KL-REGULARIZED REINFORCEMENT LEARNING IS DESIGNED TO MODE COLLAPSE

Anonymous authors
Paper under double-blind review

ABSTRACT

It is commonly believed that optimizing the reverse KL divergence result in "mode seeking", while optimizing forward KL result in "mass covering", with the latter being preferred if the goal is to sample from multiple diverse modes. We show—mathematically and empirically—that this intuition does not necessarily transfer well to doing reinforcement learning with reverse/forward KL regularization (e.g. as commonly used with language models). Instead, the choice of reverse/forward KL determines the *family* of target distributions which maximizes the objective, while mode coverage depends primarily on other factors, such as regularization strength. Further, we show commonly used settings such as low regularization strength and equal verifiable rewards tend to specify uni-modal target distributions, meaning the optimization objective is *by construction* non-diverse. We leverage these insights to construct a simple yet principled algorithm, which makes minimal changes to reward magnitudes, and theoretically prove that it optimizes for a target distribution which puts high probability over *all* high-quality sampling modes. We empirically show this simple modification works to post-train both Large Language Models and Chemical Language Models to have higher solution quality and diversity, without external signals of diversity, and works with both forward and reverse KL when using either naively fails.

1 INTRODUCTION

Reinforcement Learning (RL) is the predominant way for post-training foundation models (Ouyang et al., 2022), and the only way to train models in settings where the correct solution is not known *a priori*. Output diversity in the trained policy is important. Traditionally in chat-bot settings with Large Language Models (LLMs), diversity drives engagement for tasks such as creative writing and free-form conversation. Diversity also drives the generation of new knowledge, such as discovering new mathematical solutions (Romera-Paredes et al., 2024), cognitive science models (Castro et al., 2025), and novel algorithms and software (Surina et al., 2025; Novikov et al., 2025; Aygün et al., 2025). Further, diversity is fundamentally tied to an expression of uncertainty over possible hypotheses for scientific discovery (GX-Chen et al., 2025).

At its core, the technical problem involves solving a regularized RL problem, where the foundation model is trained to maximize some external reward, while preserving "closeness" to a base policy (as to e.g. preserve coherence). Yet, current empirical evidence suggests RL post training improves quality at the cost of diversity (Kirk et al., 2023; Cui et al., 2025). As a response, a number of recent works set out to treat this ailment, with a variety of approaches including explicit diversity rewards (Li et al., 2025), changing the KL regularization (Wang et al., 2023), selecting diverse data (Lanchantin et al., 2025), and count-based exploration bonuses (Song et al., 2025).

In this work, we take a step back to diagnose a more fundamental problem: *does the objective being optimized actually have a solution that is diverse?* We find that with current set-ups, the answer is often "no", even with unlimited compute, high quality data, and perfect optimization. We prove that under very commonly used settings (such as weak KL regularization with varied rewards, or *any* regularization using the same reward for all correct answers), the globally optimal solution is often *by construction* unimodal.

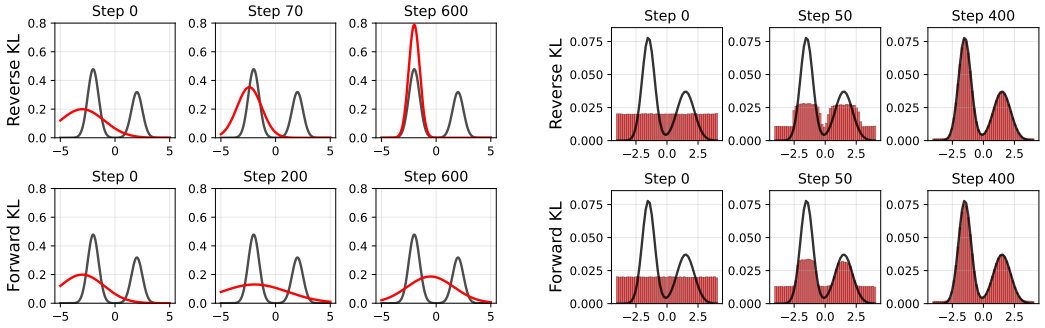
To see this, we view RL through the lens of approximate inference: by analyzing how the policy moves toward the solution distribution. Section 2 provides preliminaries about KL divergences.

Section 3 extends this to the setting of reward maximization *with* KL regularization, and derive a set of facts about the *gradient* and *optimal solution* of KL-regularized RL objectives (for both reverse and forward KL). Section 4 more deeply analyzes the *shape* of this optimal solution, how it is sculpted by the reward, reference policy, and regularization strength, particularly focusing on implication for multi-modality. This allows us to understand diversity collapse not as a quirk of post-training, but as a natural consequence of the RL objective as currently defined. Finally, in Section 5 we move beyond the current objective and suggest directly constructing the solution distribution to be diverse. We specify one such distribution which puts mass over all high reward regions above a certain threshold, and show that this requires only a small change to current algorithms. Each section is empirically supported with didactic simulations, and we also train LLMs and chemical language models to demonstrate our method works, out of the box, for complex, realistic scenarios.

The main contributions can be summarized as follows,

1. We show RL with reverse/forward KL-regularization define different *families* of solution distributions, with levels of mode coverage depending primarily on regularization strength and reward shapes, rather than the type of KL (contrary to common intuitions).
2. We show that with typical RL hyperparameters, the solution distribution is often *by definition* uni-modal, regardless of regularization types, making diversity collapse a natural consequence of solving the RL problem.
3. We derive conditions required for multi-modal solution distributions, and use this insight to construct a simple and principled RL algorithm that directly optimizes for multi-modality, without the need for any external diversity signals.

