Outcome-based Exploration for LLM Reasoning

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Reinforcement learning (RL) has become a powerful tool for improving the reasoning ability of large language models (LLMs). While outcome-based RL, which rewards models solely on the correctness of the final answer, achieves strong accuracy gains, it also causes a systematic loss of diversity in generations. This collapse undermines real-world performance, where diversity is essential for test-time scaling. We analyze this phenomenon by viewing RL post-training as a sampling process and uncover two key properties: (i) transfer of diversity degradation, where reduced diversity on solved problems propagates to unsolved ones, and (ii) tractability of the outcome space, since reasoning tasks admit only a limited set of distinct answers. Motivated by these insights, we propose outcome-based exploration, which assigns exploration bonuses based only on final outcomes. We introduce two complementary algorithms: historical exploration, which rewards rarely observed answers via UCB-style bonuses, and batch exploration, which penalizes repetition within a batch to promote test-time diversity. Experiments across multiple models and datasets show that both methods improve accuracy while mitigating diversity collapse. Together, they offer a practical path toward RL methods that enhance LLM reasoning without sacrificing the diversity critical for scalable deployment.

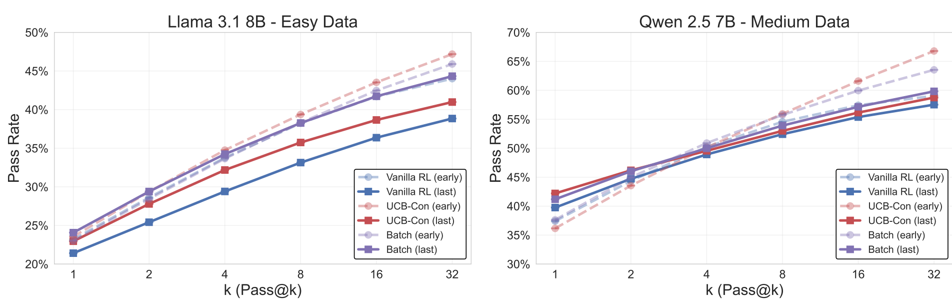

Figure 1: Test performance comparison (averaged across MATH-500, aime24/25, amc23) between our exploration methods (`UCB-Con` and `Batch`) and the `GRPO` baseline, with Llama3.1-8B-Instruct on the easy dataset (left) and Qwen2.5-7B on the medium dataset (right). We report pass@k for $k \in \{1, 2, 4, 8, 16, 32\}$ on an early checkpoint (at timestep 100) and the final checkpoint (at timestep 700). We repeat each experiment with 3 different random seeds and plot the mean performance. The exploration methods outperform the baseline on nearly all metrics across the training process (except Qwen 2.5 7B with `UCB-Con` on pass@1 on the early checkpoint due to exploration, but it has much higher pass@32 rate), and better exploitation-exploration trade-off and mitigation of overoptimization.

Submitted to the Workshop on Aligning Reinforcement Learning Experimentalists and Theorists (ARLET 2025). Do not distribute.

1 Introduction

Large language models (LLMs) are commonly post-trained with reinforcement learning (RL), both in preference alignment (Ouyang et al., 2022; Bai et al., 2022) and in reasoning tasks (Shao et al., 2024; Guo et al., 2025). A longstanding difficulty in RL is the design of reward signals: while one might hope to shape intermediate reasoning steps, recent works have shown that the seemingly crude strategy of rewarding only the final correctness (e.g., whether a math answer is correct) can be remarkably effective (Shao et al., 2024; Guo et al., 2025).

However, a growing body of evidence points to an important drawback of RL post-training: a systematic loss of diversity in model generations (Song et al., 2024; Dang et al., 2025; Yue et al., 2025; Zhao et al., 2025; Wu et al., 2025). This phenomenon is most cleanly captured by the pass@k metric: when k is large (say $k = 512$), post-trained models exhibit a lower pass@k than the base model. This raises a practical concern: in real-world deployments, diversity is often valuable and can amplify performance through test-time scaling (Wu et al., 2024; Snell et al., 2024), with different sampling processes such as directly sampling from the model or tree search. Indeed, we find that diversity degradation already manifests during training, as models collapse to a reduced set of candidate answers on unsolved problems due to a transfer effect of the diversity degradation induced by concentrating on correct answers, which we detail in Section 2.

Exploration is the canonical RL tool for combating such collapse (Bellemare et al., 2016; Azar et al., 2017; Burda et al., 2018). However, directly importing classical techniques such as Upper Confidence Bound (`UCB`) exploration (Auer et al., 2002) to token-level language modeling is intractable, as it would require searching over exponentially many sequences. Motivated by the success of outcome-based rewards, we therefore study *outcome-based exploration*, where exploration bonuses depend only on final outcomes. This perspective allows us to adapt `UCB`-style methods to LLM training, which we further refine by incorporating both positive and negative outcome signals.

A subtlety arises, however: in language models, one must distinguish between *historical exploration* (visiting a more diverse set of states and actions during training) and *batch exploration* (producing diverse outputs at test time). The latter improves pass@k but does not necessarily increase diversity during training whereas the former improves pass@1 but does not guarantee test-time diversity of the trained model. We introduce and study a batch version of outcome-based exploration, which demonstrates improved tradeoff between accuracy and diversity during test time.

Our contributions

1. We study RL post-training dynamics by framing RL as a sampling process. This perspective reveals that diversity loss is not limited to test-time behavior, but already occurs on the training set: as RL concentrates probability mass on previously solved questions, the resulting collapse propagates and reduces diversity even on unsolved ones. We term this effect the transfer of diversity degradation.
2. We propose outcome-based exploration, which adapts classical exploration bonuses (e.g. `UCB`) to the outcome space of LLM tasks. We show that naively adapting `UCB` does not lead to improved testing performance. We thus propose more refined algorithms which incorporate both positive and negative signals, we show that they improve both training exploration and test generalization.
3. We introduce a batch outcome-based exploration method that explicitly encourages diverse generations within a batch, yielding a better accuracy–diversity tradeoff at test time.
4. Through additional analysis, we clarify the interaction between historical and batch exploration, showing that they are not mutually exclusive but in fact complementary.

2 Diversity Degradation: RL as Sampling

2.1 Preliminaries

We consider LLM reasoning training with RL in a verifiable reward setting. Denote the set of questions as $\mathcal{X}$, the training question set $\mathcal{X}_{\mathsf{train}} \subseteq \mathcal{X}$ and the test question set $\mathcal{X}_{\mathsf{test}} \subseteq \mathcal{X}$. Further, define the space of intermediate text as $\mathcal{Y}$, and the answer space as $\mathcal{A}$; we consider an LLM to be a policy $\pi : \mathcal{X} \to \Delta(\mathcal{Y} \times \mathcal{A})$, i.e, given any question $x \in \mathcal{X}$, the LLM generates a sample $(y, a) \sim \pi(\cdot \mid x)$, where $y \sim \pi(\cdot \mid x)$ is the intermediate reasoning trace (chain of thought) and $a \sim \pi(\cdot \mid x, y)$ is the final answer. By following the convention in (Guo et al., 2025) we have access to a ground truth reward $r : \mathcal{X} \times \mathcal{A} \to \{0, 1\}$ that checks the correctness of the final answer. The evaluation metric for a given (dataset, policy) pair is defined in terms of the accuracy of the final answer: $J(\pi, \mathcal{X}) = \mathbb{E}_{x\sim\mathsf{unif}(\mathcal{X})}\, \mathbb{E}_{(y,a)\sim\pi(\cdot|x)}[r(x, a)]$. During RL training, we use the KL-regularized version of the objective $J(\pi, \mathcal{X}_{\mathsf{train}})$, which aims to find the $\pi^\star$ such that

