Beyond Accuracy: A Policy Gradient Reweighting Approach for Pass@K Maximization in LLMs

Sadegh Mahdavi [1] **Muchen Li** [1] **Kaiwen Liu** [1] **Renjie Liao** [1] **Christos Thrampoulidis** [1]

Abstract

Recent works have demonstrated that reinforcement learning (RL) can substantially improve large language models (LLMs) for mathematical reasoning. However, most RL fine-tuning strategies optimize for single-sample accuracy (Pass@1), despite many practical applications relying on multi-sample inference (Pass@K). In this paper, we derive a principled RL objective that directly maximizes the expected Pass@K metric. Our approach formulates Pass@K maximization as a policy gradient objective, where harder examples (i.e., those with lower probability of success) are emphasized more during training. We connect our objective to Focal Loss from supervised learning and demonstrate its effectiveness across both Rejection-Fine-Tuning and GRPO algorithms. Experiments on mathematical benchmarks and synthetic arithmetic benchmarks show improvements in Pass@K over standard RL baselines. Our method provides a simple yet effective way to better align RL fine-tuning with the practical usage of LLMs.

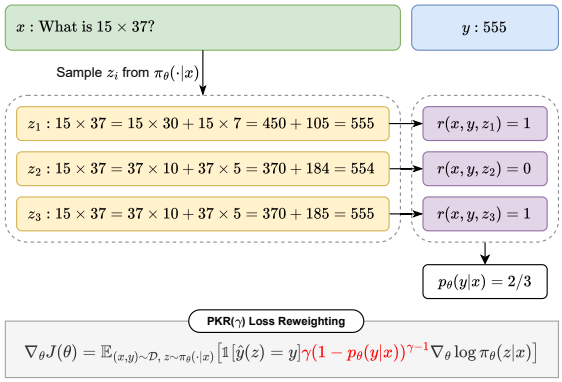


Figure 1. Overview of our proposed reweighting method. We add a reweighting factor to the reward function of the RL algorithm, which is based on the model's estimated probability of generating the correct answer. The larger the γ is, the more the model is encouraged to focus on higher K values for Pass@K maximization. We show that this reweighting works well with both RFT and GRPO algorithms.

1. Introduction

Reinforcement Learning (RL) has been shown to be effective for improving the performance of large language models (LLMs) on various tasks, including mathematical reasoning (Guo et al., 2025). However, most existing RL methods focus on maximizing the accuracy of the model on the training set, which may not necessarily lead to improved performance on harder problems.

In this work, we focus on the Pass@K metric (Chen et al., 2021), quantifying the ability of a model to generate correct answers for a given input by sampling K times from the model, and checking for a correct answer among those samples. This metric is particularly relevant for reasoning tasks, where several attempts may be needed to arrive at the correct answer, or where we have access to a reward model (Lightman et al., 2023; Cobbe et al., 2021; Wang et al., 2024) that can provide feedback on the correctness of the generated answer.

Despite the success of standard RL objectives in improving single-sample accuracy (Pass@1), they do not explicitly account for the multi-sample nature of Pass@K, and thus may underperform on harder instances where diversity across samples is crucial. In particular, when $p_\theta(y \mid x)$ (the probability that a single draw yields the correct answer) is low, maximizing expected reward under common RL reward functions does not sufficiently encourage exploration of diverse solution paths. To address this gap, we derive a closed-form expression for the Pass@K objective and show that its gradient corresponds to reweighting each correct sample by a factor proportional to $K\,(1 - p_\theta(y \mid x))^{K-1}$. Building on this insight, we introduce Pass@K-Reweighting (PKR), a simple yet theoretically grounded method that augments existing RL algorithms (RFT (Yuan et al., 2023),

[1]The University of British Columbia. Correspondence to: Sadegh Mahdavi <smahdavi@ece.ubc.ca>.

The second AI for MATH Workshop at the 42nd International Conference on Machine Learning, Vancouver, Canada.

GRPO (Shao et al., 2024)) with a tunable reweighting coefficient γ. This reweighting shifts emphasis toward harder examples (those with low $p_\theta(y \mid x)$) thereby directly optimizing for improved Pass@K performance and sample diversity.

Contributions

- **Theoretical Derivation of Pass@K Gradient.** We derive a closed-form expression for the Pass@K objective (Eq. (6)) and show that its policy gradient can be written as a reweighted version of the standard RL gradient, with weight $K(1 - p_\theta(y \mid x))^{K-1}$ on each correct sample.
- **Pass@K-Reweighting (PKR) Framework.** We propose PKR, which introduces a reweighting coefficient γ into existing RL reward functions (RFT, GRPO). Setting $\gamma = 1$ recovers the original algorithm, while $\gamma > 1$ focuses on harder examples to directly improve Pass@K.
- **Empirical Validation.** Through experiments on synthetic arithmetic and real math benchmarks, we demonstrate that PKR with $\gamma > 1$ yields consistent gains in Pass@K for $K > 1$, increases sample diversity, and imposes only a modest cost to Pass@1. We also study the effect of varying γ and show diminishing returns at very large values.

2. Related Work

Reinforcement Learning for Mathematical Reasoning. Earlier works explored the usage of model-generated solutions as training data as a bootstrapping mechanism to improve math reasoning capabilities of LLMs. These works generate solutions using a pre-trained model, then filter the solutions based on the correctness of their final answers, and use the filtered solutions to train a new model (Zelikman et al., 2022; Yuan et al., 2023; Singh et al., 2023; Anthony et al., 2017). Following the success of RLHF (Stiennon et al., 2020), several works have explored the use of RL for improving the performance of LLMs on mathematical reasoning tasks (Shao et al., 2024; Yang et al., 2024a). The main advantage of RL is that the algorithm is online (meaning that it uses the latest model's weights to generate the solutions), and it has been shown to be effective for improving the performance of LLMs on various tasks, including mathematical reasoning (Guo et al., 2025). Furthermore, RL takes into account the incorrect samples the model has generated, and it has been shown to be beneficial for mathematical reasoning tasks (Seed et al., 2025; Shao et al., 2024). In this work, we focus on online learning methods, which cover both the RFT (Yuan et al., 2023) and GRPO (Shao et al., 2024) algorithms.