2 The Kullback-Leibler (KL) Divergence



(a) Restrictive (Gaussian) approximation

(b) Flexible (categorical) approximation

Figure 1: Illustration of how the choice of approximate distribution family affects KL optimization. With a restrictive approximate distribution (e.g. two-parameter Gaussian), KL exhibit the typical "mode seeking" and "mass covering" characteristics. This intuition does not necessarily hold for flexible distributions (e.g. independent categoricals, foundational models).

The Kullback–Leibler (KL) divergence (Kullback & Leibler, 1951) measures the discrepancy between two probability distributions. In machine learning, it is commonly used in variational inference (VI), where minimizing the KL divergence enables a tractable variational distribution q to approximate an intractable posterior p (Jordan et al., 1999; Blei et al., 2017). Following Murphy (2012), we refer to $D_{KL}(q||p) = \mathbb{E}_q[\log q(y) - \log p(y)]$ as the *reverse KL divergence*, and $D_{KL}(p||q) = \mathbb{E}_p[\log p(y) - \log q(y)]$ as the *forward KL divergence*. Reverse KL is often described as "mode seeking", avoiding mass where p is small (Figure 1a, top), while forward KL is often described as "mass covering", putting mass anywhere p has mass (Figure 1a, bottom). These intuitions hold *if* the variational family is not sufficiently expressive and we can at best settle on a **local optimum** (Bishop & Nasrabadi, 2006; Murphy, 2012). With a flexible family, however, optimizing either KLs to the **global optimum** can well-approximate a complex posterior (Figure 1b).

3 KL-Regularized Reward Maximization

KL-regularized reward maximization aims to (i) maximize a reward function $R : \mathcal{Y} \to \mathbb{R}$, mapping from samples to a scalar outcome (e.g. improve human preference), while (ii) keeping the policy

π_θ close to a reference distribution π_{ref} (e.g. maintain grammatical coherence). The objective is $J(\pi_\theta) = \mathbb{E}_{\pi_\theta(y)}[R(y)] - \beta\, D\big(\pi_\theta, \pi_{\text{ref}}\big)$, where $D(\cdot,\cdot)$ denotes a divergence between the policy and reference distributions. For brevity, we consider the unconditional generation problem where the policy models distribution $\pi_\theta(y)$. The problem is the same in the case of conditional generation (e.g. question answering), where the objective is simply defined over the conditional distribution $\pi_\theta(y|x)$.

In this section, we consider the *solution / target distribution* of KL-regularization reward maximization—i.e. the distribution which maximizes the objective. The central question is:

> *If we perfectly solve the RL problem to its global optimum, what does the solution (policy) distribution look like?*

3.1 Solution of the Reverse KL Regularized Objective

The most common KL-regularized policy gradient objective uses the *reverse KL divergence*,

$$J_\beta(\pi_\theta) = \mathbb{E}_{\pi_\theta(y)}[R(y)] - \beta\, D_{KL}\big(\pi_\theta || \pi_{\text{ref}}\big)\,. \tag{1}$$

A number previous works have discussed the solution / optimal distribution of this optimization problem (Korbak et al., 2022; Go et al., 2023; Rafailov et al., 2023), which we note again below.

Remark 3.1. *The optimal solution to the reverse-KL regularized reward maximization problem,* $\arg\max_{\pi_\theta} J_\beta(\pi_\theta)$*, is given by the solution distribution* $\pi^* = G_\beta$*,*

$$G_\beta(y) = \frac{1}{\zeta}\, \pi_{ref}(y) \exp\left(\frac{R(y)}{\beta}\right), \tag{2}$$

where $\zeta = \int \pi_{ref}(y) \exp(R(y)/\beta)\, dy$ *is the normalizing constant.*

Proof. Appendix B.1. □

Remark 3.1 tells us the solution distribution maximizing Equation 1 is $\pi_\theta = G_\beta$. However, it may not be immediately obvious *how* the gradient of Equation 1, $\nabla_\theta J_\beta(\pi_\theta)$, moves π_θ toward G_β. We analyze this to understand the behaviour of optimizing Equation 1.

Remark 3.2. *The gradient of Equation 1 is a gradient of the reverse KL divergence between the current policy* π_θ *and the target distribution* G_β*,*

$$\nabla_\theta\, D_{KL}\big(\pi_\theta \,||\, G_\beta\big) \propto -\nabla_\theta\, J_\beta(\pi_\theta)\,. \tag{3}$$

Proof. Appendix B.2. □

Main Takeaway

Maximizing the reverse-KL regularized RL objective J_β (Equation 1) is equivalent to doing distribution matching by minimizing a reverse KL toward the target distribution G_β (Equation 2).

3.2 Solution of the Forward KL Regularized Objective

Alternatively, we can regularized the reward maximizaztion with a forward KL penalty,

$$J_{\text{fwd}}(\pi_\theta) = \mathbb{E}_{\pi_\theta(y)}[R(y)] - \beta\, D_{KL}\big(\pi_{\text{ref}} || \pi_\theta\big)\,. \tag{4}$$

A number of recent works have used forward KL regularization. Some are motivated explicitly by the "mass covering" intuition of the forward KL (Wang et al., 2023), while others—such as GRPO (Shao et al., 2024; Guo et al., 2025a)—may have incidentally estimated the forward KL, despite meaning to use the reverse KL (Tang & Munos, 2025).