$$\pi^\star := \arg\max_\pi \mathbb{E}_{x\sim\mathsf{unif}(\mathcal{X}_{\mathsf{train}})} \big[\mathbb{E}_{(y,a)\sim\pi(\cdot|x)}[r(x, a)] - \beta \mathrm{KL}(\pi(\cdot \mid x), \pi_{\mathsf{base}}(\cdot \mid x))\big],$$

where π_{base} is our base LLM from which the RL training is initialized. In this paper, we consider the fully on-policy GRPO algorithm (Shao et al., 2024), which optimizes the following objective:

$$\widehat{\mathbb{E}}_{x,\{y_i,a_i\}_{i=1}^n\sim\pi(\cdot|x)} \left[\frac{1}{n}\sum_{i=1}^n \hat{A}(x, \{y_i, a_i\}_{i=1}^n)_i - \beta\widehat{\mathrm{KL}}(\pi(\cdot \mid x), \pi_{\mathsf{base}}(\cdot \mid x))\right], \tag{1}$$

where $\hat{A}(x, \{y_i, a_i\}_{i=1}^n)_i = \frac{r(x,a_i)-\mu\left(\{r(x,a_{i'})\}_{i'=1}^n\right)}{\sigma\left(\{r(x,a_{i'})\}_{i'=1}^n\right)}$ and $\widehat{\mathrm{KL}}(\pi(\cdot \mid x), \pi_{\mathsf{base}}(\cdot \mid x))$ is estimated by $\log\left(\frac{\pi(y,a|x)}{\pi_{\mathsf{base}}(y,a|x)}\right) + \frac{\pi_{\mathsf{base}}(y,a|x)}{\pi(y,a|x)} - 1$, which is considered to enjoy lower variance than directly sampling $\log\left(\frac{\pi(y,a|x)}{\pi_{\mathsf{base}}(y,a|x)}\right)$ (Schulman, 2020). Notice that it is known that this objective leads to a biased gradient of the regularized objective J, and incidentally minimizes the forward KL-divergence $\mathrm{KL}(\pi_{\mathsf{base}}(\cdot \mid x), \pi(\cdot \mid x))$. (Tang & Munos, 2025). However, we will use this algorithm as it is a popular baseline and will call it vanilla RL in the following.

Finally, given question x, we define all possible reasoning traces of LLM π as $\mathcal{Y}^\pi(x) = \mathrm{supp}(\pi(\cdot \mid x))$, and thus the answer support of an LLM π as $\mathcal{A}^\pi(x) := \mathrm{supp}(\pi(\cdot \mid x, \mathcal{Y}^\pi(x)))$.

Experiment setting: In this paper, we primarily investigate LLM RL training for math reasoning. We test two models: Llama3.1-8B-Instruct (Dubey et al., 2024) and Qwen2.5-7B-Base models (Yang et al., 2024). We use two datasets: an easy dataset and a medium difficulty dataset. The easy dataset is the train split of the MATH dataset (Hendrycks et al., 2021) with a total of 7500 questions. For the medium dataset, we subsample 3840 questions from the training set of DAPO (Yu et al., 2025). To test, we use the MATH-500 (Lightman et al., 2023), AIME 2024/2025 and AMC23 datasets. To measure whether two given answers are different, we apply the $\mathrm{math\_verify}$ function in the verl (Sheng et al., 2024) codebase, which treats two answers as the same as long as they are mathematically equivalent. This defines our reward function as well since it is the indicator function of whether a given answer is equivalent to the ground truth answer.

2.2 Diversity Degradation during RL training

Recently it has been observed that, during LLM post training (either with SFT or RL), the diversity of the final policy decreases, as measured with the pass@k metric with $k > 1$, over the test dataset (Song et al., 2024; Dang et al., 2025; Yue et al., 2025; Wu et al., 2025). However, the previous analysis only focused on comparing the base model π_{base} with the final model check point π_T, a single artifact of the RL training method.

To understand how diversity degrades during RL training, we propose to examine the dynamics of RL training by considering RL as a sampling process on the training set. Specifically, in each epoch $t \in [T]$ during RL training, one samples n trajectories for each question $x \in \mathcal{X}_{\mathsf{train}}$. Thus given a base model π_{base} and RL algorithm Alg, we sample in total nT trajectories for each question x, i.e., $\{y_i, a_i\}_{i=1}^{nT} \sim \mathrm{Alg}(\pi_{\mathsf{base}}, x)$. Now this allows us to directly compare with sampling the same amount of trajectories from the base model, i.e., $\{y'_i, a'_i\}_{i=1}^k \sim \pi_{\mathsf{base}}(\mathcal{X}_{\mathsf{train}})$, where $k = nT$.

We conducted experiments on the Llama 3.1-8B-Instruct and Qwen 2.5-7B-Base models, trained on both the easy and medium difficulty datasets. To compare the RL training dynamics and the base model, we adopt two metrics: total number of questions solved and total number of distinct answers. Note that these metrics correspond to the pass@k and diff@k metrics that are used to measure a fixed model. Recall that to convert a training epoch t to k in pass@k and diff@k, we have $k = nt$, where in our experiments we use $n = 16$. We summarize the results in Figure 2, and we make the following observations:

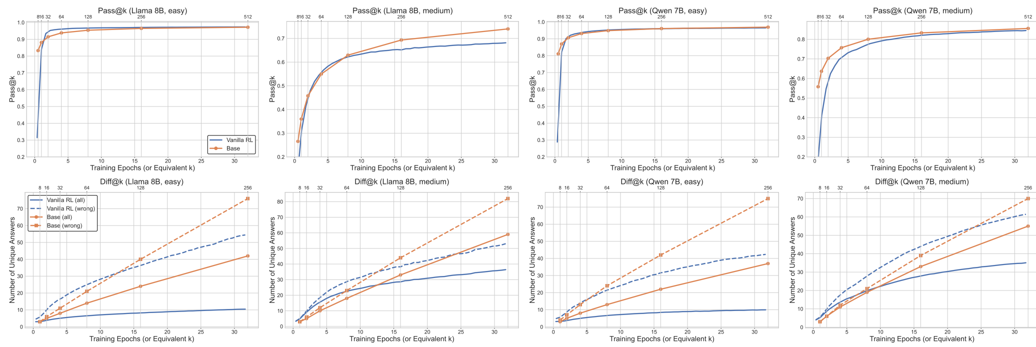

Figure 2: Comparison between RL training dynamics and base model sampling, on both easy and medium difficulty datasets, with Llama3.1-8B-Instruct and Qwen2.5-7B models. Top row: number of questions solved so far; Bottom row: number of different answers sampled so far. The bottom x-ticks are the number of epochs t for training, and the top x-ticks are the corresponding k for sampling from the base model. We convert $k = nt$ where n is the number of samples per epoch and t is the epoch index. We use $n = 16$ for pass@k comparison and $n = 8$ for diff@k comparison. In the diff@k comparison, solid lines denote the average number of different answers per all questions, and dashed lines denote the average number of different answers per unsolved questions (i.e., all answers are wrong so far). The fact that RL has lower diff@k on unsolved questions than the base model indicates the transfer of diversity degradation.