Multi-Sample Inference. A recent body of works have focused on improving multi-sample inference (such as Pass@K and Majority@K) metrics (Beirami et al., 2024). Amini et al. (2024) derive a variational method to fine-tune an LLM to align it to a best-of-K distribution under a reward model. Their goal is to fine-tune the LLM to act as closely as possible to a scenario where K samples are taken from the LLM and the best sample is taken according to a reward model. Du et al. (2025) proposes a method to efficiently tune the temperature parameter of a language model to optimize the Majority@K metric.

Hardness-Aware Training. Recent works have found that upweighting harder examples during training can lead to improved performance on harder tasks. Toshniwal et al. (2024) and Tong et al. (2024) use synthetic data generation to fine-tune LLMs, and find out that harder examples (i.e., examples that the model has a low solve-rate on) should be upweighted during the synthetic sampling process. The Focal Loss Lin et al. (2017) proposes a class-reweighting mechanism in the context of supervised learning in computer vision to reweight the cross-entropy for each sample by $(1-p)^\gamma$ (where the p is the probability the model assigns to a class). Kimi-1.5 (Team et al., 2025) resamples RL training datapoints by $1 - p$ where p is an estimate of the model's probability of generating the correct answer. In all the above works, the reweighting mechanism is applied in an empirical way, without a theoretical justification. Here, we provide a theoretical justification for the reweighting mechanism, and show that it can be used to improve the Pass@K metric in RL training of LLMs.

While writing our manuscript, we became aware of contemporaneous work by Chow et al. (2025), which also proposes a reweighting method for inference-aware training of LLMs. Their method takes samples from a best-of-n distribution, while we take our samples from the distribution of the language model itself (see Appendix C for a detailed comparison). This allows us to use gradient signals from more sampled answers, and not just the best one, as well as a simpler implementation. Furthermore, our method allows for a more flexible reweighting parameter γ, which can be chosen to be any real-valued number greater than 1.

3. Notation and Background

Notation. We consider a dataset $\mathcal{D}$ of size n with $\mathcal{D} := \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\}$ with $x^{(i)}$ being inputs (e.g., math questions), and $y^{(i)}$ being answers (e.g., final answer to the question). Let π_θ be a language model parameterized by model weights θ. Concretely, let $\pi_\theta(\cdot|x)$ be the conditional probability distribution of the model's output sequence of tokens when prompted with x.

In a multi-sample inference setting, for each input-answer

pair (x, y), we sample $K \in \mathbb{N}$ sequences from $\pi_\theta(\cdot|x)$. Let us call the final answers of these samples $\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_K$. Then, we define the Pass@K metric for question-answer pair (x, y) as:

$$\text{Pass}_K(\pi_\theta; x, y) := \mathbb{1}\left[\exists k \in [K] : \hat{y}_k = y\right].$$

Similarly, we define Pass@K for the dataset $\mathcal{D}$:

$$\text{Pass}_K(\pi_\theta; \mathcal{D}) := \frac{1}{n}\sum_{i=1}^{n} \text{Pass}_K(\pi_\theta; x^{(i)}, y^{(i)}).$$

Throughout, we denote the full chain-of-thought response from an LLM by z. The final answer $\hat{y} := \hat{y}(z)$ can be read-off from the chain-of-thought response z. An example of each variable is shown in Figure 1. In that example, we have $\hat{y}(z_1) = 555$, which in this case, equals y.

Reinforcement Learning and Policy Gradient Methods. In the context of reinforcement learning, policy gradient methods are used to optimize the expected reward of a policy given a reward function $r(x, y, z)$ (Sutton & Barto, 2018). In our case, the policy is the LLM parameterized by θ. Furthermore, the reward function $r(x, y, z)$ is the reward given to the LLM for generating a response z conditioned on input x with correct answer y. Thus, the objective is to maximize the expected reward (Sutton & Barto, 2018):

$$J(\theta) := \mathbb{E}_{(x,y)\sim\mathcal{D}, z\in\pi_\theta(\cdot|x)}\left[r(x, y, z)\right]. \tag{1}$$

To optimize Eq (1), the REINFORCE (Williams, 1992) trick is often used to compute the gradient of the objective function with respect to the parameters θ:

$$\nabla_\theta J(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}, z\in\pi_\theta(\cdot|x)}\left[r(x, y, z)\nabla_\theta \log \pi_\theta(z|x)\right]. \tag{2}$$

Different reward functions correspond to different policy gradient methods (Lambert, 2024; Shao et al., 2024). Here, we focus on the following reward functions:

$$r_{\text{RFT}}(x, y, z) := \mathbb{1}\left[\hat{y}(z) = y\right],$$
$$r_{\text{REINFORCE}}(x, y, z) := \text{sign}\left[\hat{y}(z) = y\right],$$
$$r_{\text{GRPO}}(x, y, z) := \frac{\mathbb{1}\left[\hat{y}(z) = y\right] - \mu_x}{\sigma_x}.$$

Using r_{RFT} gives an online version of the Rejection FineTuning (RFT) algorithm (Yuan et al., 2023). Using $r_{\text{REINFORCE}}$ gives a variant of REINFORCE where correct samples receive positive reward and negative samples receive negative reward (Williams, 1992), and r_{GRPO} gives a simplified version of the Group Relative Policy Optimization (GRPO) reward function (Shao et al., 2024)[1]. The RFT reward function is a binary reward function that gives a reward of 1 if the answer is correct and 0 otherwise. The REINFORCE reward function is also similar to the RFT reward function, but it gives a positive reward for correct answers and a negative reward for incorrect answers (without any normalization). The GRPO reward function is similar, but is normalized by the mean μ_x and standard deviation σ_x of the rewards for the input x in the dataset $\mathcal{D}$. Specifically, μ_x and σ_x are computed by taking multiple samples from the LLM π_θ for each input x and computing the Monte-Carlo mean and standard deviation of the rewards for each input x. The main difference between RFT and GRPO is that RFT only considers the correct samples, while GRPO considers all samples and consists of both positive and negative gradients.

[1]Concretely, this formulation is equivalent to the original GRPO with the KL-penalty set to zero, and the number of gradient iterations (μ in (Shao et al., 2024)) set to one. We adopt this practical formulation since it has been shown that the KL penalty is not necessary in RL with verifiable rewards (Yu et al., 2025) and Shao et al. (2024) already uses a single gradient iteration.

