Maximum Likelihood Reinforcement Learning

Fahim Tajwar[*1] **Guanning Zeng**[*2] **Yueer Zhou**[3] **Yuda Song**[1]
Daman Arora[1] **Yiding Jiang**[1] **Jeff Schneider**[1] **Ruslan Salakhutdinov**[1]
Haiwen Feng[4,5] **Andrea Zanette**[1]

[1]Carnegie Mellon University [2]Tsinghua University [3]Zhejiang University
[4]UC Berkeley [5]Impossible, Inc.

`{ftajwar,azanette}@andrew.cmu.edu`

Abstract

Reinforcement learning is the method of choice to train models in *sampling-based* setups with binary outcome feedback, such as navigation, code generation, and mathematical problem solving. In such settings, models implicitly induce a likelihood over correct rollouts. However, we observe that reinforcement learning does not maximize this likelihood, and instead optimizes only a lower-order approximation. Inspired by this observation, we introduce **Maximum Likelihood Reinforcement Learning (MaxRL)**, a sampling-based framework to approximate maximum likelihood using reinforcement learning techniques. MaxRL addresses the challenges of non-differentiable sampling by defining a compute-indexed family of sample-based objectives that interpolate between standard reinforcement learning and exact maximum likelihood as additional sampling compute is allocated. The resulting objectives admit a simple, unbiased policy-gradient estimator and converge to maximum likelihood optimization in the infinite-compute limit. Empirically, we show that MaxRL Pareto-dominates existing methods in all models and tasks we tested, achieving up to **20**× test-time scaling efficiency gains compared to its GRPO-trained counterpart. We also observe MaxRL to scale better with additional data and compute. Our results suggest MaxRL is a promising framework for scaling RL training in correctness based settings.[1]

1 Introduction

Maximum likelihood (ML) and reinforcement learning (RL) are two highly successful optimization paradigms that significantly shaped the landscape of modern machine learning. Maximum likelihood training is a foundational principle behind modern generative and predictive models (Bishop, 2006; Murphy, 2012); in fully differentiable settings, optimizing log-likelihood objectives has reliably translated increases in model capacity, data, and compute into consistent performance improvements (Krizhevsky et al., 2012; Radford et al., 2018). Reinforcement learning, by contrast, originated in optimal control and sequential decision-making (Bertsekas, 1995; Sutton et al., 1998), where learning proceeds through interaction with an environment and the objective is to maximize expected return. The generality of this formulation enables reinforcement learning to address problems involving *non-differentiable* intermediate sampling, and has led to models capable of superhuman performance in complex domains (Mnih et al., 2015; Silver et al., 2016).

Many modern learning problems are typically addressed with reinforcement learning even if they define an implicit likelihood over successes. Examples include navigation (Thrun et al., 2005; Anderson et al., 2018), program synthesis (Chen et al., 2018; Bunel et al., 2018), structured prediction (Smith, 2011; Mensch and Blondel, 2018), and multi-step reasoning in large language models (Wei et al., 2023). In these tasks, success is determined by an external verifier only after a stochastic generation process, yielding a binary outcome. From an end-to-end perspective, the model induces a probability of success for each input, defining an implicit likelihood over correctness. Maximizing this likelihood would be the principled approach, but non-differentiable intermediate sampling precludes direct optimization. Reinforcement learning is used instead, not because it offers a better objective, but as a workaround to this non-differentiability.

*Equal contribution.
[1]Project website and code: https://zanette-labs.github.io/MaxRL/

Suppose that both maximum likelihood and reinforcement learning could be applied to a given task, regardless of differentiability. The two objectives induce markedly different optimization behavior as we explain below. To make this distinction precise, we compare the population-level objectives implied by each framework. Let $p_\theta(x)$ denote the probability that a model with parameters θ produces a correct output for input x. The corresponding gradients of reinforcement learning, $\nabla_\theta J_{\mathrm{RL}}$, and of maximum likelihood, $\nabla_\theta J_{\mathrm{ML}}$, take the form

$$\nabla_\theta J_{\mathrm{RL}} = \mathbb{E}_x[\nabla_\theta p_\theta(x)]$$

$$\nabla_\theta J_{\mathrm{ML}} = \mathbb{E}_x[\nabla_\theta \log p_\theta(x)] = \mathbb{E}_x\left[\frac{1}{p_\theta(x)}\nabla_\theta p_\theta(x)\right]$$

The inverse-probability reweighting induced by maximum likelihood places greater emphasis on hard, low-success inputs, leading to very different optimization dynamics, as we empirically demonstrate in this paper (cf. Section 6).

When viewed end-to-end, maximum likelihood emerges as the principled objective, for the very same reasons it is the method of choice in differentiable supervised learning with binary correctness. In non-differentiable problems, however, this objective is difficult to optimize directly because correctness is observed only after a non-differentiable stochastic generation process, and the success probability $p_\theta(x)$ may be small. This computational challenge motivates a framework that leverages additional sampling compute to more faithfully approximate likelihood-based training. We call this framework **Maximum Likelihood Reinforcement Learning (MaxRL)**. At a high level, MaxRL bridges standard reinforcement learning and exact maximum likelihood by progressively incorporating higher-order correctness information as more sampling compute is utilized, ultimately recovering likelihood optimization in the infinite-compute limit. We discuss the relationship between MaxRL and recent work in this direction (Xiong et al., 2025b; Davis and Recht, 2025) in Section 7.

Our contributions are threefold:

1. We formalize correctness-based reinforcement learning as a latent-generation maximum likelihood problem and show that standard reinforcement learning optimizes only the first-order approximation of the maximum likelihood objective (cf. Section 3).

2. We introduce a compute-indexed family of objectives that interpolates between expected reward and exact maximum likelihood via a Maclaurin expansion in pass@k events (cf. Section 3).

3. We derive a simple on-policy estimator whose expected gradient exactly matches the compute-indexed approximation of the likelihood objective, implying that increased sampling improves the optimized objective itself rather than merely reducing variance.

Empirically, MaxRL Pareto-dominates standard RL objectives (RLOO (Ahmadian et al., 2024), GRPO (Shao et al., 2024)) on all settings that we tested on (cf. Section 6). Specifically, MaxRL show better scaling trends when additional compute and data are available, and on reasoning tasks, it achieves up to 20× test-time scaling efficiency gains with a perfect verifier. Together, our results illustrate MaxRL to be a promising direction for scaling RL in correctness based domains.

2 Preliminaries

In this work, we consider reinforcement learning settings that involve *generalization*, where models learn from a set of tasks and are evaluated on a heldout task distribution. We focus on correctness-based problems that can be abstracted as a *binary success or failure* outcome for each input. Formally, let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output spaces, and let $x \sim \rho$ be the distribution over tasks. For each input x, we denote $y^*(x) \in \mathcal{Y}$ as the correct label or answer. Equality between outputs is defined up to a task-dependent equivalence relation, so that $y = y^*(x)$ denotes semantic correctness rather than exact output equality. Finally, we let the learner be parameterized by θ and denote the predictive distribution induced by the model as $p_\theta(y \mid x)$, where $p_\theta(\cdot \mid x) \in \Delta(\mathcal{Y})$ is a conditional probability distribution over outputs for a fixed input x. In mathematical reasoning, x is the prompt and y is the final solution produced by the model. All logarithms use base e unless stated otherwise.

Latent generation models. In many modern settings, the model does not sample outputs directly from $\mathcal{Y}$, but instead generates a latent variable $z \in \mathcal{Z}$ according to a conditional distribution $m_\theta(z \mid x)$.

The final output $y \in \mathcal{Y}$ is then obtained via a deterministic decoding function $y = f(z)$, such as parsing a generated program or extracting a boxed answer from a chain of thought. Correctness is evaluated only on the decoded output, i.e., a trajectory z is successful if $f(z) = y^*(x)$. This induces a marginal probability of correctness

$$p_\theta(y^*(x) \mid x) = \sum_{z \in \mathcal{Z}} m_\theta(z \mid x)\, \mathbb{I}\{f(z) = y^*(x)\}. \tag{1}$$

Throughout the paper, expectations with respect to model outputs should be understood as expectations over latent samples $z \sim m_\theta(\cdot \mid x)$ followed by deterministic decoding.

Pass rate. We define the *pass rate* as the probability that the model produces the correct answer for a fixed input x:

$$p_\theta^{\text{pass}}(x) := p_\theta(y^*(x) \mid x) = \mathbb{E}_{y \sim p_\theta(\cdot|x)}[\mathbb{I}\{y = y^*(x)\}]\,.$$

Similarly, let $y_1, \ldots, y_k \overset{\text{i.i.d.}}{\sim} p_\theta(\cdot \mid x)$. We define pass@$k$ as the probability of at least one correct sample:

$$\text{pass@}k(x) := \mathbb{P}(\exists\, i \in [k] \text{ s.t. } y_i = y^*(x))\,.$$

Next, we consider two optimization frameworks for training our models: *maximum likelihood* and *reinforcement learning.*

Maximum likelihood (ML). Maximum likelihood selects parameters that maximize the log-probability of the observed data under the model. In our binary correctness setting, each input x yields a single binary observation indicating whether the model produces a correct output. Under the latent generation model $z \sim m_\theta(\cdot \mid x)$ with deterministic decoding $y = f(z)$, the probability of observing correctness is $p_\theta(y^*(x) \mid x)$. Maximizing the corresponding log-likelihood therefore yields the objective

$$J_{\text{ML}}(\theta) := \mathbb{E}_{x \sim \rho}\left[\log p_\theta(y^*(x) \mid x)\right] \quad \text{with} \quad p_\theta(y^*(x) \mid x) = \mathbb{E}_{z \sim m_\theta(\cdot|x)}[\mathbb{I}\{f(z) = y^*(x)\}]\,, \tag{2}$$

which is directly analogous to cross-entropy training in differentiable supervised learning.

Reinforcement learning (RL). For correctness based tasks, we also define a binary reward function $r(x, y) = \mathbb{I}\{y = y^*(x)\}$, and similarly under the latent variable case define $r(x, z) = \mathbb{I}\{f(z) = y^*(x)\}$. In this binary reward setting, the RL objective becomes (using the latent version without loss of generality):

$$J_{\text{RL}}(\theta) := \mathbb{E}_{x \sim \rho}\left[\mathbb{E}_{z \sim m_\theta(\cdot|x)}\left[r(x, z)\right]\right] = \mathbb{E}_{x \sim \rho}\left[p_\theta^{\text{pass}}(x)\right]. \tag{3}$$

This gives an objective equivalent to maximizing the expected population pass rate directly.

3 Maximum Likelihood Reinforcement Learning (MaxRL)

In this section, we show that reinforcement learning on expected reward optimizes only a first-order approximation of the ML objective. Specifically, the maximum likelihood objective admits a population-level expansion in terms of pass@k events, with standard RL optimizing only the first-order term. This suggests a compute-indexed family of objectives that incorporate higher-order terms, converging to ML as more compute is allocated.

3.1 Maclaurin Expansion of Maximum Likelihood

For simplicity, let us consider a single task x in the development to follow, as the final objective and gradients can be obtained by taking an expectation over $x \sim \rho$. Moreover, we write $p := p_\theta^{\text{pass}}(x)$ to simplify our notation. The maximum likelihood objective admits the *Maclaurin expansion* in terms of failure events:

$$J_{\text{ML}}(x) = \log p = -\sum_{k=1}^{\infty} \frac{(1-p)^k}{k} = -\sum_{k=1}^{\infty} \frac{\text{fail@}k(x)}{k}, \tag{4}$$

where fail@$k(x) = 1 - \text{pass@}k(x)$ denotes the probability that all k i.i.d. samples from the model fail. Differentiating (4) yields the population-level gradient identity

$$\boxed{\nabla_\theta J_{\text{ML}}(x) = \sum_{k=1}^{\infty} \frac{1}{k} \nabla_\theta \text{pass@}k(x).} \tag{5}$$

Thus, maximum likelihood optimizes an infinite harmonic mixture of pass@k gradients, with higher-order terms encoding rare success which are critical when p is small. In contrast, the classical reinforcement learning approach is to optimize only the expected pass@1 objective (Koenig and Simmons, 1993; Silver et al., 2016; Vecerik et al., 2018; Guo et al., 2025):

$$\nabla_\theta J_{\text{RL}}(x) = \nabla_\theta \text{pass@}1(x),$$

corresponding to retaining solely the leading term of (5). From this observation, we can claim:

Reinforcement learning optimizes a first-order approximation of the maximum likelihood objective.

3.2 MaxRL Objective Function

The maximum likelihood gradient in (5) is difficult to estimate with finite samples. In particular, estimating pass@k gradients for large k requires an increasing number of samples, especially when the pass rate p is small. This finite-sample difficulty is precisely what motivates Maximum Likelihood Reinforcement Learning. We define MaxRL as the *class of reinforcement-learning methods that explicitly target the maximum likelihood objective* rather than the pass rate, while remaining implementable under finite sampling and non-differentiable generation. We consider a principled way to do so below.

Consider approximating the maximum likelihood objective by truncating the Maclaurin expansion (5) to a finite order and then estimating such an objective instead. For a truncation level $T \in \mathbb{N}$, we define the truncated maximum likelihood objective for a fixed input x as

$$J^{(T)}_{\text{MaxRL}}(x) := -\sum_{k=1}^{T} \frac{(1-p)^k}{k}. \tag{6}$$

Differentiating (6) yields the truncated population gradient

$$\nabla_\theta J^{(T)}_{\text{MaxRL}}(x) = \sum_{k=1}^{T} \frac{1}{k} \nabla_\theta \text{pass@}k(x). \tag{7}$$

This defines a family of objectives: **$\mathbf{T = 1}$ recovers reinforcement learning, $\mathbf{T \to \infty}$ recovers maximum likelihood**, and intermediate T values interpolate between them. Thus, the truncation level T directly controls the order of correctness events that contribute to learning. As we will soon see, it becomes viable to estimate higher-order $\nabla_\theta J^{(T)}_{\text{MaxRL}}(x)$ as more compute is expended in terms of rollouts. In other words:

MaxRL provides a principled framework for trading additional compute for higher-fidelity approximations to the maximum likelihood objective.

The remaining question is whether these truncated objectives admit simple, unbiased estimators under finite sampling, a question that we answer affirmatively in the next section.

4 Gradient Estimators for MaxRL

Eq. (7) already provides a viable approach for constructing an unbiased estimator: approximate *each* term in the finite series using a pass@k gradient estimator, as provided in recent work (Walder and Karkhanis, 2025; Chen et al., 2025d). Under this strategy, any improvement in pass@k estimators directly translates into improved estimators for the truncated maximum likelihood objective in Eq. (7).

In this work, we take an alternate approach, one that will lead to a simpler estimator and to a new viewpoint. The key insight is that the maximum likelihood gradient can be expressed as an expectation under the *success-conditioned* distribution (Davis and Recht (2025) also recently made a similar observation), as established by the following theorem. We provide the proof in Appendix B.

Theorem 1 (Conditional Form of the Maximum Likelihood Gradient)**.** *The gradient of the maximum likelihood objective admits the following conditional expectation representation:*

$$\nabla_\theta J_{\mathrm{ML}}(x) = \mathbb{E}[\nabla_\theta \log m_\theta(z \mid x) \mid f(z) = y^*(x)] \,. \tag{8}$$

The theorem establishes that the maximum likelihood gradient is the average gradient from successful trajectories only. This interpretation naturally leads to a concrete gradient estimator by replacing the expectation with sample averages.

4.1 Empirical Gradient Estimator

Theorem 1 suggests drawing samples from the success-conditioned policy. Recent works have proposed rejection fine-tuning (Touvron et al., 2023; Yuan et al., 2023; Dong et al., 2023; Xiong et al., 2025a; Davis and Recht, 2025) and adaptive sampling (Xiong et al., 2025b) as mechanisms to sample from this conditional distribution. However, doing so is computationally demanding when the pass rate is small or requires a more complex implementation regarding adaptive sampling. Instead, we adopt a simpler approach: we sample from the unconditional policy $m_\theta(\cdot \mid x)$ and then *average over only the successful trajectories.* Fix an input x and draw N latent trajectories $z_1, \dots, z_N \sim m_\theta(\cdot \mid x)$. Let $r_i := \mathbb{I}\{f(z_i) = y^*(x)\}$ indicate success based reward, $S_i := \nabla_\theta \log m_\theta(z_i \mid x)$ denote the score function, and $K := \sum_{i=1}^N r_i$ be the number of successful samples. We average score functions *only over successful trajectories* and obtain the following REINFORCE-style estimator:

Table 1: Comparison of the REINFORCE estimator ($\nabla_\theta J_{\mathrm{RL}}(x)$) and the conditional estimator ($\nabla_\theta J^{(T)}_{\textsc{MaxRL}}(x)$). Although they differ only in normalization (N vs. K), they are unbiased for fundamentally different objectives.

	REINFORCE	**MaxRL**
Estimator	$\frac{1}{N}\sum_{i=1}^N r_i S_i$	$\frac{1}{K}\sum_{i=1}^N r_i S_i$
Unbiased for	∇_θ pass@1(x)	$\sum_{k=1}^N \frac{1}{k} \nabla_\theta \text{pass@}k(x)$

$$\widehat{g}_N(x) := \begin{cases} \dfrac{1}{K}\displaystyle\sum_{i=1}^N r_i S_i, & K \ge 1, \\ 0, & K = 0. \end{cases} \tag{9}$$

The estimator constructed in this way is such that some inputs may receive zero gradient if there are no successes within N samples, making the resulting estimator no longer unbiased with respect to Eq. (9). We show that this estimator is however unbiased for the gradient of the truncated maximum likelihood objective in Eq. (7), $\nabla_\theta J^{(T)}_{\textsc{MaxRL}}(x)$, with truncation level $T = N$:

Theorem 2 (Estimator–objective equivalence)**.** *The estimator $\widehat{g}_N(x)$ is an unbiased estimator for the* MaxRL *gradient of order $T = N$, i.e.,*

$$\mathbb{E}[\widehat{g}_N(x)] = \nabla_\theta J^{(N)}_{\textsc{MaxRL}}(x).$$

We present the proof of this result in Appendix B. Theorem 2 reveals an elegant alignment between the estimator in Eq. (9) and the gradient of the truncated Maclaurin expansion in Eq. (7). It is worth highlighting the most important property of the estimator:

Increasing compute as rollouts N leads to a better approximation of the maximum likelihood gradient.

