GMTS: Gradient Magnitude-based Token Selection Improves RLVR Training for LLM Reasoning

Outongyi Lv[1*] **Yuanwei Zhang**[1*] **Xiaoqun Zhang**[1,2†]
[1]School of Mathematical Sciences, Shanghai Jiao Tong University
[2]Institute of Natural Sciences, Shanghai Jiao Tong University
{harry_lv,sjtuzyw,xqzhang}@sjtu.edu.cn

Abstract

Reinforcement learning (RL), particularly RL with Verifiable Rewards (RLVR), has recently emerged as a central paradigm for enhancing large language models' (LLMs) reasoning abilities, demonstrating remarkable effectiveness across reasoning tasks. Recent studies suggest that high-entropy tokens play an exceptionally important role in model training, since training with only the highest 20% entropy tokens yields significant performance gains. However, why such high-entropy tokens are beneficial remains insufficiently understood. In this work, we find that although high-entropy tokens within one answer tend to correlate with large gradient magnitude, entropy alone fails to consistently reflect token importance across different answers, considering the variations in the answer-level reward signals. Based on this observation, we introduce the **G**radient **M**agnitude-based **T**oken **S**election (GMTS) method to quantify token importance, which leverages the entropy–gradient connection to approximate gradient-magnitude rankings for token selection. We find that training on the top 20% tokens ranked by GMTS consistently outperforms entropy-based token selection across three reasoning domains and various model sizes, suggesting that GMTS provides a more fine-grained estimate of token contribution for RLVR training.[1]

1 Introduction

Large language models (LLMs) have recently shown remarkable capabilities across diverse domains, such as mathematics, image quality assessment, speech, and multimodality (Achiam et al., 2023; Wu et al., 2025; Shen et al., 2025; Bie et al., 2026). Reinforcement learning (RL) has made substantial contributions to the advancement of LLMs, demonstrating outstanding effectiveness in enhancing performance (Ouyang et al., 2022; Lee et al., 2024; Shinn et al., 2023). DeepSeek introduced Group Relative Policy Optimization (GRPO) (Shao et al., 2024), a method that dispenses with the value model in favor of simple rule-based rewards and adopts group-level advantage estimation, leading to significant improvements on reasoning benchmarks. Based on GRPO, several variants, such as Dynamic sAmpling Policy Optimization (DAPO) (Yu et al., 2025) and Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025), have been proposed, showing that RLVR provides a scalable and efficient paradigm for advancing the reasoning abilities of LLMs. Nevertheless, these methods adopt a uniform objective function across all tokens, without fully accounting for the fact that different tokens contribute unevenly to RLVR.

Recent studies have focused on distinguishing the token importance by designing token-level advantage functions (Yang et al., 2025; Liu et al., 2026; Ma et al., 2026), introducing new objective functions (Cui et al., 2025; Li et al., 2025), and filtering out low importance tokens (Wang et al., 2025c; Lv et al., 2026). Concurrently, several studies have also emphasized the central role of high token entropy in enhancing RLVR (Wang et al., 2025c; Cui et al., 2025; Meng et al., 2026), and controlling entropy provides a way to strengthen the RLVR reasoning ability. In these works, Wang et al. (2025c) shows that training with only the top 20% high-entropy tokens yields substantially better performance across the reasoning tasks than training with all tokens.

However, can entropy alone sufficiently indicate token importance?

In this work, we observe that within a single answer, high-entropy tokens tend to be associated

[*]The first two authors contributed equally and are listed alphabetically.
[†]Corresponding author.
[1]Our code is available at https://github.com/outongyiLv/GMTS

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 52136–52149
October 24-29, 2026 ©2026 Association for Computational Linguistics

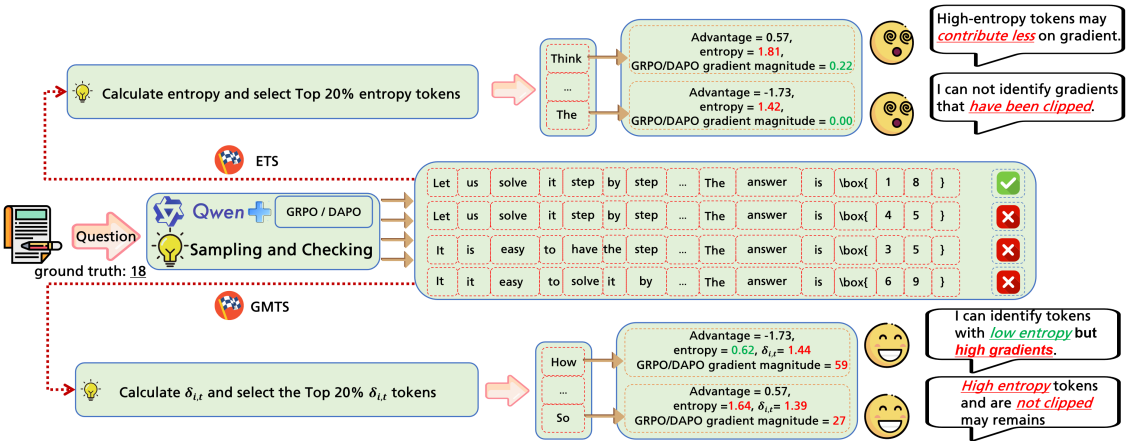


Figure 1: Overall architecture of the GMTS framework: High-entropy tokens selected by ETS may contribute less to the token gradient or be affected by gradient clipping. By computing $\delta_{i,t}$, GMTS mitigates this problem. Moreover, GMTS can identify low-entropy tokens that are overlooked by ETS but still contribute more on token gradient.

with large gradient magnitude. However, due to variations in reward signals and sample-specific characteristics, entropy alone fails to adequately capture token importance across different answers, whereas gradient magnitude provides a more reliable quantification. Based on this, we propose the **G**radient **M**agnitude-based **T**oken **S**election (GMTS) method. In contrast to **E**ntropy-based **T**oken **S**election (ETS) (Wang et al., 2025c) under the same settings, GMTS can more effectively distinguish token importance across different answers. Empirically, our experiments show that GMTS consistently improves performance across diverse reasoning tasks and model sizes. For example, GMTS achieves approximately **1-3** percentage point gains on math reasoning benchmarks. Furthermore, GMTS is also simple to implement and can be easily integrated into existing RLVR frameworks such as GRPO and DAPO. Since all the required components are already available during standard training, GMTS introduces minimal computational overhead. In summary, the contributions of this work are threefold:

- *Establish the relation between token entropy and gradient magnitude.* We observe that within a single answer, tokens with high entropy correlate strongly with large gradient magnitude, which explains why entropy is an effective indicator of token importance. However, we further demonstrate that this correlation does not hold consistently across different answers when their reward signals vary significantly. This motivates the use of gradient magnitude as a more robust and reliable metric for quantifying token importance.
- *GMTS for quantifying token importance.* Building upon the relationship between entropy and gradient magnitude, we propose GMTS for quantifying token importance. GMTS effectively retains the benefits of using entropy as a signal while overcoming its key limitations in quantifying token importance across different answers.
- *Empirical studies across reasoning domains and model scales.* Our empirical studies show that GMTS consistently outperforms the baselines across three reasoning domains and multiple model scales. Detailed experimental results and settings are provided in Section 4. These results demonstrate the effectiveness of GMTS in improving RLVR performance, as well as its adaptability across different tasks and model scales.

1.1 Related Works

Token-level advantage estimation. Chen et al. (2025) leverage the differences between positive and negative advantage-induced token-level signals, together with entropy, to redistribute token-level advantages. Tan and Pan (2025) assign entropy-weighted rewards to emphasize high-entropy tokens and down-weight low-entropy ones, mitigating model over-updating. At the same time, Cheng et al. (2026) show that high-entropy tokens correlate with exploration-facilitating behaviors and directly incorporate entropy into advantage estimation. Wang et al. (2025a) focus on high-impact strategy tokens in reasoning and amplifies

their learning signals by modifying the advantages, while Ma et al. (2026) use token-level future KL to reassess each token's contribution.