4. Method

Our goal is to derive an objective function whose maximization leads to an improvement in the Pass@K metric. We approach this by framing the problem within a reinforcement learning context, aiming to maximize the expected reward associated with generating correct answers.

4.1. Pass@1 Maximization as RFT

First, let us consider the standard accuracy metric, which corresponds to Pass@1. Recall the RFT reward function $r_{\text{RFT}}(x, y, z) := \mathbb{1}[\hat{y}(z) = y]$, that yields a reward of 1 if the generated response z produces the correct final answer y for input x, and 0 otherwise. The Pass@1 objective is the same as maximizing the expected RFT reward:

$$J_1(\theta) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) = y]\right]\right]. \tag{3}$$

Using the REINFORCE trick (similar to Eq (2)), the gradient $\nabla_\theta J_1(\theta)$ of this objective with respect to the model parameters θ is:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{E}_{z\sim\pi_\theta(\cdot|x)}[\mathbb{1}[\hat{y}(z) = y]\nabla_\theta \log \pi_\theta(z|x)]\right]. \tag{4}$$

Note this corresponds exactly to the gradient used in supervised fine-tuning (SFT) on correctly generated responses or the online RFT objective r_{RFT}: it encourages the model to increase the probability of generating correct sequences z.

4.2. Pass@K Maximization as Weighted RFT

Now, we extend this to the Pass@K metric. The Pass@K metric is 1 if at least one out of K samples $z_1, \ldots, z_K$ drawn independently from $\pi_\theta(\cdot|x)$ yields the correct answer y, and 0 otherwise. Our objective function $J_K(\theta)$ is the expected Pass@K score over the dataset $\mathcal{D}$:

$$J_K(\theta) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\text{Pass}_K(\pi_\theta; x, y)\right] \tag{5}$$
$$= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{E}_{z_1,\ldots,z_K\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}\left[\exists k \in [K] : \hat{y}(z_k) = y\right]\right]\right].$$

The inner expectation represents the probability that at least one of the K independent samples $z_1, \ldots, z_K$ drawn from

$\pi_\theta(\cdot|x)$ yields the correct answer y. Let E_k be the event that the k-th sample z_k is correct, i.e., $\hat{y}(z_k) = y$. Clearly,

$$\mathbb{P}\left(\bigcup_{k=1}^{K} E_k\right) = \mathbb{E}_{z_1,\ldots,z_K\sim\pi_\theta(\cdot|x)}\mathbb{1}\left[\exists k \in [K] : \hat{y}(z_k) = y\right].$$

It is easier to compute the probability of the complement event, where none of the samples are correct (i.e., all samples yield incorrect answers):

$$\mathbb{P}\left(\bigcap_{k=1}^{K} E_k^c\right) = \mathbb{P}_{z_1,\ldots,z_K\sim\pi_\theta(\cdot|x)}\left[\forall k \in [K] : \hat{y}(z_k) \neq y\right].$$

Let $p_\theta(y|x)$ denote the probability that a single sample z drawn from $\pi_\theta(\cdot|x)$ yields the correct answer y:

$$\begin{aligned} p_\theta(y|x) &:= \mathbb{P}_{z\sim\pi_\theta(\cdot|x)}[\hat{y}(z) = y] \\ &:= \mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) = y]\right]. \end{aligned}$$

Since the K samples $z_1, \ldots, z_K$ are drawn independently, the probability that all K samples are incorrect is the product of their individual probabilities of being incorrect:

$$\mathbb{P}\left(\bigcap_{k=1}^{K} E_k^c\right) = \prod_{k=1}^{K}\mathbb{P}(E_k^c) = (1 - p_\theta(y|x))^K.$$

Using the complement rule, the probability that at least one sample is correct is:

$$\mathbb{P}\left(\bigcup_{k=1}^{K} E_k\right) = 1 - \mathbb{P}\left(\bigcap_{k=1}^{K} E_k^c\right) = 1 - (1 - p_\theta(y|x))^K.$$

Substituting this back into the definition of $J_K(\theta)$ (Eq (5)), we arrive at the closed-form expression for the objective function in terms of the probability $p_\theta(y|x)$ that a single sample z yields the correct answer y:

$$J_K(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[1 - (1 - p_\theta(y|x))^K\right]. \tag{6}$$

To optimize this objective, we compute its gradient with respect to the model parameters θ. Using the linearity of expectation and the chain rule for differentiation:

$$\begin{aligned} &\nabla_\theta J_K(\theta) = \nabla_\theta\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[1 - (1 - p_\theta(y|x))^K\right] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\nabla_\theta\left(1 - (1 - p_\theta(y|x))^K\right)\right] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[-K\left(1 - p_\theta(y|x)\right)^{K-1}\cdot\nabla_\theta(1 - p_\theta(y|x))\right] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[-K\left(1 - p_\theta(y|x)\right)^{K-1}\cdot(-\nabla_\theta p_\theta(y|x))\right] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[K(1 - p_\theta(y|x))^{K-1}\nabla_\theta p_\theta(y|x)\right]. \end{aligned} \tag{7}$$

Recalling that $\nabla_\theta p_\theta(y|x) = \nabla_\theta\mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) = y]\right] = \mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) = y]\nabla_\theta\log\pi_\theta(z|x)\right]$ from the Pass@1 derivation (Eq (4)), and substituting into Eq (7), the gradient $\nabla_\theta J_K(\theta)$ evaluates to:

$$\mathbb{E}_{\substack{(x,y)\sim\mathcal{D}\\ z\sim\pi_\theta(\cdot|x)}}\left[\mathbb{1}[\hat{y}(z) = y]K(1 - p_\theta(y|x))^{K-1}\nabla_\theta\log\pi_\theta(z|x)\right]$$

This final expression gives the policy gradient for maximizing the Pass@K objective. Comparing this to the Pass@1 gradient (Eq (4)), we see that each correct sample z (where $\mathbb{1}[\hat{y}(z) = y] = 1$) is reweighted by a factor of $K\left(1 - p_\theta(y|x)\right)^{K-1}$. This weight depends on the current model's probability $p_\theta(y|x)$ of getting the answer correct for the specific input x. Intuitively, problems x where the model is less likely to succeed (small $p_\theta(y|x)$) receive a higher weight, encouraging the model to improve on harder instances.