Remark 3.3. *Assume the solution has the same support as* π_{ref}*, the optimal solution to the forward-KL regularized reward maximization problem,* $\arg\max_{\pi_\theta} J_{fwd}$*, is given by the distribution:*

$$G_{fwd}(y) = \frac{\beta\, \pi_{ref}(y)}{\Lambda - R(y)}\,, \quad \Lambda > \max_y R(y)\,, \tag{5}$$

where Λ *needs to be solved for each* β *such that* G_{fwd} *is a valid probability distribution.*

Proof. Appendix B.3. □

Notably, Equation 5 is a *completely different* distribution family from the reverse KL case (Equation 2), and does not have a closed form solution.

Remark 3.4. *The gradient of Equation 4 is **not** a forward KL gradient,*

$$\nabla_\theta D_{KL}(h \,||\, \pi_\theta) \not\propto -\nabla_\theta J_{fwd}(\pi_\theta)\,, \tag{6}$$

*for **any** distribution h that is defined independent of π_θ, and arbitrary reward functions R.*

Proof. Appendix B.4. □

Therefore, while Equation 4 can still be a good objective to optimize, it does not necessarily inherit the same properties as a "forward KL gradient" (e.g. intuition about "mass seeking" at sub-optimality).

What is the gradient of the forward KL $D_{KL}(G_\beta||\pi_\theta)$, then? This in fact amount to doing maximum likelihood / supervised fine-tuning on trajectories sampled from the target G_β (Remark B.1), which is intractable. However, this provides a perspective on algorithms such as STaR (Zelikman et al., 2022) and RAFT (Dong et al., 2023; Xiong et al., 2025) that filter high-reward trajectories for maximum likelihood. One can interpret filtering as approximating a target distribution (which put high mass over high-reward regions), to then optimize a forward KL towards.

Main Takeaway

Maximizing the forward-KL regularized objective J_{fwd} (Equation 4) does not yield a forward-KL gradient, so its behaviour cannot be naively equated to forward-KL optimization.

3.3 Both KL Regularization Can Have Multimodal Solution Distributions

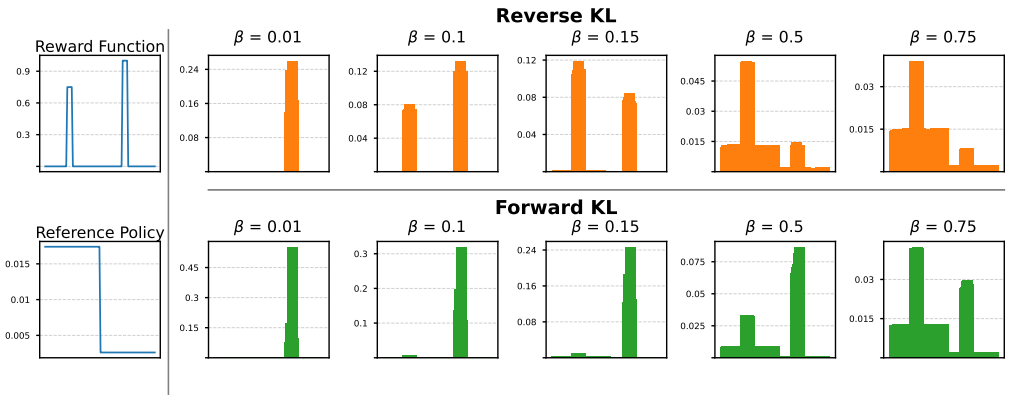


Figure 2: Final policy distribution from optimizing a reverse/forward KL regularized reward maximization objective, given the same reward function, reference policy, across a range of regularization strengths (β). Note that both KLs can lead to multi-modal solution distributions.

It is worth briefly noting that the solution distributions for both the reverse (Equation 2) and forward (Equation 5) KL regularization *can* be multi-modal. To ground the discussion, we first define a common-sense notion of "multi-modal" which we will use for the rest of the paper.

Definition 3.5. (Informal) A solution distribution for KL-regularized reward maximization is "**multi-modal**" if all high-reward samples have high probability.

We show this in a didactic example in Figure 2, where given the same reward function containing two high-reward modes, and a reference policy with support over the first half of the token space, optimizing the reverse and forward KL objectives lead to a wide variety of solutions that depend on the regularization coefficient β. Both KLs have settings of β that induce multi-modal solution distributions. We analyze the properties of the target distribution in the subsequent section, and return to the Figure 2 example in detail in Section 4.3.

4 Analysis of KL Regularized Optimal Distribution

We have seen in Section 3.3 that both KL-regularized RL objectives can have multi-modal solutions, and in Section 2 that optimizing either KL divergence to global optimum will give us policies that

well-approximate the (multi-modal) solution. However, the shape of the solution distribution depend on the reward, reference distribution, and regularization strength. This begs the central question:

> *Is the globally optimal solution we commonly define when we do KL-regularized RL actually multi-modal (Definition 3.5)?*

The central tools we use in this section is a *probability ratio* between two samples under a distribution. Intuitively, we want (i) high-reward samples to be much more probable than low-reward samples, and (ii) similarly high-reward samples to have similar high probabilities. Unless otherwise stated, we focus our analysis on the solution of the reverse-KL regularized objective (Equation 2), both for its clean form and because it is the most common way KL-regularized RL is formulated.