Objective function design. Wang et al. (2025b) introduce an entropy-aware clipping strategy that applies stricter constraints to low-entropy tokens while relaxing updates for high-entropy ones. Cui et al. (2025) observe that decreasing entropy can lead to entropy collapse and address this issue by dynamically adjusting the entropy regularization strength. Building on this direction, Li et al. (2025) identify critical positions using high-entropy tokens, resample them to construct branched trajectories, and jointly train RLVR on both the original and branched trajectories. In addition, Yao et al. (2025) systematically study diversity in RLVR reasoning and incorporate diversity-aware training into the RLVR objective.

Low importance token filtering. Wang et al. (2025c) find that high-entropy tokens often serve as critical forking thinking tokens, and leverage this observation by selecting only the top 20% entropy tokens for RLVR training, achieving significant performance gains. GMTS also lies in this direction, filtering tokens with low gradient magnitude and using remaining tokens in RLVR training.

2 Preliminaries

2.1 LLM Formulation and Notations

Let $\mathcal{V} = \{x^1, x^2, \ldots, x^V\}$ denote the finite vocabulary of tokens, where V is the size of the vocabulary. Given an input query $\boldsymbol{q}$, an LLM parameterized by θ defines an autoregressive policy π_θ that generates an output sequence $\boldsymbol{o} = (o_1, \ldots, o_T)$ token by token. At step t, conditioned on the query and the previously generated tokens $\boldsymbol{o}_{<t}$, the model predicts a distribution over the vocabulary $\pi_\theta(\cdot \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) = [p_{t,1}, p_{t,2}, \ldots, p_{t,V}]$, from which the next token o_t is sampled or selected. This process continues until an end-of-sequence token is generated or the maximum generation length is reached. We define the entropy of the predictive distribution at step t as

$$E_t := -\sum_{k=1}^{V} p_{t,k} \log p_{t,k}.$$

This notation will be used throughout our method.

2.2 RLVR Methods

This section briefly reviews two representative RLVR methods: GRPO and DAPO.

GRPO (Shao et al., 2024) is a PPO-style method (Schulman et al., 2017) that estimates advantages from a group of sampled responses, thereby removing the need for a separate value model. Given a query $\boldsymbol{q} \sim \mathcal{D}$, GRPO samples G answers $[\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_G]$ from π_{old}, with rewards $R = [R_1, R_2, \ldots, R_G]$ to compute the advantage $A_i = \frac{R_i - \text{mean}(R)}{\text{std}(R)}$. This response-level advantage is assigned to all tokens in $\boldsymbol{o}_i$, i.e., $A_{i,t} = A_i$. The optimization objective of GRPO is:

$$\max_\theta \mathbb{E}_{\boldsymbol{q}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|\boldsymbol{o}_i|} \sum_{t=1}^{|\boldsymbol{o}_i|} \ell_{i,t}(\theta) \right], \qquad (1)$$

where $\ell_{i,t}(\theta)$ denotes the token-level PPO-style objective:

$$\ell_{i,t}(\theta) := \min \big(\text{clip}(r_{i,t}(\theta), 1-\epsilon_1, 1+\epsilon_2) A_{i,t}, \\ r_{i,t}(\theta) A_{i,t} \big) - \beta \cdot \mathbb{D}_{i,t}^{KL}(\theta).$$

In this expression, $\mathbb{D}_{i,t}^{KL}(\theta)$ denotes the KL divergence between the current and reference predictive distributions $\pi_\theta(\cdot \mid q, \boldsymbol{o}_{<t})$ and $\pi_{\text{ref}}(\cdot \mid q, \boldsymbol{o}_{<t})$, π_{ref} is the reference policy and $r_{i,t}(\theta) = \pi_\theta(o_{i,t} | \boldsymbol{q}, \boldsymbol{o}_{i,<t}) / \pi_{\text{old}}(o_{i,t} | \boldsymbol{q}, \boldsymbol{o}_{i,<t})$. β is the parameter for the KL penalty term and ϵ_1, ϵ_2 are clipping hyperparameters. In practice, following Schulman (2020), we estimate the per-token KL penalty using a single-sample unbiased estimator.

DAPO builds on GRPO with several practical modifications (Yu et al., 2025). Compared with GRPO, DAPO removes the KL penalty from the objective and adopts dynamic sampling to discard uninformative groups in which all responses are either correct or incorrect. It also uses token-level loss averaging and a larger upper clipping threshold ϵ_2 to improve training stability.

3 Methodology

However, both GRPO and DAPO treat tokens in the same response equally in training objectives by assigning the same response-level advantage to all tokens, thereby ignoring their unequal contributions to policy updates. In this section, we first analyze token-level differences from the perspective of gradient magnitude and connect them to token entropy. We then introduce GMTS, a gradient-magnitude-based token selection method for RLVR training.

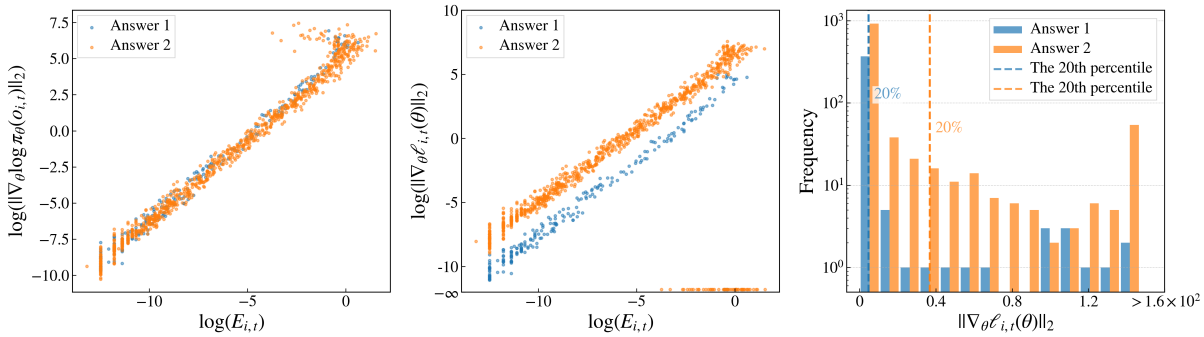


Figure 2: We use DAPO to train Qwen2.5-math-1.5B on MATH-12K dataset under the setting of $G = 16$. At training step 100, we selected two answers with different advantages in one question group, and computed the gradients $\nabla_\theta \ell_{i,t}(\theta)$, $\nabla_\theta \log \pi_\theta(o_{i,t})$, and entropy $E_{i,t}$ of all the tokens. With these collected data, we plotted entropy against the token-level log-probability gradients under the log scale **(left)**, entropy against the true gradient magnitude under the log scale **(middle)**, and the distributions of true gradient magnitude together with their 20th percentile boundaries **(right)**.