In practice, $p_\theta(y|x)$ is unknown and needs to be estimated. We use Monte Carlo estimate based on M samples drawn from $\pi_\theta(\cdot|x)$:

$$\hat{p}_\theta(y|x) := \frac{1}{M}\sum_{m=1}^{M}\mathbb{1}[\hat{y}(z_m) = y]. \tag{8}$$

4.3. Final Objective

We now define our final objective function for maximizing Pass@K. For clarity, we use γ to denote the reward weight in our objective function, distinguishing it from K which represents the number of samples in the Pass@K metric. With this, we define our final policy gradient objective function for maximizing Pass@K as follows:

Definition 4.1. PKR(γ) Objective. The Pass@K-Reweighting (PKR) gradient objective with reweighting coefficient γ is defined as:

$$\begin{aligned} \nabla J_{\text{PKR}_\gamma}(\theta) := \mathbb{E}_{(x,y)\sim\mathcal{D},\, z\sim\pi_\theta(\cdot|x)}\big[&r(x,y,z) \\ &\times\gamma\left(1 - p_\theta(y|x)\right)^{\gamma-1}\nabla_\theta\log\pi_\theta(z|x)\big]. \end{aligned} \tag{9}$$

where $r(x, y, z)$ is the specific reward function used in the RL algorithm (e.g., RFT, GRPO).

Connection to Focal Loss. The Focal Loss in supervised learning (Lin et al., 2017) is designed to address class imbalance by down-weighting easy examples and focusing on hard examples. Specifically, it reweights the loss for each example as $(1 - p_\theta(y|x))^\gamma$, where γ is a focusing parameter and $p_\theta(y|x)$ is the model's estimated probability for the true label y given input x. The common point between Focal Loss and our proposed PKR objective is that both methods base their reweighting on the hardness of the example using a similar polynomial term. However, a key difference is that in supervised learning with Focal Loss, $p_\theta(y|x) = \pi_\theta(y|x)$ directly gives the model's probability

of the correct label, while in our reinforcement learning setting, $p_\theta(y|x) = \mathbb{E}_{z\sim\pi_\theta}[\mathbb{1}[\hat{y}(z) = y]]$ requires taking an expectation over sampled sequences.

5. Experiments

5.1. Models and Training

We use Qwen models (Yang et al., 2024b) as they are state-of-the-art open-weight models for math reasoning tasks. To isolate the effect of reinforcement learning, we take the base models, and do a light round of supervised fine-tuning (SFT) on 1000 datapoints from OpenMath-Instruct2 (Toshniwal et al., 2025). We then apply reinforcement learning with both RFT and GRPO reward functions with and without our proposed reweighting. In all settings, $\gamma = 1.0$ gives the original (GRPO or RFT) objective, and $\gamma > 1.0$ gives our proposed PKR objective.

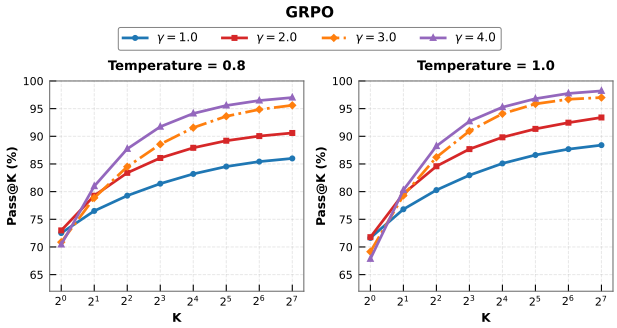
5.2. RL Datasets

Synthetic Arithmetic Dataset. We create a synthetic dataset of 5000 arithmetic problems, where each problem consists of a simple arithmetic expression of the form `What is` $a \times b + c \times d$?, where a, b, c, d are positive integers uniformly chosen from a pre-determined range, depending on the capability of the LLM. We sample 500 problems for validation and 500 for testing.

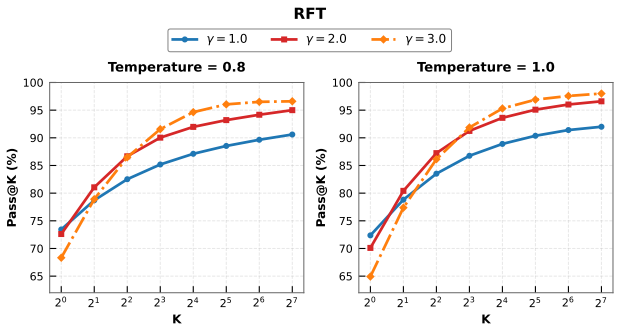
Math Dataset. We use the AoPS-Instruct dataset (Mahdavi et al., 2025) for our experiments with the math reasoning task. We take the problems that have parsable Sympy (Meurer et al., 2017) solutions and for each model, we filter out the problems that the model has a solve-rate of between 10% and 80% (inclusive) on the training set (Yang et al., 2024a). This filtering prunes the examples that are too easy or too hard (both of which hinder RL training). We use MATH dataset (Hendrycks et al., 2021) for evaluation. The MATH test dataset consists of 5,000 problems, of which 500 are designated as the MATH-500 test set (Lightman et al., 2023). We randomly sample 500 datapoints from the remaining 4,500 problems to use as a validation set for our experiments to pick the best checkpoint during the RL phase.

5.3. Results

Figure 2 reports results of our experiments on the arithmetic task with the Qwen 0.5B model. We observe several interesting trends in these results. For $K = 1$, the vanilla algorithm ($\gamma = 1$) performs best, which aligns with our theoretical understanding since higher γ values optimize for higher K's. With $\gamma = 2$, we see improvements already at $K = 2$ that continue to increase for larger K values. Similarly, $\gamma = 4$ shows even larger gains for higher K values despite worse



(a) GRPO Results



(b) RFT Results

Figure 2. Comparison of Pass@K metrics for different RL algorithms and temperatures for Qwen 0.5B on arithmetic task. Our proposed reweighting is $\gamma > 1$.

performance at $K = 1$. The curves for increasing γ values appear to saturate at progressively higher levels, though the performance gap between $\gamma = 3$ and $\gamma = 4$ narrows, suggesting diminishing returns for very large γ values.