Proposition 4.1. *The (log) probability ratio between any two samples, y_1, y_2, under the optimal solution distribution for reverse-KL regularized RL, G_β, can be written in closed form,*

$$\log \frac{G_\beta(y_1)}{G_\beta(y_2)} = \log \frac{\pi_{ref}(y_1)}{\pi_{ref}(y_2)} + \frac{1}{\beta}\Big(R(y_1) - R(y_2)\Big). \tag{7}$$

Proof. Because normalization constant ζ cancel out in ratios. See Appendix B.6. □

This let us exactly compute how likely one sample is relative to another in the *optimal solution*, using *only* π_{ref} and the reward function R. There are a number of consequential insights about this solution.

4.1 With equal supports, small reward differences lead to large probability differences

Remark 4.2. *For any two samples y_1 and y_2, if $\pi_{ref}(y_1) = \pi_{ref}(y_2)$, their probability ratio is:*

$$\log \frac{G_\beta(y_1)}{G_\beta(y_2)} = \frac{1}{\beta}\Big(R(y_1) - R(y_2)\Big). \tag{8}$$

In words, for two samples that have the same probability under the reference distribution ("equal support"), the difference in their final log probabilities is simply the difference in their rewards, scaled by $1/\beta$. Smaller β exaggerates the difference between relative probabilities. Note a *linear* difference in rewards result in an *exponential* difference in probabilities: for a 0.1 difference in rewards, and a commonly used $\beta =$ 1e-3, the higher reward sample is 2.6×10^{43} times more likely in the solution distribution (Figure 3). This suggests for commonly used hyperparameter settings, the solution distribuiton is highly concentrated around its mode.

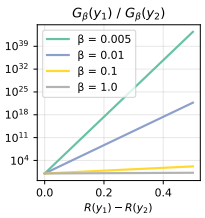


Figure 3

To build additional intuition and empirically validate the theory, we use a didactic example where we optimize a categorical distribution using KL regularized RL (details in Appendix C.1). We observe in Figure 4 that regularization strength β controls the difference in rewards, and below some threshold of regularization the solution policy becomes uni-modal.

4.2 With equal rewards, solution *never* prefers off-support samples

We now analyze the case where the correct solutions all have equal reward. This is a common set-up for the case of RL with verifiable reward (e.g. math), where a correct answer is usually given a reward of 1, and incorrect answers given reward of 0.

Remark 4.3. *For any two samples with the same reward, $R(y_1) = R(y_2)$, their probability ratio is:*

$$\log \frac{G_\beta(y_1)}{G_\beta(y_2)} = \log \frac{\pi_{ref}(y_1)}{\pi_{ref}(y_2)}. \tag{9}$$

In words, their relative probabilities in the solution is simply the relative probabilities in the reference distribution, and *do not depend on the KL-regularization strength* β.[1] In other words, with identical rewards, RL only changes the relative probability between answers with *different* rewards, but not between on- and off-support correct answers. The **RL with equal verifiable reward objective by construction discourages off-support answers**.

We empirically verify this prediction in Figure 5. We see that the final policy distribution *never* favours the (equally correct) off-support mode. This is not an issue with exploration: we will see in the subsequent section and Figure 2 that with a small change in reward we can indeed optimize for a distribution that equally weights or even prefers the off-support solution.

[1] This is the case for both reverse and forward-KL regularized RL.

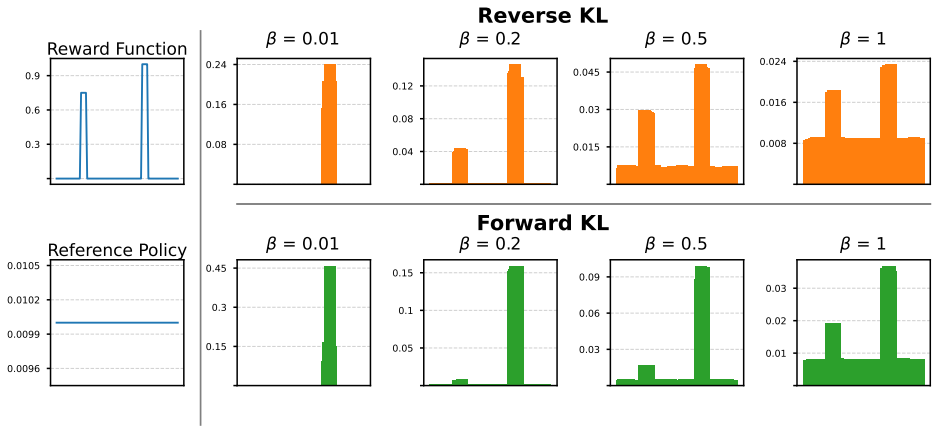


Figure 4: Final policy distribution after KL-regularized RL, where high reward regions have equal support. Here, regularization strength β controls the difference in probability between differently rewarding regions, with a moderately low β concentrating all mass on the highest reward mode.

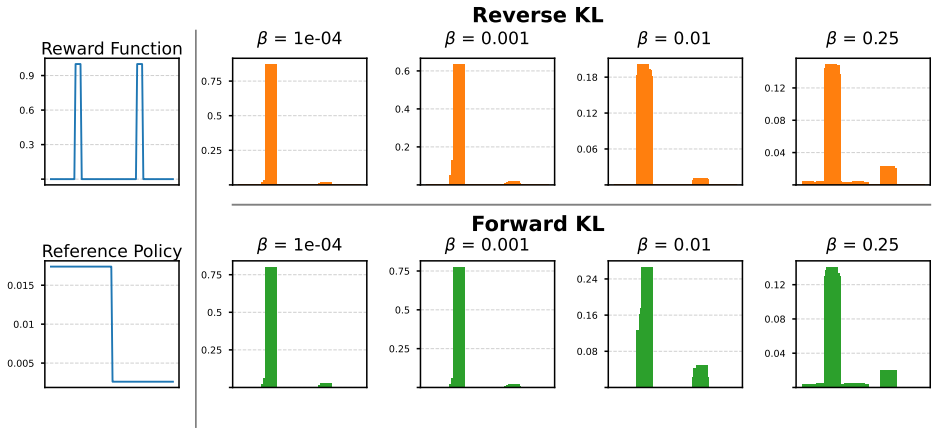


Figure 5: Final policy distribution after KL-regularized RL, with equal rewards for all correct answers. Off-support (yet equally correct) answers are never preferred over on-support answers.