- **RL eventually solves fewer questions than the base model**. At the beginning of the RL training, the rate of questions solving is faster than the base model, which is expected, as RL quickly converges to the correct answers on the ones that it can easily solve. However, as training continues, the rate of question solving decreases faster than the base model, and eventually RL solves fewer questions than the base model with the same amount of samples *on the training set*.
- **Transfer of diversity degradation across questions.** In an ideal setting where the training dynamics are independent across questions, vanilla RL training should never underperform the base model. This is because the model does not update on questions x it has not solved yet (i.e., it receives zero gradient on those questions), so its behavior on those questions is equivalent to the base model, i.e., $\mathcal{A}^{\pi_{\mathsf{RL}}}(x) = \mathcal{A}^{\pi_{\mathsf{base}}}(x)$. The observed diversity degradation can therefore be explained as follows: once the model concentrates its answers on questions it has solved, this reduced diversity propagates to unsolved questions as well. To quantify this effect, we track the cumulative number of distinct answers sampled. We find that RL training yields lower diversity across all questions on average, and, more importantly, even lower diversity on the unsolved questions. We refer to this phenomenon as the *transfer of diversity degradation*.
- **Diversity is tractable on verifiable domains**. In general it is hard to predict that, given two generations from LLMs, whether they are semantically different or not. Naively measuring in token space results in an exponentially many candidates and thus is intractable. However, in the verifiable domain, we can use the final answer as a proxy to measure the diversity of the generations. From Figure 2, we observe that given a large sample budget, we only have $|\mathcal{A}^{\pi_{\mathsf{base}}}(x)| < 50$ on average, which is tractable to measure and optimize. We refer to this property as *the tractability of the outcome space*. We will introduce our algorithms that leverage this property in the next section.

3 Outcome-based Exploration

3.1 Historical Exploration via UCB

Given the observation that there are bounded number of final answers to search over, our training objective thus becomes to explore as many different answers (and their corresponding reasoning traces) as possible, while also rewarding the correctness of the answer. This problem is well studied in the bandit and RL literature, and the canonical solution for exploration is the upper confidence

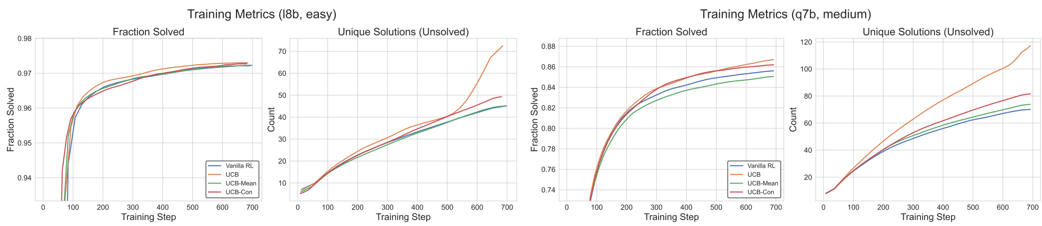

Figure 3: Training performance comparison between different `UCB` variants and the `GRPO` baseline, with Llama3.1-8B-Instruct on the easy dataset (left) and Qwen2.5-7B on the medium dataset (right). For each subplot: Left: fraction of questions solved so far; Right: number of different answers sampled on the questions that the model has yet to solve (i.e., sample one correct answer historically). The x-axis denotes the number of gradient updates as we train all models fully on-policy. We repeat each experiment with 3 different random seeds and plot the mean performance.

bound (`UCB`) method (Auer et al., 2002; Azar et al., 2017), which for each state and action adds an additional exploration bonus that is inversely proportional to its historical visitation counts, on top of the correctness reward. Thus, the training objective in Equation (1) becomes:

$$\widehat{\mathbb{E}}_{x,\{y_i,a_i\}_{i=1}^n\sim\pi(\cdot|x)}\left[\frac{1}{n}\sum_{i=1}^{n}\widehat{A}(x,\{y_i,a_i\}_{i=1}^n)_i + cb_{\mathsf{ucb}}(x,a_i) - \beta\widehat{\mathrm{KL}}(\pi(\cdot\mid x),\pi_{\mathsf{base}}(\cdot\mid x))\right], \quad (2)$$

where c is a tunable hyperparameter and

$$b_{\mathsf{ucb}}(x,a) = \min\left\{1, \sqrt{\frac{1}{N(x,a)}}\right\},$$

where $N(x,a)$ is the number of times we have sampled the answer a for the question x.).

3.1.1 Naive UCB only Improves Training Performance

We present partial training results in Figure 3 and test results in Figure 4, and defer the remaining training results to Figure 7 and test results to Figure 8. We observe that, although the `UCB` bonus improves the training performance consistently (and with a larger improvement on the harder dataset), it does not consistently improve the test performance across different models and datasets. In particular, we only observe a significant improvement on the easy dataset with the Llama3.1 8B model.

Originally, the design of `UCB` is due to the fact that, for any pair of state and action, the estimation error of the dynamics and reward scales with the order of $O(1/\sqrt{N(x,a)})$, and thus adding this bonus offsets this error and encourages the policy to explore uncertain states and actions. However, in the LLM reasoning setting, the dynamics and reward are both deterministic, and thus in the extreme case where the training dynamics is independent across questions, the policy should stop visiting an answer once it gets a reward of 0, because now it has a perfect estimation of the reward of this answer already. While in practice the training dynamics is not independent across questions, and intuitively the `UCB` bonus encourages the model to explore answers that it has not visited often and thus accelerates the training performance, we hypothesize that a redundant visitation of incorrect answers actually hurts the generalization performance.

3.2 UCB with a Baseline

The above observation suggests that providing only positive exploration signals is not the most effective strategy where test performance is concerned. Instead, we propose incorporating a baseline into the bonus calculation, so that exploration signals are defined relative to this baseline and can be either positive or negative. A natural starting point—analogous to the `GRPO` baseline—is to use the batch mean of the `UCB` bonus as the baseline. Concretely, we modify the objective in Equation (2) by replacing $b_{\mathsf{ucb}}(x,a_i)$ with:

$$\widehat{B}(x,\{y_i,a_i\}_{i=1}^n)_i = b_{\mathsf{ucb}}(x,a_i) - \frac{1}{n-1}\sum_{j\neq i}^{n} b_{\mathsf{ucb}}(x,a_j).$$

