:

A temporal dataset for robust evaluation and efficient training of code llms. *arXiv preprint arXiv:2504.14655*, 2025. 4

[56] Guofu Xie, Yunsheng Shi, Hongtao Tian, Ting Yao, and Xiao Zhang. Capo: Towards enhancing llm reasoning through verifiable generative credit assignment. *arXiv e-prints*, pages arXiv–2508, 2025. 3

[57] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. 2

[58] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024. 2, 4

[59] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 2, 4

[60] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. 3

[61] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024. 5

[62] Thomas Zeng, Shuibai Zhang, Shutong Wu, Christian Classen, Daewon Chae, Ethan Ewer, Minjae Lee, Heeju Kim, Wonjun Kang, Jackson Kunde, Ying Fan, Jungtaek Kim, Hyung Il Koo, Kannan Ramchandran, Dimitris Papailiopoulos, and Kangwook Lee. Versaprm: Multi-domain process reward model via synthetic reasoning data, 2025. 2

[63] Jianghangfan Zhang, Yibo Yan, Kening Zheng, Xin Zou, Song Dai, and Xuming Hu. Gm-prm: A generative multimodal process reward model for multimodal mathematical reasoning, 2025. 3

[64] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024. 5

[65] Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025. 3

[66] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. 2

[67] Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. Reasonflux-prm: Trajectory-aware prms for long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2506.18896*, 2025. 5