Main Takeaway

KL-regularized RL does not increase the probability of off-support samples relative to on-support ones as long as their rewards are the same. Lowering the KL regularization strength β has *no effect* on up-weighting off-support samples.

4.3 For unequal rewards *and* supports, regularization strength determines mode coverage

When two trajectories have different rewards and different probabilities under the reference policy, a unique setting of β will induce the two to have the same probability in the solution distribution.

Remark 4.4. *Two samples have the same probability in the target distribution if,*

$$R(y_2) - R(y_1) = \beta\big(\log \pi_{ref}(y_1) - \log \pi_{ref}(y_2)\big)\,. \tag{10}$$

This condition allow us to predict, given only the reward and reference policy, when two samples will have the same probabilities in the solution to the RL problem. As an example, we know in Figure 2 that the two high-reward modes have rewards 0.75 and 1.0, and reference policy probabilities of $\log \pi_{\text{ref}}(y_1) \approx -4.05$ and $\log \pi_{\text{ref}}(y_2) \approx -5.95$, respectively. This allows us to predict the setting of β which will "flip" the solution distribution's preference from the on-support mode to the off-support mode to be $(1 - 0.75)/(-4.05 + 5.95) \approx 0.132$. Indeed, we see in Figure 2 for the reverse KL case, the preference between the two modes switch as we move from $\beta = 0.15$ to $\beta = 0.10$. This

is the true role of the regularization coefficient β: it is a knob that decides between picking higher rewarding, off-support solutions, vs. lower rewarding, on-support solutions.

5 Directly Optimizing a Multi-Modal Target

Having identified the various failure cases of the KL-regularized RL objective (Section 4), and the role of regularization in balancing reward differences (Section 4.3), we now turn to the question:

> *Can we construct an objective such that when optimized, naturally give rise to a multi-modal solution distribution?*

Indeed, Remark 4.4 already provides the equality condition required to achieve this. We derive a simple procedure which will ensure we are optimizing for a solution that puts *equal* probabilities on all high-quality samples (per Definition 3.5), using the augmented reward function,

$$\bar{R}(y) = \begin{cases} R(y) & \text{if } R(y) < \tau, \\ R(z) + \beta\big(\log \pi_{\text{ref}}(z) - \log \pi_{\text{ref}}(y)\big) & \text{if } R(y) \geq \tau, \end{cases} \tag{11}$$

where $\tau \leq \max_y R(y)$ is some threshold for "goodness", and z is a fixed "anchor" sample chosen from the set of high-quality samples. We can pick it to be $z = \arg\max_y \pi_{\text{ref}}(y)$ where $R(y) \geq \tau$. Because we are choosing the "anchor" to be from a high-reward mode, we colloquially refer to this as "*mode anchoring*", and the method as ***M**ode **A**nchored **R**eward **A**ugmentation* (**MARA**, Algorithm 1).

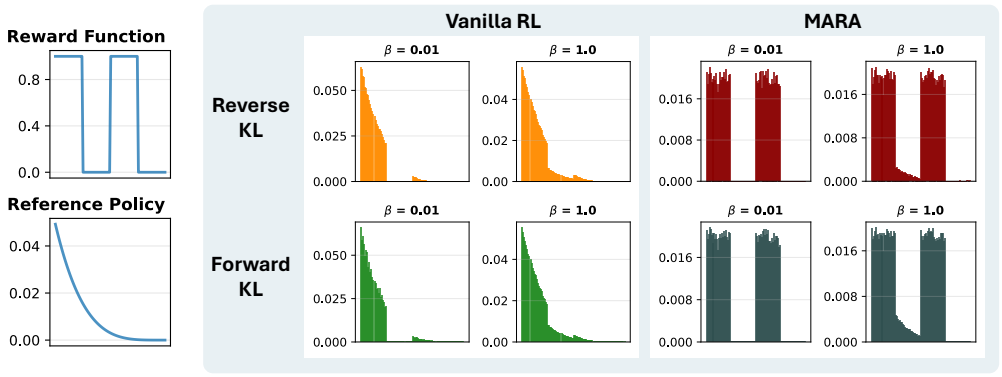


Figure 6: MARA stays close to the reference policy in low-reward areas, and puts high, uniform mass over all high-reward areas.

Intuitively, the augmented reward function constructs a new *target distribution* with *uniform* high density over regions of high reward, and stays close to the reference π_{ref} in regions of low reward (see Remark B.2 for formal proof). We see in the Figure 6 that vanilla KL-regularized RL result in a policy that heavily favours the left (on-support) mode, regardless of the choice of β or KL. On the other hand, using MARA result in solutions that put *equal* high mass over *all* high quality samples, for both KLs. Note that to setting threshold τ does require a priori knowledge of the reward range. In the case where this is not possible, we can set it in a per-batch basis by e.g. taking an upper percentile of the sampled reward.