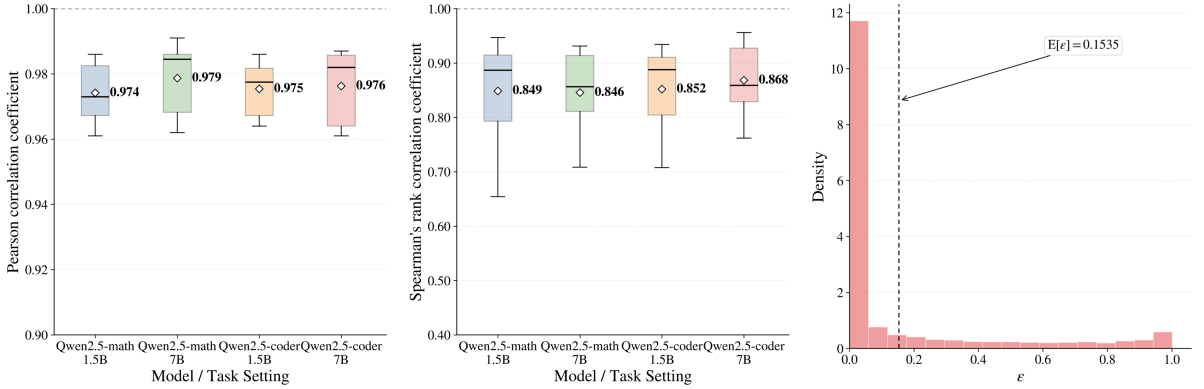


Figure 3: We select 10 questions from each of MATH-500 and KodCode, and evaluate Qwen2.5-Math and Qwen2.5-Coder at the 1.5B and 7B scales on the math and code benchmarks, respectively. For each question, we generate 16 responses. Within each group of responses to the same question, we compute the Pearson correlation coefficient between $\log(E_{i,t})$ and $\log\big(\|\nabla_\theta \log \pi_\theta(o_{i,t})\|_2\big)$ **(left)**, and the Spearman rank correlation coefficient between the GMTS score $|E_{i,t} \cdot \omega_{i,t}(\theta)|$ and the true gradient magnitude $\|\nabla_\theta \ell_{i,t}(\theta)\|_2$ **(middle)**. We further plot the distribution of ε over all 160 generated responses on MATH-500 **(right)**.

Gradient of each token. Following Yang et al. (2025), the gradient of the token-level objective $\ell_{i,t}(\theta)$ can be expressed as a scalar coefficient times the score function:

$$\nabla_\theta \ell_{i,t}(\theta) = \omega_{i,t}(\theta) \cdot \nabla_\theta \log \pi_\theta(o_{i,t}),$$

where we use $\pi_\theta(o_{i,t})$ to represent $\pi_\theta(o_{i,t}|\boldsymbol{q}, \boldsymbol{o}_{i,<t})$ for ease of exposition. The coefficient $\omega_{i,t}(\theta)$ is given by

$$\omega_{i,t}(\theta) = r_{i,t}(\theta) A_{i,t} \cdot \mathbb{I}_{\epsilon_1,\epsilon_2}(r_{i,t}(\theta), A_{i,t}) + \beta \frac{\pi_{\text{ref}}(o_{i,t})}{\pi_\theta(o_{i,t})} - \beta.$$

Here, $\mathbb{I}_{\epsilon_1,\epsilon_2}(\cdot,\cdot)$ denotes the clipping indicator function:

$$\mathbb{I}_{\epsilon_1,\epsilon_2}(a,b) = \begin{cases} 0 & \text{if } b > 0,\ a > 1+\epsilon_2, \\ 0 & \text{if } b < 0,\ a < 1-\epsilon_1, \\ 1 & \text{otherwise.} \end{cases}$$

The decomposition of $\nabla_\theta \ell_{i,t}(\theta)$ shows that $\omega_{i,t}(\theta)$ collects the effective token-level learning signals in RLVR, including the advantage value, clipping, and KL regularization terms. Therefore, it directly scales the policy gradient at each token and determines how strongly that token contributes to the overall parameter update. In DAPO, the KL term is omitted, so $\omega_{i,t}(\theta)$ reduces to $r_{i,t}(\theta) A_{i,t} \mathbb{I}_{\epsilon_1,\epsilon_2}(r_{i,t}(\theta), A_{i,t})$. When π_θ remains close to π_{old} and the clipping constraint is inactive, we have $r_{i,t}(\theta) \approx 1$ and hence $\omega_{i,t}(\theta)$ is approximately $A_{i,t}$.

Token gradient magnitude and entropy. As illustrated in the left panel of Figure 2, the magnitude of the token-level log-probability gradient $\nabla_\theta \log \pi_\theta(o_{i,t}|\boldsymbol{q}, \boldsymbol{o}_{<t})$ exhibits an approximately linear relationship with token entropy when both

quantities are plotted on a logarithmic scale. To better understand this phenomenon, we analyze the gradient structure in detail. Consider a query input $\boldsymbol{q}$ and an output sequence $\boldsymbol{o}$, and focus on the t-th generated token o_t. Let $\boldsymbol{z}_t = [z_{t,1}, z_{t,2}, \ldots, z_{t,V}]$ denote the logits produced by the final layer of the LLM at time step t, and let $\pi_\theta(\cdot \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) = \mathrm{Softmax}(\boldsymbol{z}_t) := [p_{t,1}, p_{t,2}, \ldots, p_{t,V}]$ be the resulting predictive distribution over the vocabulary.

Proposition 3.1. *Let G_t be the ℓ_2-norm of the gradient of the log-probability $\log \pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t})$ with respect to the logits z_t, E_t is the entropy of distribution $\pi_\theta(\cdot|\boldsymbol{q}, \boldsymbol{o}_{<t})$. Let $1 - \pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t}) = \varepsilon$, then we have:*

$$\lim_{\varepsilon \to 0} \frac{\log G_t}{\log E_t} = 1.$$

The detailed derivation for Theorem 3.1 is provided in Section A.2. It establishes a direct connection between token entropy and the magnitude of the log-probability gradient with respect to the logits. As shown by the distribution of ε on the right panel of Figure 3, most tokens have ε values close to 0. This provides a theoretical explanation for the linearity observed in Figure 2, as well as its slope being close to 1. Furthermore, let $\frac{\partial \boldsymbol{z}_t}{\partial \theta}$ denote the Jacobian of the logits with respect to the model parameters. By the chain rule $\nabla_\theta \log \pi_\theta(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) = \frac{\partial \boldsymbol{z}_t}{\partial \theta} \cdot \nabla_{\boldsymbol{z}_t} \log \pi_\theta(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})$, which implies that tokens with higher entropy tend to induce larger gradients with respect to the model parameters. This analysis provides a principled explanation for the empirical observation that high-entropy tokens may contribute more significantly to parameter updates in RLVR training (Wang et al., 2025c).

Gradient magnitude token selection. However, the coefficients $\omega_{i,t}(\theta)$ in the token gradients can exhibit substantial variability and are particularly influenced by the clipping constraints and the advantage values derived from answer-level reward signals. As shown in the middle and right panels of Figure 2, tokens belonging to answers with different advantages display notable differences in gradient magnitudes, along with distinct percentile distributions. These observations suggest that entropy alone is insufficient to reflect a token's contribution in RLVR training. To address this limitation, we propose the GMTS method, which assesses token importance directly from the gradient magnitude. Since directly computing the true gradient norm is computationally intractable during RL training, we approximate $\nabla_\theta \log \pi_\theta(o_{i,t}|\boldsymbol{q}, \boldsymbol{o}_{i,<t})$ using token entropy. And we define the GMTS score of token $o_{i,t}$ as:

$$\delta_{i,t} = |E_{i,t} \cdot \omega_{i,t}(\theta)|,$$

which serves as a tractable measure of token importance. Based on that, the training objective of GMTS is as follows:

$$\max_\theta \mathbb{E}_{\boldsymbol{q}} \left[\frac{1}{S_\rho} \sum_{i=1}^{G} \sum_{t=1}^{|\boldsymbol{o}_i|} \mathbb{I}[\delta_{i,t} \geq \tau_\rho] \cdot \ell_{i,t}(\theta) \right]. \quad (2)$$

where $\mathbb{I}(\cdot)$ is the indicator function that evaluates to 1 if the condition inside holds and 0 otherwise. ρ is a predefined ratio specifying the top proportion to be selected, and τ_ρ is the corresponding threshold such that only tokens with $\delta_{i,t}(\theta) \geq \tau_\rho$, comprising the top-$\rho$ fraction of all tokens in the batch, are used to compute the gradient, S_ρ denotes the number of selected tokens in the group. Thus the loss is averaged only over the selected tokens.