Table 1 compares our estimator with the REINFORCE[2] estimator, whose expected value underlies most RL algorithms. At the estimator level, the difference is simple: both average score functions over

[2]Modern policy-gradient methods such as PPO (Schulman et al., 2017) introduce additional mechanisms (importance weight truncation via clipping) that trade bias for robustness. In the fully on-policy setting, these reduce to REINFORCE, our canonical baseline. GRPO (Shao et al., 2024) is a notable exception due to its division by standard deviation in the advantage calculation, which we discuss further in Section 5.

sampled trajectories, but REINFORCE normalizes by total samples N while MaxRL normalizes by successful samples K. This difference in normalization determines the objective each estimator is unbiased for.

Consequently, increasing number of samples, N, for the two estimators has different effects: REINFORCE reduces variance of a fixed objective (pass@1), while MaxRL increases the approximation order to maximum likelihood. Additional compute thus improves the *objective itself* for MaxRL, not just estimation quality.

4.2 Variance Reduction via Control Variates

Like REINFORCE, the estimator (9) can exhibit high variance when successful samples K is small. Policy-gradient baselines are typically introduced to reduce variance without changing the expected gradient (Sutton, 1988). However, standard arguments for policy-gradient baselines are not directly applicable in this setting, as the estimator normalizes by the random variable K which depends on all samples which makes it correlated with the observed rollouts.

We instead proceed from first principles and use a simple zero-mean control variate, the unconditional average score:

$$V_N := \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log m_\theta(z_i \mid x),$$

which satisfies $\mathbb{E}[V_N] = 0$. Subtracting V_N preserves unbiasedness while reducing variance:

$$\widetilde{g}_N(x) = \frac{1}{K}\sum_{i=1}^{N} r_i S_i - \frac{1}{N}\sum_{i=1}^{N} S_i = \sum_{i=1}^{N}\left(\frac{r_i}{K} - \frac{1}{N}\right) S_i, \tag{10}$$

with the convention that the first term, $\left(\sum_{i=1}^{N} r_i S_i\right)/K$, is zero when $K = 0$.

4.3 On-Policy Implementation

In Algorithm 1, we present a simple *on-policy* implementation of MaxRL that differs from standard REINFORCE-style policy gradient methods by a *single-line* modification to the advantage calculation. We adopt the variance-reduced formulation in Eq. (10), and drop both terms when $K = 0$, consistent with standard policy-gradient practice in which no gradient is computed for tasks with no successful rollouts: this choice is simpler and performs better empirically. Concretely, the advantage is normalized by the per-task mean reward, rather than left unnormalized as in RLOO (Ahmadian et al., 2024) or normalized by the reward standard deviation as in GRPO (Shao et al., 2024). The modified line is highlighted in blue.

Algorithm 1 On-Policy Implementation of MaxRL.

require Batch of inputs B, number of rollouts N, latent policy $m_\theta(\cdot \mid \cdot)$
for each input $x \in B$ **do**
  Sample $z_1, \ldots, z_N \sim m_\theta(\cdot \mid x)$
  for $j = 1$ to N **do**
    $r_j \leftarrow \mathbb{I}\{f(z_j) = y^*(x)\}$
    $S_j \leftarrow \nabla_\theta \log m_\theta(z_j \mid x)$
  end for
  $\widehat{r}(x) \leftarrow \frac{1}{N}\sum_{j=1}^{N} r_j$
  $\widehat{g}(x) \leftarrow \begin{cases} \dfrac{1}{N\,\widehat{r}(x)} \displaystyle\sum_{j=1}^{N}(r_j - \widehat{r}(x))S_j, & \widehat{r}(x) > 0, \\ 0, & \text{otherwise} \end{cases}$
end for
$\widehat{g} \leftarrow \frac{1}{|B|}\sum_{x \in B} \widehat{g}(x)$
return $\widehat{g}$

5 A Unifying Weight-Function View

Maximum likelihood, MaxRL, classical reinforcement learning, and GRPO all admit population-level gradients of the form

$$\nabla_\theta J = \mathbb{E}_{x\sim\rho}[w(p_\theta(x))\ \nabla_\theta p_\theta(x)]\,, \tag{11}$$

where $p_\theta(x) = p_\theta^{\text{pass}}(x)$ and $w(p)$ is a scalar weight that depends only on the pass rate. The function $w(p)$ determines how learning signal is allocated across inputs of varying difficulty and fully characterizes the differences between these objectives at the population level. Figure 1 plots the resulting weight functions for each method; derivations are provided in Appendix C. The key distinction among objectives is *how strongly they emphasize hard, low-pass-rate inputs.* As T increases, MaxRL uniquely approaches maximum likelihood weighting in the low pass rate regime.

This weight perspective also provides a useful reinterpretation of GRPO (Shao et al., 2024). Although GRPO is heuristically motivated by Z-normalization using the empirical standard deviation, such normalization induces a fundamentally different population-level objective from REINFORCE, a conclusion also reached by recent works (Davis and Recht, 2025; Liu et al., 2025b; Xiong et al., 2025b). Table 2 summarizes the population-level weighting functions; relative to standard expected-reward optimization, GRPO upweights low-pass-rate inputs approximately as $1/\sqrt{p}$ when p is small, placing it between classical reinforcement learning and maximum likelihood. However, increasing compute via additional sampling under GRPO does not yield a better approximation to the maximum likelihood objective, as the induced population loss is fundamentally distinct. Moreover, as shown in Figure 1, the GRPO weighting function *inverts* for sufficiently large pass rates, increasing as $p \to 1$, unlike likelihood-based objectives. Consequently, GRPO assigns increased weight to very easy inputs when they are present, in contrast to the other formulations.[3]

Table 2: Population-level weighting functions $w(p)$.

	RL	GRPO	MaxRL (T)	ML
$w(p)$	1	$\frac{1}{\sqrt{p(1-p)}}$	$\frac{1-(1-p)^T}{p}$	$\frac{1}{p}$

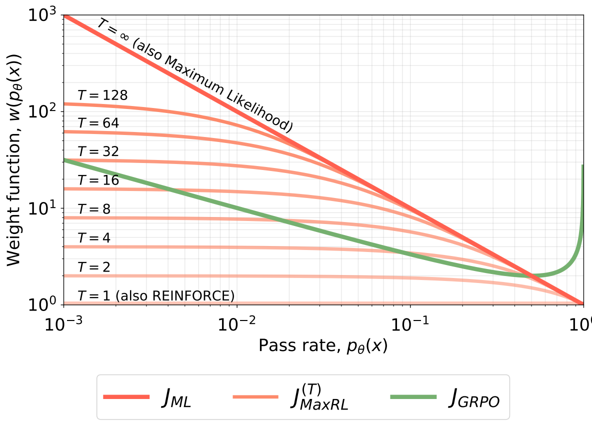

Figure 1: Population-level weighting functions $w(p)$ as a function of pass rate p. Truncated objectives $J^{(T)}_{\text{MaxRL}}$ interpolate between REINFORCE and maximum likelihood as T increases.

6 Experiments

We now turn to the empirical evaluation of MaxRL. We begin in Section 6.1 with a controlled setting where exact maximum likelihood optimization is possible, allowing direct comparison with MaxRL as compute increases. We then study non-differentiable correctness-based tasks in two regimes: (i) an *effectively infinite-data* setting with large number of novel tasks (Section 6.2), and (ii) a *data-scarce* setting with a fixed training dataset where we can scale compute nonetheless by training for many epochs over the same dataset (Section 6.3). Finally, in Section 6.4, we train and evaluate billion-parameter reasoning models on mathematical problem-solving, testing whether the benefits of MaxRL extend to larger-scale LLM training.

Because we compare training objectives rather than algorithms, all methods are trained *on-policy*. We compare against REINFORCE with a leave-one-out baseline (RLOO) (Ahmadian et al., 2024) and Group Relative Policy Optimization (GRPO) (Shao et al., 2024) as primary baselines.

6.1 Comparisons with Exact Maximum Likelihood

As a first step, we evaluate how closely MaxRL approximates *exact* maximum likelihood in a setting where the latter can be implemented exactly. We compare three objectives: (i) reinforcement learning on expected reward, (ii) MaxRL, and (iii) exact maximum likelihood training. We consider a standard image

[3]We conjecture that this inversion may contribute to distribution sharpening (Yue et al., 2025; Wu et al., 2026) when datasets contain a substantial fraction of overly easy inputs, and leave a detailed analysis to future work.

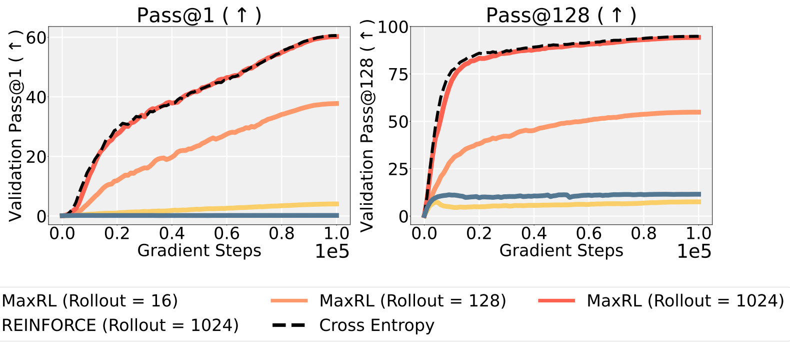

Figure 2: (ImageNet) Comparison of training dynamics under exact maximum likelihood, MaxRL, and REINFORCE in a controlled image classification setting. With sufficient rollouts, MaxRL closely matches cross-entropy training, while REINFORCE fails to make progress from low initial pass rates even with high number of rollouts.

classification task, where maximum likelihood corresponds to minimizing cross-entropy. The reinforcement learning reward is defined as 1 if the predicted class matches the ground-truth label and 0 otherwise.

We instantiate this comparison on ImageNet (Deng et al., 2009) using a ResNet-50 (He et al., 2015) trained under each objective; full experimental details are provided in Appendix D. Figure 2 summarizes the results: REINFORCE (with a standard baseline) fails to achieve meaningful improvements even with very high per-input sampling budget, whereas exact maximum likelihood training yields steady gains.

In contrast, MaxRL is trained on the same samples and observes the same sparse set of successful trajectories as REINFORCE, but makes more effective use of this limited learning signal through likelihood-inspired reweighting. As the compute increases by means of higher rollout counts, *MaxRL improves consistently and closely tracks exact maximum likelihood.* We also analyze the gradient norm resulting from different objectives in Figure 8: MaxRL and cross-entropy concentrate learning signal on harder tasks and are characteristically similar given sufficient compute for MaxRL, whereas GRPO and REINFORCE exhibit very different behavior. For additional experiments such as comparison to GRPO, we refer the reader to Appendix D.

Takeaway 1: MaxRL approaches exact maximum likelihood given infinite compute.

When direct maximum-likelihood optimization is available, MaxRL converges to it as sampling compute increases.

6.2 Infinite Data Regime

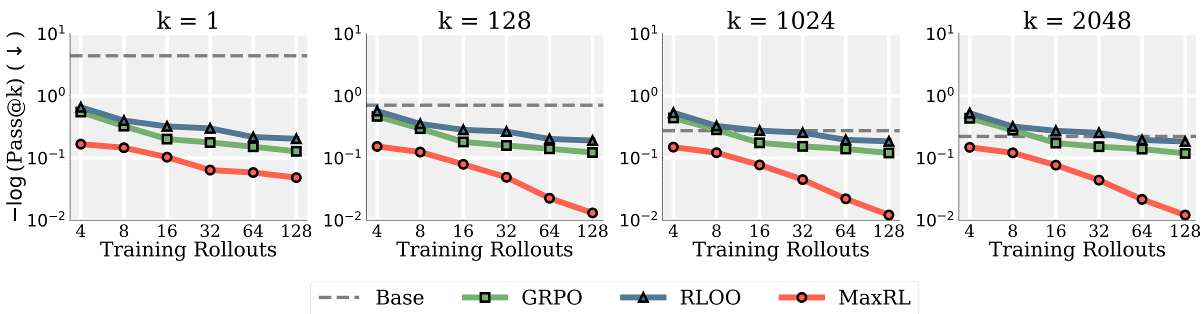

Figure 3: (Maze) Motivated by Schaeffer et al. (2025), we record $-\log(\text{Pass@}k)$ (lower is better) as a function of training rollouts for different objectives. We see that for across different all inference rollout budgets (k), MaxRL exhibit better scaling compared to GRPO and RLOO as we increase number of training rollouts.

Next, we study MaxRL in non-differentiable settings. For the first experiment, we study how MaxRL behave in data-rich domains. To simulate training with infinite data, we construct a procedurally generated maze-navigation environment with 1 million unique 17×17 mazes for training where multiple valid solution paths might exist for a given task. We reserve a held-out set of 256 mazes for evaluation and apply a brief supervised pretraining phase to ensure a non-zero initial pass rate. The complete details of the task are provided in the Appendix E.1.

We train a lightweight transformer model (Vaswani et al., 2017) with approximately 3M parameters and simulate extended training by running 9K RL steps with up to 128 rollouts per prompt, varying the number of rollouts to control compute. We report performance after 9K steps in Figure 3 as a function of training compute, implemented as different number of rollouts per prompt from 4 to 128. Notice that this is a substantial amount of compute for the size of the model.

All three objectives (RLOO, GRPO, and MaxRL) improve upon the base model, but the magnitude of improvement differs markedly across methods. Even at the highest compute budget (128 rollouts per prompt), RLOO fails to match the performance achieved by MaxRL at the lowest budget (4 rollouts) across all pass@k metrics. GRPO-trained models at the highest compute similarly trail MaxRL trained with only 16 rollouts per prompt at every pass@k evaluation. These results highlight that MaxRL *scales with compute* far more effectively than competing frameworks when large amounts of unique training data is available. Comparisons with additional baselines are provided in Table 3 and Appendix L.

Table 3: Performance comparison in maze with additional baselines.

Method	Pass@1	Pass@128	Pass@256
GRPO (Shao et al., 2024)	43.6	49.0	49.6
RLOO (Ahmadian et al., 2024)	25.2	28.7	29.0
GRPO (with entropy bonus)	47.2	53.4	54.0
PKPO (T = 16) (Walder and Karkhanis, 2025)	74.5	77.6	77.9
Differential Smoothing (Gai et al., 2025)	50.0	57.8	58.7
MaxRL	**84.4**	**92.0**	**94.3**

Takeaway 2: MaxRL scales better with additional compute in the infinite data regime.

In a data-rich training regime, MaxRL scales more favorably with additional compute compared to existing methods.

6.3 Data-Scarce Regime

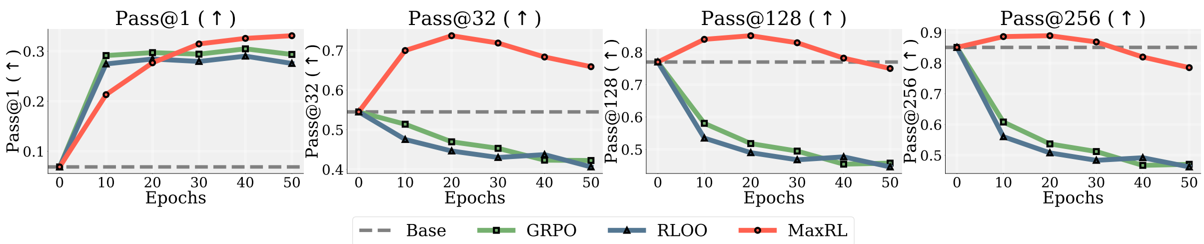

Figure 4: (GMS8K) Training dynamics on GSM8K with a fixed dataset and increasing training compute in terms of RL steps. MaxRL shows slower initial gains but ultimately achieves higher performance with substantially less pass@k degradation compared to GRPO and REINFORCE.

We next consider a data-scarce regime where models are trained for many epochs on a fixed dataset until peak performance is reached, with the goal of identifying which method extracts the highest attainable performance. Unlike the infinite-data setting in Section 6.2, longer training in this regime does not necessarily translate into improved performance due to the increased risk of overfitting. Specifically, we train a SmolLM2-360M-Instruct model (Allal et al., 2025) on GSM8K (Cobbe et al., 2021) for up to 50 epochs. Training dynamics are reported in Figure 4, with additional experimental details provided in Appendix E.2.

All methods improve upon the base model in terms of pass@1 performance; however, only MaxRL consistently exceeds the base model in pass@k metrics. In contrast, RLOO and GRPO exhibit massive pass@k degradation with extended training, mirroring the behavior observed by Yue et al. (2025) and exacerbated here by prolonged training on a fixed dataset. RLOO and GRPO reach their peak pass@1 performance faster than MaxRL (around 10 epochs), but MaxRL overtakes competing methods at approximately 30 epochs and continues to increase through the end of training, reaching a higher peak. At the same time, pass@k remains substantially healthier for MaxRL and exceeds the base model for a large portion of training. This behavior suggests *MaxRL is more resistant to overfitting*, particularly with regards to output diversity. Additional comparisons with baseline methods are reported in Table 4.

Table 4: Performance comparison across methods on GSM8K.

Method	Pass@1	Pass@128	Pass@1024
GRPO (Shao et al., 2024)	29.3	45.8	48.8
RLOO (Ahmadian et al., 2024)	27.5	44.6	48.5
GRPO (with entropy bonus)	31.1	48.1	51.6
PKPO (T = 16) (Walder and Karkhanis, 2025)	30.7	67.2	75.9
Differential Smoothing (Gai et al., 2025)	31.4	48.5	52.3
MaxRL	**33.2**	**75.0**	**83.4**

Takeaway 3: MaxRL is more resistant to overfitting.

