Do Not Step Into the Same River Twice: Learning to Reason from Trial and Error

Anonymous ACL submission

Abstract

Reinforcement learning with verifiable rewards (RLVR) has significantly boosted the reasoning capability of language models (LMs) recently. However, existing RLVR approaches merely train LMs based on their own generated on-policy responses and are constrained by the initial capability of LMs, thus prone to exploration stagnation, in which LMs fail to solve more training problems and cannot further learn from the training data. Some work tries to address this by leveraging off-policy solutions to training problems, but relies on external expert guidance that is limited in availability and scalability. In this work, we propose LTE (Learning to reason from Trial and Error), an approach that hints LMs with their previously self-made mistakes, not requiring any external expert guidance. Experiments validate the effectiveness of LTE, which outperforms the normal group relative policy optimization (GRPO) by 5.02 in Pass@1 and 9.96 in Pass@k on average across six mathematical reasoning benchmarks for Qwen3-8B-Base and even performs better than methods that require external gold solutions as guidance after aligning the experimental setup. Further analysis confirms that LTE successfully mitigates exploration stagnation and enhances both exploitation and exploration during training. Our code is available at https://anonymous.4open.science/r/Learning-from-Trial-and-Error.

1 Introduction

You cannot step into the same river twice.
—Heraclitus

Recent language models (LMs) have made significant breakthroughs in reasoning capability (DeepSeek-AI, 2025; Yang et al., 2025a; Kimi Team, 2025), leveraging long chain-of-thoughts to perform test-time computing and remarkably benefiting reasoning-intensive tasks like mathematics and coding. The awesome reasoning capability of LMs is unlocked primarily by reinforcement learning with verifiable rewards (RLVR), in which the correctness of LMs' responses can be objectively verified and thus the rewards can be automatically given. However, existing RLVR approaches optimize the LM based on its own on-policy rollouts and are constrained by the LM's initial capability, thus suffering from a limitation named *exploration stagnation* (Zhang et al., 2025a). To be specific, if the LM encounters a training sample sufficiently difficult to be beyond its capability upper bound, it fails to produce any correct solutions and thus receives no positive training signal from this sample. In this way, the LM can never solve problems that it could not initially (with reasonably many rollouts) solve and is bounded by itself.

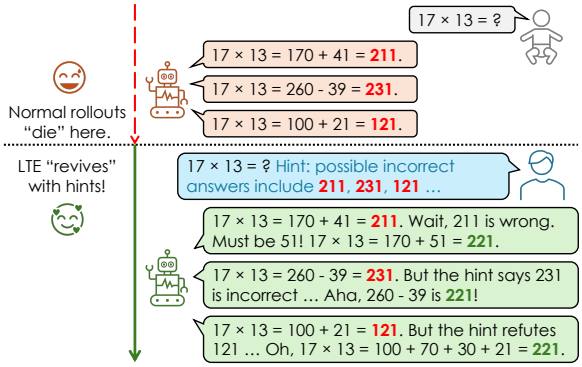


Figure 1: For difficult problems the policy model fails to solve, conventional methods simply go on with nothing gained. On the contrary, LTE tries to fix the mess with immediate hints from the model's own errors.

To address this, some work proposes to introduce external guidance to help the LM break through its upper bound (Yan et al., 2025; Ma et al., 2025; Zhang et al., 2025a,b; Lv et al., 2025). However, these either utilize human-annotated ground truth solutions that suffer from high cost and limited scalability, or require correct reasoning chains generated by stronger LMs, which can be unavailable

in certain cases like training flagship models. In other words, these methods are more like a palliative than a remedy for exploration stagnation.

In this work, we propose LTE (**L**earning to reason from **T**rial and **E**rror), a novel method that leverages the LM's own trials as a hint, without explicit external guidance [1]. For problems that no rollout passes the verification (namely *none-pass* samples), we collect the incorrect answers generated from the rollouts and perform extra rollouts with these as in-context hints, warning the LM not to step into the same river twice (*i.e.*, fall into errors it is prone to). If there are overly long responses that were truncated in the initial rollouts, we further prompt the LM to think concisely to reduce prolixity. In this way, LTE increases the chance of obtaining correct solutions and provides meaningful training signals for these stagnated samples (Figure 1). This is similar to the learning process of human. When a student is hinted with mistakes previously made, he will keep away from these and be more likely to obtain the correct solution.

Our experiments validate the effectiveness of LTE, which outperforms baseline methods across two LMs in both Pass@1 and Pass@k. For Qwen3-8B-Base, LTE outperforms its counterpart which simply performs extra rollouts by **+7.29** and **+10.04** in Pass@1 and Pass@k averaged across six mathematical benchmarks, respectively. The entropy-loss-enabled variant LTE† even exhibits higher scores (**+2.41** Pass@1 and **+2.15** Pass@k) than LUFFY (Yan et al., 2025), a mixed-policy method requiring external solutions from stronger LMs.

Analysis on the training data proves that LTE mitigates exploration stagnation by consistently reducing the number of unsolved samples while keeping more learnable *some-pass* samples. Also, LTE achieves a higher upper bound of both Pass@1 and Pass@k during training, keeps a relatively considerable level of entropy in the long tail, and encourages exploration based on test-time deep thinking by increasing the response length. These confirm LTE implicitly elicits the internal exploration capability in LMs while also enhancing exploitation.

Our contributions are as follows:

- We present LTE, which addresses exploration stagnation of RLVR using LMs' own trial and error, independent of any explicit external guidance from humans or stronger LMs.
- We empirically validate the effectiveness of LTE, which showcases the most outstanding performance among compared methods across two LMs in both Pass@1 and Pass@k.
- We confirm that LTE successfully mitigates exploration stagnation by remarkably reducing *none-pass* samples in the training dataset and elicits the internal exploration capability in LMs by keeping a relatively considerable entropy while encouraging test-time deep thinking with more tokens.

[1] In this paper, "explicit external guidance" refers to guidance of detailed reasoning traces that is less accessible than merely ground truth answers.