From a computational perspective, calculating $\omega_{i,t}(\theta)$ involves the advantage signal $A_{i,t}$ and the predicted probability from π_θ, π_{old} and π_{ref} for each token. These quantities are already available during standard RLVR training. Therefore, the additional computational overhead introduced by GMTS is minimal. Moreover, GMTS can be readily integrated into other RLVR training frameworks, with the only difference being the specific formulation of the coefficient $\omega_{i,t}(\theta)$. By leveraging only a small subset of influential tokens for RLVR updates, GMTS offers the potential for more efficient RLVR training.

Comparison with ETS. Both ETS and GMTS retain only the top-ρ fraction of tokens for policy updates, but they differ in how token importance is measured. ETS ranks tokens solely by entropy, whereas GMTS scores each token by $\delta_{i,t}$, which incorporate the effective scalar coefficient in the token gradient. This difference matters in three aspects: (1) Tokens with similar entropy can have very different effects when they belong to responses with different advantages. As shown in the middle panels of Figure 2, the same level of entropy may correspond to distinct gradient magnitudes due to different response-level reward signals. (2) PPO-style clipping can suppress the policy-gradient term when the probability ratio falls outside the clipping range. In such cases, ETS may still select high-entropy tokens whose gradients are inactive or largely reduced, while GMTS downweights them

through the clipping indicator in $\omega_{i,t}(\theta)$. (3) When KL regularization is used, the effective gradient is also affected by the reference-policy correction term, which is not reflected by entropy alone. Thus, GMTS prioritizes tokens that contribute more directly to the actual RLVR update.

4 Numerical Experiments

4.1 Training and Evaluation Settings

We implement and evaluate GMTS and ETS within the verl framework[2] across GRPO and DAPO. Furthermore, we provide a standalone implementation[3] that not only gives a clear understanding of GMTS but also supports training under limited GPUs. In our experiments, unless otherwise specified, both ETS and GMTS are trained using the **Top 20%** of tokens, consistent with the selection ratio recommended by ETS (Wang et al., 2025c).

Training Details. We evaluate GMTS across three distinct domains: **MATH**, **CODE**, and **COMMONSENSE (CS)**. Primary results are reported for the 1.5B and 7B model variants, with additional experiments conducted on the 8B model specifically for **MATH** domain. We use the following training sets: For MATH domain, we utilize the MATH-12K (Lightman et al., 2023) dataset (1.5B and 7B) and DAPO-MATH-17K (Wang et al., 2025c) (8B). For CODE domain, we utilize the KodCode (Xu et al., 2025). For CS domain, we utilize the CS-QA (Talmor et al., 2019). Other details for training on these tasks are all summarized in Section A.3.

Evaluation Details. For **MATH** domain, we evaluate the 1.5B and 7B models on five benchmarks: AIME2024, AMC23, MATH-500, Minerva, and OlympiadBench. For the 8B model, we additionally report results on AIME2025, since it is particularly challenging for small models (1.5B and 7B). For **CODE** domain, we utilize standard benchmarks widely adopted in the Qwen3-Coder evaluation [4]: LiveCodeBench (202407 – 202411), MBPP, HumanEval, and Bigcode-Bench. For **CS** domain, we utilize the test of CS-QA and more general CS-QA2 (Talmor et al., 2022). For **MATH** and **CS** domains, we generate 16 candidate answers for each question and report the average accuracy as *average@16*. All generations are performed with the temperature of $T = 1.0$. For **CODE** domain, we employ the greedy strategy for evaluation.

[2] https://github.com/volcengine/verl

[3] https://github.com/policy-gradient/GRPO-Zero

[4] https://github.com/QwenLM/Qwen3-Coder/

4.2 Main Results

In Figure 4, we illustrate the global performance of ETS and GMTS across different domains. Next, we show more details of these results on each domain.

4.2.1 Results on MATH Domain

We implement ETS/GMTS on GRPO/DAPO, and apply them to the Qwen2.5-math-1.5B and 7B model. All the detailed results are in Tables 1 and 2.

Tables 1 and 2 show that GMTS Top (20%) consistently outperforms ETS across both DAPO and GRPO backbones (1.5B-DAPO: **+1.55**, 1.5B-GRPO: **+1.30**, 7B-DAPO: **+1.33**, 7B-GRPO: **+3.41**). To evaluate the performance of GMTS on larger reasoning models, we extended our experiments to Qwen3-8B, using results from Wang et al. (2025c) as a baseline. As shown in Table 3, GMTS consistently delivers notable improvements over ETS across a wide range of math benchmarks, with substantial improvements in challenging tasks such as AIME2024 **(+5.21)** and AIME2025 **(+3.75)**. With an overall average gain of **1.85%**, these results demonstrate the generality of GMTS, suggesting it has the potential to scale effectively to larger and more complex reasoning models.

4.2.2 Results on CODE and CS Domains

We implement ETS/GMTS on GRPO/DAPO, for code domain, we apply them to the Qwen2.5-coder-1.5B and 7B model. For commonsense domain, we apply them to the Qwen2.5-1.5B-base and 7B model. All the detailed results are in Tables 4 and 5.

For HumanEval and MBPP, which consist of Base and Plus subsets, the table reports the average accuracy across both. Similarly, for BigCodeBench, the reported values represent the mean accuracy across the Full and Hard subsets. As shown in Tables 4 and 5, GMTS achieves consistent performance gains within the code domain (1.5B-DAPO: **+1.87**, 1.5B-GRPO: **+1.90**, 7B-DAPO: **+0.74**, 7B-GRPO: **+0.69**). Furthermore, in the commonsense domain (1.5B-DAPO: **+0.87**, 1.5B-GRPO: **+0.67**, 7B-DAPO: **+1.53**, 7B-GRPO: **+1.26**). These results validate the generalization of the GMTS method across diverse reasoning tasks.

4.3 Ablation Study

Linearity and monotonicity consistency. We evaluated the linear and monotonic behavior of GMTS using Qwen2.5-math and Qwen2.5-coder at both

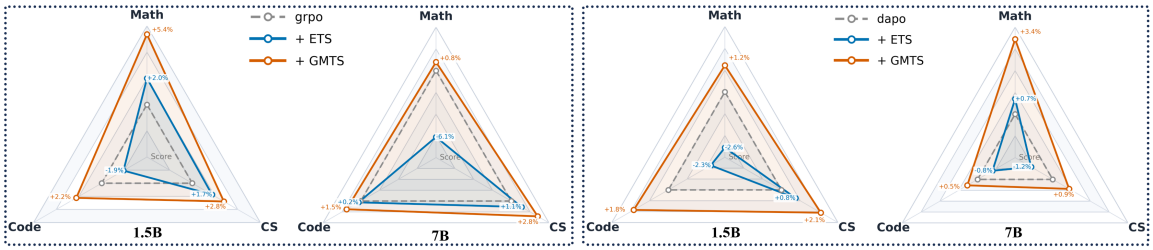


Figure 4: Overall performance comparison. GMTS consistently outperforms GRPO and DAPO baselines as well as the ETS method across multiple model scales (1.5B and 7B).

Table 1: Evaluation on 5 math-reasoning benchmarks (ignore AIME2025) for ETS/GMTS with DAPO/GRPO on **Qwen2.5-math-1.5B** under *average@16* accuracy (%).

	AIME2024	AMC23	MATH-500	Minerva	OlympiadBench	Avg.
Qwen2.5-math-1.5B	2.70	18.75	23.40	6.20	14.10	13.03
DAPO	14.17	50.62	**73.85**	29.47	37.17	41.06
+ ETS	14.17	48.44	72.25	29.01	36.16	40.01
+ GMTS	**14.58**	50.63	73.83	**30.51**	**38.23**	**41.56**
GRPO	10.63	48.12	71.59	28.81	34.90	38.81
+ ETS	12.71	48.89	72.39	28.41	35.54	39.59
+ GMTS	11.88	**53.33**	**73.06**	**30.11**	**36.09**	**40.89**

Table 2: Evaluation on 5 math-reasoning benchmarks (ignore AIME2025) for ETS/GMTS with DAPO/GRPO on **Qwen2.5-math-7B** under *average@16* accuracy (%).