In a data-scarce regime, MaxRL can sustain improvement over a large number of epochs, demonstrating less pass@k degradation (overfitting) and converging to a higher average performance.

6.4 Large Reasoning Model Training

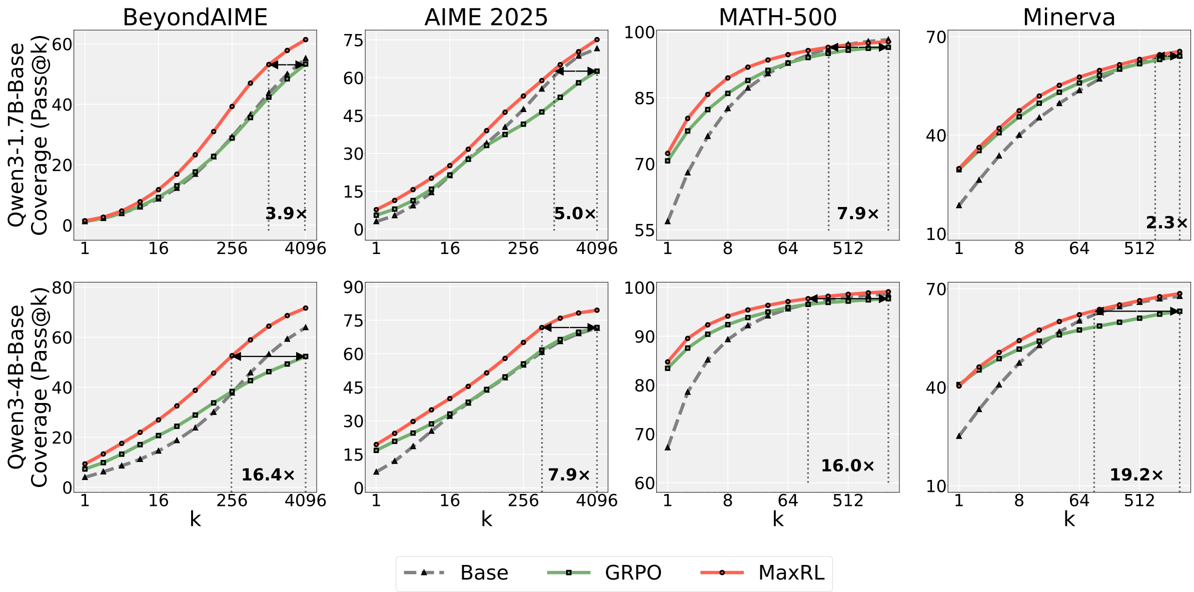

Figure 5: (Qwen3 training results) Evaluation of final checkpoints from training Qwen3-1.7B-Base and Qwen3-4B-Base models, on 4 benchmarks: AIME 2025, BeyondAIME, MATH-500 and Minerva. MaxRL match or ourperform GRPO in all 4 evaluation datasets and shows little to no degradation at coverage (pass@k) for very high k values. We also note the increase in inference efficiency: MaxRL can provide 2.3× - 19.2× speedup compared to GRPO while generating multiple samples with a perfect verifier and maintains similar or better pass@1 performance.

We next demonstrate that the benefits of MaxRL extend to larger-scale LLM reasoning training. We train Qwen3-1.7B-Base and Qwen3-4B-Base models on POLARIS-53K (An et al., 2025), a dataset of approximately 50K mathematical reasoning prompts, using 256 prompts per batch, 16 rollouts per prompt, and 1000 RL steps. Notice that this setup utilizes lower rollout counts and RL steps than prior experiments to allow training larger models within our compute budget. We evaluate on four standard math benchmarks: AIME 2025, BeyondAIME (ByteDance-Seed, 2025), MATH-500 (Hendrycks et al., 2021; Lightman et al., 2023), and Minerva (Lewkowycz et al., 2022). We compare against GRPO (Shao et al., 2024), a widely used baseline for large-scale reasoning (Guo et al., 2025; Yang et al., 2025). Additional details are provided in Appendix E.3.

Figure 5 summarizes our main results. Across both model sizes, MaxRL consistently Pareto dominates GRPO, achieving higher pass@1 while simultaneously improving pass@k. Consistent with prior work (Yue et al., 2025; Wu et al., 2026), GRPO exhibits pronounced pass@k degradation at larger k. In contrast, MaxRL improves pass@k relative to both the pretrained base model and the GRPO-trained checkpoint in 7 out of 8 evaluation settings. Improved pass@k directly translates into inference efficiency under repeated sampling. As shown in Figure 5, MaxRL achieves up to 20× test-time scaling efficiency gains when using a perfect verifier to filter wrong solutions, yielding substantial practical savings at inference time. We note that many settings admit a strong verifier, such as programming (Chen et al., 2021) or Lean (de Moura et al., 2015), where MaxRL can show strong benefits over other RL objectives. We provide additional results, such as evaluation on 4 additional benchmarks, and training dynamics statistics such as mean generated response length, entropy and gradient norm, in Appendix H, K, and J.

Takeaway 4: MaxRL's benefits transfer to larger scale mathematical reasoning

On larger scale mathematical reasoning, MaxRL Pareto-dominates GRPO, shows little to no diversity degradation with respect to the base model, and leads to strong (up to 20×) test-time scaling efficiency gains.

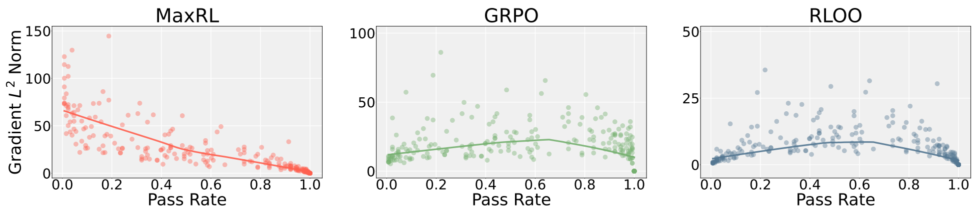

Figure 6: (Gradient norm analysis) To compare different objectives qualitatively, we show a scatter plot of gradient L^2 norm vs pass rate over individual prompts. We use Qwen2.5-1.5B-Instruct on MATH-500 dataset for this analysis. MaxRL generate larger gradient norms over prompts with close to 0 pass rates.

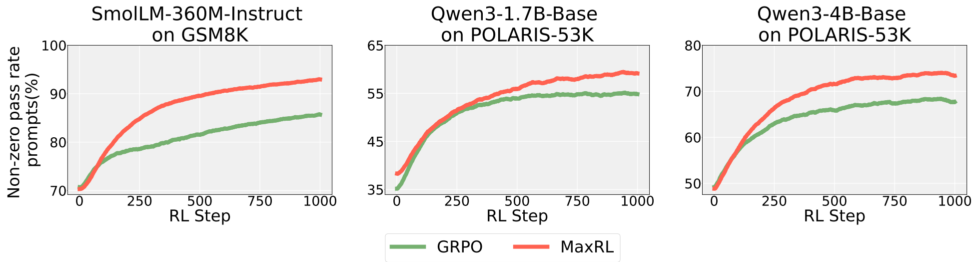

Figure 7: (Training dynamics comparison) Fraction of prompts where the model generates at least one correct rollout (out of 128, 16, and 16 rollouts for SmolLM-360M-Instruct, Qwen3-1.7B-Base, and Qwen3-4B-Base, respectively) during training. MaxRL consistently produces at least one correct rollout for more prompts across all settings, demonstrating its effectiveness at extracting more learning signal from the training dataset.

6.5 MaxRL Behaves Characteristically Different from Other RL Objectives

Finally, we study whether MaxRL show different characteristics compared to commonly used RL objectives besides performance metrics. We first study the gradient norms produced by different objectives: Figure 6 illustrates that MaxRL generates higher gradient norms on harder prompts and lower gradient norms on easier prompts. This behavior matches that of cross-entropy in fully differentiable image classification setting (Figure 8): showing that MaxRL concentrate learning signal on harder problems unlike GRPO and RLOO. This larger gradient norms on more difficult prompts then translates into the model's capability to generate correct solutions for a larger fraction of problems during training: Figure 7 demonstrates this for 3 different base models where MaxRL consistently generate at least one correct rollout for a larger fraction of training prompts, and the gap between MaxRL and GRPO persists as we train longer. We have also run this analysis for our maze and GSM8K training settings: Section I shows similar behavior in these setups as well. Overall, we demonstrate that MaxRL exhibit interesting differences compared to other RL objectives, and we leave their further study to future work.

Takeaway 5: MaxRL shows characteristically different optimization dynamics.

Besides performance metrics, MaxRL also exhibits different optimization dynamics. Most notably, it produces stronger gradients on harder prompts, and also leads to a larger fraction of prompts with at least one correct rollout during training.

7 Related Works

Supervised training vs reinforcement learning. Supervised learning and reinforcement learning (RL) are complementary but fundamentally different paradigms. Supervised training is stable, sample-efficient, and well-calibrated within the training distribution (Ng and Jordan, 2001), but it is limited by the quality and scope of available data and cannot directly optimize non-differentiable objectives such as correctness or preferences. In contrast, RL can optimize such objectives directly, typically via policy gradients (Williams, 1992; Sutton et al., 1999, 1998; Schulman et al., 2017; Guo et al., 2017), and improve performance beyond available demonstrations by having access to interactions with an environment and the resulting reward-based feedback. Although recent work has reframed RL objectives as supervised ones (Rafailov et al., 2023), on-policy learning, characteristic of online RL algorithms, appears crucial for optimal performance (Tajwar et al., 2024; Xu et al., 2024). Modern foundation model training often combines supervised learning on human data with subsequent RL (Ouyang et al., 2022). Unlike these approaches, we assume no access to high-quality demonstrations or a stronger model, and instead study a purely interactive RL setting that nonetheless optimizes an objective that mimics cross-entropy. We discuss additional related works in Appendix A.

Training LLMs for strong reasoning abilities. Reinforcement learning from verifiable rewards (RLVR), where LLMs receive reward from a ground truth verifier instead of using a trained reward model, has emerged as the dominant paradigm for instilling strong reasoning capabilities into LLMs (OpenAI et al., 2024; Guo et al., 2025; Team et al., 2025; Lambert et al., 2025; Yang et al., 2025). Whereas supervised training learns better behavior from fixed static datasets, reinforcement learning uses policy gradient algorithms (e.g., PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), RLOO (Ahmadian et al., 2024)) to learn from self-generated responses and non-differentiable rewards. However, these algorithms and their variants (Zheng et al., 2025; Liu et al., 2025b; MiniMax et al., 2025) optimize expected reward or pass rate and only differs in how the advantage or off-policy updates are calculated. In contrast, the goal of our work is to propose a fundamentally different objective for RL training.

RL training causes distribution sharpening. Despite its usefulness, questions remain on whether RLVR teaches LLMs fundamentally new behavior/skills, or simply sharpens existing good behavior from the pretrained model. Prior works (Liu et al., 2025b; Zhao et al., 2025; AI et al., 2025) demonstrated that certain reasoning skills like reflection already exist in the pretrained model, and Gandhi et al. (2025) shows that good reasoning behaviors learned from pretraining is crucial for the success of RLVR in the post-training phase. More recently, studies (Yue et al., 2025; Dang et al., 2025; Wu et al., 2026) found that RLVR decreases the model's diversity by reducing pass@k. In our paper, we confirm these findings and attribute this to the RL objective itself as optimizing expected reward tends to marginalize learning signal from harder prompts, which results in distribution sharpening.

Learning to solve hard problems. Due to RLVR's shrinking of model coverage, significant attention has been drawn to new RL algorithms mitigating pass@k collapse. Approaches range from directly optimizing for pass@k during training (Walder and Karkhanis, 2025; Tang et al., 2025) to employing exploration bonuses in RL (Song et al., 2025; Tuyls et al., 2025). We show in our work that pass@k optimization objectives are a special case of our objective, since it optimizes an infinite harmonic series of pass@k objectives. On the other hand, the other works carry the fundamental limitation of RLVR of maximizing expected reward or pass rate over a batch of prompts, which we demonstrate to have vanishing gradient for prompts with low pass rate. This issue is also recognized by Nguyen et al. (2025b), which introduces selective learning only on prompts where greedy response fails, but unlike us, does not weigh prompts differently based on their pass rate.

Closely related works. One closely related line of work is that of Xiong et al. (2025b), which also considers non-linear functions of the pass rate in reinforcement learning. Both works are motivated by the observation that expected-reward objectives can underweight low-pass-rate prompts, and maximizing a log-likelihood like objective can mitigate this issue. However, the focus of Xiong et al. (2025b) is on adaptive rollout budget allocation and sampling strategies, treating the choice of non-linear weighting as part of an algorithmic design space. In contrast, we do not employ adaptive sampling, and instead derive a sampling-based on-policy estimator that approaches maximum likelihood as sampling compute budget is increased. We also empirically focus on demonstrating better data and compute scaling with our framework, whereas Xiong et al. (2025b) focuses on comparing against other adaptive sampling frameworks (Yu et al., 2025a). Furthermore, Davis and Recht (2025) provides a theoretical argument characterizing the population-level objectives approached by specific binary-reward reinforcement-learning algorithms in asymptotic limits, showing that certain procedures induce log-like weighting of the pass rate. Our work addresses a complementary question: how finite sampling defines explicit population-level objectives. We establish an exact estimator–objective equivalence at each finite rollout count and empirically evaluate how this objective-level interpolation manifests as compute is increased in both controlled settings and large-scale LLM post-training experiments. We discuss additional related works in Appendix A.

8 Conclusion

In this work, we establish Maximum Likelihood Reinforcement Learning as a principled optimization framework for non-differentiable binary reward settings. We showed that MaxRL approaches maximum likelihood in differentiable settings as compute increases and that in non-differentiable settings it offers key advantages over traditional RL, scaling more effectively with additional compute and data. More broadly, our results suggest that some limitations attributed to reinforcement learning with foundation models arise from objective choice rather than optimization or sampling. Our work currently assumes a binary reward setting and does not extend directly to continuous or arbitrarily valued rewards. Moreover, one can also consider objectives other than maximum likelihood to optimize following our framework. Generalizing MaxRL to continuous rewards, multi-turn reinforcement learning, and off-policy settings such as PPO-style training are promising directions for future work.

Acknowledgements

This work has greatly benefited from the use of Delta's advanced computing and data resource supported by the National Science Foundation (OAC 2005572) and the State of Illinois, as part of ACCESS-approved compute grants (Boerner et al., 2023). The authors also appreciate the computing resources of Bridges-2 (Brown et al., 2021) at Pittsburgh Supercomputing Center through ACCESS allocation CIS240901 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. Overall, this project used ACCESS grants CIS240901, CIS250216, CIS250428, CIS250651, CIS250835, CIS250560, CIS251063, and CIS251385 for its compute resources. The authors also thank the CMU FLAME center and the CMU Babel Compute Cluster for additional compute support during the early phases of this project.

This work was supported in part by ONR N000142312368 and ONR MURI N00014-25-1-2116. Moreover, Fahim Tajwar was partially supported by the U.S. Army Futures Command under Contract No. W519TC-23-C-0030 during the project. Yiding Jiang gratefully acknowledges the support of the Google PhD Fellowship. Daman Arora was supported by the National Science Foundation under Grants CCF-2106778.

The authors thank Brandon Pusateri, Jillian Lehosky, and Greg Bauer from ACCESS Support Staff for their incredible help at approving supplements and renewals for ACCESS compute grants throughout this project. Moreover, the work would not have finished in a timely manner without the help of Brett Bode from NCSA Delta Support Staff, who provided the authors with critical help in properly utilizing the Delta cluster. The authors are also grateful to Zhang et al. (2025) for their latex template, which was used to write this paper. The authors gratefully acknowledge Qianqian Wang, Yifei Zhou, Samuel Sokota, Yutong He, Lili Chen, Stephan Xie, Haque Ishfaq, and other members of the Zanette, Russ, and Auton lab for feedback and suggestions received on earlier versions of this work.

References

Jonathan Kwaku Afriyie, Kassim Tawiah, Wilhemina Adoma Pels, Sandra Addai-Henne, Harriet Achiaa Dwamena, Emmanuel Odame Owiredu, Samuel Amening Ayeh, and John Eshun. A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions. *Decision Analytics Journal*, 6:100163, 2023. ISSN 2772-6622. doi: https://doi.org/10.1016/j.dajour.2023.100163. URL https://www.sciencedirect.com/science/article/pii/S2772662223000036.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024. URL https://arxiv.org/abs/2402.14740.

Essential AI, :, Darsh J Shah, Peter Rushton, Somanshu Singla, Mohit Parmar, Kurt Smith, Yash Vanjani, Ashish Vaswani, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, Anil Thomas, Anthony Polloreno, Ashish Tanwer, Burhan Drak Sibai, Divya S Mansingka, Divya Shivaprasad, Ishaan Shah, Karl Stratos, Khoi Nguyen, Michael Callahan, Michael Pust, Mrinal Iyer, Philip Monk, Platon Mazarakis, Ritvik Kapila, Saurabh Srivastava, and Tim Romanski. Rethinking reflection in pre-training, 2025. URL https://arxiv.org/abs/2504.04022.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025. URL https://arxiv.org/abs/2502.02737.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL https://hkunlp.github.io/blog/2025/Polaris.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments, 2018. URL https://arxiv.org/abs/1711.07280.

Anonymous. Entropy-preserving reinforcement learning. In *Submitted to The Fourteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=E8MR8jgEeZ. under review.

Anonymous. Exploratory memory-augmented LLM agent via hybrid on- and off-policy optimization. In *Submitted to The Fourteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=UOzxviKVFO. under review.

Daman Arora, Himanshu Gaurav Singh, and Mausam . Have LLMs advanced enough? a challenging problem solving benchmark for large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=YHWXlESeS8.

Sanjar Atamuradov. Evaluating model-agnostic meta-learning on metaworld ml10 benchmark: Fast adaptation in robotic manipulation tasks, 2025. URL https://arxiv.org/abs/2511.12383.

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

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025a. URL https://matharena.ai/.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, November 2025b. URL https://matharena.ai/.

Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning, 2023. URL https://arxiv.org/abs/2306.02437.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation, 2016. URL https://arxiv.org/abs/1606.01868.

Richard Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957. ISSN 00959057, 19435274. URL http://www.jstor.org/stable/24900506.

Emmanuel Bengio, Joelle Pineau, and Doina Precup. Interference and generalization in temporal difference learning, 2020. URL https://arxiv.org/abs/2003.06350.

Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific, 1995.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN 978-0387310732.

Timothy J. Boerner, Stephen Deems, Thomas R. Furlani, Shelley L. Knuth, and John Towns. Access: Advancing innovation: Nsf's advanced cyberinfrastructure coordination ecosystem: Services & support. In *Practice and Experience in Advanced Research Computing 2023: Computing for the Common Good*, PEARC '23, page 173–176, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450399852. doi: 10.1145/3569951.3597559. URL https://doi.org/10.1145/3569951.3597559.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars, 2016. URL https://arxiv.org/abs/1604.07316.

Rémy Hosseinkhan Boucher, Onofrio Semeraro, and Lionel Mathelin. Evidence on the regularisation properties of maximum-entropy reinforcement learning, 2025. URL https://arxiv.org/abs/2501.17115.

Shawn T. Brown, Paola Buitrago, Edward Hanna, Sergiu Sanielevici, Robin Scibek, and Nicholas A. Nystrom. Bridges-2: A platform for rapidly-evolving and data intensive research. In *Practice and Experience in Advanced Research Computing 2021: Evolution Across All Dimensions*, PEARC '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450382922. doi: 10.1145/3437359.3465593. URL https://doi.org/10.1145/3437359.3465593.

Emma Brunskill and Lihong Li. Sample complexity of multi-task reinforcement learning, 2013. URL https://arxiv.org/abs/1309.6821.

Rudy Bunel, Matthew Hausknecht, Jacob Devlin, Rishabh Singh, and Pushmeet Kohli. Leveraging grammar and reinforcement learning for neural program synthesis, 2018. URL https://arxiv.org/abs/1805.04276.

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

ByteDance-Seed. Beyondaime: Advancing math reasoning evaluation beyond high school olympiads. https://huggingface.co/datasets/ByteDance-Seed/BeyondAIME, 2025.

George Casella and Roger L Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

Jonathan D. Chang, Wenhao Zhan, Owen Oertell, Kianté Brantley, Dipendra Misra, Jason D. Lee, and Wen Sun. Dataset reset policy optimization for rlhf, 2024. URL https://arxiv.org/abs/2404.08495.

Justin Chih-Yao Chen, Becky Xiangyu Peng, Prafulla Kumar Choubey, Kung-Hsiang Huang, Jiaxin Zhang, Mohit Bansal, and Chien-Sheng Wu. Nudging the boundaries of llm reasoning, 2025a. URL https://arxiv.org/abs/2509.25666.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Peter Chen, Xiaopeng Li, Ziniu Li, Wotao Yin, Xi Chen, and Tianyi Lin. Exploration vs exploitation: Rethinking rlvr through clipping, entropy, and spurious reward, 2025b. URL https://arxiv.org/abs/2512.16912.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning, 2025c. URL https://arxiv.org/abs/2505.14970.

Xinyun Chen, Chang Liu, and Dawn Song. Towards synthesizing complex programs from input-output examples, 2018. URL https://arxiv.org/abs/1706.01284.

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, 2025d. URL https://arxiv.org/abs/2508.10751.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. Reasoning with exploration: An entropy perspective, 2025a. URL https://arxiv.org/abs/2506.14758.

Zhoujun Cheng, Shibo Hao, Tianyang Liu, Fan Zhou, Yutao Xie, Feng Yao, Yuexin Bian, Yonghao Zhuang, Nilabjo Dey, Yuheng Zha, Yi Gu, Kun Zhou, Yuqi Wang, Yuan Li, Richard Fan, Jianshu She, Chengqian Gao, Abulhair Saparov, Haonan Li, Taylor W. Killian, Mikhail Yurochkin, Zhengzhong Liu, Eric P. Xing, and Zhiting Hu. Revisiting reinforcement learning for llm reasoning from a cross-domain perspective, 2025b. URL https://arxiv.org/abs/2506.14965.

Pierre Clavier, Stéphanie Allassonière, and Erwan Le Pennec. Robust reinforcement learning with distributional risk-averse formulation, 2022. URL https://arxiv.org/abs/2206.06841.

Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning, 2020. URL https://arxiv.org/abs/1912.01588.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning, 2018. URL https://arxiv.org/abs/1710.02410.

Runpeng Dai, Linfeng Song, Haolin Liu, Zhenwen Liang, Dian Yu, Haitao Mi, Zhaopeng Tu, Rui Liu, Tong Zheng, Hongtu Zhu, and Dong Yu. Cde: Curiosity-driven exploration for efficient reinforcement learning in large language models, 2025. URL https://arxiv.org/abs/2509.09675.

Xingyu Dang, Christina Baek, J Zico Kolter, and Aditi Raghunathan. Assessing diversity collapse in reasoning. In *Scaling Self-Improving Foundation Models without Human Supervision*, 2025. URL https://openreview.net/forum?id=AMiKsHLjQh.

Damek Davis and Benjamin Recht. What is the objective of reasoning with reinforcement learning?, 2025. URL https://arxiv.org/abs/2510.13651.

Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris van Doorn, and Jakob von Raumer. The lean theorem prover (system description). In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction - CADE-25*, pages 378–388, Cham, 2015. Springer International Publishing. ISBN 978-3-319-21401-6.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment, 2023. URL https://arxiv.org/abs/2304.06767.

Xiaoyi Dong, Jian Cheng, and Xi Sheryl Zhang. Maximum entropy reinforcement learning with diffusion policy, 2025. URL https://arxiv.org/abs/2502.11612.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning, 2016. URL https://arxiv.org/abs/1611.02779.

Arda Surya Editya, Moch. Machlul Alamin, Anggay Lury Pramana, and Neny Kurniati. Fraud classification in online payments using supervised machine learning algorithms. *Green Intelligent Systems and Applications*, 5(1):40–50, Mar. 2025. doi: 10.53623/gisa.v5i1.552. URL https://tecnoscientifica.com/journal/gisa/article/view/552.

Benjamin Eysenbach and Sergey Levine. Maximum entropy rl (provably) solves some robust rl problems, 2022. URL https://arxiv.org/abs/2103.06257.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. URL https://arxiv.org/abs/1703.03400.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg. Noisy networks for exploration, 2017. URL https://arxiv.org/abs/1706.10295.

Jingchu Gai, Guanning Zeng, Huaqing Zhang, and Aditi Raghunathan. Differential smoothing mitigates sharpening and improves llm reasoning, 2025. URL https://arxiv.org/abs/2511.19942.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D. Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars, 2025. URL https://arxiv.org/abs/2503.01307.

François Ged and Maria Han Veiga. Matryoshka policy gradient for entropy-regularized rl: Convergence and global optimality, 2024. URL https://arxiv.org/abs/2303.12785.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Irving John Good. Rational decisions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 14(1):107–114, 1992.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling, 2023. URL https://arxiv.org/abs/2308.08998.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017. URL https://arxiv.org/abs/1706.04599.

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.

Anthony GX-Chen, Jatin Prakash, Jeff Guo, Rob Fergus, and Rajesh Ranganath. Kl-regularized reinforcement learning is designed to mode collapse, 2025. URL https://arxiv.org/abs/2510.20817.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

Zhezheng Hao, Hong Wang, Haoyang Liu, Jian Luo, Jiarui Yu, Hande Dong, Qiang Lin, Can Wang, and Jiawei Chen. Rethinking entropy interventions in rlvr: An entropy change perspective, 2025. URL https://arxiv.org/abs/2510.10150.

Horace He and Thinking Machines Lab. Defeating nondeterminism in llm inference, September 2025.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

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, 2021. URL https://arxiv.org/abs/2103.03874.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Gabriel Dulac-Arnold, Ian Osband, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep q-learning from demonstrations, 2017. URL https://arxiv.org/abs/1704.03732.

Irina Higgins, Arka Pal, Andrei A. Rusu, Loic Matthey, Christopher P Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. Darla: Improving zero-shot transfer in reinforcement learning, 2018. URL https://arxiv.org/abs/1707.08475.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning, 2016. URL https://arxiv.org/abs/1606.03476.

Rein Houthooft, Xi Chen, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/abd815286ba1007abfbb8415b83ae2cf-Paper.pdf.

Jian Hu, Mingjie Liu, Ximing Lu, Fang Wu, Zaid Harchaoui, Shizhe Diao, Yejin Choi, Pavlo Molchanov, Jun Yang, Jan Kautz, and Yi Dong. Brorl: Scaling reinforcement learning via broadened exploration, 2025. URL https://arxiv.org/abs/2510.01180.

Wenke Huang, Quan Zhang, Yiyang Fang, Jian Liang, Xuankun Rong, Huanjin Yao, Guancheng Wan, Ke Liang, Wenwen He, Mingjun Li, Leszek Rutkowski, Mang Ye, Bo Du, and Dacheng Tao. Mapo: Mixed advantage policy optimization, 2025. URL https://arxiv.org/abs/2509.18849.

Haque Ishfaq, Qingfeng Lan, Pan Xu, A. Rupam Mahmood, Doina Precup, Anima Anandkumar, and Kamyar Azizzadenesheli. Provable and practical: Efficient exploration in reinforcement learning via langevin monte carlo, 2024a. URL https://arxiv.org/abs/2305.18246.

Haque Ishfaq, Yixin Tan, Yu Yang, Qingfeng Lan, Jianfeng Lu, A. Rupam Mahmood, Doina Precup, and Pan Xu. More efficient randomized exploration for reinforcement learning via approximate sampling, 2024b. URL https://arxiv.org/abs/2406.12241.

Haque Ishfaq, Guangyuan Wang, Sami Nur Islam, and Doina Precup. Langevin soft actor-critic: Efficient exploration through uncertainty-driven critic learning, 2025. URL https://arxiv.org/abs/2501.17827.

Kaito Ito and Kenji Kashima. Risk-sensitive control as inference with rényi divergence. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=LUIXdWn6Z5.

Garud N. Iyengar. Robust dynamic programming. *Math. Oper. Res.*, 30(2):257–280, May 2005. ISSN 0364-765X. doi: 10.1287/moor.1040.0129. URL https://doi.org/10.1287/moor.1040.0129.

Tarun Jain, Payal Garg, Namita Chalil, Aditya Sinha, Vivek Kumar Verma, and Rishi Gupta. Sms spam classification using machine learning techniques. In *2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 273–279, 2022. doi: 10.1109/Confluence52989.2022.9734128.

Mahnoor Jamil, Hristina Mihajloska Trpcheska, Aleksandra Popovska-Mitrovikj, Vesna Dimitrova, and Reiner Creutzburg. Advancing image spam detection: Evaluating machine learning models through comparative analysis. *Applied Sciences*, 15(11), 2025. ISSN 2076-3417. doi: 10.3390/app15116158. URL https://www.mdpi.com/2076-3417/15/11/6158.

Andrew Jesson and Yiding Jiang. Improving generalization on the procgen benchmark with simple architectural changes and scale, 2024. URL https://arxiv.org/abs/2410.10905.

Chen Jia. Generalizing reward modeling for out-of-distribution preference learning, 2024. URL https://arxiv.org/abs/2402.14760.

Yiding Jiang, J. Zico Kolter, and Roberta Raileanu. On the importance of exploration for generalization in reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 12951–12986. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/2a4310c4fd24bd336aa2f64f93cb5d39-Paper-Conference.pdf.

Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S. Dhillon, David Brandfonbrener, and Rishabh Agarwal. The art of scaling reinforcement learning compute for llms, 2025. URL https://arxiv.org/abs/2510.13786.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

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, 2024. URL https://arxiv.org/abs/2310.06452.

Sven Koenig and Reid G Simmons. Complexity analysis of real-time reinforcement learning. In *AAAI*, volume 93, pages 99–105, 1993.

Simon Kornblith, Honglak Lee, Ting Chen, and Mohammad Norouzi. Demystifying loss functions for classification, 2021. URL https://openreview.net/forum?id=jNTeYscgSw8.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels, 2021. URL https://arxiv.org/abs/2004.13649.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. Rewardbench: Evaluating reward models for language modeling, 2024. URL https://arxiv.org/abs/2403.13787.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19884–19895. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/e615c82aba461681ade82da2da38004a-Paper.pdf.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

E.L. Lehmann and G. Casella. *Theory of Point Estimation*. Springer Texts in Statistics. Springer New York, 2006. ISBN 9780387227283. URL https://books.google.com/books?id=4f24CgAAQBAJ.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review, 2018. URL https://arxiv.org/abs/1805.00909.

Sergey Levine and Vladlen Koltun. Guided policy search. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1–9, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/levine13.html.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models, 2022. URL https://arxiv.org/abs/2206.14858.

Caiyi Li, Kaishuai Liu, and Shuai Liu. A survey of loss functions in deep learning. *Mathematics*, 13(15), 2025. ISSN 2227-7390. doi: 10.3390/math13152417. URL https://www.mdpi.com/2227-7390/13/15/2417.

Zhaochun Li, Mingyang Yi, Yue Wang, Shisheng Cui, and Yong Liu. Towards a theoretical understanding to the generalization of rlhf, 2026. URL https://arxiv.org/abs/2601.16403.

Anthony Liang, Guy Tennenholtz, Chih wei Hsu, Yinlam Chow, Erdem Bıyık, and Craig Boutilier. Dynamite-rl: A dynamic model for improved temporal meta-reinforcement learning, 2024. URL https://arxiv.org/abs/2402.15957.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, 2015. URL https://arxiv.org/abs/1509.02971.

Yong Lin, Skyler Seto, Maartje Ter Hoeve, Katherine Metcalf, Barry-John Theobald, Xuan Wang, Yizhe Zhang, Chen Huang, and Tong Zhang. On the limited generalization capability of the implicit reward model induced by direct preference optimization. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16015–16026, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.940. URL https://aclanthology.org/2024.findings-emnlp.940/.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models, 2025a. URL https://arxiv.org/abs/2505.24864.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective, 2025b. URL https://arxiv.org/abs/2503.20783.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017. URL https://arxiv.org/abs/1608.03983.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Miao Lu, Han Zhong, Tong Zhang, and Jose Blanchet. Distributionally robust reinforcement learning with interactive data collection: Fundamental hardness and near-optimal algorithm, 2024. URL https://arxiv.org/abs/2404.03578.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. Cross-entropy loss functions: Theoretical analysis and applications, 2023. URL https://arxiv.org/abs/2304.07288.

Arthur Mensch and Mathieu Blondel. Differentiable dynamic programming for structured prediction and attention, 2018. URL https://arxiv.org/abs/1802.03676.

Beren Millidge, Alexander Tschantz, Anil K Seth, and Christopher L Buckley. On the relationship between active inference and control as inference, 2020. URL https://arxiv.org/abs/2006.12964.

MiniMax, :, Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, Chengjun Xiao, Chengyu Du, et al. Minimax-m1: Scaling test-time compute efficiently with lightning attention, 2025. URL https://arxiv.org/abs/2506.13585.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Sumeet Ramesh Motwani, Alesia Ivanova, Ziyang Cai, Philip Torr, Riashat Islam, Shital Shah, Christian Schroeder de Witt, and Charles London. h1: Bootstrapping llms to reason over longer horizons via reinforcement learning, 2025. URL https://arxiv.org/abs/2510.07312.

Kevin P. Murphy. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2012. ISBN 978-0262018029.

Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations, 2018. URL https://arxiv.org/abs/1709.10089.

Andrew Ng and Michael Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/7b7a53e239400a13bd6be6c91c4f6c4e-Paper.pdf.

Minh Nhat Nguyen, Andrew Baker, Clement Neo, Allen Roush, Andreas Kirsch, and Ravid Shwartz-Ziv. Turning up the heat: Min-p sampling for creative and coherent llm outputs, 2025a. URL https://arxiv.org/abs/2407.01082.

Phuc Minh Nguyen, Chinh D. La, Duy M. H. Nguyen, Nitesh V. Chawla, Binh T. Nguyen, and Khoa D. Doan. The reasoning boundary paradox: How reinforcement learning constrains language models, 2025b. URL https://arxiv.org/abs/2510.02230.

Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 625–632, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595931805. doi: 10.1145/1102351.1102430. URL https://doi.org/10.1145/1102351.1102430.

Arnab Nilim and Laurent El Ghaoui. Robust markov decision processes with uncertain transition matrices, 2004. AAI3165509.

Brendan O'Donoghue, Ian Osband, and Catalin Ionescu. Making sense of reinforcement learning and probabilistic inference, 2020. URL https://arxiv.org/abs/2001.00805.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, et al. Olmo 3, 2025. URL https://arxiv.org/abs/2512.13961.

OpenAI, 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.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization, 2024. URL https://arxiv.org/abs/2404.19733.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019.

Dean A. Pomerleau. Alvinn: an autonomous land vehicle in a neural network. In *Proceedings of the 2nd International Conference on Neural Information Processing Systems*, NIPS'88, page 305–313, Cambridge, MA, USA, 1988. MIT Press.

Robert Clay Prim. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957.

Yuxiao Qu, Anikait Singh, Yoonho Lee, Amrith Setlur, Ruslan Salakhutdinov, Chelsea Finn, and Aviral Kumar. Rlad: Training llms to discover abstractions for solving reasoning problems, 2025. URL https://arxiv.org/abs/2510.02263.

Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. Pope: Learning to reason on hard problems via privileged on-policy exploration, 2026. URL https://arxiv.org/abs/2601.18779.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, et al. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023. URL https://arxiv.org/abs/2305.18290.

Roberta Raileanu, Max Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic data augmentation for generalization in deep reinforcement learning, 2021. URL https://arxiv.org/abs/2006.12862.

Konrad Rawlik, Marc Toussaint, and Sethu Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. *Proceedings of Robotics: Science and Systems VIII*, 2012.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958. URL https://api.semanticscholar.org/CorpusID:12781225.