2 Preliminary

2.1 Reinforcement Learning with Verifiable Rewards (RLVR)

RLVR is an emergent reinforcement paradigm for the post-training of LMs, aiming to enhance the capability of LMs, especially reasoning. In the RLVR framework, an LM π_θ is trained on tasks where the correctness of an outcome can be automatically verified. Group relative policy optimization (GRPO) (Shao et al., 2024) is the *de facto* standard algorithm for RLVR due to its effectiveness and simplicity. Given an input query $q \sim \mathcal{D}$, G rollouts $\{o_1, o_2, ..., o_G\}$ are sampled from $\pi_\theta(\cdot|q)$ and rewarded by a reward function R. Then GRPO, adopting the token-level loss proposed by Yu et al. (2025), maximizes the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q,\{o_i\}}\Big[\frac{1}{Z}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\Big(\text{CLIP}(r_{i,t}(\theta), \hat{A}_{i,t}, \varepsilon) - \beta\, \mathrm{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})\Big)\Big], \tag{1}$$

where Z is the normalization factor:

$$Z = \sum_{i=1}^{G} |o_i|, \tag{2}$$

$r_{i,t}(\theta)$ is the importance sampling ratio:

$$r_{i,t}(\theta) = \frac{\pi_\theta\big(o_{i,t} \mid q,\, o_{i,<t}\big)}{\pi_{\theta_{\text{old}}}\big(o_{i,t} \mid q,\, o_{i,<t}\big)}, \tag{3}$$

$\hat{A}_{i,t}$ is the group-relative advantage:

$$\hat{A}_{i,t} = \frac{R(o_i) \;-\; \text{Mean}\big(\{R(o_j)\}_{j=1}^{G}\big)}{\text{Std}\big(\{R(o_j)\}_{j=1}^{G}\big)}, \tag{4}$$

and $\mathrm{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$ is the KL penalty term.

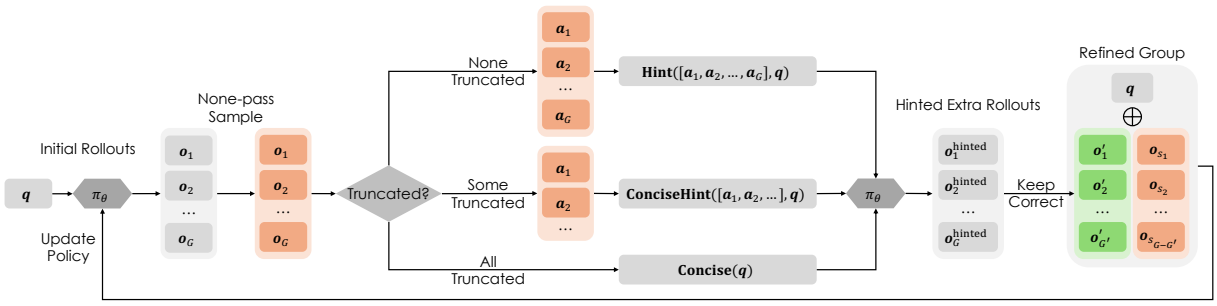


Figure 2: The framework of LTE. For *none-pass* samples, we extract the LM's generated incorrect answers as hints for the extra rollouts. We do nothing to other samples which are omitted in the figure for simplicity.

2.2 Exploration Stagnation

A major limitation of existing RLVR approaches is that all the rollouts are on-policy and the LM is constrained by its initial capability, failing to break through its own upper bound. This is known as *exploration stagnation* (Zhang et al., 2025a).

To be specific, in the typical $0/1$ reward setting, if a training problem q is too difficult for the LM to answer (*i.e.*, beyond its capability upper bound), each of the G rollouts receives a zero reward. Then all the group-relative advantages degenerate to zero, which means that the policy learns nothing from such a difficult problem.

Formally, after ignoring the clip function and KL term, the GRPO gradient can be represented as:

$$\nabla_\theta \mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q,\{o_i\}} \left[\sum_{i=1}^{G} \left(\sum_{t=1}^{|o_i|} \nabla_\theta \log \pi_\theta(o_{i,t} \mid q,\, o_{i,<t}) \cdot \hat{A}_{i,t} \right) \right]. \quad (5)$$

In *none-pass* samples, we have $R(o_i) = 0$ for all i. Thus, we get $\hat{A}_{i,t} = 0$ for all i and t. In this way, $\nabla_\theta \mathcal{J}_{\text{GRPO}} = 0$, which means the policy learns nothing from such samples.

3 LTE: Learning from Trial and Error

The framework of LTE is shown in Figure 2 and its training workflow is summarized in Algorithm 1. We provide a theoretical analysis in Appendix A demonstrating that LTE increases the chance of reaching the correct solutions.

3.1 Hinted Extra Rollouts

To address the exploration stagnation issue, a trivial approach is to perform extra rollouts (Hu et al., 2025). For example, when all the G rollouts fail to pass the problem q, another G rollouts $\{o_{G+1}, o_{G+2}, ..., o_{2G}\}$ are sampled to increase the chance of obtaining any correct rollouts with positive rewards.

Although the approach of vanilla extra rollouts is sensible, it makes use of no information from existing rollouts and can be inefficient as can be seen from the plain prompt template in Figure 3a. Instead, we propose hinted extra rollouts with guidance from existing self-generated trials, making full use of the existing rollouts.

Concretely, when all the G rollouts fail, we decide the type of hint based on the number of truncated overlong responses in the G outputs.

If the problem is *all-truncated*, in which all the responses are truncated due to length exceeding, we attribute the failure to the prolixity of responses and hint the LM to think concisely. The prompt template employed in the extra rollouts is demonstrated in Figure 3b. Note that we treat all truncated responses as they do not contain any valid answers for the completeness of responses and consideration of implementation simplicity, albeit there may still exist extractable answers in truncated responses.

If it is *some-truncated* or *none-truncated*, in which there exist responses that are not overlong, we collect the extracted answers $\{a_1, a_2, ...\}$ in these, which are all valuable possible incorrect answers that the LM is prone to. Then, we include these answers in the prompt as a hint to prevent the LM from falling into the same incorrect answers again (*i.e.*, step into the same river twice). This reduces the size of the solution space for the LM, making it easier to reach the correct solution. For *some-truncated* problems, we also hint the LM to think concisely to reduce prolixity. The prompt templates for *some-truncated* and *none-truncated* problems are demonstrated in Figure 3c and 3d, respectively, where the model is instructed not to explicitly use or mention the hint so that the reasoning chain is kept as clean as possible.

q

Let's think step by step and output the final answer within \boxed{}.

(a) The normal prompt template, namely **Prompt**$(\cdot)$.

q

Let's think **concisely** and output the final answer within \boxed{}.

(b) The prompt template for *all-truncated none-pass* samples, namely **Concise**$(\cdot)$.

q

Hint: possible incorrect answers include

a_1 a_2 a_3 ...

Do not use or mention the hint explicitly in any of your intermediate steps. Let's think **concisely** and output the final answer within \boxed{}.

(c) The prompt template for *some-truncated none-pass* samples, namely **ConciseHint**$(\cdot)$.

q

Hint: possible incorrect answers include

a_1 a_2 a_3 ...