	AIME2024	AMC23	MATH-500	Minerva	OlympiadBench	Avg.
Qwen2.5-math-7B	7.29	21.09	34.45	6.09	10.92	15.97
DAPO	18.12	62.50	80.91	36.49	44.35	48.47
+ ETS	20.00	63.75	80.04	36.72	43.56	48.81
+ GMTS	**25.00**	**65.16**	80.40	**37.39**	42.75	**50.14**
GRPO	**27.08**	62.34	79.76	35.18	42.87	49.45
+ ETS	19.17	60.67	77.72	33.65	40.96	46.43
+ GMTS	23.33	**64.83**	**81.13**	**36.32**	**43.60**	**49.84**

Table 3: Evaluation on 6 math-reasoning benchmarks for ETS/GMTS with DAPO on **Qwen3-8B** under *average@16* accuracy (%).

	AIME2024	AIME2025	AMC23	MATH-500	Minerva	OlympiadBench	Avg.
Qwen3-8B	3.75	6.04	33.91	62.99	22.29	23.66	25.44
DAPO	33.33	25.42	77.81	89.24	39.77	56.67	53.71
+ ETS	34.58	26.25	77.19	89.70	40.26	**57.43**	54.23
+ GMTS	**39.79**	**30.00**	**79.38**	**91.14**	**42.42**	53.76	**56.08**

the 1.5B and 7B scales. As shown in Figure 3 right and middle panels, the linear relationship consistently holds across domains and model scales, indicating that it is not an isolated empirical artifact. Moreover, GMTS exhibits strong monotonicity, suggesting that gradient-magnitude-based selection preserves the ordering of token contributions.

Bottom selection. To examine the role of low gradient magnitude tokens, we follow Wang et al. (2025c) to evaluate DAPO with bottom selection on Qwen2.5-math-1.5B. As shown in Figure 5 left and Table 6, GMTS consistently performs worse than ETS, with average drops of **-1.08** and **-1.28** at the

Table 4: Evaluation on 4 code-reasoning and 2 CS benchmarks for ETS/GMTS with DAPO/GRPO on **Qwen2.5-coder-1.5B** under *greedy* accuracy (%) (left) and **Qwen2.5-base-1.5B** under *average@16* accuracy (%) (right).

	LiveCodeBench	HumanEval	MBPP	Bigcode-Bench	Avg.	CS-QA	CS-QA2	Avg.
1.5B Base Model	1.33	22.04	12.21	4.20	9.95	10.26	4.14	7.20
DAPO	11.02	72.50	73.14	26.31	45.74	77.21	52.73	64.97
+ ETS	9.00	73.51	73.12	23.22	44.71	78.09	52.82	65.46
+ GMTS	11.02	**74.72**	**73.49**	**27.06**	**46.58**	**79.15**	**53.51**	**66.33**
GRPO	10.31	72.91	74.14	25.10	45.62	77.43	51.07	64.25
+ ETS	9.00	72.92	71.34	25.71	44.74	79.82	50.92	65.37
+ GMTS	**11.02**	**75.03**	**74.21**	**26.33**	**46.64**	**79.95**	**52.16**	**66.04**

Table 5: Evaluation on 4 code-reasoning and 2 CS benchmarks for ETS/GMTS with DAPO/GRPO on **Qwen2.5-coder-7B** under *greedy* accuracy (%) (left) and **Qwen2.5-base-7B** under *average@16* accuracy (%) (right).

	LiveCodeBench	HumanEval	MBPP	Bigcode-Bench	Avg.	CS-QA	CS-QA2	Avg.
7B Base Model	3.20	36.33	39.92	13.50	23.24	31.56	14.85	23.21
DAPO	19.94	85.10	79.02	36.84	55.20	84.70	66.57	75.64
+ ETS	19.06	84.39	77.18	**38.42**	54.76	83.44	66.09	74.77
+ GMTS	**20.13**	**85.62**	**79.89**	36.37	**55.50**	**85.65**	**66.95**	**76.30**
GRPO	20.00	84.21	76.71	36.31	54.31	84.00	62.39	73.20
+ ETS	17.31	85.46	77.02	**37.98**	54.44	84.17	63.82	74.00
+ GMTS	**20.60**	**85.91**	**77.43**	36.58	**55.13**	**84.60**	**65.92**	**75.26**

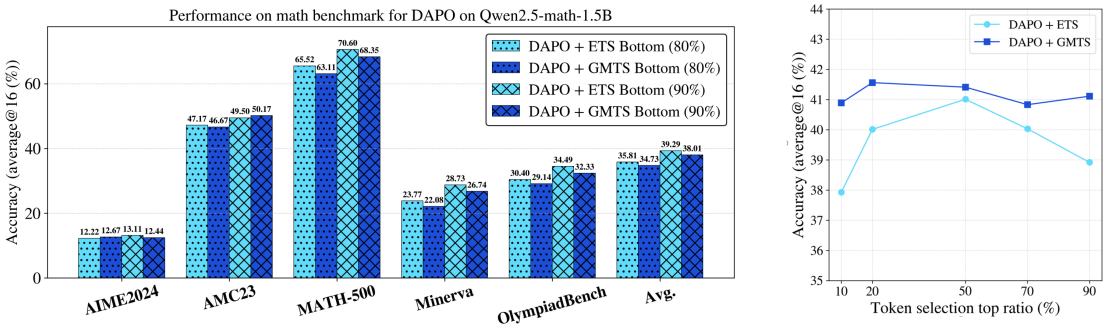


Figure 5: The *average@16* performance of DAPO, DAPO + ETS Bottom (80%, 90%) and DAPO + GMTS Bottom (80%, 90%) on Qwen2.5-math-1.5B (**left**) and the *average@16* overall performance of varying selected ratios for DAPO, DAPO + ETS Top, and DAPO + GMTS Top on Qwen2.5-math-1.5B (**right**).

80% and 90% ratios, respectively. This suggests that low gradient magnitude tokens contribute less to RLVR training, whereas low-entropy tokens may still contain useful signals.

Sensitivity to selection ratios. To evaluate the impact of selection intensity, we evaluate both DAPO and GRPO on Qwen2.5-math-1.5B with ratios 0.1, 0.2, 0.5, 0.7, and 0.9, with the results reported in Table 7 and Figure 5 right. It shows that GMTS consistently outperforms ETS in nine out of ten configurations. These findings suggest that the advantages of GMTS are not confined to a specific regime but extend across a broad spectrum of training conditions, providing a more reliable selection mechanism than ETS.

5 Conclusion

In this work, we propose GMTS for improving RLVR in LLM reasoning. Although gradient magnitude is correlated with entropy within one answer, entropy alone cannot fully capture token importance across different answers, where the contribution of each token to policy update can vary substantially. GMTS addresses this limitation by leveraging a gradient-magnitude proxy for token selection. Empirically, GMTS achieves consistent improvements over ETS and remains robust across multiple selection ratios, model scales, and evaluation domains. These results provide empirical evidence supporting GMTS as an effective and efficient choice for RLVR.

Limitations

Despite the consistent performance gains of GMTS in RLVR, several limitations remain and point to directions for future research.

First, although extensive empirical evaluations support the effectiveness of GMTS, its theoretical foundation remains only partially understood. In this work, we provide preliminary evidence on the relationship between token entropy and gradient magnitude, but a rigorous theoretical justification for why GMTS outperforms ETS, along with a deeper analytical characterisation of its behaviour, remains an important direction for future work.

Second, due to computational constraints, we have not yet evaluated GMTS on larger models, such as 14B and 32B models. While our empirical results across different reasoning domains and model scales suggest that GMTS has the potential to extend to larger-scale models, a more comprehensive evaluation on such models is left for future work.

Acknowledgments

We thank the SJTU supercomputer center for providing the computing services and the anonymous reviewers very much for their careful reading and valuable comments. This work was supported by the National Natural Science Foundation of China (Grant No. 125B2026) and the Natural Science Foundation of Chongqing, China (No. CSTB2023NSCQ-LZX0054).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and et al. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, and et al. 2026. Llada2. 1: Speeding up text diffusion via token editing. arXiv preprint arXiv:2602.08676.