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/ross10a.html.

Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning, 2010. URL https://arxiv.org/abs/1011.0686.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi: 10.1038/323533a0.

Leonard J. Savage. Elicitation of personal probabilities and expectations. *Journal of the American Statistical Association*, 66(336):783–801, 1971. ISSN 01621459, 1537274X. URL http://www.jstor.org/stable/2284229.

Rylan Schaeffer, Joshua Kazdan, John Hughes, Jordan Juravsky, Sara Price, Aengus Lynch, Erik Jones, Robert Kirk, Azalia Mirhoseini, and Sanmi Koyejo. How do large language monkeys get their power (laws)?, 2025. URL https://arxiv.org/abs/2502.17578.

Tom Schaul, Diana Borsa, Joseph Modayil, and Razvan Pascanu. Ray interference: a source of plateaus in deep reinforcement learning, 2019. URL https://arxiv.org/abs/1904.11455.

Jürgen Schmidhuber. Curious model-building control systems. In *Proc. international joint conference on neural networks*, pages 1458–1463, 1991.

Jürgen Schmidhuber. Gödel machines: Fully self-referential optimal universal self-improvers. In *Artificial general intelligence*, pages 199–226. Springer, 2007.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2015. URL https://arxiv.org/abs/1502.05477.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Amrith Setlur, Matthew Y. R. 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, 2025. URL https://arxiv.org/abs/2506.09026.

Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. Can large reasoning models self-train?, 2025. URL https://arxiv.org/abs/2505.21444.

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

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

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.

Laixi Shi, Gen Li, Yuting Wei, Yuxin Chen, Matthieu Geist, and Yuejie Chi. The curious price of distributional robustness in reinforcement learning with a generative model, 2025. URL https://arxiv.org/abs/2305.16589.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Noah A. Smith. *Linguistic Structure Prediction*. Morgan & Claypool Publishers, 1st edition, 2011. ISBN 1608454053.

Yuda Song, Julia Kempe, and Remi Munos. Outcome-based exploration for llm reasoning, 2025. URL https://arxiv.org/abs/2509.06941.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning, 2020. URL https://arxiv.org/abs/2004.04136.

Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards, 2025. URL https://arxiv.org/abs/2505.24760.

Alexander L. Strehl, Lihong Li, and Michael L. Littman. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 10(84):2413–2444, 2009. URL http://jmlr.org/papers/v10/strehl09a.html.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021. URL https://arxiv.org/abs/2104.09864.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 1988.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of llms should leverage suboptimal, on-policy data, 2024. URL https://arxiv.org/abs/2404.14367.

Fahim Tajwar, Yiding Jiang, Abitha Thankaraj, Sumaita Sadia Rahman, J Zico Kolter, Jeff Schneider, and Ruslan Salakhutdinov. Training a generally curious agent, 2025. URL https://arxiv.org/abs/2502.17543.

Yunhao Tang, Kunhao Zheng, Gabriel Synnaeve, and Rémi Munos. Optimizing language models for inference time objectives using reinforcement learning, 2025. URL https://arxiv.org/abs/2503.19595.

Jean Tarbouriech, Tor Lattimore, and Brendan O'Donoghue. Probabilistic inference in reinforcement learning done right, 2023. URL https://arxiv.org/abs/2311.13294.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, et al. Kimi k1.5: Scaling reinforcement learning with llms, 2025. URL https://arxiv.org/abs/2501.12599.

Juan Terven, Diana-Margarita Cordova-Esparza, Julio-Alejandro Romero-González, Alfonso Ramírez-Pedraza, and E. A. Chávez-Urbiola. A comprehensive survey of loss functions and metrics in deep learning. *Artificial Intelligence Review*, 58(7), April 2025. ISSN 1573-7462. doi: 10.1007/s10462-025-11198-7. URL http://dx.doi.org/10.1007/s10462-025-11198-7.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, et al. Llama 2: Open foundation and fine-tuned chat models, 2023. URL https://arxiv.org/abs/2307.09288.

Jens Tuyls, Dylan J. Foster, Akshay Krishnamurthy, and Jordan T. Ash. Representation-based exploration for language models: From test-time to post-training, 2025. URL https://arxiv.org/abs/2510.11686.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Mel Vecerik, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards, 2018. URL https://arxiv.org/abs/1707.08817.

Joshua Vendrow, Edward Vendrow, Sara Beery, and Aleksander Madry. Do large language model benchmarks test reliability?, 2025. URL https://arxiv.org/abs/2502.03461.

Kartik Waghmare and Johanna Ziegel. Proper scoring rules for estimation and forecast evaluation, 2025. URL https://arxiv.org/abs/2504.01781.

Christian Walder and Deep Karkhanis. Pass@k policy optimization: Solving harder reinforcement learning problems, 2025. URL https://arxiv.org/abs/2505.15201.

KAIXIN WANG, Bingyi Kang, Jie Shao, and Jiashi Feng. Improving generalization in reinforcement learning with mixture regularization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7968–7978. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/5a751d6a0b6ef05cfe51b86e5d1458e6-Paper.pdf.

Qi Wang, Yue Ma, Kun Zhao, and Ying jie Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, 9:187 – 212, 2020.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=yfcpdY4gMP.

Xudong Wang, Long Lian, and Stella X. Yu. Unsupervised visual attention and invariance for reinforcement learning, 2021. URL https://arxiv.org/abs/2104.02921.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023a. URL https://arxiv.org/abs/2203.11171.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. How far can camels go? exploring the state of instruction tuning on open resources, 2023b. URL https://arxiv.org/abs/2306.04751.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3–4):279–292, 1992. doi: 10.1007/BF00992698. URL https://doi.org/10.1007/BF00992698.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Fang Wu, Weihao Xuan, Ximing Lu, Mingjie Liu, Yi Dong, Zaid Harchaoui, and Yejin Choi. The invisible leash: Why rlvr may or may not escape its origin, 2026. URL https://arxiv.org/abs/2507.14843.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and Hanze Dong. A minimalist approach to llm reasoning: from rejection sampling to reinforce, 2025a. URL https://arxiv.org/abs/2504.11343.

Wei Xiong, Chenlu Ye, Baohao Liao, Hanze Dong, Xinxing Xu, Christof Monz, Jiang Bian, Nan Jiang, and Tong Zhang. Reinforce-ada: An adaptive sampling framework under non-linear rl objectives, 2025b. URL https://arxiv.org/abs/2510.04996.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. Is dpo superior to ppo for llm alignment? a comprehensive study, 2024. URL https://arxiv.org/abs/2404.10719.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024a. URL https://arxiv.org/abs/2407.10671.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, et al. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024b. URL https://arxiv.org/abs/2409.12122.

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, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Kai Yang, Jian Tao, Jiafei Lyu, and Xiu Li. Exploration and anti-exploration with distributional random network distillation, 2024c. URL https://arxiv.org/abs/2401.09750.

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, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025a. URL https://arxiv.org/abs/2503.14476.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Avnish Narayan, Hayden Shively, Adithya Bellathur, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning, 2021. URL https://arxiv.org/abs/1910.10897.

Zhaoning Yu, Will Su, Leitian Tao, Haozhu Wang, Aashu Singh, Hanchao Yu, Jianyu Wang, Hongyang Gao, Weizhe Yuan, Jason Weston, Ping Yu, and Jing Xu. Restrain: From spurious votes to signals – self-driven rl with self-penalization, 2025b. URL https://arxiv.org/abs/2510.02172.

Lifan Yuan, Weize Chen, Yuchen Zhang, Ganqu Cui, Hanbin Wang, Ziming You, Ning Ding, Zhiyuan Liu, Maosong Sun, and Hao Peng. From $f(x)$ and $g(x)$ to $f(g(x))$: Llms learn new skills in rl by composing old ones, 2025. URL https://arxiv.org/abs/2509.25123.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models, 2023. URL https://arxiv.org/abs/2308.01825.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?, 2025. URL https://arxiv.org/abs/2504.13837.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022. URL https://arxiv.org/abs/2203.14465.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild, 2025. URL https://arxiv.org/abs/2503.18892.

Amy Zhang, Nicolas Ballas, and Joelle Pineau. A dissection of overfitting and generalization in continuous reinforcement learning, 2018a. URL https://arxiv.org/abs/1806.07937.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction, 2021. URL https://arxiv.org/abs/2006.10742.

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and rl on reasoning language models, 2025. URL https://arxiv.org/abs/2512.07783.

Chi Zhang, Guangming Sheng, Siyao Liu, Jiahao Li, Ziyuan Feng, Zherui Liu, Xin Liu, Xiaoying Jia, Yanghua Peng, Haibin Lin, and Chuan Wu. A framework for training large language models for code generation via proximal policy optimization, 2024. URL https://i.cs.hku.hk/~cwu/papers/gmsheng-NL2Code24.pdf. PDF available online.

Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning, 2018b. URL https://arxiv.org/abs/1804.06893.

Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis : A survey, 2018c. URL https://arxiv.org/abs/1801.07883.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey, 2023. URL https://arxiv.org/abs/2308.10792.

Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *J. Mach. Learn. Res.*, 5:1225–1251, December 2004. ISSN 1532-4435.

Rosie Zhao, Alexandru Meterez, Sham Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo chamber: Rl post-training amplifies behaviors learned in pretraining, 2025. URL https://arxiv.org/abs/2504.07912.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, page 1433–1438. AAAI Press, 2008. ISBN 9781577353683.

A Extended Related Works

Cross-entropy objective. Maximizing log-likelihood via optimizing cross-entropy is a widely used framework in machine learning due to its simplicity and favorable theoretical properties. In particular, cross-entropy is a strictly proper scoring rule, meaning that the expected loss is uniquely minimized by the true probability distribution, which encourages statistically calibrated predictions (Good, 1992; Savage, 1971; Gneiting and Raftery, 2007; Waghmare and Ziegel, 2025). Moreover, it yields statistically efficient estimators under standard assumptions (Vaart, 1998; Casella and Berger, 2002; Lehmann and Casella, 2006), and induces gradients that concentrate learning signal on low-probability or uncertain outcomes through logarithmic weighting (Wang et al., 2020). As a result, cross-entropy (or log-loss) often yields consistent estimators for classification and tends to generalize well in practice (Ng and Jordan, 2001; Zhang, 2004). However, more recent work has shown that models trained to maximize log-likelihood can still overfit and exhibit miscalibration, motivating post-hoc techniques such as temperature scaling (Niculescu-Mizil and Caruana, 2005; Guo et al., 2017). Moreover, the unbounded nature of cross-entropy and its sensitivity to small perturbations in the predicted distribution suggest that alternative strictly proper scoring rules may be more suitable in certain settings (Kornblith et al., 2021). Because cross-entropy is extensively studied, we refer interested readers to Mao et al. (2023); Li et al. (2025); Terven et al. (2025) for a comprehensive review.

Supervised training vs reinforcement learning. Supervised learning has been the go-to training paradigm in machine learning, beginning with early "learning with a teacher" neural network based systems such as the perceptron (Rosenblatt, 1958) and later becoming practical with backpropagation-based neural network training (Rumelhart et al., 1986). It has been used to tackle a broad range of problems, from financial fraud detection (Afriyie et al., 2023; Editya et al., 2025), to sentiment analysis (Zhang et al., 2018c) and spam detection (Jain et al., 2022; Jamil et al., 2025). More recently, supervised training has been used in modern image classification systems (Lecun et al., 1998; Krizhevsky et al., 2012) to achieve strong performance. In modern foundation models, "pretraining" is typically done by supervised learning via minimizing cross-entropy over next-token prediction (Radford et al., 2018, 2019) on large corpus of text, followed by additional supervised training on high quality human written demonstrations (Ouyang et al., 2022; Zhang et al., 2023) to teach models how to respond to prompts, often known as instruction-tuning. In sequential decision making domains, supervised training appears as behavior cloning from expert demonstration, used in early autonomous driving and robotic systems (Pomerleau, 1988; Bojarski et al., 2016; Codevilla et al., 2018). However, supervised learning in sequential decision making breaks the i.i.d. assumption since the learned policy's actions affect which states it visits during execution, causing compounding errors over time (Ross and Bagnell, 2010; Belkhale et al., 2023). The classic no-regret reductions line (DAgger) makes this explicit and addresses drift by iteratively querying the expert on states visited by the learner, turning the sequential problem into an online supervised learning loop with improved guarantees (Ross et al., 2010).

In such scenarios, reinforcement learning is an attractive alternative paradigm that formalizes learning under delayed, sparse, and evaluative feedback in Markov decision processes (MDPs), with foundational roots in dynamic programming and MDP theory (Bellman, 1957). Mechanistically, the key contrast with supervised learning / behavior cloning is that supervised learning assumes (or benefits strongly from) an i.i.d. dataset of correct targets under a fixed data distribution, whereas RL's data distribution is policy-induced and nonstationary, and gradients arise from credit assignment through rewards rather than direct target labels. This mismatch shows up starkly in sequential prediction/imitation: naive behavior cloning trains on expert state distributions but at test time visits states induced by its own errors, causing compounding error (a form of distribution shift / "state drift"). Classical RL algorithms include temporal-difference learning (Sutton, 1988; Sutton et al., 1998) and value-based control such as Q-learning (Watkins and Dayan, 1992), while policy-gradient methods (Williams, 1992; Sutton et al., 1999) directly optimize expected return via likelihood-ratio gradients (REINFORCE) and later stabilized large-scale learning through variants like trust-region policy optimization (TRPO) (Schulman et al., 2015) and off-policy actor–critic methods for continuous control (e.g., DDPG (Lillicrap et al., 2015)) and maximum-entropy actor–critic (e.g., SAC (Haarnoja et al., 2018)). Empirically, deep RL's modern resurgence is often associated with representation learning + RL on high-dimensional inputs (e.g., DQN (Mnih et al., 2013)).

A large body of work blends supervised and RL to get the best of both: (i) Imitation + online correction methods like DAgger (Ross et al., 2010) explicitly combine supervised learning with interactive data collection to mitigate distribution shift ; (ii) Inverse RL / MaxEnt IRL reframes imitation as learning a reward/cost model that explains expert behavior, with maximum-entropy formulations giving a principled probabilistic objective (Ziebart et al., 2008); (iii) Adversarial imitation (GAIL) (Ho and Ermon, 2016) avoids explicit reward learning by matching occupancy measures via a GAN-like discriminator, typically trained with policy optimization; (iv) Learning from demonstrations in deep RL injects supervised losses and/or demonstration replay into RL to improve exploration and sample efficiency—e.g., DQfD (Hester et al., 2017) combines TD learning with supervised large-margin imitation terms, and demonstration-augmented continuous-control methods address sparse-reward exploration failures (Nair et al., 2018); and (v) Trajectory-optimization–guided policy learning (guided policy search (Levine and Koltun, 2013)) explicitly produces supervised targets for a policy network from trajectory optimization / local controllers, bridging optimal control, RL, and supervised regression. In modern LLM alignment, the same hybrid template appears as "Supervised fine-tuning + preference-based RL": InstructGPT (Ouyang et al., 2022) first performs supervised fine-tuning on demonstrations and then applies RL from human feedback (RLHF), while more recent approaches like Direct Preference Optimization (DPO) (Rafailov et al., 2023) recast parts of RLHF into a supervised-style classification objective — illustrating an active trend of recovering supervised-like training signals even when the underlying goal is preference/reward optimization.

Finally, control as inference is also a closely related topic (Millidge et al., 2020; O'Donoghue et al., 2020; Rawlik et al., 2012; Ito and Kashima, 2024; Tarbouriech et al., 2023), and we point the reader to Levine (2018) for more details.

Training LLMs for strong reasoning abilities. A few different approaches for post-training have demonstrated success, including supervised fine-tuning on human-crafted high quality demonstrations (Wang et al., 2023b), iterative supervised training on self-generated good quality responses (Zelikman et al., 2022; Gulcehre et al., 2023), reinforcement learning from a learned reward model on human preferences (Ouyang et al., 2022), and more recently preference-based contrastive learning (Rafailov et al., 2023; Pang et al., 2024). In our work, we focus on recovering the cross-entropy based classification objective in an RL training pipeline, fundamentally differing from the prior works. Since the recent advent of RLVR, multiple followup works have studied the RLVR pipeline (Zeng et al., 2025; Liu et al., 2025b; Khatri et al., 2025) and proposed alternative algorithms such as Dr.GRPO (Liu et al., 2025b), DAPO (Yu et al., 2025a), GSPO (Zheng et al., 2025) and CISPO (MiniMax et al., 2025), RAFT (Xiong et al., 2025a). The idea of normalizing advantages by mean reward, similar to ours, has been explored in (Huang et al., 2025), but whereas we normalize advantage by group mean reward (mean reward over the rollouts associated with a particular prompt), Huang et al. (2025) normalizes by the batch mean reward (mean reward over all prompts in a batch of policy gradient updates). Finally, recent work such as Zhang et al. (2025) has also studied how RL training is influenced by pretraining and midtraining in toy didactic settings, establishing the importance of good pretraining/midtraining for the success of RL, similar to Gandhi et al. (2025).

On exploration for reinforcement learning for LLMs. Exploration, or taking actions to discover new information, is a widely studied topic in reinforcement learning. A closely related topic is *curiosity*, where an agent seeks new information about its environment via interactions. *Intrinsic motivation* is a popular notion for curiosity, where the agent is driven by an exploration bonus that is not necessarily related to the task to be achieved (Schmidhuber, 1991, 2007). Followup works have built on this notion to mitigate problems of sparse reward (reward is observed at a very belated phase of interactions) or no reward at all (Pathak et al., 2017, 2019; Eysenbach et al., 2018; Burda et al., 2018; Sharma et al., 2019; Yang et al., 2024c; Houthooft et al., 2016). Count-based bonuses have also been introduced as a way of computing intrinsic motivation (Bellemare et al., 2016). Prompt-level reweighting of gradients has also been studied (Yu et al., 2025b), though under a different context (self-training) and a different weighting mechanism. Finally, adding noise to network parameters or optimization has been another line of work to improve exploration during RL training (Fortunato et al., 2017; Ishfaq et al., 2024b,a, 2025). Maximum entropy RL, the principle where one attempts to recover an agent that achieves high reward but is as stochastic as possible, can be seen as another attempt at solving exploration for classical RL (Haarnoja et al., 2018; Boucher et al., 2025; Eysenbach and Levine, 2022; Dong et al., 2025). In summary, exploration-exploitation tradeoff (Sutton, 1988; Auer et al., 2002; Thompson, 1933) has been a crucial topic for ensuring RL agents' success.