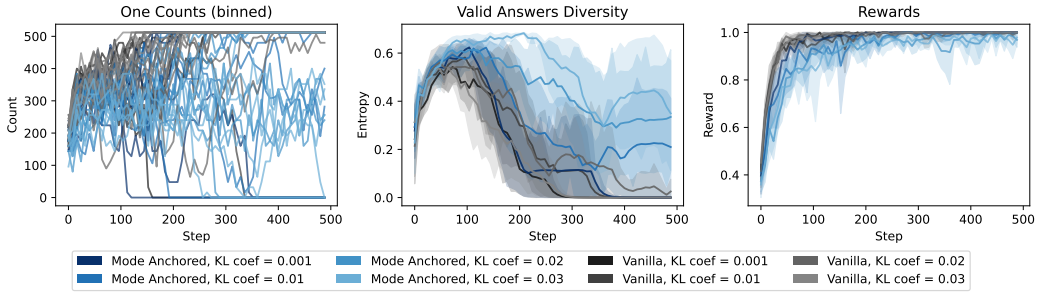
6 Empirical Validations

We now evaluate MARA as a drop-in method in a variety of post-training tasks. While our theory are mainly about the final solution, we empirically investigate whether training, even if stopped early, can still benefit from a more diverse global optimum. We evaluate MARA in (i) verifiable LLM task with multiple answers, (ii) non-verifiable task with reward models, and (iii) chemical language model task for drug discovery, where mode collapse is detrimental.

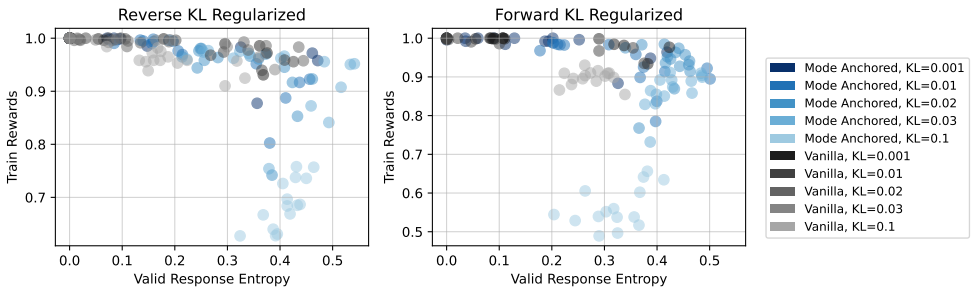
6.1 Verifiable 1-2 Task for LLM

We train an LM (`Qwen2.5 3B`) to generate uniform random integer that is either 1 or 2. It gets a reward of 1.0 for correct (producing "1" or "2" in XML), and 0.0 otherwise (details in Appendix C.2).

Most runs are able to optimize the reward well and get a reward of ~ 1 (Figure 7a, right). Figure 7a (left) shows the number of correctly formatted 1's the LM generates over the course of training. We see that for naive KL regularization (grey), across a range of β's and seeds, all but one run collapse into generating only a single answer as a result of RL, and most collapse into generating 1's, which has higher likelihood under the base policy. MARA (blue), on the other hand, is able to preserve the diversity in the correct answers, with many runs learning to generate 1's and 2's with near uniform probability, while still correctly learning to generate with the correct format (Figure 7a, middle). Further, the Pareto front of model checkpoints at different points in training shows that for both reverse and forward KL regularization, MARA is able to match vanilla training in terms of correctness, while exceeding vanilla training in terms of generation diversity.



(a) Valid answer entropy and rewards



(b) Pareto front of reward (quality) and entrpy (diversity)

Figure 7: Performance on verifiable task with multiple solutions, against both reverse & forward KL

6.2 Creative Question Answering for Chat LLM

We test MARA in a non-verifiable alignment task. We train `Qwen3-1.7B` on a subset of WildChat text (Zhao et al., 2024), using a parametric reward model (`Skywork-Reward-V2-Qwen3-4B`). We evaluate the model on a curated test set (Zhang et al., 2025) and report both the training reward (`In dist Reward`), and test set reward from a different reward model (`Out dist Reward`). We also report diversity metrics in terms of n-grams (`Ngrams`), semantic embeddings (`Semantic Div`), and "distinct functional classes" (`Mean Distinct`). Details in Appendix C.3. Here, MARA is used as a drop-in replacement in an RLOO style algorithm (Ahmadian et al., 2024). We observe that MARA out-performs both GRPO and RLOO in terms of out-of-distribution rewards, and all-but-one diversity metrics (Table 1).

6.3 Drug Discovery with Chemical Language Models

Finally, we apply MARA to a distinctively different domain where diversity and quality is crucial: drug discovery. Chemical language models (CLMs) have seen success in discovering molecules in clinical trials. We adpat two realistic reward functions from Guo et al. (2025b): `SYNTH` and `ALL-AMIDE` that jointly reward binding potency and synthesizability. The core CLM optimization problem is also a regularized RL problem: maximize reward, while staying close to a pretrained "prior" model to ensure chemical validity. Unlike the traditional RL setting, CLMs are evaluated based on their ability to generate *unique* molecules given a *fixed* number of reward function evaluations (which are expensive simulations and/or experiments), making diversity an essential quality for any performant CLMs. The `REINVENT` algorithm (Olivecrona et al., 2017; Guo & Schwaller, 2024b) is