Furthermore, we test our method on real math problems using the AoPS dataset with Qwen 3B model, where the trends are more nuanced but still show benefits from our reweighting approach. Figure 3 shows the results of our experiments on the AoPS dataset with Qwen 3B model. This confirms that our proposed PKR method slightly improves the Pass@K metric for $K > 1$ in GRPO algorithm across all temperatures.

5.4. Analysis

We observe that the pass@K metric improves with higher γ values. To understand this, we analyze the generated samples and count for each K generated samples, how many of their generated final answers are unique. Figure 4 shows the results of our analysis on the arithmetic task with Qwen 0.5B model. As shown in the figure, higher values of γ lead to more unique final answers generated at inference time. This is in-line with the intuition that higher γ values encourage the model to optimize pass@K metric by generating more diverse samples, which in turn leads to higher pass@K

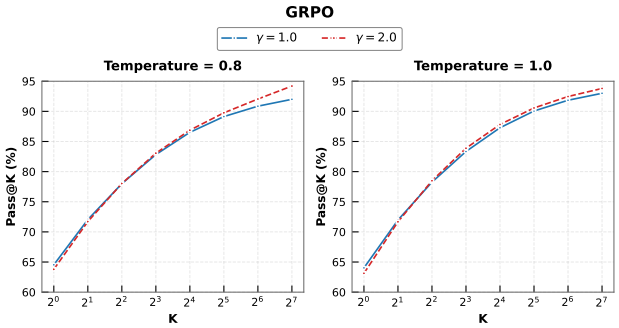


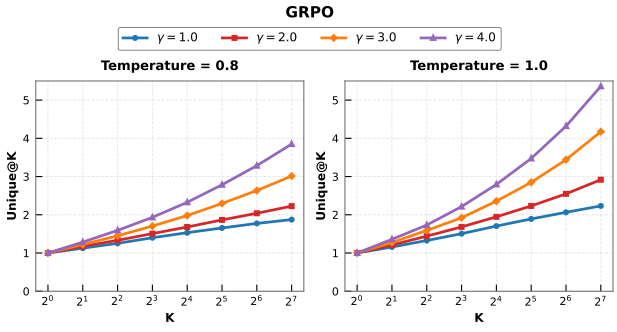
Figure 3. Comparison of Pass@K metrics for GRPO and temperatures for Qwen 3B on Math-500 Benchmark. Our proposed PKR is $\gamma > 1$.

scores.

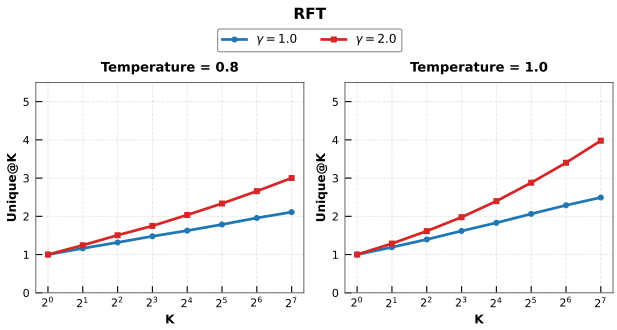
6. Conclusion and Future Work

In this work, we introduced Pass@K-Reweighting (PKR), a theoretically grounded method to directly optimize the Pass@K metric in reinforcement learning for large language models. By deriving a closed-form expression for Pass@K and computing its gradient, we showed that maximizing Pass@K corresponds to reweighting each correct sample by $K\,(1 - p_\theta(y|x))^{K-1}$. Our PKR objective, with the tunable parameter γ, recovers standard RL algorithms (RFT and GRPO) when $\gamma = 1$ and prioritizes harder instances when $\gamma > 1$. Empirically, we demonstrated on synthetic arithmetic and AoPS math datasets that $\gamma > 1$ yields improvements in Pass@K for $K > 1$, at a modest cost to Pass@1, and that higher γ also promotes greater sample diversity.

Future work includes testing and scaling up PKR on larger and more diverse reasoning benchmarks to assess its robustness and generality. Scheduling strategies, such as focusing on Pass@K optimization early in training before shifting emphasis to Pass@1, could help balance multi-sample performance with single-sample accuracy. In addition, exploring adaptive γ schedules and integration with richer reward functions may further improve model performance and sample diversity.



(a) GRPO Results



(b) RFT Results

Figure 4. Comparison of unique samples for different RL algorithms and temperatures for Qwen 0.5B on arithmetic task. Higher values of γ lead to more unique samples generated at inference time.

References

Amini, A., Vieira, T., Ash, E., and Cotterell, R. Variational best-of-n alignment. *arXiv preprint arXiv:2407.06057*, 2024.

Anthony, T., Tian, Z., and Barber, D. Thinking fast and slow with deep learning and tree search. *Advances in neural information processing systems*, 30, 2017.

Beirami, A., Agarwal, A., Berant, J., D'Amour, A., Eisenstein, J., Nagpal, C., and Suresh, A. T. Theoretical guarantees on the best-of-n alignment policy. *arXiv preprint arXiv:2401.01879*, 2024.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chow, Y., Tennenholtz, G., Gur, I., Zhuang, V., Dai, B., Kumar, A., Agarwal, R., Thiagarajan, S., Boutilier, C., and Faust, A. Inference-aware fine-tuning for best-of-n sampling in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=77gQUdQhE7.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Du, W., Yang, Y., and Welleck, S. Optimizing temperature for language models with multi-sample inference. *arXiv preprint arXiv:2502.05234*, 2025.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Lambert, N. *Reinforcement Learning from Human Feedback*. Online, 2024. URL https://rlhfbook.com.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Mahdavi, S., Li, M., Liu, K., Thrampoulidis, C., Sigal, L., and Liao, R. Leveraging online olympiad-level math problems for llms training and contamination-resistant evaluation, 2025. URL https://arxiv.org/abs/2501.14275.

Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J. K., Singh, S., Rathnayake, T., Vig, S., Granger, B. E., Muller, R. P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., Curry, M. J., Terrel, A. R., Roučka, v., Saboo, A., Fernando, I., Kulal, S., Cimrman, R., and Scopatz, A. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, January 2017. ISSN 2376-5992. doi: 10.7717/peerj-cs.103. URL https://doi.org/10.7717/peerj-cs.103.