Do not use or mention the hint explicitly in any of your intermediate steps. Let's think step by step and output the final answer within \boxed{}.

(d) The prompt template for *none-truncated none-pass* samples, namely **Hint**$(\cdot)$.

Figure 3: The normal and hinted prompt templates.

Based on the chosen prompt template, we sample G extra rollouts $\{o_1^{\text{hinted}}, o_2^{\text{hinted}}, ..., o_G^{\text{hinted}}\}$.

3.2 Mixed-policy Optimization

After the hints, if there exist G' correct solutions $\{o'_1, o'_2, ..., o'_{G'}\}$ in the extra rollouts, we randomly replace G' responses in the initial rollouts with these correct ones when updating the policy, ensuring the policy receives training signal from the initially insurmountable problem. The final responses for policy updating are denoted by $\{o'_1, o'_2, ..., o'_{G'}, o_{s_1}, o_{s_2}, ..., o_{s_{G-G'}}\}$, where s_i denotes the index of the i-th remaining initial rollout after the random replacement.

Note that since the correct solutions are indeed generated conditioned on the hinted prompt (denoted by $(\mathcal{H}_q, q)$ where $\mathcal{H}_q$ refers to the hint information for q) while the policy we want to optimize should generate solutions conditioning on the normal prompt only, they should be treated in an off-policy manner. The importance sampling for these samples is given by:

$$\hat{r}'_{i,t}(\theta) = \frac{\pi_\theta\left(o'_{i,t} \mid q,\, o_{i,<t}\right)}{\pi_{\theta_{\text{old}}}\left(o'_{i,t} \mid \mathcal{H}_q, q,\, o_{i,<t}\right)}. \tag{6}$$

Following Yan et al. (2025), we adopt regularized importance sampling for policy shaping on these samples:

$$f(\hat{r}'_{i,t}(\theta)) = \frac{\hat{r}'_{i,t}(\theta)}{\hat{r}'_{i,t}(\theta) + \gamma}, \tag{7}$$

and perform mixed-policy GRPO:

$$\begin{aligned}\mathcal{J}_{\text{Mixed}}(\theta) =& \mathbb{E}_{q,\{o'_i, o_{s_i}\}}\Big[\frac{1}{Z'}\sum_{i=1}^{G'}\sum_{t=1}^{|o'_i|}(f(\hat{r}'_{i,t}(\theta)) \cdot \hat{A}'_{i,t}) \\ &+ \frac{1}{Z}\sum_{i=1}^{G-G'}\sum_{t=1}^{|o_{s_i}|}\Big(\text{CLIP}(r_{s_i,t}(\theta), \hat{A}_{s_i,t}, \varepsilon)\Big)\Big],\end{aligned} \tag{8}$$

where the normalization factors Z' and Z follow the definition in Equation 2, and the KL term is omitted in the formula for simplicity.

In line with Yan et al. (2025), we set $\gamma = 0.1$. For computational efficiency and simplicity, we treat the old policy term $\pi_{\theta_{\text{old}}}\left(o'_{i,t} \mid \mathcal{H}_q, q,\, o_{i,<t}\right)$ in the importance sampling ratio as 1.

Algorithm 1 LTE: Learning from Trial and Error

Require: Policy LM π_θ, number of rollouts G, batch size n, number of training steps T, training data $\mathcal{D}$
Ensure: Updated policy LM π_θ

```
for t = 1 to T do
    Q ← {q_i ~ D}_{i=1}^n                                   ▷ training batch
    O ← ∅; Â ← ∅                                            ▷ outputs and advantages
    for q ∈ Q do
        O_q ← {o_i ~ π_θ(· | q)}_{i=1}^G                    ▷ response sampling
        A_q ← {EXTRACT(o_i)}_{i=1}^G                        ▷ answer extraction
        R_q ← {EVAL(A_q^(i), q)}_{i=1}^G                    ▷ evaluation
        if ∀r ∈ R_q, r = 0 then                             ▷ none-pass sample
            if ∀o ∈ O_q, TRUNCATED(o) = 1 then
                q' ← CONCISE(q)                             ▷ all-truncated
            else if ∀o ∈ O_q, TRUNCATED(o) = 0 then
                q' ← CONCISEHINT(q, A_q)                    ▷ some-truncated
            else
                q' ← HINT(q, A_q)                           ▷ none-truncated
            end if
            O_q^hinted ← {o_i ~ π_θ(· | q')}_{i=1}^G        ▷ hinted rollouts
            A_q^hinted ← {EXTRACT(o_i)}_{i=1}^G             ▷ answer extraction
            R_q^hinted ← {EVAL(A_q^(i), q)}_{i=1}^G         ▷ evaluation
            O_q^* ← {o_i | R_q^(i) = 1}_{i=1}^G             ▷ correct outputs
            R_q^* ← {R_q^(i) | R_q^(i) = 1}_{i=1}^G         ▷ corresponding rewards
            O_q, R_q ← REPLACE(O_q, R_q, O_q^*, R_q^*)      ▷ replacement
        end if
        O ← O ∪ {O_q}                                       ▷ append current outputs
        Â ← Â ∪ {ADVANTAGE(R_q)}                            ▷ append current advantages
    end for
    π_θ ← MIXEDPOLICYUPDATE(π_θ, Q, O, Â)                   ▷ update policy
end for
return π_θ
```

4 Experimental Setup

4.1 Language Models

We experiment with Qwen3-4B-Base and Qwen3-8B-Base (Yang et al., 2025a). We do not use the Qwen2.5 herd of models (Yang et al., 2025b) because they may undergo a benchmark contamination issue (Wu et al., 2025). We do not include