Model	In-dist. Reward (↑)	Out-dist. Reward (↑)	Ngrams EAD (↑)	Semantic Div (↑)	Mean Distinct (↑)
Base Model	10.94	1.166 ±0.076	0.413 ±0.015	**0.220 ±0.009**	4.01 ±0.254
GRPO	14.80	1.317 ±0.102	0.497 ±0.014	0.193 ±0.009	3.96 ±0.249
RLOO	**15.56**	1.280 ±0.100	0.514 ±0.014	0.192 ±0.008	3.88 ±0.243
MARA (rev)	15.42	1.451 ±0.103	0.543 ±0.014	0.186 ±0.008	4.14 ±0.233
MARA (fwd)	15.33	**1.604 ±0.113**	**0.568 ±0.012**	0.193 ±0.009	**4.62 ±0.258**

Table 1: Performance on non-verifiable creative task. Mean ± bootstrap SEM.

a state-of-the-art RL-based method on standard benchmarks (Gao et al., 2022). We apply MARA as a *drop-in replacement* to its rewards. Evaluation details are in Appendix C.4.

Table 2 shows MARA consistently results in higher average `Yield` and lower `OB100`, indicating that it finds many *unique* high-reward molecules, and more efficiently with less reward function calls. Going further, we also assess "global" diversity (which MARA does not explicitly optimize for) in terms of `IntDiv1` and `#Circles`. Both define more macroscopic differences based on molecular sub-structures. We fine MARA is competitive with the baseline here. Overall, we see MARA further boosts REINVENT's optimization efficiency, while maintaining diversity.

Threshold	Algorithm	Yield (↑)	OB100 (↓)	IntDiv1 (↑)	Circles (↑)
0.80	REINVENT	6569 ± 186	1042 ± 66	0.766 ± 0.011	$\mathbf{67 \pm 3}$
	MARA	$\mathbf{6834 \pm 78}$	1015 ± 55	0.761 ± 0.009	59 ± 8
0.85	REINVENT	1614 ± 407	4114 ± 109	0.701 ± 0.018	7 ± 1
	MARA	1796 ± 210	$\mathbf{3654 \pm 272}$	0.716 ± 0.015	6 ± 1

(a) SYNTH task

Threshold	Algorithm	Yield (↑)	OB100 (↓)	IntDiv1 (↑)	Circles (↑)
0.80	REINVENT	5433 ± 184	1427 ± 63	0.768 ± 0.012	35 ± 1
	MARA	5635 ± 249	1407 ± 123	0.766 ± 0.008	36 ± 3
0.85	REINVENT	1098 ± 88	4360 ± 257	0.721 ± 0.016	8 ± 1
	MARA	$\mathbf{1235 \pm 130}$	$\mathbf{3943 \pm 303}$	0.733 ± 0.009	8 ± 1

(b) ALL-AMIDE task

Table 2: Results for different tasks and evaluation reward thresholds for two challenging drug discovery tasks. Error bars (±) denote standard deviation over 5 independent seeds. Bold indicates if the performance is statistically significantly better than the alternative method for that threshold (one-sided student's t-test, $p < 0.05$).

7 Conclusion

Over the past decade, the lesson of Artificial Intelligence has been that simple, flexible objectives combined with scale, are the recipe to continued progress. In this work, we provide an in-depth understanding of the KL-regularized RL objective, particularly in terms of its diversity. This opens up a number of exciting future directions: a deeper analysis into the properties of the forward KL regularized gradient, better gradients that optimizes the MARA objective, and even a wider class of solution distributions. All in all, we emphasize that regularized RL is inherently a distribution matching problem, and more thoughts should go into defining a *good* distribution to match to.

Reproducibility Statement

We use open-source, publicly available libraries for all experimental code. Didactic experiments are constructed in PyTorch (Paszke et al., 2019). Reinforcement learning on LLM training is done using the `nano-aha-moment` (Kazemnejad et al., 2025) and verl (`https://github.com/volcengine/verl`) github repos. Chemical language model experiments use the official `saturn`

github repo (Guo & Schwaller, 2024b). We provide detailed experimental information in Appendix C. Pseudo-code is provided in Algorithm 1 and Algorithm 2.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*, 2024.

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. *arXiv preprint arXiv:2402.14740*, 2024.

Amr Alhossary, Stephanus Daniel Handoko, Yuguang Mu, and Chee-Keong Kwoh. Fast, accurate, and reliable molecular docking with quickvina 2. *Bioinformatics*, 31(13):2214–2216, 2015.

Eser Aygün, Anastasiya Belyaeva, Gheorghe Comanici, Marc Coram, Hao Cui, Jake Garrison, Renee Johnston Anton Kast, Cory Y McLean, Peter Norgaard, Zahra Shamsi, et al. An ai system to help scientists write expert-level empirical software. *arXiv preprint arXiv:2509.06503*, 2025.

G Richard Bickerton, Gaia V Paolini, Jérémy Besnard, Sorel Muresan, and Andrew L Hopkins. Quantifying the chemical beauty of drugs. *Nature chemistry*, 4(2):90–98, 2012.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Esben Jannik Bjerrum. Smiles enumeration as data augmentation for neural network modeling of molecules. *arXiv preprint arXiv:1703.07076*, 2017.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Dean G Brown and Jonas Bostrom. Analysis of past and present synthetic methodologies on medicinal chemistry: where have all the new reactions gone? miniperspective. *Journal of medicinal chemistry*, 59(10):4443–4458, 2016.

Pablo Samuel Castro, Nenad Tomasev, Ankit Anand, Navodita Sharma, Rishika Mohanta, Aparna Dev, Kuba Perlin, Siddhant Jain, Kyle Levin, Noémi Éltető, et al. Discovering symbolic cognitive models from human and animal behavior. *bioRxiv*, pp. 2025–02, 2025.