Seed, B., Yuan, Y., Yue, Y., Wang, M., Zuo, X., Chen, J., Yan, L., Xu, W., Zhang, C., Liu, X., et al. Seed-thinking-v1. 5: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*, 2025.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Singh, A., Co-Reyes, J. D., Agarwal, R., Anand, A., Patil, P., Garcia, X., Liu, P. J., Harrison, J., Lee, J., Xu, K., et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Team, K., Du, A., Gao, B., Xing, B., Jiang, C., Chen, C., Li, C., Xiao, C., Du, C., Liao, C., et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Tong, Y., Zhang, X., Wang, R., Wu, R., and He, J. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *Advances in Neural Information Processing Systems*, 37:7821–7846, 2024.

Toshniwal, S., Moshkov, I., Narenthiran, S., Gitman, D., Jia, F., and Gitman, I. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. *Advances in Neural Information Processing Systems*, 37:34737–34774, 2024.

Toshniwal, S., Du, W., Moshkov, I., Kisacanin, B., Ayrapetyan, A., and Gitman, I. Openmathinstruct-2: Accelerating AI for math with massive open-source instruction data. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mTCbq2QssD.

Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Yang, A., Zhang, B., Hui, B., Gao, B., Yu, B., Li, C., Liu, D., Tu, J., Zhou, J., Lin, J., et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024a.

Yang, Q. A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Dong, G., Wei,

H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y.-C., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., Qiu, Z., Quan, S., and Wang, Z. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024b. URL https://api.semanticscholar.org/CorpusID:274859421.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Fan, T., Liu, G., Liu, L., Liu, X., et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yuan, Z., Yuan, H., Li, C., Dong, G., Lu, K., Tan, C., Zhou, C., and Zhou, J. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*, 2023.

Zelikman, E., Wu, Y., Mu, J., and Goodman, N. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

A. Experiment Details

A.1. Synthetic Data Generation

The dataset is generated by sampling a, b, c, d uniformly from the range $[1, MaxN]$, where $MaxN$ is a hyperparameter that controls the maximum number in the arithmetic expression. We choose $MaxN$ to be a power of 10,, depending on the capability of the LLM. For the 0.5B model, we choose $MAxN = 1000$, and for 1.5B math model, we choose $MaxN = 10000$.

B. Extending the reweighting to negative gradients

In Section 4, we derived the reweighting factor $K(1 - p_\theta(y|x))^{K-1}$ by maximizing the Pass@K success probability $J_K(\theta) = \mathbb{E}[1 - (1 - p_\theta(y|x))^K]$. This objective leads to a gradient update that only involves correct samples (positive gradients). Here, we show that the same reweighting factor applies when considering an objective based on minimizing the probability of failure, which involves incorrect samples (negative gradients).

Let us define the objective function for minimizing the probability that *all* K samples are incorrect:

$$\bar{J}_K(\theta) := \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[(1 - p_\theta(y|x))^K\right]. \tag{10}$$

This represents the probability of failing the Pass@K check. Our goal is to minimize this objective. We compute the gradient with respect to θ:

$$\nabla_\theta \bar{J}_K(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[K(1 - p_\theta(y|x))^{K-1}\nabla_\theta(1 - p_\theta(y|x))\right]. \tag{11}$$

We know that

$$1 - p_\theta(y|x) = \mathbb{P}_{z\sim\pi_\theta(\cdot|x)}[\hat{y}(z) \neq y] = \mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) \neq y]\right].$$

Using the policy gradient theorem, the gradient of this term is:

$$\begin{aligned}\nabla_\theta(1 - p_\theta(y|x)) &= \nabla_\theta \mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) \neq y]\right] \\ &= \mathbb{E}_{z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) \neq y]\nabla_\theta \log \pi_\theta(z|x)\right].\end{aligned} \tag{12}$$

Substituting Eq (12) into Eq (11):

$$\nabla_\theta \bar{J}_K(\theta) = K\,\mathbb{E}_{(x,y)\sim\mathcal{D},\, z\sim\pi_\theta(\cdot|x)}\left[\mathbb{1}[\hat{y}(z) \neq y](1 - p_\theta(y|x))^{K-1} \times \nabla_\theta \log \pi_\theta(z|x)\right]. \tag{13}$$

Minimizing $\bar{J}_K(\theta)$ involves taking gradient descent steps, i.e., moving in the direction of $-\nabla_\theta \bar{J}_K(\theta)$. This gradient applies updates based on the *incorrect* samples z (where $\mathbb{1}[\hat{y}(z) \neq y] = 1$).

Crucially, comparing Eq (13) with the gradient for maximizing Pass@K success (Eq (7) in the main text), we observe that the reweighting factor $K(1 - p_\theta(y|x))^{K-1}$ is identical for both positive gradients (from correct samples) and negative gradients (from incorrect samples).

C. Comparison to Chow et al. (2025)

In this section, we compare our method to the approach proposed by Chow et al. (2025). In Corollary 4 of their paper, their method involves a reweighting factor of the form (in our notation):

$$\frac{K(1 - p_\theta(y|x))^{K-1} p_\theta(y|x)}{1 - (1 - p_\theta(y|x))^K}, \tag{14}$$

where each response z is sampled from a distribution $\pi_{\text{bon}}(z|x)$ defined as follows:

$$\pi_{\text{bon}}(z|x) = \begin{cases} \pi_\theta(z|x) \cdot (1 - p_\theta(y|x))^{K-1} & \text{if } \hat{y}(z) \neq y, \\ \frac{\pi_\theta(z|x)}{p_\theta(y|x)} \cdot (1 - (1 - p_\theta(y|x))^K) & \text{if } \hat{y}(z) = y. \end{cases} \tag{15}$$

This distribution is the distribution given by taking K samples $z_1, z_2, \ldots, z_K$ from $\pi_\theta(z|x)$, and then taking the first correct sample z_i as the output. If no correct sample is found, the output is the last sample z_K. Chow et al. (2025) propose to sample from this distribution $\pi_{\text{bon}}(z|x)$, and then apply the reweighting factor in Eq (14) to the sampled response z.