Method	EC	MATH-500	Minerva	Olympiad	AMC'23	AIME'24	AIME'25	Avg.
QWEN3-4B-BASE	-	45.40	19.49	22.81	35.31	8.75	3.75	22.59
GRPO	-	70.45	**34.28**	35.74	47.03	11.88	9.17	34.76
$\text{GRPO}_{\text{Extra}}$	-	71.00	31.43	35.11	54.22	13.33	13.75	36.47
EvoCoT	-	67.05	30.06	31.33	49.84	9.79	8.75	32.80
LTE	-	**74.55**	34.19	**39.78**	**56.72**	**18.33**	**14.17**	**39.62**
ReLIFT	EL	51.20	27.30	22.11	35.31	7.29	5.00	24.70
DAPO	CH	76.10	35.29	42.96	67.34	22.08	**22.92**	44.45
LUFFY	EL	76.85	32.26	40.89	64.53	**25.62**	18.33	43.08
LTE†	EL	**77.25**	**37.04**	**44.15**	**69.38**	**25.62**	22.50	**45.99**
QWEN3-8B-BASE	-	60.85	26.10	27.41	47.34	11.46	9.17	30.39
GRPO	-	77.70	37.32	42.07	68.91	22.08	15.83	43.99
$\text{GRPO}_{\text{Extra}}$	-	75.75	36.95	40.07	60.47	22.29	14.79	41.72
EvoCoT	-	70.45	33.09	36.37	56.41	16.46	14.79	37.93
LTE	-	**78.85**	**37.87**	**45.52**	**73.91**	**30.62**	**27.29**	**49.01**
ReLIFT	EL	68.15	33.73	32.89	55.62	15.21	10.21	35.97
DAPO	CH	76.85	**38.97**	41.81	70.00	23.54	21.25	45.40
LUFFY	EL	**80.20**	36.76	44.74	70.78	29.17	21.25	47.15
LTE†	EL	79.80	38.79	**47.59**	**74.53**	**30.83**	**25.83**	**49.56**

Table 1: Pass@1 results, with best results **bolded**. "EC", "EL" and "CH" refers to entropy control, entropy loss and clip-higher, respectively. "-" means the method does not explicitly control the policy entropy.

the Llama herd of models (Llama Team, 2024) because they show significant instability in reinforcement training and even exhibit a degenerated performance after training in our early experiments.

4.2 Training

We focus on mathematical reasoning and adopt OpenR1-Math-46k-8192 (Yan et al., 2025) as our training dataset, which contains 45,792 instances. We adopt `verl` (Sheng et al., 2025) as the training framework. We sample 8 rollouts per prompt and set the temperature as 1.0. The maximum response length is 8,192 following Yan et al. (2025). We conduct training for 500 steps, with a batch size of 128. More details are in Appendix B.

4.3 Evaluation

We focus on six widely used mathematics benchmarks including MATH-500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), AMC'23, AIME'24 and AIME'25. For MATH-500, Minerva and OlympiadBench, we sample 4 times and report Mean@4 (for Pass@1) and Pass@4. For AMC'23, AIME'24 and AIME'25, we sample 16 times and report Mean@16 (for Pass@1) and Pass@16.

Note that the hints of LTE are only applied in the training stage. In evaluation, we strictly follow the standard protocol. The context length is set as 32,768 (*i.e.*, the default maximum context length of the Qwen3 series of models) for evaluation to minimize the interference caused by truncated overlong responses. We set temperature as 0.6, top-p as 0.95 and top-k as 20 following Yang et al. (2025a).

For both training and evaluation, we verify the correctness of answers with `Math-Verify` [2].

4.4 Baselines and Methods

Our evaluated baselines and methods include: (1) **Base Model**: the performance of the base LM; (2) **GRPO**: the normal baseline of GRPO; (3) **$\text{GRPO}_{\text{Extra}}$**: simply performing vanilla extra rollouts for *none-pass* samples without hints. (4) **EvoCoT** (Liu et al., 2025a): adopting the ground truth answer as a hint to overcome the exploration bottleneck in reinforcement learning. (5) **ReLIFT** (Ma et al., 2025): alternating between reinforcement learning and supervised finetuning to inject off-policy reasoning traces for difficult problems. (6) **DAPO** (Yu et al., 2025): an advanced policy optimization algorithm with several techniques like clip-higher, dynamic sampling and overlong reward shaping. (7) **LUFFY** (Yan et al., 2025): mixed-policy optimization with the guidance from reasoning traces generated by stronger models. (8) **LTE**: our proposed method. (9) **LTE†**: our proposed LTE with entropy loss enabled, aiming to ensure a fairer comparison with methods involving entropy control (*e.g.*, ReLIFT, DAPO and LUFFY) and study how LTE performs with better exploration

[2] https://github.com/huggingface/Math-Verify

Method	EC	MATH-500	Minerva	Olympiad	AMC'23	AIME'24	AIME'25	Avg.
QWEN3-4B-BASE	-	69.80	37.87	39.70	82.50	33.33	26.67	48.31
GRPO	-	76.40	40.81	43.85	**85.00**	**33.33**	23.33	50.45
$\text{GRPO}_{\text{Extra}}$	-	78.20	40.81	42.67	82.50	26.67	33.33	50.70
EvoCoT	-	77.60	39.34	40.44	82.50	26.67	30.00	49.43
LTE	-	**80.60**	**41.91**	**49.48**	**85.00**	30.00	**46.67**	**55.61**
ReLIFT	EL	76.60	42.28	39.26	85.00	30.00	26.67	49.97
DAPO	CH	82.40	44.85	52.00	92.50	46.67	40.00	59.74
LUFFY	EL	**83.20**	42.28	50.22	**95.00**	**50.00**	40.00	60.12
LTE†	EL	82.40	**46.69**	**53.48**	90.00	**50.00**	**46.67**	**61.54**
QWEN3-8B-BASE	-	77.80	43.38	43.56	82.50	40.00	30.00	52.87
GRPO	-	82.40	44.49	50.67	90.00	43.33	30.00	56.82
$\text{GRPO}_{\text{Extra}}$	-	82.00	43.75	48.00	90.00	46.67	30.00	56.74
EvoCoT	-	76.40	40.81	46.22	87.50	43.33	23.33	52.93
LTE	-	**85.20**	**47.06**	**54.22**	**97.50**	**70.00**	**46.67**	**66.78**
ReLIFT	EL	79.40	47.43	45.63	87.50	50.00	33.33	57.22
DAPO	CH	82.80	**47.79**	50.37	92.50	53.33	36.67	60.58
LUFFY	EL	**85.60**	47.43	54.52	95.00	60.00	50.00	65.43
LTE†	EL	84.60	**47.79**	**55.56**	**97.50**	**66.67**	**53.33**	**67.58**

Table 2: Pass@k results, with best results **bolded**. "EC", "EL" and "CH" refers to entropy control, entropy loss and clip-higher, respectively. "-" means the method does not explicitly control the policy entropy.

promoted by entropy control.

5 Results and Analysis

5.1 Main Results

Table 1 presents the Pass@1 results. When trained without an entropy loss, LTE outperforms GRPO and $\text{GRPO}_{\text{Extra}}$ with **+4.94** and **+5.22** average improvements on the two LMs across the six benchmarks. This confirms that with the hint based on the LM's own trial and error, LTE brings effective guidance in the extra rollouts and successfully yields a superior final policy after the training process. On the contrary, merely performing vanilla extra rollouts, which makes no use of existing experience of trial and error, only brings marginal (even negative) improvement to GRPO, demonstrating the limitations of simple rollout scaling.