Minghan Chen, Guikun Chen, Wenguan Wang, and Yi Yang. 2025. Seed-grpo: Semantic entropy enhanced grpo for uncertainty-aware policy optimization. arXiv preprint arXiv:2505.12346.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Xin Zhao, Zhenliang Zhang, and Furu Wei. 2026. Reasoning with exploration: An entropy perspective. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 40, pages 30377–30385.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, and et al. 2025. The entropy mechanism of reinforcement learning for reasoning language models. arXiv preprint arXiv:2505.22617.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Ren Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and et al. 2024. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. In International Conference on Machine Learning, pages 26874–26901. PMLR.

Qingbin Li, Rongkun Xue, Jie Wang, Ming Zhou, Zhi Li, Xiaofeng Ji, Yongqi Wang, Miao Liu, Zheming Yang, Minghui Qiu, and et al. 2025. Cure: Critical-token-guided re-concatenation for entropy-collapse prevention. arXiv preprint arXiv:2508.11016.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In The Twelfth International Conference on Learning Representations.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and et al. 2026. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. arXiv preprint arXiv:2601.05242.

Outongyi Lv, Yanzhao Zheng, Yuanwei Zhang, Zhenghao Huang, Xingjun Wang, Baohua Dong, Hangcheng Zhu, and Yingda Chen. 2026. Which tokens matter? adaptive token selection for rlvr with the relative surprisal index. arXiv preprint arXiv:2606.31575.

Chiyu Ma, Shuo Yang, Kexin Huang, Jinda Lu, Haoming Meng, Shangshang Wang, Bolin Ding, Soroush Vosoughi, Guoyin Wang, and Jingren Zhou. 2026. Fipo: Eliciting deep reasoning with future-kl influenced policy optimization. arXiv preprint arXiv:2603.19835.

Haoming Meng, Kexin Huang, Shaohang Wei, Chiyu Ma, Shuo Yang, Xue Wang, Guoyin Wang, Bolin Ding, and Jingren Zhou. 2026. Sparse but critical: A token-level analysis of distributional shifts in rlvr fine-tuning of llms. In The Fourteenth International Conference on Learning Representations.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and et al. 2022. Training language models to follow instructions with human feedback. Advances in neural information processing systems, 35:27730–27744.

John Schulman. 2020. Approximating kl divergence. John Schulman's Homepage.

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

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

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, and et al. 2025. Vlm-r1: A stable and generalizable r1-style large vision-language model. arXiv preprint arXiv:2504.07615.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems, 36:8634–8652.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4149–4158.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2022. Commonsenseqa 2.0: Exposing the limits of ai through gamification. arXiv preprint arXiv:2201.05320.

Hongze Tan and Jianfei Pan. 2025. Gtpo and grpo-s: Token and sequence-level reward shaping with policy entropy. arXiv preprint arXiv:2508.04349.

Haozhe Wang, Qixin Xu, Che Liu, Junhong Wu, Fangzhen Lin, and Wenhu Chen. 2025a. Emergent hierarchical reasoning in llms through reinforcement learning. arXiv preprint arXiv:2509.03646.

Jiakang Wang, Runze Liu, Fuzheng Zhang, Xiu Li, and Guorui Zhou. 2025b. Stabilizing knowledge, promoting reasoning: Dual-token constraints for rlvr. arXiv preprint arXiv:2507.15778.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and et al. 2025c. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. arXiv preprint arXiv:2506.01939.

Tianhe Wu, Jian Zou, Jie Liang, Lei Zhang, and Kede Ma. 2025. Visualquality-r1: Reasoning-induced image quality assessment via reinforcement learning to rank. arXiv preprint arXiv:2505.14460.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. 2025. Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. In Findings of the Association for Computational Linguistics: ACL 2025, pages 6980–7008.

Zhihe Yang, Xufang Luo, Zilong Wang, Dongqi Han, Zhiyuan He, Dongsheng Li, and Yunjian Xu. 2025. Do not let low-probability tokens over-dominate in rl for llms. arXiv preprint arXiv:2505.12929.

Jian Yao, Ran Cheng, Xingyu Wu, Jibin Wu, and Kay Chen Tan. 2025. Diversity-aware policy optimization for large language model reasoning. arXiv preprint arXiv:2505.23433.

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

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, and et al. 2025. Group sequence policy optimization. arXiv preprint arXiv:2507.18071.

A Appendix

A.1 More Experimental Results

We provide additional complete numerical results for Section 4.3 in the main text, with detailed values reported in Tables 6 and 7. The Bottom selection experiment in Table 6 is designed to empirically examine whether the top ranked tokens under GMTS contain more training beneficial information than those selected by ETS. In addition, as shown in Table 7, GMTS clearly outperforms ETS in 9 out of 10 GRPO/DAPO settings, which empirically shows that GMTS generally remains superior to ETS across different top-selection ratios.

A.2 Derivation of token gradient

Proposition 3.1: Let G_t be the ℓ_2-norm of the gradient of the log-probability $\log \pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t})$ with respect to the logits z_t, E_t is the entropy of distribution $\pi_\theta(\cdot|\boldsymbol{q}, \boldsymbol{o}_{<t})$. Let $1 - \pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t}) = \varepsilon$, then we have:

$$\lim_{\varepsilon \to 0} \frac{\log G_t}{\log E_t} = 1.$$

Proof. Let $p_t = \pi_\theta(\cdot \mid \boldsymbol{q}, \boldsymbol{o}_{<t})$

$$\begin{aligned} \nabla_z \log(p_{t,i}) &= \frac{1}{p_{t,i}} \nabla_z p_{t,i} \\ &= \frac{1}{p_{t,i}} \left[p_{t,i}(e_i - p_t) \right] \\ &= e_i - p_t, \end{aligned}$$

Table 6: Evaluation on five math-reasoning benchmarks (ignore AIME2025) for ETS/GMTS with DAPO/GRPO on **Qwen2.5-math-1.5B** under *average@16* accuracy (%).

	AIME2024	AMC23	MATH-500	Minerva	OlympiadBench	Avg.
Qwen2.5-math-1.5B	2.70	18.75	23.40	6.20	14.10	13.03
DAPO	14.17	50.62	73.85	29.47	37.17	41.06
+ ETS Bottom (80%)	12.22	47.17	65.52	23.77	30.40	35.81
+ ETS Bottom (90%)	13.11	49.50	70.60	28.73	34.49	39.29
+ GMTS Bottom (80%)	12.67	46.67	63.11	22.08	29.14	34.73
+ GMTS Bottom (90%)	12.44	50.17	68.35	26.74	32.33	38.01

Table 7: Evaluation on five math-reasoning benchmarks for ETS/GMTS with DAPO/GRPO on **Qwen2.5-math-1.5B** under *average@16* accuracy (%) in different selected ratios.