In our derivation, we perform reweighting of $K(1-p_\theta(y|x))^{K-1}$ on the distribution $\pi_\theta(z|x)$. The gradient we optimize is:

$$\nabla_\theta J_K(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D},\, z\sim\pi_\theta(\cdot|x)}\big[\mathbb{1}[\hat{y}(z)=y]K(1-p_\theta(y|x))^{K-1}\nabla_\theta \log \pi_\theta(z|x)\big]. \tag{16}$$

In contrast, Chow et al. (2025) optimize:

$$\nabla_\theta J_K(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D},\, z\sim\pi_{\text{bon}}(\cdot|x)}\big[\mathbb{1}[\hat{y}(z)=y]\frac{K(1-p_\theta(y|x))^{K-1}p_\theta(y|x)}{1-(1-p_\theta(y|x))^K}\nabla_\theta \log \pi_\theta(z|x)\big]. \tag{17}$$

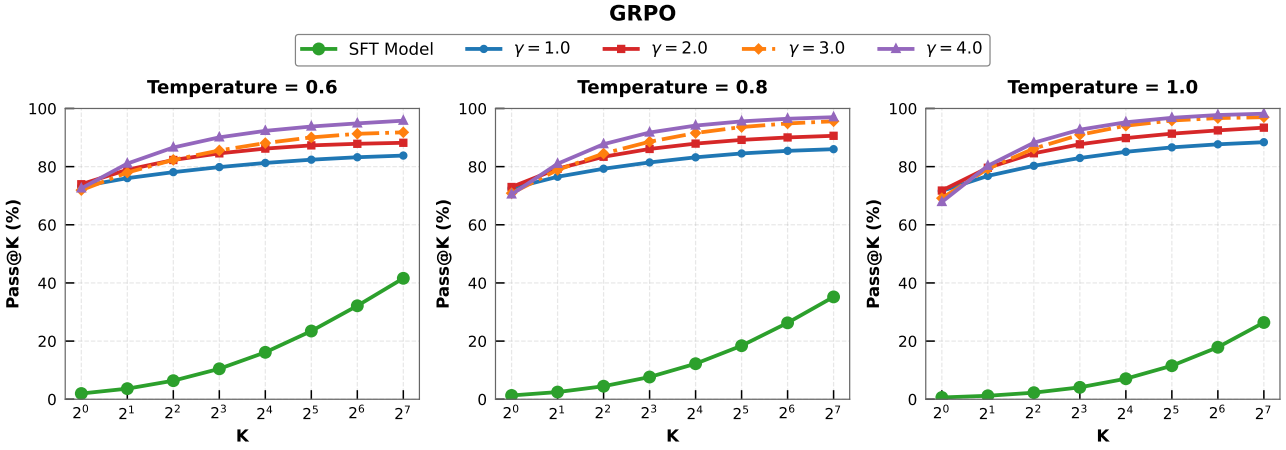
Theoretically, the two distributions and reweightings are equivalent:

$$\mathbb{E}_{z\sim\pi_{\text{bon}}(z|x)}\left[\mathbb{1}[\hat{y}(z)=y]\frac{K(1-p_\theta(y|x))^{K-1}p_\theta(y|x)}{1-(1-p_\theta(y|x))^K}f(z,x)\right] = \mathbb{E}_{z\sim\pi_\theta(z|x)}\left[\mathbb{1}[\hat{y}(z)=y]K(1-p_\theta(y|x))^{K-1}f(z,x)\right], \tag{18}$$

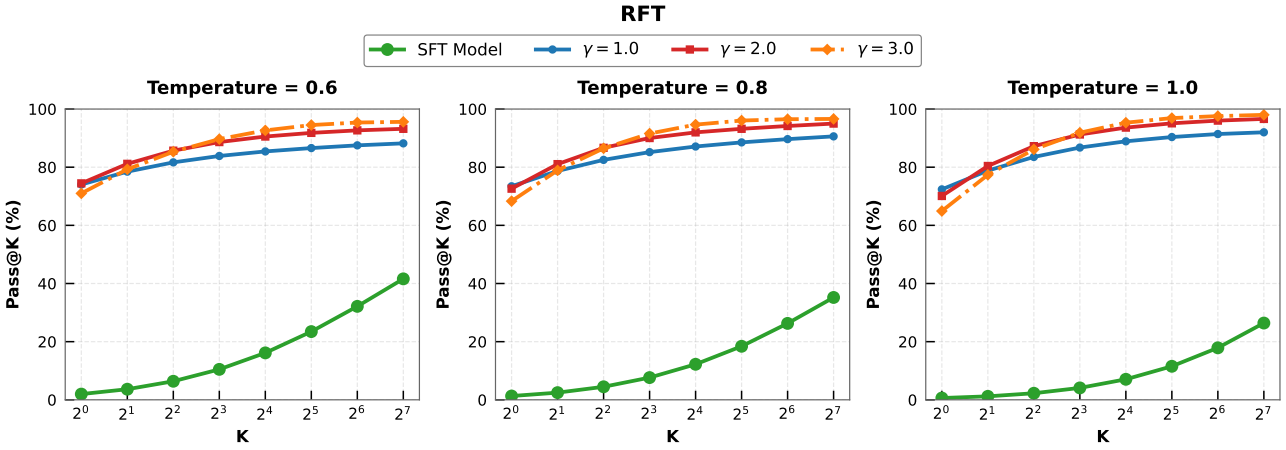
for any function $f(z,z)$. However, practically, the method of Chow et al. (2025) only uses gradient signal from a single sample z from the K sampled responses, while our method uses the gradient signal from all K samples that have correct answer. Furthermore, our method allows for a simpler implementation, as well as allowing for any real-valued reweighting factor γ.

D. Additional Experiments

We present additional experimental results across different model sizes and datasets to illustrate the performance of our reweighting approach under various settings. Figure 5 shows results for Qwen 0.5B on the arithmetic task, comparing GRPO and RFT methods across different sampling temperatures. Figure 6 provides analogous comparisons for the larger Qwen 1.5B model. Finally, Figure 7 presents GRPO results for Qwen 3B trained on the AoPS-Instruct dataset and evaluated on MATH-500, demonstrating how our method generalizes to larger models and more complex datasets.