With the help of entropy loss, which brings better exploration, the power of LTE† is further amplified, which exhibits the highest average scores across the six benchmarks among all the compared methods. It is noteworthy that LTE†, without any explicit external guidance, even performs better than LUFFY, which requires reasoning traces from stronger models, with an average gain of **+2.66** points.

Table 2 presents the Pass@k results which focus on the exploration upper bound (to a certain degree). Compared with the base model, GRPO and $\text{GRPO}_{\text{Extra}}$ improve Pass@k relatively marginally. For instance, the latter brings improvements of **+2.39** and **+3.87** for the two base models, respectively. On the contrary, LTE significantly enhances exploration with **+7.30** and **+13.91** higher Pass@k scores for the two LMs, which are **3.1×** and **3.6×** of the improvements provided by $\text{GRPO}_{\text{Extra}}$. Furthermore, LTE† unlocks a higher exploration upper bound, with **+13.23** and **+14.71** improvements over the two base models, outperforming all the compared baselines including DAPO (by **+4.40** on average) and LUFFY (by **+1.79** on average) which either explicitly encourage exploration or utilize guidance of external gold reasoning traces.

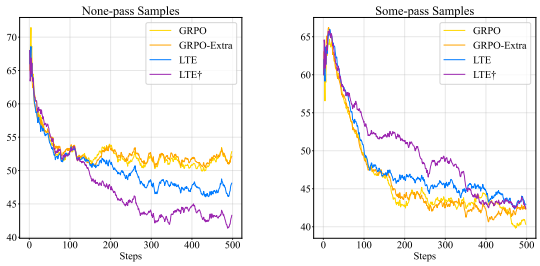
Note that we run the exact training scripts provided by Ma et al. (2025) and Liu et al. (2025a) but get relatively poor results for ReLIFT and EvoCoT. We attribute these to the issue of compatibility of foundation models and leave the investigation of the underlying causes for future work.

Combining the Pass@1 and Pass@k results, we conclude that, with the guidance from the LM's own behavior only, LTE encourages both exploitation and exploration during the RLVR process and unlocks higher performance without any explicit external expert guidance.

5.2 Analysis on Different Training Samples

We analyze the number of *none-pass* and *some-pass* samples of the training data to study whether and how exploration stagnation is addressed.

In Figure 4a, the normal GRPO becomes stuck



(a) The number of *none-pass* samples in the initial rollouts. (b) The number of *some-pass* samples in the initial rollouts.

Figure 4: Performance of Qwen3-4B-Base on the training data, smoothed with exponential moving average.

in exploration stagnation after about 150 training steps, in which the policy model cannot further solve a larger number of *none-pass* samples any more. $\text{GRPO}_{\text{Extra}}$ also fails to break the spell. On the contrary, LTE and LTE† consistently keep the LM solving more *none-pass* samples during training, significantly reducing the number of unsolved problems even in the second half of training. This directly proves that LTE is successful in addressing the exploration stagnation issue in RLVR without any explicit external expert guidance.

Figure 4b presents the number of *some-pass* samples. During training, LTE and LTE† keep a relatively higher level of number of *some-pass* samples, which are non-zero-gradient groups that primarily bring learning signal. This indicates that LTE keeps a higher level of learnability (Bae et al., 2025) during training, with more moderately uncertain *some-pass* samples benefiting the learning.

The above observations prove that, only with hints from the LM's own experience, LTE successfully mitigates the issue of exploration stagnation and keeps a high learnability level during training.

5.3 Analysis on the Training Dynamics

In Figure 5a, the Pass@1 on the validation data of GRPO and $\text{GRPO}_{\text{Extra}}$ get stagnated after 100 training steps with no further improvement. LTE and LTE†, however, continue to increase the validation score and converge to a higher level. In Figure 5b, the Pass@4 scores of the two baselines even slightly shrink after 300 steps as the policy becomes over-certain. On the contrary, LTE and LTE† keep unlocking the upper bound and achieving higher Pass@4. This further validates LTE's advantage in both exploitation and exploration.

As shown in Figure 5c, although all the three methods suffer from a trend of entropy collapse due to explicit entropy control, the entropy of LTE remains relatively high throughout the long tail after 100 steps, indicating that it helps the LM keep a certain level of uncertainty and exploration.

From the perspective of response length presented in Figure 5d, we observe a noticeable trend. While GRPO and $\text{GRPO}_{\text{Extra}}$ exhibit a slow increase, LTE and LTE†, especially the latter, raise the LM's response length more significantly. This demonstrates that LTE implicitly encourages test-time deep thinking (DeepSeek-AI, 2025) and drives the LM to spend more tokens on exploration, thus shaping a much more explorative policy.

The above analysis confirms that LTE effectively elicits the internal exploration capability in LMs while keeping an excellent level of exploitation.

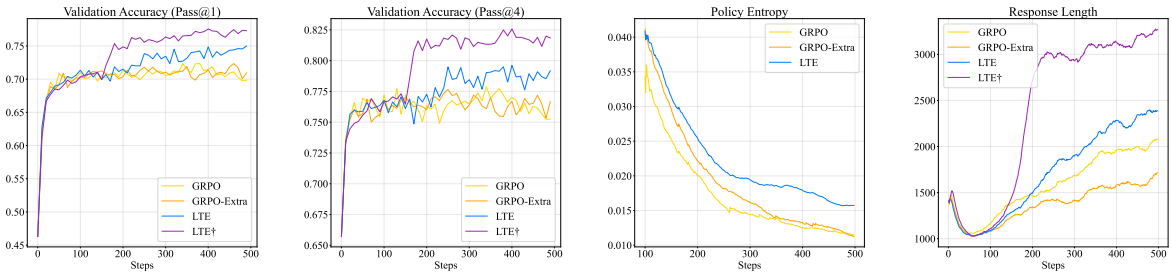
5.4 Ablation Study

We assess the indispensability of the types of hints in LTE by ablating the source of hints (*i.e.*, incorrect answer and overlong response) with Qwen3-8B-Base. As shown in Table 3, both types of hints contribute to the outstanding performance of LTE, especially the hint of incorrect answer, which leads to a performance decline of **4.54** and **4.30** on average for Pass@1 and Pass@k when ablated. The hint of overlong response is also essential with an averaged contribution of **1.82** and **1.71** for Pass@1 and Pass@k, respectively. Finally, when both types are ablated (*i.e.*, $\text{GRPO}_{\text{Extra}}$), the trained policy suffers from a drastic degeneration (**7.29** for Pass@1 and **10.04** for Pass@k).