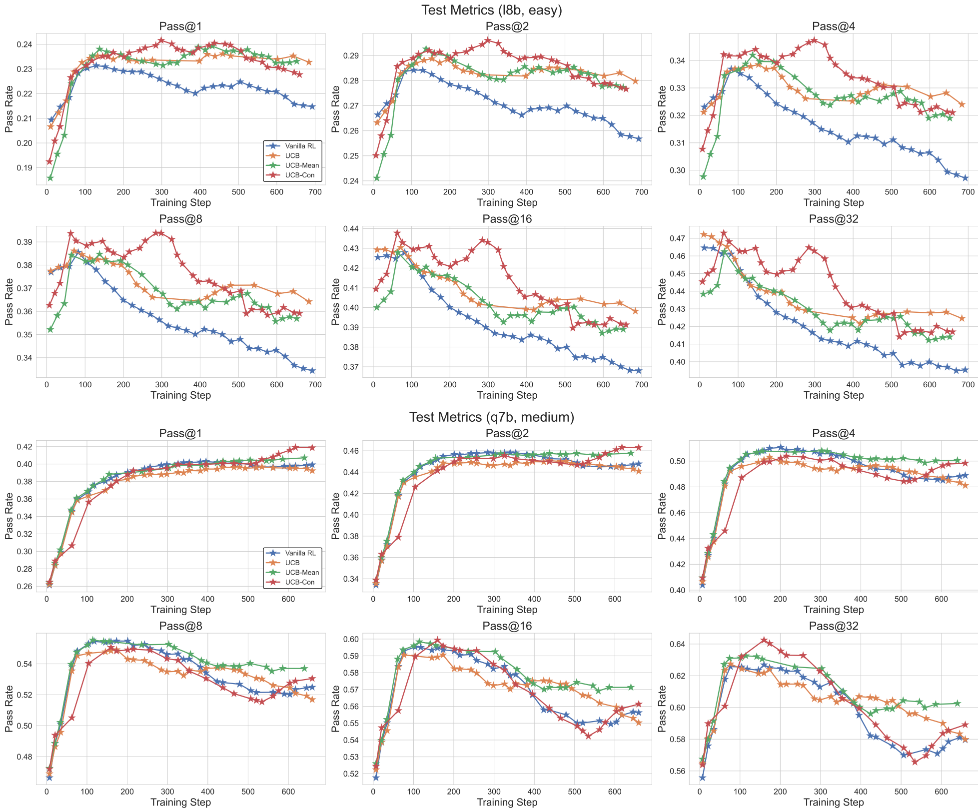

Figure 4: Test performance comparison between different `UCB` variants and the `GRPO` baseline, with Llama3.1-8B-Instruct on the easy dataset (top) and Qwen2.5-7B on the medium dataset (bottom). We report pass@k for $k \in \{1, 2, 4, 8, 16, 32\}$ at every 20 training steps. We repeat each experiment with 3 different random seeds and plot the mean performance. The metrics are calculated based on 32 samples per question during evaluation.

We refer to this method as `UCB` with a mean baseline (`UCB-Mean`). Intuitively, it encourages the model to explore answers that are less frequent in the current batch while penalizing those that appear more often. Although historically frequent answers tend to receive a negative signal, the batch-level baseline means that an answer can still receive a positive exploration signal if it is relatively underrepresented within the current batch.

To avoid this issue, we propose a third method, `UCB` with a constant baseline (`UCB-Con`), where we simply use a constant as the baseline, i.e.,

$$\widehat{B}(x, \{y_i, a_i\}_{i=1}^n)_i = b_{\mathsf{ucb}}(x, a_i) - b_0,$$

where b_0 is a tunable hyperparameter. Note that this gives easy control over the tradeoff between positive and negative exploration signal (Arnal et al., 2025). For example, if we set $b_0 = 0.5$, then an answer will get a positive exploration signal if it has been visited less than 4 times, and a negative signal otherwise. One issue with the baseline formulation is that, in the case where all answers in the batch are correct, then we have $A_i = 0$ for all i, and thus the exploration signal will dominate the training objective. After the beginning of the training, this objective will assign a negative gradient to the batch where all the answers are correct. To prevent this undesirable behavior, in this case (for both `UCB-Mean` and `UCB-Con`) we simply assign zero exploration bonus to all answers in the batch, thus recovering the regular `GRPO` objective.

3.2.1 UCB with a Baseline Generalizes towards Test Performance

We compare these three variants with the `GRPO` baseline, and the partial results are summarized in Figure 3 and Figure 4. For the training performance, we observe that adding a baseline slightly hurts

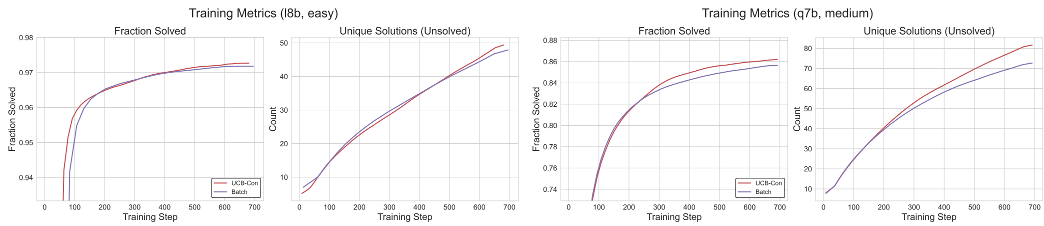

Figure 5: Training performance comparison between `Batch` and `UCB-Con`, Llama3.1-8B-Instruct on the easy dataset (left) and Qwen2.5-7B on the medium dataset (right). For each subplot: Left: fraction of questions solved so far; Right: number of different answers sampled on the questions that the model has yet to solve (i.e., sample one correct answer historically). The x-axis denotes the number of gradient updates as we train all models fully on-policy. We repeat each experiment with 3 different random seeds and plot the mean performance.

the training performance, but `UCB-Con` still outperforms `GRPO`. On the other hand, both `UCB-Mean` and `UCB-Con` consistently improve the test performance across different models and datasets. While `UCB-Mean` improves over `UCB` and `GRPO`, `UCB-Con` achieves the best frontier performance as it achieves the best pass@k performance for all k's in most of the settings. Another observation is that under our large number of epochs training setup, vanilla RL (`GRPO`) sometimes suffers from overoptimization as even the pass@k performance degrades after a certain number of epochs, while RL with exploration mitigates this issue. See Table 3 and Table 4 for a quantitative comparison.

However, in general, one should not expect global exploration to achieve a high pass@k when k is large, especially at the end of the training. A pedagogical example is that, to maximize the exploration bonus, the model can generate a batch of the same answers that currently has the least visitation counts. Indeed, in the theoretical RL literature, the goal of exploration is usually to return a policy that is deterministic (and optimal) (Azar et al., 2017). While in general adding exploration bonus does provide better pass@k with large k than vanilla RL, we observe that in certain settings the final pass@k for $k = 8, 16, 32$ is similar to that of vanilla RL.

3.3 Batch Exploration

The above issue suggests a fundamental but subtle difference between the goal of traditional RL exploration and the goal of exploration in the LLM reasoning setting. In traditional RL, the goal of exploration is to find the optimal policy which maximizes the expected return (corresponding to pass@1), while in the LLM reasoning setting, in addition to pass@1, sometimes we also care about the diversity of the generation which determines the model's capacity towards test-time scaling (Wu et al., 2024). To encourage the model to generate diverse answers, we consider a different exploration strategy, *batch exploration*, which directly rewards the model to generate diverse answers regardless of their historical behavior. In particular, in batch exploration we propose the (`Batch`) objective, with $b_{\mathsf{ucb}}(x, a_i)$ in Equation (2) replaced by:

$$b_{\mathsf{batch}}(x, \{y_i, a_i\}_{i=1}^n)_i = -\frac{1}{n}\sum_{j \neq i} \mathbb{1}\{a_i = a_j\},$$

where we simply penalize each answer based on how repetitive it is in the batch. We remark that we also experimented with the positive version of the batch exploration bonus where we provide a bonus of 1 for unique answers in the batch, but our result shows that such positive batch exploration bonus does not provide meaningful improvement in either training or test results.

We summarize the experimental results in Figure 5 and Figure 6, and defer the remaining train results to Figure 9 and test results to Figure 10. We focus on comparing `Batch` with `UCB-Con` for a cleaner presentation since both methods outperform the `GRPO` baseline consistently. We observe that in general, `Batch` achieves worse performance during the training, as measured by both the fraction of questions solved and the number of different answers generated. However, note that the objective of `Batch` is not designed to explicitly optimize these two metrics. One can also consider a pedagogical failure of batch exploration as the model keeps sampling the same n distinct answers in each epoch for each question it could not solve yet, and thus no real exploration is performed during training.