More recently, exploration has emerged as an important topic for building modern LLM based systems. There are two types of exploration to consider. The first is *inference-time exploration*, where an agent has to efficiently gather information during deployment by strategically choosing its interactions with its environment, Tajwar et al. (2025) is an important work in this line of research. More importantly, pass@k degradation (mode collapse) during RLVR (Yue et al., 2025; Wu et al., 2026; GX-Chen et al., 2025) has prompted research into *train-time exploration*, where the challenge is to go beyond the pretrained model's capabilities and discover new knowledge. Primary approaches include directly optimizing for pass@k (Walder and Karkhanis, 2025; Tang et al., 2025; Chen et al., 2025d), curriculum learning (Tajwar et al., 2025; Chen et al., 2025c; Setlur et al., 2025; Motwani et al., 2025), learning from additional hints or abstractions (Qu et al., 2025; Chen et al., 2025a; Anonymous, 2025b), increasing number of rollouts to prevent RL gains from saturating (Hu et al., 2025), employing data curation algorithm to redirect effort to problems with low success rate (Nguyen et al., 2025b), leveraging expert guidance (Chang et al., 2024; Qu et al., 2026), or differential smoothing by penalizing entropy on low reward trajectories and encouraging entropy on high reward trajectories. Entropy based bonuses to encourage exploration during RL training (Hao et al., 2025; Chen et al., 2025b; Cheng et al., 2025a; Wang et al., 2025; Anonymous, 2025a; Ged and Veiga, 2024) is another popular line of work for improving exploration. A few modern approaches for exploration bonus utilized for LLM training are Song et al. (2025); Tuyls et al. (2025). The idea of curiosity-driven exploration from classical RL discussed above has also been adopted for LLMs (Dai et al., 2025). Although some works have reported pass@k degradation during RL training, others have found the opposite results. For example, ProRL (Liu et al., 2025a) has shown that RL training on a mixture of reasoning puzzles (Stojanovski et al., 2025) can improve pass@k on a heldout reasoning task. Similarly, Yuan et al. (2025) has shown that LLMs can learn new skills via RL by composing old ones, showing the promise of going beyond pre-training knowledge, and Cheng et al. (2025b) also found pass@k to improve, particularly on tasks less likely to appear during the pre-training stage. Ray interference (Schaul et al., 2019) has been proposed as an explanation for the observed pass@k degradation. Overall, this line of research remains important as focus moves to LLMs discovering new information during RL training and it is therefore an ongoing field of research.

Generalization in reinforcement learning. Research on generalization in reinforcement learning asks a core question: does an agent learn principles that transfer beyond the exact environments it trained in, or does it just memorize experience (Zhang et al., 2018b,a; Schaul et al., 2019; Bengio et al., 2020)? Empirical work shows deep RL agents often overfit to training seeds, visuals, or dynamics, performing poorly on new levels, layouts, or slightly shifted physics. To study this, researchers built procedural benchmarks (like CoinRun/Procgen (Cobbe et al., 2020)) and multi-task suites (e.g., robotics task collections (Yu et al., 2021; Atamuradov, 2025) or LLM sequential decision-making task suites (Tajwar et al., 2025)) that separate train and test environments. A major line of work improves generalization through regularization (Kostrikov et al., 2021) and invariances (Zhang et al., 2021) — especially data augmentation (Laskin et al., 2020; Raileanu et al., 2021), mixup-style methods (WANG et al., 2020), and representation learning tricks (Higgins et al., 2018; Srinivas et al., 2020; Wang et al., 2021) that make policies rely less on superficial visual details. Another branch focuses on task and domain shift, using meta-RL (Duan et al., 2016; Finn et al., 2017; Liang et al., 2024), multi-task learning (Brunskill and Li, 2013), domain randomization, and distributionally robust RL (Clavier et al., 2022; Lu et al., 2024; Shi et al., 2025) to handle new tasks or uncertain dynamics. Recent lines of work has also directly studied exploration as a mean to achieve generalization in RL (Jiang et al., 2023), and have shown that simple architectural changes and scale can often improve generalization in the ProcGen benchmark (Jesson and Jiang, 2024). On the theory side, classical PAC-MDP (Strehl et al., 2009) and robust MDP frameworks (Nilim and Ghaoui, 2004; Iyengar, 2005) formalize when policies learned from limited samples can be expected to work in new situations. In the LLM settings, analogous questions appear in RLHF (Ouyang et al., 2022), where reinforcement learning is used to align models with human preferences, and researchers now study how this training affects generalization to unseen prompts, behaviors, and user distributions (Kirk et al., 2024; Lin et al., 2024; Lambert et al., 2024; Jia, 2024; Li et al., 2026). Overall, the field has moved from "can RL learn?" to "what exactly does it learn, and when does that knowledge transfer?"

B Theoretical Results

Here we present the proofs of theorems mentioned in the main paper. First we restate and prove Theorem 1.

Theorem 3 (Restatement of Theorem 1)**.** *The gradient of the maximum likelihood objective admits the following conditional expectation representation:*

$$\nabla_\theta J_{\mathrm{ML}}(x) = \mathbb{E}[\nabla_\theta \log m_\theta(z \mid x) \mid f(z) = y^*(x)] .$$

Proof. Recall the standard REINFORCE identity for the gradient of the pass rate:

$$\nabla_\theta p_\theta^{\mathrm{pass}}(x) = \nabla_\theta \mathbb{E}_{z \sim m_\theta(\cdot|x)}[\mathbb{I}\{f(z) = y^*(x)\}] = \mathbb{E}_{z \sim m_\theta(\cdot|x)}[\mathbb{I}\{f(z) = y^*(x)\} \nabla_\theta \log m_\theta(z \mid x)].$$

The gradient of the maximum likelihood objective is:

$$\nabla_\theta J_{\mathrm{ML}}(x) = \nabla_\theta \log p_\theta^{\mathrm{pass}}(x) = \frac{\nabla_\theta p_\theta^{\mathrm{pass}}(x)}{p_\theta^{\mathrm{pass}}(x)} = \frac{\mathbb{E}_{z \sim m_\theta(\cdot|x)}[\mathbb{I}\{f(z) = y^*(x)\} \nabla_\theta \log m_\theta(z \mid x)]}{\mathbb{E}_{z \sim m_\theta(\cdot|x)}[\mathbb{I}\{f(z) = y^*(x)\}]}.$$

By the definition of conditional expectation for an event A with $\mathbb{P}(A) > 0$:

$$\mathbb{E}[X \mid A] = \frac{\mathbb{E}[X \cdot \mathbb{I}_A]}{\mathbb{P}(A)}.$$

Letting $X = \nabla_\theta \log m_\theta(z \mid x)$ and $A = \{z : f(z) = y^*(x)\}$, and noting that $p_\theta^{\mathrm{pass}}(x) = \mathbb{P}(A)$, we obtain:

$$\nabla_\theta J_{\mathrm{ML}}(x) = \mathbb{E}[\nabla_\theta \log m_\theta(z \mid x) \mid f(z) = y^*(x)] .$$

□

Next, we restate and prove Theorem 2.

Theorem 4 (Restatement of Theorem 2)**.** *The estimator $\widehat{g}_N(x)$ is an unbiased estimator for the* MaxRL *gradient of order $T = N$, i.e.,*

$$\mathbb{E}[\widehat{g}_N(x)] = \nabla_\theta J_{\mathrm{MaxRL}}^{(N)}(x).$$

Proof. Conditioned on $K \geq 1$, the successful samples are i.i.d. draws from the success-conditioned distribution, so by Theorem 1:

$$\mathbb{E}[\widehat{g}_N(x) \mid K \geq 1] = \nabla_\theta \log p_\theta^{\mathrm{pass}}(x).$$

Since $\widehat{g}_N(x) = 0$ when $K = 0$:

$$\mathbb{E}[\widehat{g}_N(x)] = \nabla_\theta \log p_\theta^{\mathrm{pass}}(x) \cdot \mathbb{P}(K \geq 1) = \nabla_\theta \log p_\theta^{\mathrm{pass}}(x) \cdot \mathrm{pass@}N(x).$$

Writing $p = p_\theta^{\mathrm{pass}}(x)$ and using $\mathrm{pass@}k(x) = 1 - (1-p)^k$:

$$\frac{\nabla_\theta p}{p} \cdot (1 - (1-p)^N) = \nabla_\theta p \sum_{k=1}^{N} (1-p)^{k-1} = \sum_{k=1}^{N} \frac{1}{k} \nabla_\theta \mathrm{pass@}k(x) = \nabla_\theta J_{\mathrm{MaxRL}}^{(N)}(x),$$

where the second equality uses $\nabla_\theta \mathrm{pass@}k(x) = k(1-p)^{k-1} \nabla_\theta p$. □

C More on Unifying Weight-Function View on RL Objectives

Here we provide full derivations following Section 5. Recall that we want to express population-level gradients of different objectives in the following form:

$$\nabla_\theta J = \mathbb{E}_{x\sim\rho}\left[w(p_\theta(x))\nabla_\theta p_\theta(x)\right]$$

where $p_\theta(x) = p_\theta^{\text{pass}}(x)$ and $w(p_\theta(x))$ is the weighting function. In this section, we show that all objectives of our consideration can be written in this form. Furthermore, we will derive the weighting function $w(p_\theta(x))$ for each of them.

Classical RL (REINFORCE). For classical reinforcement learning, i.e., the REINFORCE objective, we have:

$$\begin{aligned}
J_{\text{RL}} &= \mathbb{E}_{x\sim\rho}[\mathbb{E}_{z\sim m_\theta(\cdot|x)}[r(x,z)]] \\
&= \mathbb{E}_{x\sim\rho}[\mathbb{E}_{z\sim m_\theta(\cdot|x)}[\mathbb{I}\left\{f(z) = y^*(x)\right\}]] \\
&= \mathbb{E}_{x\sim\rho}[p_\theta^{\text{pass}}(x)]
\end{aligned}$$

Therefore, its gradient is:

$$\nabla_\theta J_{\text{RL}} = \mathbb{E}_{x\sim\rho}[\nabla_\theta p_\theta^{\text{pass}}(x)]$$

giving the corresponding w_{RL} to be 1.

GRPO. Our analysis is similar to that of Davis and Recht (2025). The gradient of the population level GRPO objective gradient can be written as:

$$\nabla_\theta J_{\text{GRPO}} = \mathbb{E}_{x\sim\rho}\left[\mathbb{E}_{z\sim m_\theta(\cdot|x)}\left[\left(\frac{r(x,z) - \mathbb{E}_{z\sim m_\theta(\cdot|x)}[r(x,z)]}{\sqrt{\mathrm{Var_{z\sim m_\theta(\cdot|x)}[r(x,z]}}}\right)\nabla_\theta \log m_\theta(z|x)\right]\right]$$

Since we consider a binary reward setting, we have $\mathbb{E}_{z\sim m_\theta(\cdot|x)}[r(x,z)] = p_\theta^{\text{pass}}(x)$. Similarly, considering the variance of a Bernoulli random variable, we get:

$$\mathrm{Var_{z\sim m_\theta(\cdot|x)}} = p_\theta^{\text{pass}}(x)\,(1 - p_\theta^{\text{pass}}(x))$$

Therefore, the objective becomes:

$$\nabla_\theta J_{\text{GRPO}} = \mathbb{E}_{x\sim\rho}\left[\frac{1}{\sqrt{p_\theta^{\text{pass}}(x)\,(1 - p_\theta^{\text{pass}}(x))}}\nabla_\theta p_\theta^{\text{pass}}(x)\right]$$

which thereby gives us the weighting function to be $1/\sqrt{p_\theta(x)(1 - p_\theta(x))}$, as desired.

Maximum Likelihood (ML). The maximum likelihood objective is given by

$$J_{\text{ML}} = \mathbb{E}_{x\sim\rho}[\log p_\theta^{\text{pass}}(x)].$$

Taking its gradient with respect to θ and applying the chain rule, we obtain

$$\begin{aligned}
\nabla_\theta J_{\text{ML}} &= \mathbb{E}_{x\sim\rho}\left[\nabla_\theta \log p_\theta^{\text{pass}}(x)\right] \\
&= \mathbb{E}_{x\sim\rho}\left[\frac{1}{p_\theta^{\text{pass}}(x)}\nabla_\theta p_\theta^{\text{pass}}(x)\right]
\end{aligned}$$

This shows that the weighting function for the maximum likelihood objective is $1/p_\theta^{\text{pass}}(x)$, as we claimed in Table 2.

MaxRL. Finally, we consider the objective $J^{(T)}_{\textsc{MaxRL}}$.

Proposition 5. *For MaxRL with order T, we can rewrite it as*

$$\nabla_\theta J^{(T)}_{\textsc{MaxRL}} = \mathbb{E}_{x\sim\rho}[w(p_\theta(x))\ \nabla_\theta p_\theta(x)]\,,$$

where

$$w_T(p) = \sum_{k=1}^{T}(1-p)^{k-1} = \frac{1-(1-p)^T}{p}.$$

Proof. *From Equation (7), we have:*

$$\nabla_\theta J^{(T)}_{\textsc{MaxRL}}(x) = \sum_{k=1}^{T}\frac{1}{k}\nabla_\theta \mathrm{pass@}k(x).$$

Using $\mathrm{pass@}k(x) = 1-(1-p)^k$ *where* $p = p_\theta^{\mathrm{pass}}(x)$*:*

$$\nabla_\theta \mathrm{pass@}k(x) = k(1-p)^{k-1}\nabla_\theta p.$$

Substituting:

$$\nabla_\theta J^{(T)}_{\textsc{MaxRL}}(x) = \sum_{k=1}^{T}\frac{1}{k}\cdot k(1-p)^{k-1}\nabla_\theta p = \left(\sum_{k=1}^{T}(1-p)^{k-1}\right)\nabla_\theta p = w_T(p)\nabla_\theta p_\theta^{\mathrm{pass}}(x).$$

Taking the expectation over $x\sim\rho$ completes the proof. $\square$

D Additional Details on ImageNet Experiments

D.1 Training Procedure

Let $\mathcal{X}$ be the input space and $\mathcal{Y}$ be the label space. Let π_θ denote our model: given an input image $x \in \mathcal{X}$, $\pi_\theta(y|x)$ is model's predicted probability of image x belonging to class $y \in \mathcal{Y}$. For an input image and label pair $(x, y^\star(x))$, the cross-entropy loss is:

$$\mathcal{L}_{\mathrm{CE}}(x, y^\star(x); \pi_\theta) = -\log \pi_\theta(y^\star(x)|x)$$

On the other hand, the corresponding RL objective for the same pair is:

$$\mathcal{L}_{\mathrm{RL}}(x, y^\star; \pi_\theta) = -\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[-\log \pi_\theta(y|x) \cdot \hat{A}(y|x)]$$

where the expectation is computed using Monte-Carlo sampling K rollouts of y from $\pi_\theta(\cdot|x)$. GRPO, REINFORCE and MaxRL vary only in the calculation of the advantage $A(y|x)$. Concretely, let $y^{(1)}, \ldots, y^{(K)}$ be our K rollouts, sampled from the conditional probability distribution $\pi_\theta(\cdot|x)$. We operate under a binary reward setting, meaning the reward function $r(x, y)$ is:

$$r(x,y) = \mathbb{I}[y = y^\star(x)] = \begin{cases} 1, & \text{if } y = y^\star(x) \\ 0, & \text{otherwise} \end{cases}$$

Given this reward, we calculate advantage under GRPO, REINFORCE and MaxRL as follows:

$$\hat{A}_{\mathrm{GRPO}}(x,y) = \frac{r(x,y) - \hat{\mu}}{\hat{\sigma}}$$

$$\hat{A}_{\mathrm{REINFORCE}}(x,y) = r(x,y) - \hat{\mu}$$

$$\hat{A}_{\mathrm{MAXRL}}(x,y) = \frac{r(x,y) - \hat{\mu}}{\hat{\mu}}$$

where $\hat{\mu} = \frac{\sum_{i=1}^K r(x,y^{(i)})}{K}$, $\hat{\sigma} = \sqrt{\frac{\sum_{i=1}^K (r(x,y^{(i)}) - \hat{\mu})^2}{K}}$ is the mean and standard deviation of rewards of the sampled rollouts.

Finally, at each training step, a batch of (input image, label) pairs are collected from the training dataset. The above computation gives us per (input image, label) loss, we average them over all the pairs in a given batch to calculate the final loss which is then used to update the model via gradient descent.

D.2 Training Hyperparameters

We use the following set of hyperparameters in all ImageNet experiments:

- **Batch size**: 256
- **Number of epochs**: 20
- **Optimizer**: SGD with momentum 0.9, no Nesterov momentum, initial learning rate 0.1. We run a sweep over the learning rate over 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 0.7, and 1.0. We find the standard learning rate, 0.1, generally works well for all objectives, and report that in our experiments.
- **Learning rate scheduler**: Cosine scheduler (Loshchilov and Hutter, 2017), with linear warmup for the first epoch.
- **Image augmentations**: No augmentations are used for evaluation, we only resize each image to 224x224 and normalize the images by mean and standard deviation of pixel values. For training, in addition to the same resizing and normalizing steps, we also add a random horizontal flip (with probability 0.5) and a random resized crop to 224 (with scale (0.08, 1.0)).
- **Number of rollouts, K**: This is usually varied for different experiments.

All training is done on single L40S GPUs for 15 hours.