6 Related Work

6.1 Learning for Better Exploration

Policy entropy is regarded as a metric indicating the exploration capability of LMs (Cui et al., 2025). Some work focuses on preventing entropy collapse to enhance exploration. Cui et al. (2025) propose `Clip-Cov` and `KL-Cov` to clip and apply KL penalty to high-covariance tokens. Yu et al. (2025) propose `Clip-Higher` to promote exploratory tokens. He et al. (2025) employ an adaptive entropy loss coefficient. Cheng et al. (2025) add an entropy term in the advantage. Meanwhile, some work does not directly control the entropy. Dou et al. (2025) use historical replay to resume LMs' exploration capability. Chen et al. (2025b) propose Pass@k as the reward for training to balance exploration and exploitation. Song et al. (2025) promote historical and batch diversity to encourage exploration.



(a) MATH-500 Pass@1. (b) MATH-500 Pass@4. (c) Policy entropy. (d) Response length.

Figure 5: Qwen3-4B-Base training dynamics, with entropy and length smoothed by exponential moving average.

Setting	MATH-500	Minerva	Olympiad	AMC'23	AIME'24	AIME'25	Avg.
LTE	78.85/**85.20**	**37.87/47.06**	**45.52/54.22**	**73.91/97.50**	**30.62/70.00**	**27.29/46.67**	**49.01/66.78**
w/o Incorrect	76.95/83.20	36.67/44.85	41.59/51.85	69.53/95.00	25.00/60.00	17.08/40.00	44.47/62.48
w/o Overlong	**79.75**/85.00	37.22/46.32	45.30/54.07	70.00/95.00	27.71/63.33	23.13/**46.67**	47.19/65.07
w/o Either ($\text{GRPO}_{\text{Extra}}$)	75.75/82.00	36.95/43.75	40.07/48.00	60.47/90.00	22.29/46.67	14.79/30.00	41.72/56.74

Table 3: Ablation results (Pass@1/Pass@k) of Qwen3-8B-Base. Best results are **bolded**.

6.2 Learning from Guidance

Some work enhances RLVR using external reasoning traces which are either human-annotated or from stronger LMs by means of hybrid training (Yan et al., 2025; Ma et al., 2025; Zhang et al., 2025b; Lv et al., 2025) or hinting (Zhang et al., 2025a; Chen et al., 2025a), which suffer from limited accessibility and scalability as afore discussed. Liu et al. (2025a) propose EvoCoT, a guidance-based method that leverages only the ground truth label of training data as an explicit hint. LTE is different in that its hints consist of the LM's own trails instead of the ground truth, which fully exploits the LMs' previous experience and is more reward-hacking-free. Yang et al. (2025c) improve RLVR performance via the information from LMs' self-verification, which is another kind of self-guidance.

6.3 Learning with More Rollouts

Hu et al. (2025) perform extremely extended rollouts to break the exploration bottleneck. Li et al. (2025) and Zhang et al. (2025c) both adopt an adaptive rollout strategy to assign more rollouts to difficult prompts, thus achieving training efficiency and exploration breakthrough at the same time. LTE is different from these works in that it leverages the information from trial and error.

6.4 Learning from Failures

Some work also employs the idea of learning from failures but are essentially different from ours. Thakur et al. (2024) include in-context feedback from the execution to perform multi-turn theorem-proving. Alazraki et al. (2025) include off-line mistakes generated by other LMs as in-context demonstrations to improve training-free reasoning. Chung et al. (2025) conducts an iterative manner of evaluation, verifying LMs' trial at test time and requiring to try again if it is incorrect, whose evaluation is indeed a variant of Pass@k. Similarly, Liu et al. (2025b) focuses on multi-turn reasoning, in which the user says "try again" when the LM makes a mistake at test time. This setting assumes the user knows the ground truth answer which can be impractical in reality while LTE does not require any test-time verification.

7 Conclusion

We have presented LTE, a novel approach that aims to address the exploration stagnation of RLVR without explicit external expert guidance. It hints LMs with its previously generated incorrect answers to help better solve *none-pass* samples. Experiments validate the effectiveness of LTE and further analysis confirms that LTE successfully mitigates exploration stagnation and elicits the internal exploration capability in LMs. There remain several avenues for future research including extending LTE to a broader set of tasks besides mathematics, applying hints at more fine-grained intervals, and incorporating the correct ground truth into the hint while avoiding reward hacking.

Limitations

First, the current LTE method only fits mathematical reasoning due to the concise format of answers to math problems and requires further modification to support other tasks like coding. Second, due to limited time, we only train the LMs with a relatively short maximum response length (*i.e.*, 8,192), which may not fully unlock the reasoning capability of the LMs. Third, due to limited computational resources, we have not experimented with larger-scale language models to study the performance of LTE on stronger policies.

Ethical Considerations

Use of AI Assistants The algorithmic design and core methodology of this work were derived through manual research and human reasoning. For coding, we work with the assistance of GitHub Copilot [3]. We certify that the substance of our code is our own work.

Computational Budget All our experiments are conducted on a machine with CentOS 8, 384 AMD® EPYC™ 9K84 96-Core Processor CPUs and 2.2TiB memory. We use 8× NVIDIA H20 GPUs for all the experiments. The training of LTE/LTE† takes around 6/7.5 and 9/9 days for the 4B and 8B models, respectively.

Reproducibility Our work is reproducible because we have provided our source code and implementation details.

Potential Risks To the best of our knowledge, there are no potential risks concerning our work.

Artifact	License
Qwen3 Models	Apache-2.0
verl	Apache-2.0
Math-Verify	Apache-2.0
Elliott/Openr1-Math-46k-8192	MIT
MATH-500	MIT
Minerva	MIT
OlympiadBench	MIT
AMC'23	N/A
AIME'24	N/A
AIME'25	N/A
EvoCoT	Apache-2.0
ReLIFT	N/A
DAPO	Apache-2.0
LUFFY	N/A

Table 4: Licenses of scientific artifacts we use.

[3] https://github.com/features/copilot

Scientific Artifacts We cite all the creators of scientific artifacts we use in this paper. Licenses of these scientific artifacts are shown in Table 4. Our use of these artifacts is consistent with their intended use.

References

Lisa Alazraki, Maximilian Mozes, Jon Ander Campos, Tan Yi-Chern, Marek Rei, and Max Bartolo. 2025. No need for explanations: LLMs can implicitly learn from mistakes in-context. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*.

Sanghwan Bae, Jiwoo Hong, Min Young Lee, Hanbyul Kim, JeongYeon Nam, and Donghyun Kwak. 2025. Online difficulty filtering for reasoning oriented reinforcement learning. *Preprint*, arXiv:2504.03380.