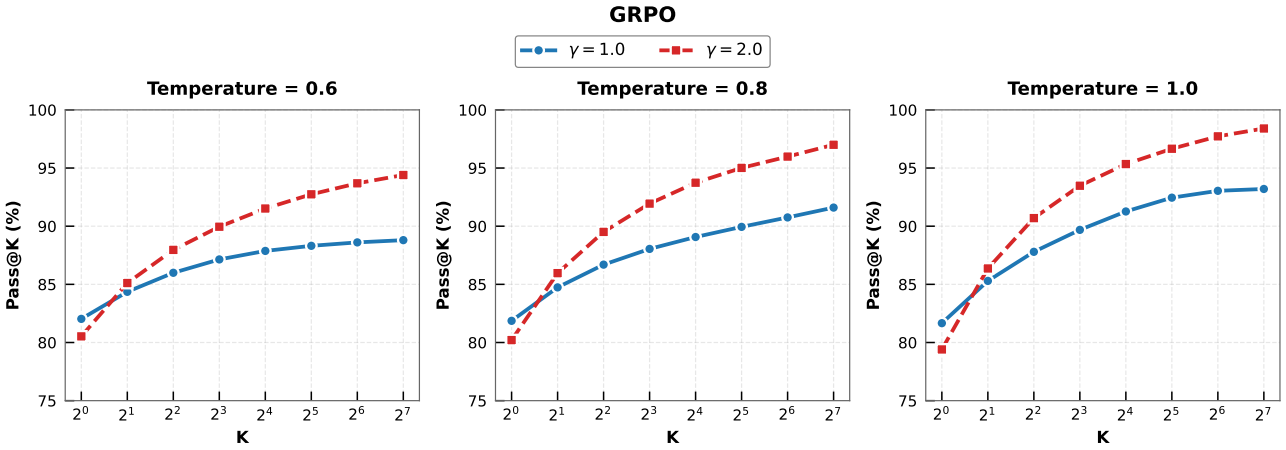


(a) GRPO Results

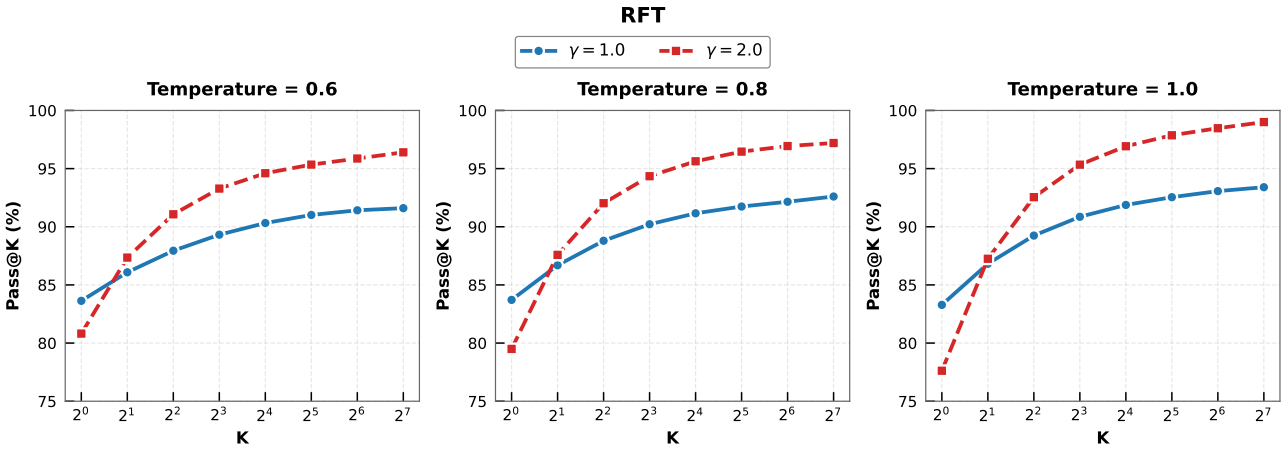


(b) RFT Results

Figure 5. Comparison of Pass@K metrics for different RL algorithms and temperatures for Qwen 0.5B on arithmetic task. Our proposed reweighting is $\gamma = 2.0$. The SFT Model is the model before RL training.



(a) GRPO Results



(b) RFT Results

Figure 6. Comparison of Pass@K metrics for different RL algorithms and temperatures for Qwen 1.5B on arithmetic task. Our proposed reweighting is $\gamma = 2.0$.

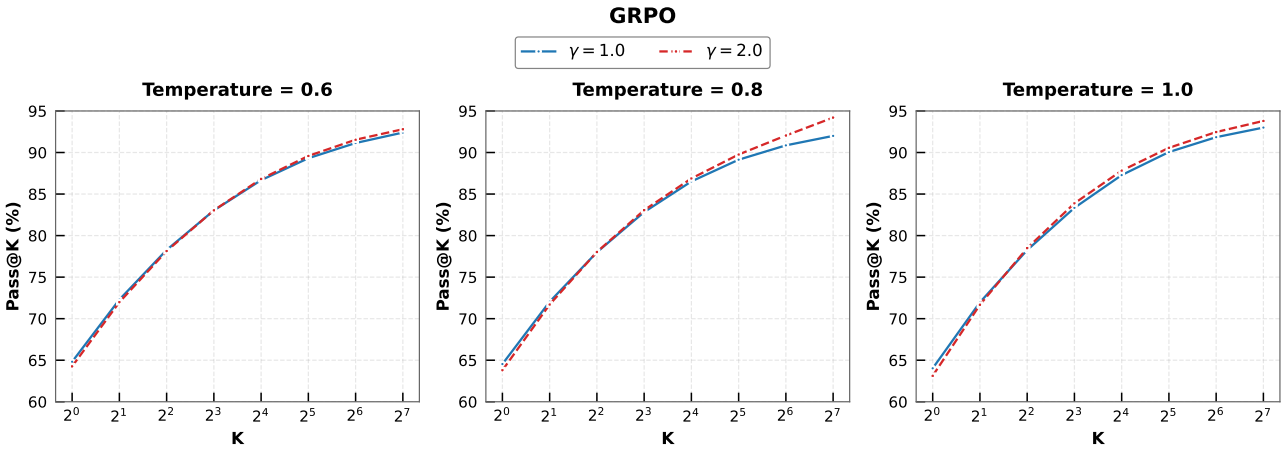


Figure 7. Comparison of Pass@K metrics for GRPO on various temperatures for Qwen 3B, trained on AoPS-Instruct dataset and evaluated on MATH-500.