D.3 Equivalence of Validation Top-1 Accuracy and Majority Voting Accuracy

In this section, we discuss the validation top-1 accuracy metric, which is the traditional metric used in image classification. Formally, validation accuracy for a single image and label pair $(x, y^\star(x))$ is defined as:

$$\text{Accuracy}(x, y^\star(x); \pi_\theta) = \mathbb{I}\left[\arg\max_{y \in \mathcal{Y}} \pi_\theta(y|x) = y^\star(x)\right] = \begin{cases} 1, & \text{if } \arg\max_{y\in\mathcal{Y}} \pi_\theta(y|x) = y^\star(x) \\ 0, & \text{otherwise} \end{cases}$$

which is then averaged over all validation examples for the final metric. In other words, validation accuracy is the same as majority voting accuracy (Wang et al., 2023a) in traditional LLM chain-of-thought reasoning tasks.

D.4 Pass@k Calculation

To calculate pass@k from a generative model, one usually samples $T \geq k$ rollouts from the model, calculate success or failure from each of them, and then uses an appropriate statistical estimator for pass@k (Chen et al., 2021; Yue et al., 2025). However, since there is no latent reasoning process involved in our didactic ImageNet experiments and since we can directly calculate the model likelihood of label $y \in \mathcal{Y}$ for an input image $x \in \mathcal{X}$, namely $\pi_\theta(y|x)$, we can also analytically compute pass@k without sampling as well. Formally, in all ImageNet experiments, we calculate pass@k for an example (image, label) pair $(x, y^\star(x))$ as follows:

$$\text{Pass@}k(x, y^\star(x); \pi_\theta) = 1 - (1 - \pi_\theta(y^\star(x)|x))^k$$

The average pass@k is then obtained by averaging the above quantity over all example pairs in the validation dataset.

D.5 Gradient Norm Analysis

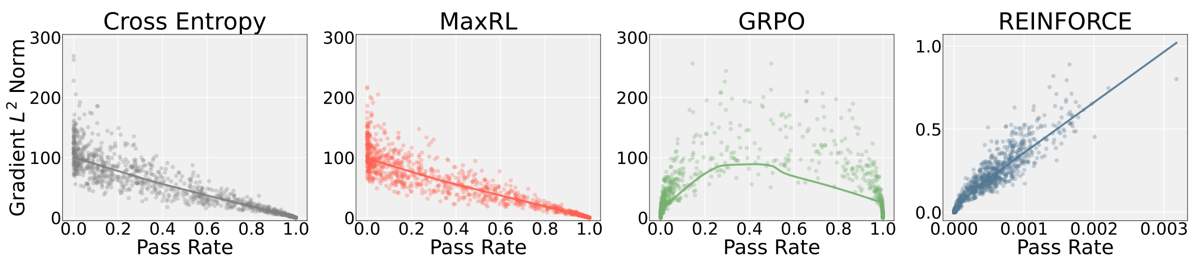

Figure 8: (ImageNet Gradient Norm Analysis) Scatter plot, where each point has the pass rate (model's predicted probability of the correct class) of a particular image in the x-axis, and gradient L^2 norm for that image in the y-axis, for 1000 randomly selected images from the ImageNet validation dataset after 1500 steps of training on a ResNet-50 model. Sampling based algorithms' (MaxRL, GRPO and REINFORCE) gradients are calculated using 131,072 rollouts per example to reduce sampling error and estimate the population level gradient. Cross Entropy and MaxRL have similar scatter plot: with high gradient norm for hard inputs (pass rate close to 0) and lower gradient norm for the easier ones (pass rate close to 1). In contrast, highest gradient norm for GRPO is on medium difficulty (pass rate close to 0.5) inputs, with hard inputs having very low gradient norm. Finally, REINFORCE fails to produce any significant gradient norm and its pass rate is confined below 0.003 after 1500 steps, demonstrating its difficulty to learn in this setting.

Figure 8 shows the correlation between gradient norm and pass rate (model's predicted probability of the correct class) for a particular image on different objectives. We see that cross-entropy and MaxRL have similar scatter plot: with high gradient norm for hard inputs (pass rate close to 0) and lower gradient norm for the easier ones (pass rate close to 1). In contrast, highest gradient norm for GRPO is on medium difficulty (pass rate close to 0.5) inputs, with hard inputs having very low gradient norm. Finally, REINFORCE fails to produce any significant gradient norm compared to the other objectives and its pass rate is confined below 0.003 after 1500 steps, demonstrating its difficulty to learn in this setting. This is also reflected in our other results, where REINFORCE does not show any signs of learning. We attribute this to the very low gradient norm: since the randomly initialized model has pass rate 0.001 in expectation over all inputs, REINFORCE fails to produce sufficiently large gradients during training and therefore stalls in model improvement. One caveat: REINFORCE's failure maybe due to us training the model from scratch — on a pretrained model, it indeed produces gradients but still shows poor gradient norm on hard inputs (see Figure 6).

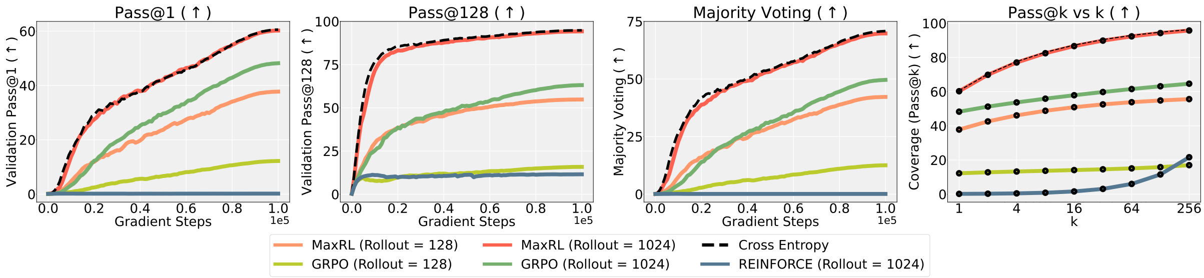

Figure 9: (Additional ImageNet results) On the didactic image classification setting, MaxRL outperforms and scales better than GRPO with additional compute, and approaches the same performance as maximum likelihood training via cross-entropy given sufficient number of rollouts (≥ 1024). Note that REINFORCE remains flat, since the initial model's pass rate is low ($\sim 0.1\%$) and REINFORCE fails to generate significant gradient signal (Figure 8). From left, the plots show Pass@1, Pass@128, Majority Voting Accuracy (equivalent to traditional validation top-1 accuracy in image classification, see Appendix D.3), and coverage of the final checkpoint, respectively.

D.6 More Experimental Results

Here we present additional experimental results. In particular, **(1)** we compare against GRPO with varying number of rollouts, **(2)** record additional metrics such as majority voting accuracy (i.e., validation top-1 accuracy), and **(3)** show the resulting coverage (pass@k vs k) from different objectives. Figure 9 records our findings: MaxRL outperforms and scales better than GRPO with additional compute. While GRPO improves performance if given more compute unlike REINFORCE, it remains suboptimal compared to MaxRL and supervised cross-entropy training. Moreover, both GRPO and REINFORCE exhibit worse coverage as their pass@k values are significantly lower compared to MaxRL, corroborating our experiments from other sections.

E Details on Other Training Settings

E.1 Maze

E.1.1 Model Architecture

We adopt a lightweight decoder-only Transformer model following the Qwen2 architecture (Yang et al., 2024a), with a total of approximately $3M$ parameters. The model consists of 4 Transformer layers, each using full self-attention. The hidden size is set to 256, with an intermediate (feed-forward) dimension of 1024, and 4 attention heads per layer. We use grouped query attention with 2 key-value heads. The model employs RMSNorm with $\sigma = 1 \times 10^{-6}$ and uses the SiLU activation function in the feed-forward networks. Rotary positional embeddings (RoPE) (Su et al., 2021) are applied with $\theta = 1,000,000$, and the maximum sequence length is 512 tokens. The vocabulary size is 32 tokens, and input and output embeddings are tied. The model is trained and evaluated using bfloat16 precision, with attention dropout set to 0. The architecture follows a standard causal language modeling setup with autoregressive decoding.

E.1.2 Task Description

Mazes are procedurally generated using Prim's algorithm (Prim, 1957), and task difficulty is controlled by the grid size. We use a symbolic tokenization to represent both the maze layout and the navigation policy, with tokens drawn from a small, discrete vocabulary.

The input sequence describes a two-dimensional grid in row-major order. Each cell is represented by a single token indicating its type (e.g., WALL, PATH, START, or GOAL). Rows are separated by a dedicated NEWLINE token, and the entire grid is delimited by special boundary tokens marking the beginning (GRID_START) and end (GRID_END) of the grid description. Following the maze specification, the model autoregressively generates a sequence of navigation actions drawn from a fixed action vocabulary (e.g., directional moves) and terminates by a DONE token.

Below, we provide an example data instance following this format.

7*7 Maze Example Model Input and Output Format

Input:

```
<bos> GRID_START WALL WALL WALL WALL WALL WALL WALL NEWLINE WALL START WALL PATH PATH PATH WALL NEWLINE WALL PATH WALL PATH WALL WALL WALL NEWLINE WALL PATH PATH PATH PATH PATH WALL NEWLINE WALL PATH WALL WALL WALL PATH WALL NEWLINE WALL PATH WALL PATH PATH GOAL WALL NEWLINE WALL WALL WALL WALL WALL WALL WALL NEWLINE GRID_END PATH_START
```

Output:

```
RIGHT RIGHT RIGHT RIGHT DOWN DOWN DOWN DOWN DONE <eos>
```

For reference, we also visualize one typical successful trajectory and one representative failed prediction in Figure 10.

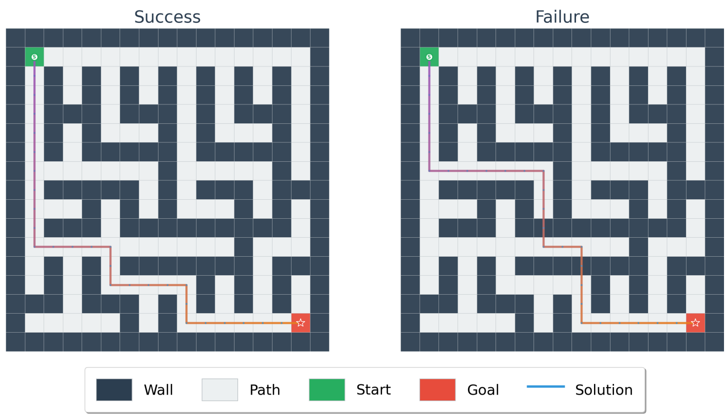

Figure 10: (Maze Data Visualization) The left plot shows a successful navigation trajectory, while the right plot illustrates a failure case produced by the trained model, where the generated action sequence deviates from the correct path before reaching the goal.

E.1.3 Training Setups

To ensure sufficient task complexity and rigorous evaluation, we construct a training set of 1 million distinct 17×17 mazes and a test set of 256 non-overlapping samples. We first pretrain the model from scratch, where it is trained to follow a provided ground-truth trajectory for each maze. During SFT, we use a learning rate of 5×10^{-4} with the `AdamW` optimizer (Kingma and Ba, 2017; Loshchilov and Hutter, 2019) and train for 1,500 steps with a batch size of 32. This pretraining stage initializes the model with the basic output format required for representing maze-solving trajectories. Subsequently, we perform reinforcement learning (RL) training. By default, we use a data batch size of 32, a rollout number of 128, a learning rate of 1×10^{-4}. We update the model parameters only once per RL step (fully on-policy setting (Tajwar et al., 2024)) to ensure all trajectories are on-policy, with 20,000 training steps. On this basis, we compare the performance of different RL algorithms.

In addition, we fix the data batch size to 256 and vary the rollout number from 4, 8, 16, 32, 64 to 128 when illustrating the scaling trends of GRPO and our MaxRL algorithm. Each training run uses 4 RTX 4090 GPUs and is conducted for at least 9,000 steps. Given the small number of model parameters, the model is no longer memory-bound, so we modified the rollout engine (instead of using the default vLLM (Kwon et al., 2023) engine) in the `verl` (Zhang et al., 2024; Sheng et al., 2024) codebase to increase sampling parallelism and speed up training.

E.2 GSM8K

E.2.1 Prompt template

We conduct reinforcement finetuning for the `SmolLM2-360M-Instruct` model (Allal et al., 2025) on GSM8K (Cobbe et al., 2021) training set for 1500 steps. We use GSM8K-Platinum (Vendrow et al., 2025), a revised version of the original GSM8K test that is non-overlapping with the GSM8K train dataset, for evaluation. An example of prompt and the model's response are shown below. In particular, we take each individual problem, append it with " Let's think step by step and output the final answer within `\\boxed{}`." and process the string through the SmolLM2-360M-Instruct model's tokenizer chat-template to obtain the final prompt for the model.

SmolLM2-360M-Instruct

```
<|im_start|>system
You are a helpful AI assistant named SmolLM, trained by Hugging Face<|im_end|>
<|im_start|>user
Jeannie hikes the 12 miles to Mount Overlook at a pace of 4 miles per hour, and then returns at
a pace of 6 miles per hour. How long did her hike take, in hours? Let's think step by step and
output the final answer within \\boxed{}. <|im_end|>
<|im_start|>assistant
To find the time it takes for Jeannie to hike the entire trip, we need to calculate her total
travel time and then divide it by the rate at which she travels.
Jeannie hikes the 12 miles at a pace of 4 miles per hour. So, the time taken is 12 miles / 4
miles per hour = 3 hours.
Next, she returns at a pace of 6 miles per hour. So, the time taken to return is 12 miles / 6
miles per hour = 2 hours.
Therefore, the total time taken is 3 hours + 2 hours = 5 hours.
So, the final answer is 5 hours. \\boxed{5} <|im_end|>
```

E.2.2 Hyperparameters

Detailed hyperparameter setup is provided in Table 5. For other baselines, we use $T = 16$ for PKPO, which means we optimize Pass@16 as the objective, and $\gamma = 0.05$ for differential smoothing.

Table 5: Training hyperparameters for SmolLM2-360M-Instruct on GSM8K.

Training hyperparameters for SmolLM2-360M-Instruct

Parameter	Value	Parameter	Value
Base model	SmoLM2-360M-Instruct	Training set	GSM8K
Test set	GSM8K	Prompts per batch	256
Generations per prompt	128	Grad update per RL step	1
Max prompt length	512	Max response len	2048
Learning rate	1×10^{-5}	Training Steps	1500
KL coeff	0.0	Entropy coeff	0.0
Rollout temp	1.0	Validation top_p	0.95
Validation temp	0.6	Device	8 × Nvidia GH200

E.3 Qwen3 Training

E.3.1 Prompt template

We use the Qwen-math template (Yang et al., 2024a; Qwen et al., 2025; Yang et al., 2024b) for formatting our prompts. We show an example prompt (Yu et al., 2025a) after formatting through our template below. In particular, we take each individual problem, append it with "\nPlease reason step by step, and put your final answer within \\boxed{{}}." and process the string through the SmolLM2-360M-Instruct model's tokenizer chat-template to obtain the final prompt for the model.

Qwen Math Prompt Template

```
<|im_start|>system
Please reason step by step and put the final answer in \\boxed{}. <|im_end|>
<|im_start|>user
Denote by S(n) the sum of the digits of the positive integer n. Find all the solutions of the
equation n(S(n) − 1) = 2010. Let's think step by step and output the final answer within \\boxed{}.
<|im_end|>
<|im_start|>assistant
```

E.3.2 Hyperparameters

Next, we describe the default hyperparameters for our training setup. Since there are many possible alternatives to handle off-policy updates and corresponding importance ratio (Schulman et al., 2017; Shao et al., 2024; Zheng et al., 2025; MiniMax et al., 2025; Yu et al., 2025a), to keep things simple, we choose to train in the fully on-policy setup, meaning we have no importance ratio or associated clipping. Similarly, to avoid tuning additional hyperparameters for each algorithm, following Olmo et al. (2025), we remove KL penalty and also entropy bonus in our default training comparison. Note: we train with GRPO and entropy bonus as a baseline in our SmolLM2-360M-Instruct training on GSM8K, results are recorded in Table 4: MaxRL outperform this variant, showing that entropy bonus does not fully mitigate issues resulting from GRPO though it can slightly mitigate it, as also observed by Yue et al. (2025).

We generate all training rollouts using temperature 1.0, and do not use special sampling techniques. Similarly, we also do not use any adaptive sampling (Yu et al., 2025a) or fixes for inference-training logit mismatch (He and Lab, 2025; Khatri et al., 2025) Finally, for evaluation, we follow the same protocol as Yue et al. (2025), and we run inference with temperature 0.6, top-p sampling parameter 0.95, no top-k or min-p sampling (Nguyen et al., 2025a).

Table 6 shows our default hyperparameter setting.

Table 6: Training hyperparameters for Qwen3-1.7B-Base and Qwen3-4B-Base training.

Training hyperparameters for Qwen3-1.7B-Base and Qwen3-4B-Base

Parameter	Value	Parameter	Value
Base model	Qwen3-1.7B-Base, Qwen3-4B-Base	Prompts per batch	256
Generations per prompt	16	Grad update per RL step	1
Max prompt length	1024	Max response len	4096
Learning rate	1×10^{-6}	Training Steps	1000
KL coeff	0.0	Entropy coeff	0.0
Rollout temp	1.0	Validation top-p	0.95
Validation temp	0.6	Device	32 $\times$ Nvidia H200

F Implementation Details on RL Training for LLM Experiments

Our discussion here follows that of (Shafayat et al., 2025). For continuity with existing literature, we use slightly different notations from the rest of the paper for this section. Let x represent a prompt, and let $y \sim \pi(\cdot|x)$ represent sequence of tokens autoregressively sampled from the language model π conditioned on the prompt x. Let π_θ be the current policy, and $\pi_{\theta_{\text{old}}}$ be an older policy (from earlier iterations in training) used for data generation. In our implementation (based on `verl` (Zhang et al., 2024; Sheng et al., 2024)), we use the following general RL objective:

$$\mathcal{J}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\ \{y_i\}_{i=1}^G \sim \pi_{\theta_{\text{old}}}(\cdot|x)} \left[\frac{1}{T} \sum_{i=1}^{G} \sum_{t=1}^{|y_i|} \min \left(w_{i,t}(\theta) \hat{A}_{i,t}, \text{clip}(w_{i,t}(\theta),\ 1-\varepsilon,\ 1+\varepsilon) \hat{A}_{i,t} \right) \right]$$

where T is the total number of tokens in the mini-batch (excluding tokens in the prompt etc., since we only compute loss on the model generated tokens), π_θ represents the current LLMs autoregressive probability distribution, $\pi_{\theta_{\text{old}}}$ denote the behavior policy/data generation policy's probability distribution, $w_{i,t}(\theta)$ is the importance ratio, defined as:

$$w_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t} \mid x,\ y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t} \mid x,\ y_{i,<t})}$$

Since we operate fully on-policy, i.e., one RL step per one batch of generated rollouts, this is always one in our experiments, and the clipping parameter ϵ has no effect on our training. $\hat{A}_{i,t}$ represents the advantage for the t-th token in the sequence y_i. The same advantage defined at a sequence level is applied to each token in the sequence, so henceforth we will drop the t from the notation as well.