	AIME2024	AMC23	MATH-500	Minerva	OlympiadBench	Avg.
Qwen2.5-math-1.5B	2.70	18.75	23.40	6.20	14.10	13.03
DAPO	14.17	50.62	73.85	29.47	37.17	41.06
+ ETS Top (90%)	**13.96**	51.72	73.97	**30.36**	24.58	38.92
+ GMTS Top (90%)	11.04	**52.34**	**74.20**	**30.36**	**37.61**	**41.11**
+ ETS Top (70%)	**12.08**	48.44	**74.07**	**29.26**	36.31	40.03
+ GMTS Top (70%)	11.46	**52.50**	74.00	28.80	**37.40**	**40.83**
+ ETS Top (50%)	12.50	52.81	73.22	29.55	**36.97**	41.01
+ GMTS Top (50%)	**13.54**	**53.44**	**73.66**	**29.89**	36.50	**41.41**
+ ETS Top (10%)	12.71	46.88	70.03	27.08	32.93	37.93
+ GMTS Top (10%)	**13.33**	**52.19**	**73.53**	**29.72**	**35.68**	**40.89**
GRPO	10.63	48.12	71.59	28.81	34.90	38.81
+ ETS Top (90%)	11.56	50.00	**72.73**	29.61	**36.07**	40.00
+ GMTS Top (90%)	**15.33**	**51.00**	72.68	**30.51**	35.91	**41.09**
+ ETS Top (70%)	11.88	50.62	73.51	30.31	36.61	40.59
+ GMTS Top (70%)	**12.92**	**52.50**	**74.22**	**30.61**	**37.11**	**41.47**
+ ETS Top (50%)	**12.92**	**50.78**	**73.46**	**29.25**	**37.86**	**40.85**
+ GMTS Top (50%)	10.63	49.38	72.95	28.75	36.09	39.56
+ ETS Top (10%)	10.63	45.94	70.00	27.30	32.97	37.37
+ GMTS Top (10%)	**15.00**	**49.84**	**72.24**	**29.60**	**35.65**	**40.47**

where e_i is the unit vector whose i-th entry is 1 and all other entries are 0. Let $\pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t}) = p_{t,i}$, we can get:

$$G_t = \sqrt{(1-p_{t,i})^2 + \sum_{j\neq i} p_{t,j}^2}.$$

According to $1 - \pi_\theta(o_t|\boldsymbol{q}, \boldsymbol{o}_{<t}) = \varepsilon$, then we will have

$$\log G_t = \log\Big(\sqrt{\varepsilon^2\left(1 + \sum_{j\neq i} q_{t,j}^2\right)}\Big) = \log\varepsilon + \log\Big(\sqrt{1 + \sum_{j\neq i} q_{t,j}^2}\Big).$$

where $q_{t,k} = \frac{p_{t,k}}{\varepsilon}$.

For entropy E_t, we have:

$$\begin{aligned} E_t &= -\sum_j p_{t,j} \log p_{t,j} \\ &= -p_{t,i} \log p_{t,i} - (1 - p_{t,i}) \log(1 - p_{t,i}) \\ &\quad - (1 - p_{t,i}) \sum_{j \neq i} \frac{p_{t,j}}{1 - p_{t,i}} \log \frac{p_{t,j}}{1 - p_{t,i}} \end{aligned}$$

Notice that $-\sum_{j \neq i} \frac{p_{t,j}}{1-p_{t,i}} \log \frac{p_{t,j}}{1-p_{t,i}}$ is the entropy of the renormalized residual distribution after removing the i-th probability, we call it $E(p_t/p_{t,i})$, then we have:

$$E_t = -(1 - \varepsilon) \log(1 - \varepsilon) - \varepsilon \log \varepsilon + \varepsilon E(p_t/p_{t,i})$$

Using the Taylor expansion $\log(1 - \varepsilon) = -\varepsilon - \frac{\varepsilon^2}{2} + O(\varepsilon^3)$. We get

$$\begin{aligned} &\log E_t \\ &= \log(\varepsilon - \varepsilon \log \varepsilon + \varepsilon E(p_t/p_{t,i}) - \frac{\varepsilon^2}{2} + O(\varepsilon^3)) \\ &= \log \varepsilon + \log(1 - \log \varepsilon + E(p_t/p_{t,i}) - \frac{\varepsilon}{2} + O(\varepsilon^2)). \end{aligned}$$

Thus, we have:

$$\begin{aligned} &\lim_{\varepsilon \to 0} \frac{\log G_t}{\log E_t} = \\ &\lim_{\varepsilon \to 0} \frac{\log \varepsilon + \log(\sqrt{1 + \sum_{j \neq i} q_{t,j}^2})}{\log \varepsilon + \log(1 - \log \varepsilon + E(p_t/p_{t,i}) - \frac{\varepsilon}{2} + O(\varepsilon^2))} \\ &= 1. \end{aligned}$$

□

A.3 Training Details

For fairness and reproducibility, we fix the training seed to **0** in all the experiments, which are performed on eight NVIDIA H100 GPUs, each equipped with 80 GB of memory.

Hyperparameter settings for Qwen2.5-math-1.5B and Qwen2.5-math-7B.

For the Qwen2.5-math-1.5B and Qwen2.5-math-7B models, the training configurations are as follows: the maximum response length is set to 2048 tokens, and the maximum prompt length to 1024 tokens. Each training batch contained 64 questions, with a mini-batch size of 64. For each question, 16 rollouts are sampled with a temperature of 1.0. The clipping parameters ϵ_1 and ϵ_2 are set to 0.2 and 0.28. The learning rate is fixed at 3×10^{-5} without applying any learning rate warm-up or scheduling.

Hyperparameter settings for Qwen3-8B.

Due to computational constraints, we could not fully adopt the settings of (Wang et al., 2025c), and therefore adjusted the maximum output length. Specifically, we set the maximum response length to 4096 tokens and the maximum prompt length to 1024 tokens. Each training batch contained 512 questions and a mini-batch size of 32, resulting in 16 gradient steps per training batch. For each question, 16 rollouts are sampled with a temperature of 1.0. The clipping parameters ϵ_1 and ϵ_2 are set to 0.2 and 0.28. The learning rate is fixed at 1×10^{-6} without applying any learning rate warm-up or scheduling.

Hyperparameter settings for Qwen2.5-coder-1.5B and Qwen2.5-coder-7B.

For the Qwen2.5-coder-1.5B and Qwen2.5-coder-7B models, the training configurations are as follows: the maximum response length is set to 32768 tokens, and the maximum prompt length to 2048 tokens. Each training batch contained 512 questions, with a mini-batch size of 32. For each question, 16 rollouts are sampled with a temperature of 1.0. The clipping parameters ϵ_1 and ϵ_2 are set to 0.2 and 0.28. The learning rate is fixed at 1×10^{-6} without applying any learning rate warm-up or scheduling.

Hyperparameter settings for Qwen2.5-base-1.5B and Qwen2.5-base-7B.

For the Qwen2.5-base-1.5B and Qwen2.5-base-7B models, the training configurations are as follows: the maximum response length is set to 4096 tokens, and the maximum prompt length to 1024 tokens. Each training batch contained 512 questions, with a mini-batch size of 32. For each question, 16 rollouts are sampled with a temperature of 1.0. The clipping parameters ϵ_1 and ϵ_2 are set to 0.2 and 0.28. The learning rate is fixed at 1×10^{-6} without applying any learning rate warm-up or scheduling.

Dialogue template

For the math and CS domains, we adopt the following dialogue template:

```
<|im_start|>system\nPlease reason step
by step, and put your final answer
within \boxed{}.<|im_end|>\n<|im_start|>
user\n"<|im_end|>\n<|im_start|>assistant
\n
```

For the code domain, we adopt the following dialogue template:

```
<|im_start|>system\nYou are Qwen,
created by Alibaba Cloud. You are a
helpful assistant.<|im_end|>\n<|im_start
|>user\n Please provide a self-contained
 Python script that solves the following
```

```
 problem:\n {question}\nYou will use the
 following starter code to write the
solution to the problem:\n```python\n{
starter_code}\n```\n Please reason step
by step, and put your code answer in a
markdown code block.
```

A.4 An Example for ETS Top (20%) and GMTS (20%) with DAPO on Qwen3-8B under AIME2025

In DAPO on Qwen3-8B, we selected an example under the **ETS Top (20%)** and **GMTS Top (20%)** settings to illustrate that the output of GMTS is not only more concise but also more accurate compared to ETS:

Question and Ground-Truth

Question: Find the number of ordered pairs (x, y), where both x and y are integers between -100 and 100, inclusive, such that $12x^2 - xy - 6y^2 = 0$.
Ground-Truth:

$$\boxed{117}$$

ETS Top (20%)