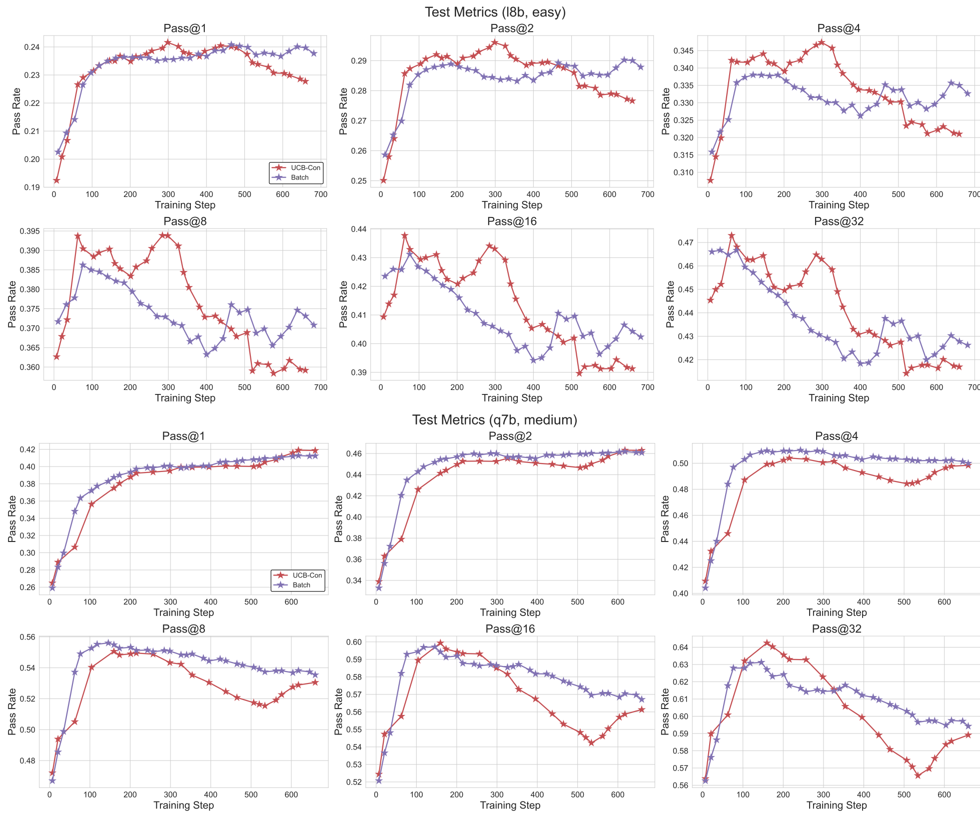

Figure 6: Test performance comparison between `Batch` and `UCB-Con`, with Llama3.1-8B-Instruct on the easy dataset (top) and Qwen2.5-7B on the medium dataset (bottom). We report pass@k for $k \in \{1, 2, 4, 8, 16, 32\}$ at every 20 training steps. We repeat each experiment with 3 different random seeds and plot the mean performance. The metrics are calculated based on 32 samples per question during evaluation.

As for test performance, in general `Batch` achieves similar peak pass@k performance as `UCB-Con` (with slight degradation in some settings), but `Batch` consistently achieves better diversity at the end of the training, as measured by the pass@k performance for large k. See Table 4 for a quantitative comparison. This suggests that batch exploration might be preferable if the objective is to achieve tradeoff between generation accuracy and diversity at test time.

4 Additional Analysis on Historical and Batch Explorations

In the previous section, we compared historical exploration and batch exploration in terms of their training dynamics. Overall, historical exploration is superior, as it solves more questions and accumulates more diverse answers over time. This is expected, since both metrics are inherently historical in nature. In this section, we turn to additional aspects of exploration, focusing in particular on batch-level statistics during training.

Generation Entropy. Entropy has been used as a measure of model diversity and as a tool to encourage exploration (Cheng et al., 2025; Zheng et al., 2025). We compare entropy at training step 400 of the Qwen 2.5-7B model on the medium dataset, trained with `GRPO`, `UCB-Con`, and `Batch`. For each method, we report the average entropy of correct and incorrect generations separately (Table 1). As expected, correct generations have lower entropy than incorrect ones. Among incorrect generations, however, `Batch` achieves substantially higher entropy than both `GRPO` and `UCB-Con`. Since entropy is measured on the current model rather than accumulated over training, this suggests that batch exploration produces more entropic, and potentially more diverse, generations when considering a single checkpoint. That said, the absolute entropy values remain low across all methods, consistent with the fact that we do not explicitly optimize for entropy, unlike entropy-regularized exploration approaches.

Table 1: Entropy comparison of GRPO, UCB-Con and Batch, measured on correct generation, incorrect generation and all generations. We repeat for 2 random seeds and report the mean and standard deviation (in parentheses).

	Correct Generation	Incorrect Generation	All
GRPO	0.080 (0.01)	0.096 (0.04)	0.095 (0.02)
UCB-Con	0.084 (0.01)	0.103 (0.03)	0.100 (0.02)
Batch	0.086 (0.01)	0.153 (0.07)	0.125 (0.03)

Batch Generation Diversity. To directly measure batch-level diversity, we consider the number of distinct answers sampled within each batch. Results are shown in Table 2. As expected, Batch consistently produces more distinct answers than UCB-Con, since it directly optimizes for batch diversity.

Table 2: Comparison of different exploration strategies based on the number of different answers sampled in a batch with size of 8. We additional cluster the statistics based on whether the question has been solved. We repeat for 2 random seeds and report the mean and standard deviation (in parentheses).

	Solved Question	Unsolved Question	All
GRPO	2.279 (0.018)	4.805 (0.075)	2.883 (0.024)
UCB-Con	2.272 (0.020)	4.855 (0.084)	2.926 (0.035)
Batch	2.284 (0.057)	5.390 (0.102)	3.230 (0.062)

Finally, we remark on the interaction between historical and batch exploration. In principle, it is possible to construct counterexamples where historical exploration converges to a nearly deterministic policy—thus sacrificing test-time diversity—or where batch exploration cycles through a small set of answers without improving training dynamics. These pathologies highlight that the two notions are not guaranteed to substitute for one another. In practice, however, our empirical results suggest a complementary relationship: historical exploration, by encouraging broader coverage of the training space, naturally increases the diversity available to each batch, while batch exploration, by promoting variation within each batch, in turn helps prevent premature collapse during training. Taken together, these findings indicate that historical and batch exploration are not mutually exclusive.

5 Conclusion and Discussion

In this paper, we study the diversity degradation problem in LLM reasoning post-training through the analysis of RL as sampling. We observe two key phenomena: the generalization of diversity degradation and the tractability of outcome space in verifiable reasoning tasks. Based on these observations, we adopt the classical RL exploration strategy UCB in the outcome space, and a careful treatment between positive and negative exploration signals achieves improvement in test performance in the pass@k metrics for all k. We also identify the distinction of the historical exploration in traditional RL and batch exploration that is more specific in the LLM reasoning setting, and derive the outcome-based batch exploration algorithm, which achieves better accuracy-diversity tradeoff at test time. Finally we provide more in-depth analysis on the connection of historical exploration and batch exploration.

There are a few limitations of our work. First, our current algorithms only apply to the verifiable domain, and problems with a tractable outcome space, extending them to more general settings is an interesting future direction. Second, currently we only evaluate our methods on the single-turn benchmarks, and we believe exploration plays an even more significant role under the multi-turn settings.