Justin Chih-Yao Chen, Becky Xiangyu Peng, Prafulla Kumar Choubey, Kung-Hsiang Huang, Jiaxin Zhang, Mohit Bansal, and Chien-Sheng Wu. 2025a. Nudging the boundaries of llm reasoning. *Preprint*, arXiv:2509.25666.

Zhipeng Chen, Xiaobo Qin, Youbin Wu, Yue Ling, Qinghao Ye, Wayne Xin Zhao, and Guang Shi. 2025b. Pass@k training for adaptively balancing exploration and exploitation of large reasoning models. *Preprint*, arXiv:2508.10751.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. 2025. Reasoning with exploration: An entropy perspective on reinforcement learning for llms. *Preprint*, arXiv:2506.14758.

Stephen Chung, Wenyu Du, and Jie Fu. 2025. Learning from failures in multi-attempt reinforcement learning. *Preprint*, arXiv:2503.04808.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. 2025. The entropy mechanism of reinforcement learning for reasoning language models. *Preprint*, arXiv:2505.22617.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Shihan Dou, Muling Wu, Jingwen Xu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. 2025. Improving rl exploration for llm reasoning through retrospective replay. *Preprint*, arXiv:2504.14363.

Zeyu Gan, Hao Yi, and Yong Liu. 2025. Cot-space: A theoretical framework for internal slow-thinking via reinforcement learning. *Preprint*, arXiv:2509.04027.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. 2025. Skywork open reasoner 1 technical report. *Preprint*, arXiv:2505.22312.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Jian Hu, Mingjie Liu, Ximing Lu, Fang Wu, Zaid Harchaoui, Shizhe Diao, Yejin Choi, Pavlo Molchanov, Jun Yang, Jan Kautz, and Yi Dong. 2025. Brorl: Scaling reinforcement learning via broadened exploration. *Preprint*, arXiv:2510.01180.

Kimi Team. 2025. Kimi k2: Open agentic intelligence. *Preprint*, arXiv:2507.20534.

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

Ziniu Li, Congliang Chen, Tianyun Yang, Tian Ding, Ruoyu Sun, Ge Zhang, Wenhao Huang, and Zhi-Quan Luo. 2025. Knapsack rl: Unlocking exploration of llms via optimizing budget allocation. *Preprint*, arXiv:2509.25849.

Huanyu Liu, Jia Li, Chang Yu, Taozhi Chen, Yihong Dong, Lecheng Wang, XiaoLong Hu, and Ge Li. 2025a. Evocot: Overcoming the exploration bottleneck in reinforcement learning. *Preprint*, arXiv:2508.07809.

Licheng Liu, Zihan Wang, Linjie Li, Chenwei Xu, Yiping Lu, Han Liu, Avirup Sil, and Manling Li. 2025b. A simple "try again" can elicit multi-turn llm reasoning. *Preprint*, arXiv:2507.14295.

Llama Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Xingtai Lv, Yuxin Zuo, Youbang Sun, Hongyi Liu, Yuntian Wei, Zhekai Chen, Lixuan He, Xuekai Zhu, Kaiyan Zhang, Bingning Wang, Ning Ding, and Bowen Zhou. 2025. Towards a unified view of large language model post-training. *Preprint*, arXiv:2509.04419.

Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Bin Cui, and Wentao Zhang. 2025. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *Preprint*, arXiv:2506.07527.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*.

Yuda Song, Julia Kempe, and Remi Munos. 2025. Outcome-based exploration for llm reasoning. *Preprint*, arXiv:2509.06941.

Amitayush Thakur, George Tsoukalas, Yeming Wen, Jimmy Xin, and Swarat Chaudhuri. 2024. An in-context learning agent for formal theorem-proving. In *First Conference on Language Modeling*.

Mingqi Wu, Zhihao Zhang, Qiaole Dong, Zhiheng Xi, Jun Zhao, Senjie Jin, Xiaoran Fan, Yuhao Zhou, Huijie Lv, Ming Zhang, Yanwei Fu, Qin Liu, Songyang Zhang, and Qi Zhang. 2025. Reasoning or memorization? unreliable results of reinforcement learning due to data contamination. *Preprint*, arXiv:2507.10532.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2025. Learning to reason under off-policy guidance. *Preprint*, arXiv:2504.14945.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025b. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Wenkai Yang, Weijie Liu, Ruobing Xie, Yiju Guo, Lulu Wu, Saiyong Yang, and Yankai Lin. 2025c. Laser: Reinforcement learning with last-token self-rewarding. *Preprint*, arXiv:2510.14943.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin,

Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. 2025a. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *Preprint*, arXiv:2507.02841.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. 2025b. On-policy rl meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. *Preprint*, arXiv:2508.11408.

Yuheng Zhang, Wenlin Yao, Changlong Yu, Yao Liu, Qingyu Yin, Bing Yin, Hyokun Yun, and Lihong Li. 2025c. Improving sampling efficiency in rlvr through adaptive rollout and response reuse. *Preprint*, arXiv:2509.25808.

A Theoretical Analysis

A.1 Definitions

We follow the theoretical framework of CoT-Space (Gan et al., 2025), which reformulates the reasoning process of LMs as optimization in a continuous semantic manifold, to conduct theoretical analysis on LTE.

Definition A.1 (Reasoning State Space) *Given a query q, $\mathcal{S}_q$ is the reasoning state space consisting of all possible intermediate states $s = (q, \xi)$, where $\xi = (\xi_1, \xi_2, ..., \xi_t)$ represents a partial reasoning chain with t conceptual steps realized through multiple token generations.*

Definition A.2 (Minimums in Reasoning Space) *Given a query q and its grount truth answer o^*, assuming Ξ_q is the set of all reasonable intermediate reasoning process for query q, the minimums of the reasoning space are defined as $\mathcal{M}_q = \{(q, \xi, o^*) \mid \xi \in \Xi_q\}$.*

Definition A.3 (Failure and Pruned Subspace) *Given a query q with the incorrect answers $\mathcal{A}_q$ generated by the LM (for simplicity, we treat the answer of a truncated response as $\emptyset$), the failure subspace is $\mathcal{S}_q^f = \{s \in \mathcal{S}_q : Extract(s) \in \mathcal{A}_q\}$, where $Extract(\cdot)$ is an abstract function extracting the final answer from s. Then, the pruned subspace is $\mathcal{S}_q^{'} = \mathcal{S}_q \setminus \mathcal{S}_q^f$.*

A.2 State Space Pruning

In this section, we theoretically demonstrate that LTE achieves state space pruning with its hint of previous incorrect answers.