Okay, so I need to find the number of ordered pairs (x, y) where both x and y are integers between -100 and 100 inclusive, and they satisfy the equation

$$12x^2 - xy - 6y^2 = 0.$$

Let me think about how to approach this.
First, the equation is quadratic in both x and y. Maybe I can factor it or find some relationship between x and y. Let me try rearranging the equation. Let me write it again:

$$12x^2 - xy - 6y^2 = 0.$$

Hmm. Maybe I can factor this quadratic equation. Let me see. If I treat this as a quadratic in x, then maybe I can factor it or use the quadratic formula. Alternatively, maybe factor it as a quadratic in y. Let me try both approaches.
Let me try treating it as a quadratic in x. So,

$$12x^2 - yx - 6y^2 = 0.$$

So, in the form $ax^2 + bx + c = 0$, where $a = 12$, $b = -y$, and $c = -6y^2$. Then, using quadratic formula:

$$x = \frac{y \pm \sqrt{y^2 - 4 \cdot 12 \cdot (-6y^2)}}{2 \cdot 12}.$$

Calculating discriminant:

$$D = y^2 - 4 \cdot 12 \cdot (-6y^2) = y^2 + 288y^2 = 289y^2.$$

So

$$\sqrt{D} = \sqrt{289y^2} = 17|y|.$$

Since y is an integer, $|y|$ is non-negative, so $\sqrt{D}$ is $17|y|$.

Therefore,

$$x = \frac{y \pm 17|y|}{24}.$$

—
Case 1: $y \geq 0$. Then $|y| = y$. So

$$x = \frac{y \pm 17y}{24}.$$

So two possibilities:

$$x = \frac{y + 17y}{24} = \frac{18y}{24} = \frac{3y}{4},$$

or

$$x = \frac{y - 17y}{24} = \frac{-16y}{24} = -\frac{2y}{3}.$$

Case 2: $y < 0$. Then $|y| = -y$. So

$$x = \frac{y \pm 17(-y)}{24}.$$

Let's compute:
First, with the plus sign:

$$\frac{y + (-17y)}{24} = \frac{-16y}{24} = -\frac{2y}{3}.$$

With the minus sign:

$$\frac{y - (-17y)}{24} = \frac{y + 17y}{24} = \frac{18y}{24} = \frac{3y}{4}.$$

So regardless of the sign of y, the solutions for x are

$$x = \frac{3y}{4} \quad \text{or} \quad x = -\frac{2y}{3}.$$

Therefore, for each y, x must be either $\frac{3y}{4}$ or $-\frac{2y}{3}$. But since x has to be an integer, this imposes conditions on y. Therefore, $3y/4$ must be integer or $-2y/3$ must be integer. Therefore:
Either $4 \mid 3y$ or $3 \mid 2y$.
But since 3 and 4 are coprime, $4 \mid 3y$ implies $4 \mid y$. Similarly, $3 \mid 2y$ implies $3 \mid y$. Therefore, for x to be integer:
Either y is divisible by 4, or y is divisible by 3. Therefore, the solutions are:
- For each y divisible by 4, $x = 3y/4$ is integer. - For each y divisible by 3, $x = -2y/3$ is integer.
If y is divisible by both 3 and 4 (i.e., by 12), then both x's would be integers.
—
Therefore, the total number of solutions would be the number of y's divisible by 4 plus the number of y's divisible by 3. However, we have to be careful not to double count the cases where y is divisible by both 3 and 4, i.e., by 12. Because those y's would be counted in both sets. Therefore, using inclusion-exclusion, total solutions would be:

$$N = (\#\{y : 4 \mid y\}) + (\#\{y : 3 \mid y\}) - (\#\{y : 12 \mid y\}).$$

—
Now we compute.
Range: $-100 \leq y \leq 100$, i.e., 201 integers.
- Number of y divisible by 4: $\lfloor 100/4 \rfloor \times 2 + 1 = 25 \times 2 + 1 = 51$. - Number of y divisible by 3: $\lfloor 100/3 \rfloor \times$

$2 + 1 = 33 \times 2 + 1 = 67$. - Number of y divisible by 12: $\lfloor 100/12 \rfloor \times 2 + 1 = 8 \times 2 + 1 = 17$.
Thus

$$N = 51 + 67 - 17 = 101.$$

—
Check special cases:
- $y = 0$. Then the equation is $12x^2 = 0 \implies x = 0$. So $(0, 0)$ is one solution. This is correctly counted once by the formula.
- $y = 4$. Then $x = 3$. Solution: $(3, 4)$ works.
- $y = 3$. Then $x = -2$. Solution: $(-2, 3)$ works.
- $y = 12$. Then $x = 9$ or $x = -8$, both valid.
All checks out.
—
Finally, also check x range: $-100 \le x \le 100$. For $|y| \le 100$, $x = \frac{3y}{4}$ or $x = -\frac{2y}{3}$ also lies in $[-100, 100]$. Verified.
—

$$\boxed{101}$$

GMTS Top (20%)

Okay, so I need to find the number of ordered pairs (x, y) where both x and y are integers between -100 and 100 inclusive, and they satisfy the equation $12x^2 - xy - 6y^2 = 0$. Hmm, let me think about how to approach this. First, maybe I can try to factor the equation or simplify it somehow. Let me write down the equation again:

$$12x^2 - xy - 6y^2 = 0.$$

Hmm, quadratic in terms of x and y. Maybe I can treat this as a quadratic equation in x or y? Let me try treating it as a quadratic in x. Let me rearrange the terms:

$$12x^2 - xy - 6y^2 = 0.$$

If I consider this as a quadratic in x, then it's of the form $ax^2 + bx + c = 0$, where:

$$a = 12, \quad b = -y, \quad c = -6y^2.$$

Then, using the quadratic formula,

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \frac{y \pm \sqrt{y^2 - 4 \cdot 12 \cdot (-6y^2)}}{24}.$$

Simplify inside the square root:

$$y^2 - 4 \cdot 12 \cdot (-6y^2) = y^2 + 288y^2 = 289y^2.$$

So

$$\sqrt{289y^2} = 17|y|.$$

Therefore,

$$x = \frac{y \pm 17|y|}{24}.$$

—
Case 1: $y \ge 0$. Then $|y| = y$. Therefore:

$$x = \frac{y \pm 17y}{24}.$$

So two possibilities:

$$x = \frac{18y}{24} = \frac{3y}{4}, \quad \text{or} \quad x = \frac{-16y}{24} = -\frac{2y}{3}.$$

Case 2: $y < 0$. Then $|y| = -y$. Therefore:

$$x = \frac{y \pm 17(-y)}{24}.$$

So:
First possibility: $y + (-17y) = -16y$, hence $x = -\frac{2y}{3}$.
Second possibility: $y - (-17y) = 18y$, hence $x = \frac{3y}{4}$.
—
So regardless of the sign of y, the solutions for x are:

$$x = \frac{3y}{4} \quad \text{or} \quad x = -\frac{2y}{3}.$$

Therefore, the solutions to the equation are all pairs (x, y) with $x, y \in [-100, 100]$ such that either $x = 3y/4$ or $x = -2y/3$, with integrality conditions.
—
Case 1: $x = 3y/4$.
For x to be integer, y must be divisible by 4. Let $y = 4k$, then $x = 3k$. Range: $-100 \le 4k \le 100 \implies -25 \le k \le 25$. That gives $25 - (-25) + 1 = 51$ solutions.
Case 2: $x = -2y/3$.
For x to be integer, y must be divisible by 3. Let $y = 3m$, then $x = -2m$. Range: $-100 \le 3m \le 100 \implies -33 \le m \le 33$. That gives $33 - (-33) + 1 = 67$ solutions.
—
Overlap check:
Suppose (x, y) satisfies both forms: $(x, y) = (3k, 4k)$ and $(x, y) = (-2m, 3m)$. Then $3k = -2m$ and $4k = 3m$. Solving gives $k = 0, m = 0$, hence $(0, 0)$. So there is exactly one overlap.
—
Final count:

$$51 + 67 - 1 = 117.$$

$$\boxed{117}$$