References

Alekh Agarwal, Yujia Jin, and Tong Zhang. Vo q l: Towards optimal regret in model-free rl with nonlinear function approximation. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 987–1063. PMLR, 2023. 14

Charles Arnal, GaĂŤtan Narozniak, Vivien Cabannes, Yunhao Tang, Julia Kempe, and Remi Munos. Asymmetric reinforce for off-policy reinforcement learning: Balancing positive and negative rewards. *arXiv preprint arXiv:2506.20520*, 2025. 6

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002. 2, 5

Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin Yang. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pp. 463–474. PMLR, 2020. 14

Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. In *International Conference on Machine Learning*, 2017. 2, 5, 7, 14

Chenjia Bai, Yang Zhang, Shuang Qiu, Qiaosheng Zhang, Kang Xu, and Xuelong Li. Online preference alignment for language models via count-based exploration. *arXiv preprint arXiv:2501.12735*, 2025. 14

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022. 2

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016. 2

Ronen I Brafman and Moshe Tennenholtz. R-max-a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3(Oct):213–231, 2002. 14

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018. 2, 14

Shicong Cen, Jincheng Mei, Katayoon Goshvadi, Hanjun Dai, Tong Yang, Sherry Yang, Dale Schuurmans, Yuejie Chi, and Bo Dai. Value-incentivized preference optimization: A unified approach to online and offline rlhf. *arXiv preprint arXiv:2405.19320*, 2024. 14

Zhipeng Chen, Xiaobo Qin, Youbin Wu, Yue Ling, Qinghao Ye, Wayne Xin Zhao, and Guang Shi. Pass@ k training for adaptively balancing exploration and exploitation of large reasoning models. *arXiv preprint arXiv:2508.10751*, 2025. 14

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. Reasoning with exploration: An entropy perspective. *arXiv preprint arXiv:2506.14758*, 2025. 8, 14

Xingyu Dang, Christina Baek, Kaiyue Wen, Zico Kolter, and Aditi Raghunathan. Weight ensembling improves reasoning in language models. *arXiv preprint arXiv:2504.10478*, 2025. 2, 3, 14

Simon S Du, Sham M Kakade, Jason D Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear classes: A structural framework for provable generalization in RL. *International Conference on Machine Learning*, 2021. 14

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

Dylan J Foster, Sham M Kakade, Jian Qian, and Alexander Rakhlin. The statistical complexity of interactive decision making. *arXiv:2112.13487*, 2021. 14

Jingtong Gao, Ling Pan, Yejing Wang, Rui Zhong, Chi Lu, Qingpeng Cai, Peng Jiang, and Xiangyu Zhao. Navigate the unknown: Enhancing llm reasoning with intrinsic motivation guided exploration. *arXiv preprint arXiv:2505.17621*, 2025. 14

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 2, 3, 14

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021. 3

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024. 14

Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, John Langford, and Robert E Schapire. Contextual decision processes with low Bellman rank are PAC-learnable. In *International Conference on Machine Learning*, 2017. 14

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? *Advances in neural information processing systems*, 31, 2018. 14

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In *Conference on Learning Theory*, pp. 2137–2143, 2020. 14

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine learning*, 49:209–232, 2002. 14

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the effects of rlhf on llm generalisation and diversity. *arXiv preprint arXiv:2310.06452*, 2023. 14

Akshay Krishnamurthy, Alekh Agarwal, and John Langford. PAC reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, 2016. 14

Jack Lanchantin, Angelica Chen, Shehzaad Dhuliawala, Ping Yu, Jason Weston, Sainbayar Sukhbaatar, and Ilia Kulikov. Diverse preference optimization. *arXiv preprint arXiv:2501.18101*, 2025. 14

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023. 3

Aditya Modi and Ambuj Tewari. No-regret exploration in contextual reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, pp. 829–838. PMLR, 2020. 14

Antoine Moulin, Gergely Neu, and Luca Viano. Optimistically optimistic exploration for provably efficient infinite-horizon reinforcement and imitation learning. *arXiv preprint arXiv:2502.13900*, 2025. 14

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems*, 26:3003–3011, 2013. 14

Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. In *Advances in neural information processing systems*, pp. 4026–4034, 2016. 14

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022. 2, 14

Laura O'Mahony, Leo Grinsztajn, Hailey Schoelkopf, and Stella Biderman. Attributing mode collapse in the fine-tuning of large language models. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, volume 2, 2024. 14

Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014. 14

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. *Operations Research*, 66(1):230–252, 2018. 14

John Schulman. Approximating kl divergence, 2020. *URL http://joschu. net/blog/kl-approx. html*, 2020. 3

Amrith Setlur, Matthew YR Yang, Charlie Snell, Jeremy Greer, Ian Wu, Virginia Smith, Max Simchowitz, and Aviral Kumar. e3: Learning to explore enables extrapolation of test-time compute for llms. *arXiv preprint arXiv:2506.09026*, 2025. 14

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 2, 3, 14

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024. 3, 18

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. 2

Yuda Song and Wen Sun. PC-MLP: Model-based reinforcement learning with policy cover guided exploration. In *International Conference on Machine Learning*, 2021. 14

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. *arXiv preprint arXiv:2412.02674*, 2024. 2, 3, 14

Fahim Tajwar, Yiding Jiang, Abitha Thankaraj, Sumaita Sadia Rahman, J Zico Kolter, Jeff Schneider, and Ruslan Salakhutdinov. Training a generally curious agent. *arXiv preprint arXiv:2502.17543*, 2025. 14

Yunhao Tang and Rémi Munos. On a few pitfalls in kl divergence gradient estimation for rl. *arXiv preprint arXiv:2506.09477*, 2025. 3

Yunhao Tang, Kunhao Zheng, Gabriel Synnaeve, and Rémi Munos. Optimizing language models for inference time objectives using reinforcement learning. *arXiv preprint arXiv:2503.19595*, 2025. 14

Fang Wu, Weihao Xuan, Ximing Lu, Zaid Harchaoui, and Yejin Choi. The invisible leash: Why rlvr may not escape its origin. *arXiv preprint arXiv:2507.14843*, 2025. 2, 3, 14

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. An empirical analysis of compute-optimal inference for problem-solving with language models. 2024. 2, 7

Tengyang Xie, Dylan J Foster, Akshay Krishnamurthy, Corby Rosset, Ahmed Awadallah, and Alexander Rakhlin. Exploratory preference optimization: Harnessing implicit q*-approximation for sample-efficient rlhf. *arXiv preprint arXiv:2405.21046*, 2024. 14

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *arXiv preprint arXiv:2312.11456*, 2023. 14

Lin Yang and Mengdi Wang. Sample-optimal parametric Q-learning using linearly additive features. In *International Conference on Machine Learning*, pp. 6995–7004. PMLR, 2019. 14

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024. 3

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

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025. 2, 3, 14

Longfei Yun, Chenyang An, Zilong Wang, Letian Peng, and Jingbo Shang. The price of format: Diversity collapse in llms. *arXiv preprint arXiv:2505.18949*, 2025. 14

Shenao Zhang, Donghan Yu, Hiteshi Sharma, Han Zhong, Zhihan Liu, Ziyi Yang, Shuohang Wang, Hany Hassan, and Zhaoran Wang. Self-exploring language models: Active preference elicitation for online alignment. *arXiv preprint arXiv:2405.19332*, 2024a. 14

Zihan Zhang, Yuxin Chen, Jason D Lee, and Simon S Du. Settling the sample complexity of online reinforcement learning. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 5213–5219. PMLR, 2024b. 14

Rosie Zhao, Alexandru Meterez, Sham Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo chamber: Rl post-training amplifies behaviors learned in pretraining. *arXiv preprint arXiv:2504.07912*, 2025. 2

Tianyu Zheng, Tianshun Xing, Qingshui Gu, Taoran Liang, Xingwei Qu, Xin Zhou, Yizhi Li, Zhoufutu Wen, Chenghua Lin, Wenhao Huang, et al. First return, entropy-eliciting explore. *arXiv preprint arXiv:2507.07017*, 2025. 8, 14

Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Conference on Learning Theory*, pp. 4532–4576. PMLR, 2021. 14

A Related Work

A.1 Diversity Degradation in LLM Post Training

Reinforcement Learning has become the de facto method for finetuning large language models (LLMs) towards specific objectives, such as maximizing human preference (Ouyang et al., 2022), or improving the reasoning ability of LLMs (Jaech et al., 2024). In the reasoning domain, it has been shown that simply rewarding the model based on the correctness of the final answer, without any intermediate reward, can significantly improve the final accuracy (Shao et al., 2024; Guo et al., 2025). However, it has been observed that, during the RL training (or even SFT), the diversity of the generations decreases significantly (Song et al., 2024; Dang et al., 2025; Yue et al., 2025; Wu et al., 2025), as measured with the pass@k metric. In the non-reasoning domain, similar observations have also been made, where post-training improves the performance of the model on the main metric, but at the cost of losing diversity, measured by either semantic or syntactic metrics (Kirk et al., 2023; O'Mahony et al., 2024; Yun et al., 2025).