The main difference between GRPO (Shao et al., 2024), RLOO (Ahmadian et al., 2024) and MaxRL come from their use of different advantage functions. RLOO objective uses the following advantage function:

$$\frac{1}{G} \sum_{i=1}^{G} [R(y_{(i)}, x) - \frac{1}{G-1} \sum_{j\neq k} R(y_{(j)}, x)]$$

whereas GRPO uses the following advantage function:

$$\hat{A}_i = \frac{r(x, y_i) - \text{mean}\left(\{r(x, y_i)\}_{i=1}^G\right)}{\text{std}\left(\{r(x, y_i)\}_{i=1}^G\right) + \epsilon}$$

where ϵ is a small number (1×10^{-6}) added to avoid division by zero. Finally, the advantage for MaxRL is follows:

$$\hat{A}_i = \frac{r(x, y_i) - \text{mean}\left(\{r(x, y_i)\}_{i=1}^G\right)}{\text{mean}\left(\{r(x, y_i)\}_{i=1}^G\right) + \epsilon}$$

Here G is the number of online samples generated. RLOO, GRPO and MaxRL create a dynamic baseline for each sample without needing a separate value function (unlike PPO (Schulman et al., 2017)), effectively estimating the expected return on-the-fly during training. Not having a value networks makes the training much simpler for all three algorithms.

G Pass@k Calculation for Tasks with Sampling

Unlike the ImageNet setting, we can't usually directly calculate pass@k via accessing the true probability of the correct action. Therefore, we use the default pass@k calculation mechanism in `verl` (Sheng et al., 2024; Zhang et al., 2024), using the bootstrapping low variance unbiased estimator introduced by Chen et al. (2021). This employs generating $n \geq k$ samples per task, counting the number of correct samples $c(x)$ among the n samples, and estimate pass@k as:

$$\text{Pass@}k = \mathbb{E}_{x\sim\rho} \left[1 - \frac{\binom{n-c(x)}{k}}{\binom{n}{k}} \right]$$

H Qwen3 Model Evaluation on Additional Benchmarks

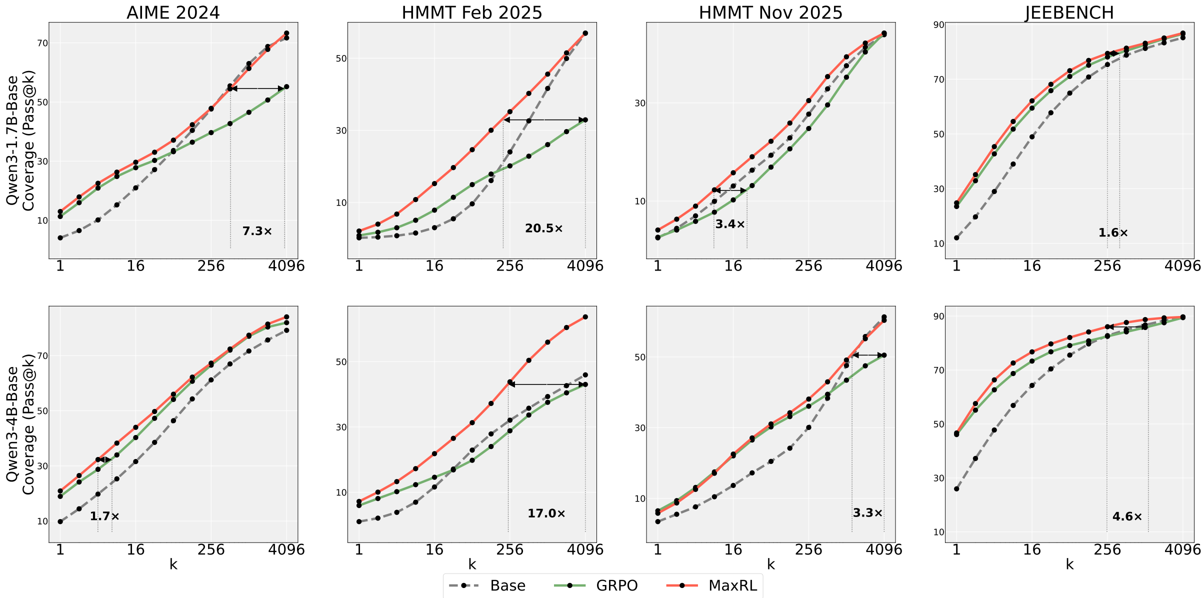

Figure 11: (Evaluation of Qwen3 model training on additional benchmarks) Here we report coverage on 4 additional benchmarks, namely AIME 2024, HMMT Feb 2025 (Balunović et al., 2025a), HMMT Nov 2025 (Balunović et al., 2025b), and JEEBENCH (Arora et al., 2023). MaxRL match or outperform both base model and GRPO, leading up to 20.5× speedup compared to GRPO while generating multiple samples with a perfect verifier and maintains similar or better pass@1 performance.

I More on MaxRL Extracting Better Learning Signal During Training

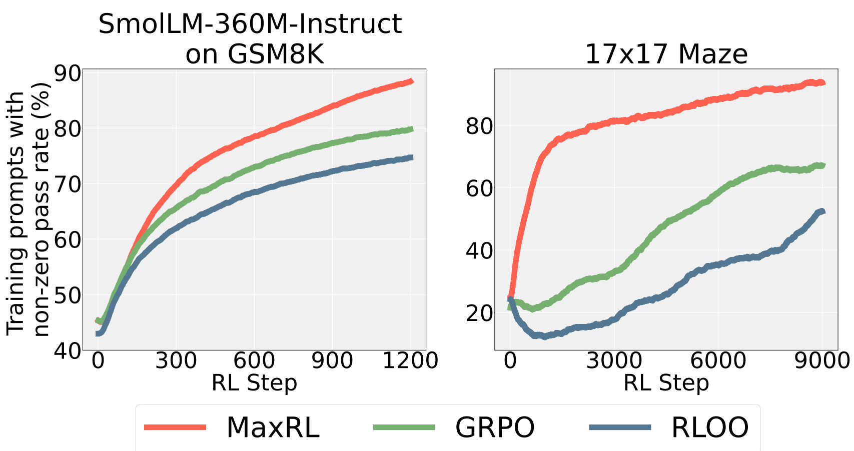

Figure 12: (Fraction of training tasks with non-zero pass rate) Similar to Figure 6, we also record the fraction of training tasks where the model generates at least one correct rollout on maze and SmolLM2-360M-Instruct training on GSM8K. We see the same trends as Figure 7, and MaxRL consistently outperforms both GRPO and RLOO, demonstrating MaxRL ability to generate better learning signal during training, as tasks with zero pass-rate contributes no gradients.

J Additional Results on Qwen3-4B-Base

J.1 Majority Voting Performance

Table 7: (Majority Voting Performance Comparison on Qwen3-4B-Base) We compare the performance of MaxRL in terms of majority voting against the pretrained base model and GRPO.

	AIME 2024 (majority@4096)	AIME 2025 (majority@4096)	BeyondAIME (majority@4096)	MATH-500 (majority@2048)	Minerva (majority@2048)
Base	23.3	23.3	7.0	69.8	18.8
GRPO	23.3	23.3	7.0	72.4	27.2
MaxRL	**26.7**	**26.7**	**14.0**	**74.0**	**28.7**

Here we present comparisons across one other metric, majority voting (Wang et al., 2023a), a commonly used verifier free method for scaling test-time compute, where we generate N i.i.d. rollouts from the model for a single task x, group the responses by the final answer, and take the most frequent answer as our outcome. Table 7 shows our results across all five benchmarks, we outperform both the pre-trained base model and GRPO trained model on majority voting across all benchmarks.

J.2 Training Dynamics

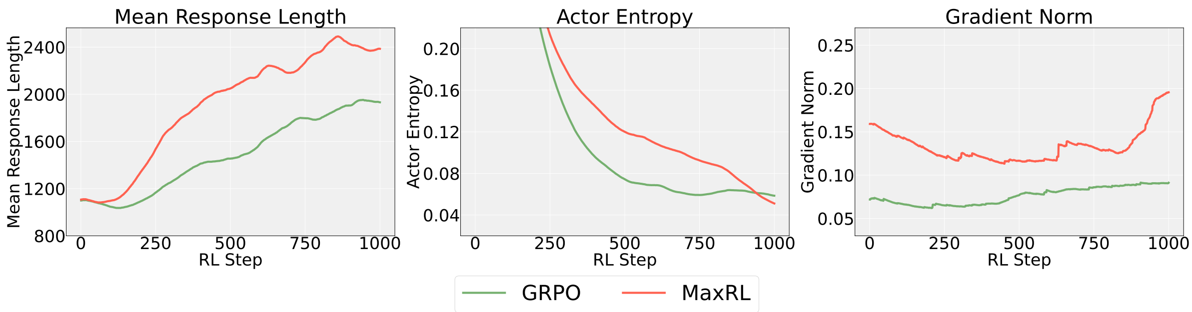

Figure 13: (Additional training dynamics metrics for Qwen3-4B-Base) We show comparison between GRPO and MaxRL in terms of mean response length, entropy of the actor, and gradient norm during training for the Qwen3-4B-Base model. MaxRL generally produces longer chains-of-thought, and also retains higher actor entropy during training. MaxRL also produce larger gradient norm during training.

J.3 Validation Accuracy During Training

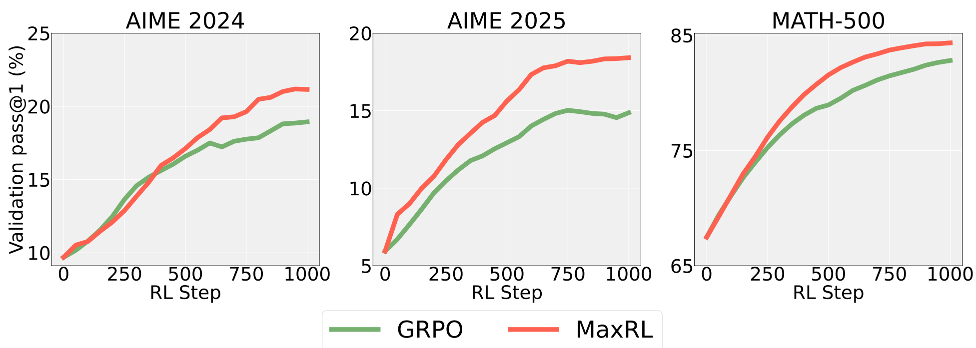

Figure 14: (Qwen3-4B-Base validation pass@1 during training) Pass@1 (estimated using 32 samples) during training of Qwen3-4B-Base, on 3 different evaluation dataset. MaxRL consistently outperform GRPO during training.

K Additional Results on Qwen3-1.7B-Base

K.1 Validation Accuracy During Training

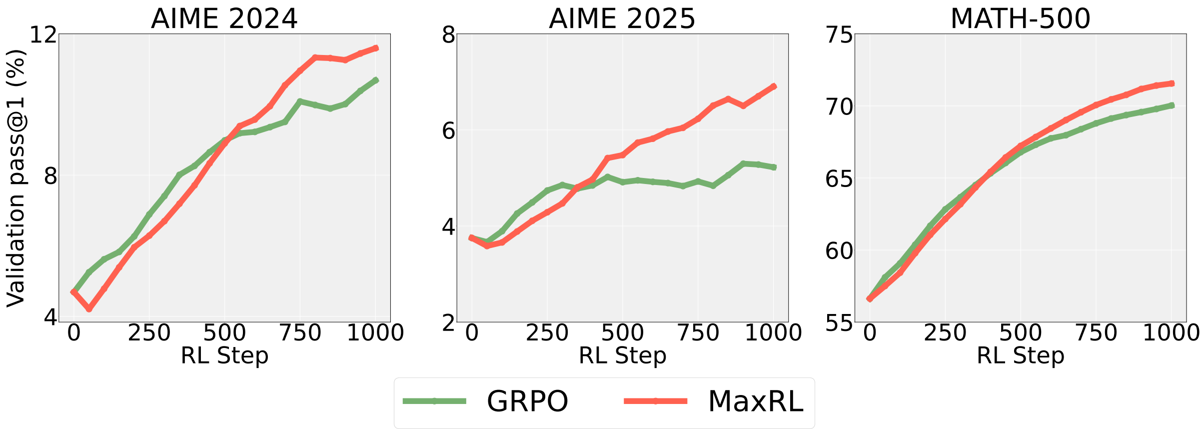

Figure 15: (Qwen3-1.7B-Base validation accuracy during intermediate training) We record validation pass@1 (using mean over 32 rollouts per prompt) over AIME 2024, AIME 2025 and MATH-500 during Qwen3-1.7B-Base model training. Similar to Figure 4, we observe that MaxRL initially trail behind GRPO at pass@1, but catches up with extended training and then converges to a higher value.

K.2 Training Dynamics

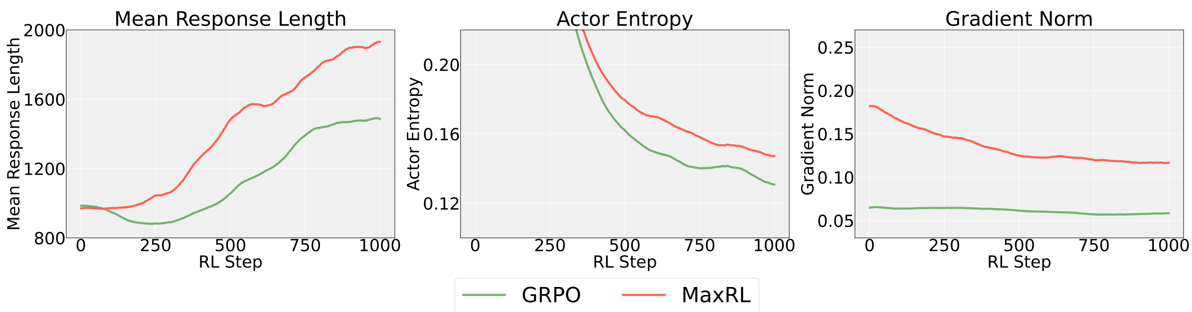

Figure 16: (Additional training dynamics metrics for Qwen3-1.7B-Base) We show comparison between GRPO and MaxRL in terms of mean response length, entropy of the actor, and gradient norm during training for the Qwen3-1.7B-Base model. MaxRL generally produces longer chains-of-thought, and also retains higher actor entropy during training. MaxRL also produce larger gradient norm during training.

L Additional Experimental Results on Maze

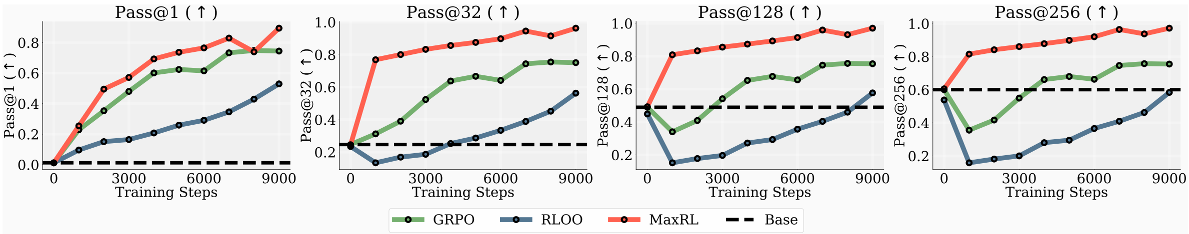

Figure 17: (Infinite training compute in maze experiment) We investigate how different objectives perform when we train a 3M model to solve 17x17 maze puzzles. MaxRL performs significantly better compared with GRPO and REINFORCE in Pass@1, Pass@32, Pass@128 and Pass@256. These results signify MaxRL's effectiveness in computation scaling during RL.

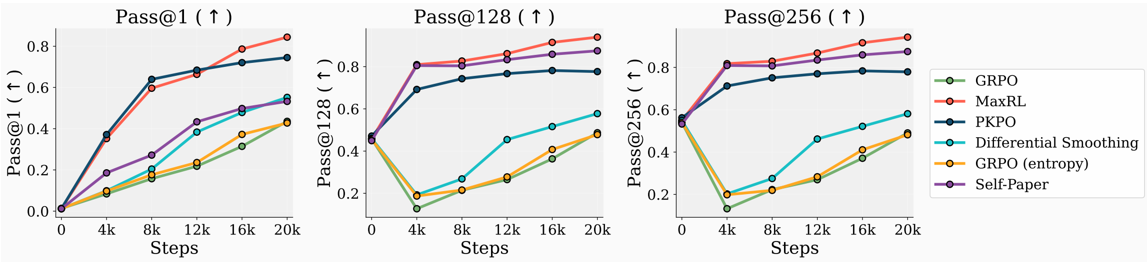

Figure 18: (Training curves compared with other baselines.) We compare MaxRL with other RL algorithms, including entropy regularization, PKPO (Walder and Karkhanis, 2025), Differential Smoothing (Gai et al., 2025) and SELF (Nguyen et al., 2025b). MaxRL significantly outperforms other methods in all metrics, and is the only method to maintain both good average performance (pass@1) and coverage (pass@k).