Starting from the law of total probability, we have

$$\begin{aligned} &P(s \in \mathcal{M}_q) \\ &= P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}) \cdot P(s \in \mathcal{S}_q^{'}) \\ &\quad + P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^f) \cdot P(s \in \mathcal{S}_q^f). \end{aligned} \tag{9}$$

Since a correct answer cannot be in $\mathcal{S}_q^f$, we have

$$P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^f) = 0. \tag{10}$$

Therefore, Equation 9 can be simplified to:

$$P(s \in \mathcal{M}_q) = P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}) \cdot P(s \in \mathcal{S}_q^{'}) \tag{11}$$

The probability of reaching a correct answer for π_θ given q is:

$$\begin{aligned} &P(s \in \mathcal{M}_q | q, \pi_\theta) \\ &= P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}, q, \pi_\theta) \cdot P(s \in \mathcal{S}_q^{'}, q, \pi_\theta). \end{aligned} \tag{12}$$

When augmented with the hint information $\mathcal{H}_q$ containing $\mathcal{A}_q$ and the instruction of thinking concisely if q is *some-truncated* or *all-truncated*, it becomes:

$$\begin{aligned} &P(s \in \mathcal{M}_q | q, \mathcal{H}_q, \pi_\theta) \\ &= P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}, q, \mathcal{H}_q, \pi_\theta) \cdot P(s \in \mathcal{S}_q^{'}, q, \mathcal{H}_q, \pi_\theta). \end{aligned} \tag{13}$$

The ratio of the above probabilities is:

$$\begin{aligned} &\frac{P(s \in \mathcal{M}_q | q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | q, \pi_\theta)} \\ &= \frac{P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}, q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}, q, \pi_\theta)} \cdot \frac{P(s \in \mathcal{S}_q^{'}, q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{S}_q^{'}, q, \pi_\theta)}. \end{aligned} \tag{14}$$

Based on the instruction following capability of LMs, when given the hint $\mathcal{H}_q$, we may assume that the probability of the policy π_θ reaching a state s in the failure subspace $\mathcal{S}_q^f$ decreases:

$$P(s \in \mathcal{S}_q^f | q, \mathcal{H}_q, \pi_\theta) = P(s \in \mathcal{S}_q^f | q, \pi_\theta) - \delta, \tag{15}$$

where $\delta > 0$. Meanwhile, the probability of a space in the pruned subspace increases:

$$P(s \in \mathcal{S}_q^{'} | q, \mathcal{H}_q, \pi_\theta) = P(s \in \mathcal{S}_q^{'} | q, \pi_\theta) + \delta. \tag{16}$$

Let α denote $\frac{P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}, q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | s \in \mathcal{S}_q^{'}, q, \pi_\theta)}$, which indicates the ratio of the probability of reaching states in $\mathcal{M}_q$ when the policy reaches a state in $\mathcal{S}_q^{'}$ with

or without $\mathcal{H}_q$. Then Equation 14 becomes:

$$\begin{aligned}
&\frac{P(s \in \mathcal{M}_q | q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | q, \pi_\theta)} \\
&= \alpha \cdot \frac{P(s \in \mathcal{S}_q^{'}, q, \pi_\theta) + \delta}{P(s \in \mathcal{S}_q^{'}, q, \pi_\theta)} \\
&= \alpha \cdot \left(1 + \frac{\delta}{P(s \in \mathcal{S}_q^{'}, q, \pi_\theta)}\right).
\end{aligned} \tag{17}$$

For a *none-pass* query q, all the n initial rollouts lies in the failure subspace $\mathcal{S}_q^f$ according to the definitions. This allows us set a confidence level τ (where $0 < \tau < 1$) indicating the probability of observing n consecutive failures:

$$P(s \in \mathcal{S}_q^f, q, \pi_\theta)^n \geq \tau. \tag{18}$$

According to the definitions of $\mathcal{S}_q^f$ and $\mathcal{S}_q^{'}$, we have $P(s \in \mathcal{S}_q^f, q, \pi_\theta) + P(s \in \mathcal{S}_q^{'}, q, \pi_\theta) = 1$. Thus

$$P(s \in \mathcal{S}_q^{'}, q, \pi_\theta) \leq 1 - \tau^{1/n}. \tag{19}$$

Combining with Equation 17, we have

$$\frac{P(s \in \mathcal{M}_q | q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | q, \pi_\theta)} \geq \alpha \cdot \left(1 + \frac{\delta}{1 - \tau^{1/n}}\right). \tag{20}$$

Since the hint $\mathcal{H}_q$ should have little negative effect on the policy's reasoning capability in the pruned state space $\mathcal{S}_q^{'}$, we can reasonably assume that the possibility of $s \in \mathcal{M}_q$ when $s \in \mathcal{S}_q^{'}$ does not significantly decrease when it is conditioned on $\mathcal{H}_q$. This indicates that α should not be much less than 1 (*i.e.*, we may at least assume $\alpha > \Omega(\frac{1}{n})$). According to Equation 20, if $\frac{P(s \in \mathcal{M}_q | q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | q, \pi_\theta)} \leq 1$, then $\alpha \leq 1/(1 + \frac{\delta}{1 - \tau^{1/n}})$. Using the approximation $\tau^{1/n} = e^{\frac{\ln \tau}{n}} \approx 1 + \frac{\ln \tau}{n}$, we approximately have $\alpha \leq 1/(1 + \frac{\delta}{|\ln \tau|} \cdot n) \sim \Theta(\frac{1}{n})$, which contradicts to our assumption. Therefore, we have

$$\frac{P(s \in \mathcal{M}_q | q, \mathcal{H}_q, \pi_\theta)}{P(s \in \mathcal{M}_q | q, \pi_\theta)} > 1. \tag{21}$$

This theoretically demonstrates that with the help of hint based on self-generated incorrect answers, LTE increases the possibility of reaching the correct answer in the extra rollouts, thus increasing the chance of the policy receiving meaningful training signals from previously *none-pass* samples.

Hyper-parameter	**Value**
Learning Rate	1e-6
Totel Steps	500
Batch Size	128
Mini Batch Size	32
Micro Batch Size	1.0
KL Loss Coefficient	0.001
Clip Ratio	0.2
Temperature	1
Number of Rollouts	8
Maximum Prompt Length	2048
Maximum Response Length	8192

Table 5: Full hyper-parameters for training.

B Implementation Details

The full hyper-parameters used in our training are listed in Table 5. We set the coefficient of the entropy loss as 0.003 for both LUFFY and LTE†. For all the methods, we evaluate their final checkpoints after 500 steps.