A.2 Exploration in LLM Post Training

Enhancing exploration during RL training has been considered the key towards addressing diversity issues during either training or testing. In the preference fine-tuning domain, Xie et al. (2024); Cen et al. (2024); Zhang et al. (2024a) propose to use the likelihood of the base model as an exploration bonus. Lanchantin et al. (2025) proposes to label ranking of the data based on their diversity in the preference learning process. Xiong et al. (2023); Bai et al. (2025) theoretically analyzes the guarantees of RL with exploration under the linear setting. In the reasoning domain, (Gao et al., 2025) directly uses Random Network Distillation (Burda et al., 2018), a canonical exploration bonus in Deep RL, as an exploration bonus to encourage the model to explore different traces. Cheng et al. (2025); Zheng et al. (2025) proposes to leverage entropy to encourage exploration. Chen et al. (2025) leverages pass@k training objective (Tang et al., 2025) to improve the batch diversity during training. Setlur et al. (2025) proposes to improve model's length generalization towards self-correction. Finally, Tajwar et al. (2025) uses multi-task training towards multi-turn exploration. Note that outcome based method is complimentary to all these methods, and can be potentially combined with them to further improve the diversity.

A.3 Exploration in theoretical RL

In the tabular setting, exploration has been studied extensively through count-based methods such as R-max (Brafman & Tennenholtz, 2002), E3 (Kearns & Singh, 2002), culminating in near-optimal PAC and regret guarantees via optimism in the face of uncertainty (Azar et al., 2017; Jin et al., 2018; Zhang et al., 2024b). These approaches rely on visitation counts to construct exploration bonuses. In linear MDPs, counts are replaced by confidence sets in feature space. Algorithms such as UCRL-VTR (Yang & Wang, 2019) and LSVI-UCB (Jin et al., 2020) establish polynomial sample complexity, achieving regret bounds scaling with the feature dimension d rather than the number of states. Subsequent refinements obtained nearly minimax-optimal guarantees (Ayoub et al., 2020; Zhou et al., 2021; Agarwal et al., 2023), with extensions to the discounted setting (Moulin et al., 2025) or model-based setting (Song & Sun, 2021) showing that the principle of optimism extends naturally from tabular counts to linear function approximation. The majority of these works share the same bonus-based exploration approach as our historical exploration method. Thompson sampling (Russo & Van Roy, 2014, 2018) is another popular exploration strategy that has been shown to achieve similar theoretical guarantees in both tabular and linear settings (Osband et al., 2013, 2016; Modi & Tewari, 2020). Finally, beyond tabular and linear settings, exploration in RL with general function approximation has been studied under various structural assumptions (Krishnamurthy et al., 2016; Jiang et al., 2017; Du et al., 2021; Foster et al., 2021). However, these methods do not enjoy computational efficiency as opposed to the bonus-based methods.

B Omitted Plots

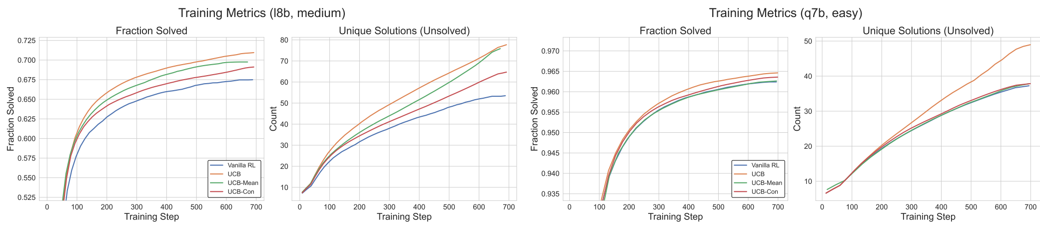

Figure 7: Training performance comparison between different UCB variants and the GRPO baseline, with Llama3.1-8B-Instruct on the medium dataset (left) and Qwen2.5-7B on the easy dataset (right). For each subplot: Left: fraction of questions solved so far; Right: number of different answers sampled on the questions that the model has yet to solve (i.e., sample one correct answer historically). The x-axis denotes the number of gradient updates as we train all models fully on-policy. We repeat each experiment with 3 different random seeds and plot the mean performance.

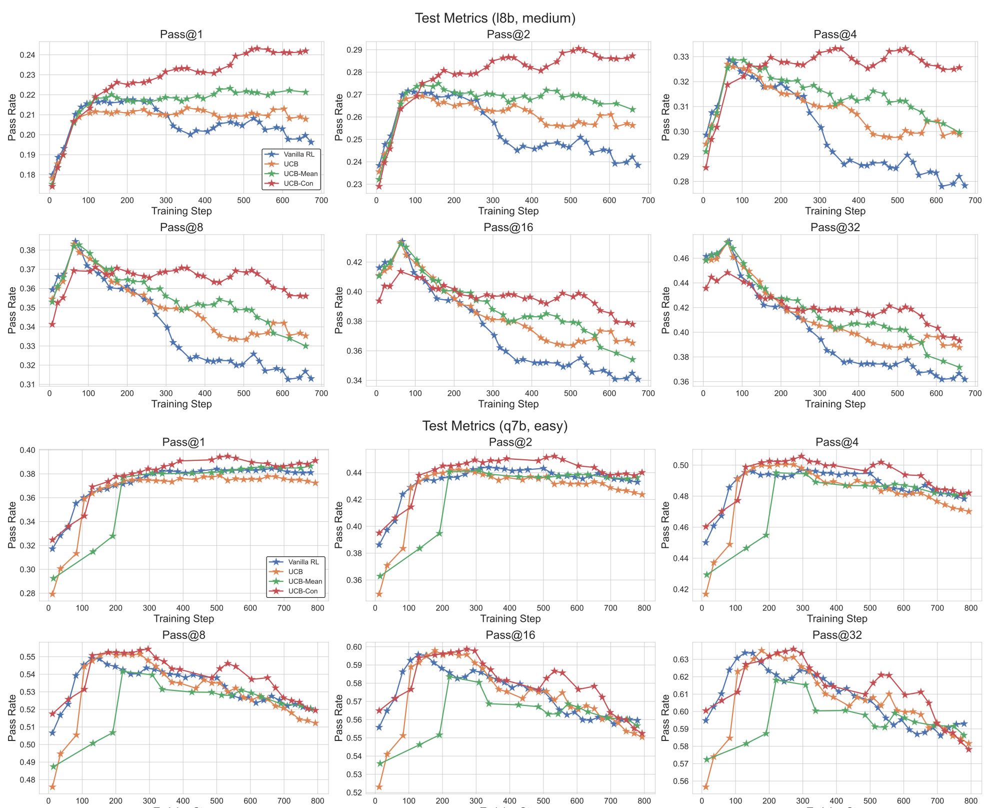

Figure 8: Test performance comparison between different UCB variants and the GRPO baseline, with Llama3.1-8B-Instruct on the medium dataset (top) and Qwen2.5-7B on the easy dataset (bottom). We report pass@k for $k \in \{1, 2, 4, 8, 16, 32\}$ at every 20 training steps. We repeat each experiment with 3 different random seeds and plot the mean performance. The metrics are calculated based on 32 samples per question during evaluation.

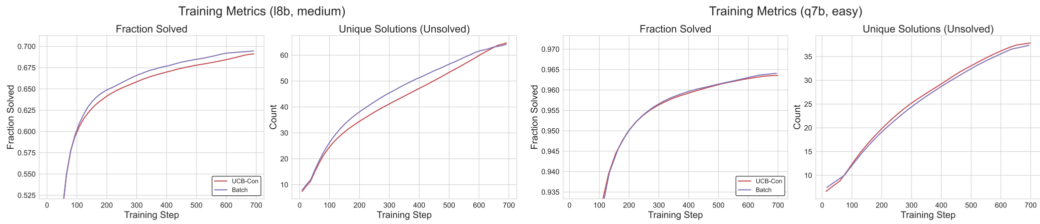

Figure 9: Training performance comparison between `Batch` and `UCB-Con`, Llama3.1-8B-Instruct on the medium dataset (left) and Qwen2.5-7B on the easy dataset (right). For each subplot: Left: fraction of questions solved so far; Right: number of different answers sampled on the questions that the model has yet to solve (i.e., sample one correct answer historically). The x-axis denotes the number of gradient updates as we train all models fully on-policy. We repeat each experiment with 3 different random seeds and plot the mean performance.

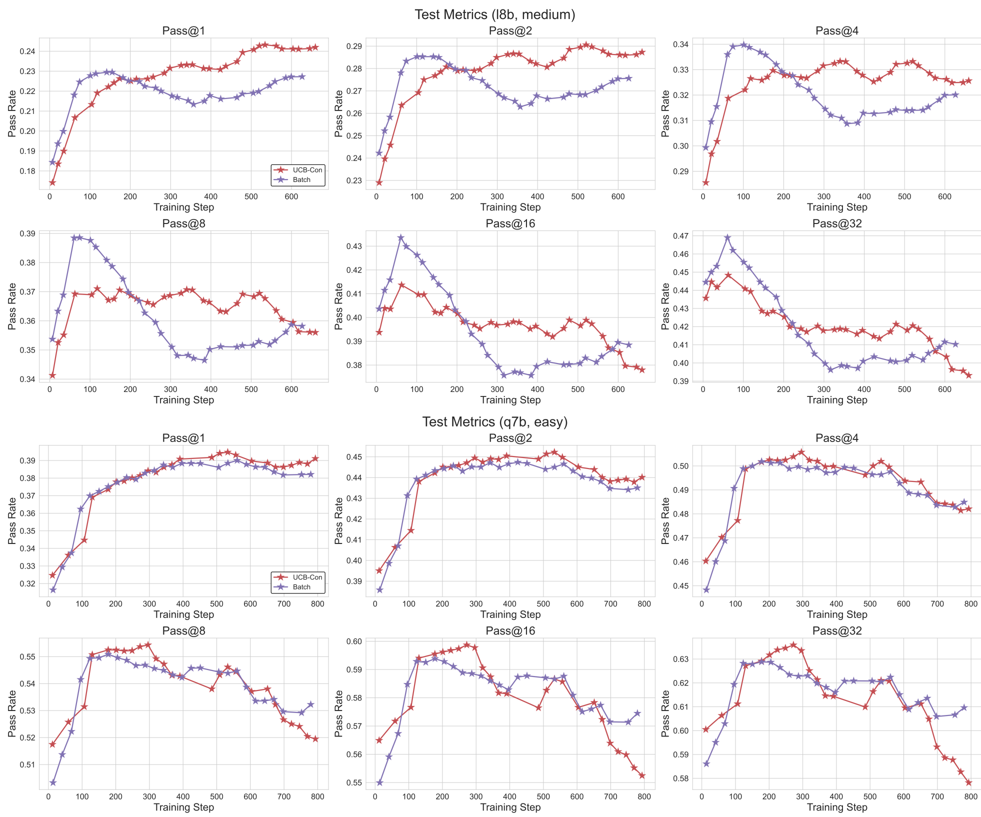

Figure 10: Test performance comparison between `Batch` and `UCB-Con`, with Llama3.1-8B-Instruct on the medium dataset (top) and Qwen2.5-7B on the easy dataset (bottom). We report pass@k for $k \in \{1, 2, 4, 8, 16, 32\}$ at every 20 training steps. We repeat each experiment with 3 different random seeds and plot the mean performance. The metrics are calculated based on 32 samples per question during evaluation.

C Quantitative Results

Table 3: Quantitative comparison of different baselines on pass@1 and pass@32 at the best checkpoint. The best results are in bold. Note that `UCB-Con` in general achieves the best peak performance.

Method	L8B				Q7B			
	Math		DAPO		Math		DAPO	
	Pass@1	Pass@32	Pass@1	Pass@32	Pass@1	Pass@32	Pass@1	Pass@32
Vanilla RL	0.231	0.465	0.218	**0.474**	0.385	0.634	0.403	0.627
`UCB`	0.236	0.472	0.214	0.473	0.379	0.635	0.397	0.627
`UCB-Mean`	0.239	0.462	0.223	0.473	0.387	0.618	0.407	0.633
`UCB-Con`	**0.242**	**0.473**	**0.243**	0.448	**0.395**	**0.636**	**0.419**	**0.642**
`Batch`	0.241	0.467	0.229	0.469	0.390	0.629	0.413	0.631

Table 4: Quantitative comparison of different baselines on pass@1 and pass@32 at the final checkpoint. The best results are in bold. Note that `Batch` in general achieves the best final performance in terms of pass@32.

Method	L8B				Q7B			
	Math		DAPO		Math		DAPO	
	Pass@1	Pass@32	Pass@1	Pass@32	Pass@1	Pass@32	Pass@1	Pass@32
Vanilla RL	0.215	0.395	0.196	0.362	0.381	0.593	0.399	0.580
`UCB`	0.233	0.425	0.208	0.388	0.372	0.582	0.392	0.580
`UCB-Mean`	0.233	0.414	0.221	0.372	0.387	0.586	0.407	**0.603**
`UCB-Con`	0.228	0.417	**0.242**	0.393	**0.391**	0.578	**0.419**	0.589
`Batch`	**0.238**	**0.426**	0.227	**0.410**	0.382	**0.610**	0.412	0.594

D Implementation Details

Our codebase is developed based on the verl codebase (Sheng et al., 2024). Thus we use the verl naming convention for the hyperparameters. For all our experiments, we use the hyperparameters in Table 5 unless otherwise specified. For all Llama experiments, we set bonus coefficient $c = 0.1$, and for all Qwen experiments, we set $c = 0.2$. For `UCB-Con`, we set $b_0 = 1$ for easy dataset and $b_0 = 0.5$ for medium dataset.

Table 5: Comparison of different exploration strategies based on the number of different answers sampled in a batch. We repeat for 2 random seeds and report the mean and standard deviation (in parentheses).

Name	Value
train batch size	256
learning rate	1e-6
ppo mini batch size	256
kl loss coef	0.001
entropy coeff	0
rollout.n	8
rollout.val_kwargs.temperature	1