Binghong Chen, Chengtao Li, Hanjun Dai, and Le Song. Retro*: learning retrosynthetic planning with neural guided a* search. In *International conference on machine learning*, pp. 1608–1616. PMLR, 2020.

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

John Joon Young Chung, Vishakh Padmakumar, Melissa Roemmele, Yuqian Sun, and Max Kreminski. Modifying large language model post-training for diverse creative writing. *arXiv preprint arXiv:2503.17126*, 2025.

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

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

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. *arXiv preprint arXiv:2304.06767*, 2023.

Yuanqi Du, Arian R Jamasb, Jeff Guo, Tianfan Fu, Charles Harris, Yingheng Wang, Chenru Duan, Pietro Liò, Philippe Schwaller, and Tom L Blundell. Machine learning-aided generative molecular design. *Nature Machine Intelligence*, pp. 1–16, 2024.

Wenhao Gao, Tianfan Fu, Jimeng Sun, and Connor Coley. Sample efficiency matters: a benchmark for practical molecular optimization. *Advances in neural information processing systems*, 35: 21342–21357, 2022.

Dongyoung Go, Tomasz Korbak, Germán Kruszewski, Jos Rozen, Nahyeon Ryu, and Marc Dymetman. Aligning language models with preferences through f-divergence minimization. *arXiv preprint arXiv:2302.08215*, 2023.

Jean-Bastien Grill, Florent Altché, Yunhao Tang, Thomas Hubert, Michal Valko, Ioannis Antonoglou, and Rémi Munos. Monte-carlo tree search as regularized policy optimization. In *International Conference on Machine Learning*, pp. 3769–3778. PMLR, 2020.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025a.

Jeff Guo and Philippe Schwaller. Augmented memory: Sample-efficient generative molecular design with reinforcement learning. *JACS Au*, 2024a.

Jeff Guo and Philippe Schwaller. Saturn: Sample-efficient generative molecular design using memory manipulation. *arXiv preprint arXiv:2405.17066*, 2024b.

Jeff Guo, Víctor Sabanza-Gil, Zlatko Jončev, Jeremy S Luterbacher, and Philippe Schwaller. Generative molecular design with steerable and granular synthesizability control. *arXiv preprint arXiv:2505.08774*, 2025b.

Anthony GX-Chen, Dongyan Lin, Mandana Samiei, Doina Precup, Blake Aaron Richards, Rob Fergus, and Kenneth Marino. Language agents mirror human causal reasoning biases. how can we help them think like scientists? In *Second Conference on Language Modeling*, 2025.

Andre He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting grpo beyond distribution sharpening. *arXiv preprint arXiv:2506.02355*, 2025.

Aspen K Hopkins, Alex Renda, and Michael Carbin. Can LLMs generate random numbers? evaluating LLM sampling in controlled domains. In *ICML 2023 Workshop: Sampling and Optimization in Discrete Space*, 2023. URL https://openreview.net/forum?id=Vhh1K9LjVI.

Edward J Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. Amortizing intractable inference in large language models. *arXiv preprint arXiv:2310.04363*, 2023.

Mete Ismayilzada, Antonio Laverghetta Jr, Simone A Luchini, Reet Patel, Antoine Bosselut, Lonneke van der Plas, and Roger Beaty. Creative preference optimization. *arXiv preprint arXiv:2505.14442*, 2025.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Amirhossein Kazemnejad, Milad Aghajohari, Alessandro Sordoni, Aaron Courville, and Siva Reddy. Nano aha! moment: Single file "rl for llm" library. https://github.com/McGill-NLP/nano-aha-moment, 2025. GitHub repository.

Peter W Kenny. Hydrogen-bond donors in drug design. *Journal of medicinal chemistry*, 65(21): 14261–14275, 2022.

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

Tomasz Korbak, Hady Elsahar, Germán Kruszewski, and Marc Dymetman. On reinforcement learning and distribution matching for fine-tuning language models with no catastrophic forgetting. *Advances in Neural Information Processing Systems*, 35:16203–16220, 2022.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

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

Tianjian Li, Yiming Zhang, Ping Yu, Swarnadeep Saha, Daniel Khashabi, Jason Weston, Jack Lanchantin, and Tianlu Wang. Jointly reinforcing diversity and quality in language model generations. *arXiv preprint arXiv:2509.02534*, 2025.

Siyang Liu, Sahand Sabour, Yinhe Zheng, Pei Ke, Xiaoyan Zhu, and Minlie Huang. Rethinking and refining the distinct metric. *arXiv preprint arXiv:2202.13587*, 2022.

Mark F Mabanglo, Keith S Wong, Marim M Barghash, Elisa Leung, Stephanie HW Chuang, Afshan Ardalan, Emily M Majaesic, Cassandra J Wong, Shen Zhang, Henk Lang, et al. Potent clpp agonists with anticancer properties bind with improved structural complementarity and alter the mitochondrial n-terminome. *Structure*, 31(2):185–200, 2023.

Krzysztof Maziarz, Austin Tripp, Guoqing Liu, Megan Stanley, Shufang Xie, Piotr Gaiński, Philipp Seidl, and Marwin HS Segler. Re-evaluating retrosynthesis algorithms with syntheseus. *Faraday Discussions*, 256:568–586, 2025.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9:1–14, 2017.

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

Sven Michael Papidocha, Andreas Burger, Varinia Bernales, and Alán Aspuru-Guzik. The elephant in the lab: Synthesizability in generative small-molecule design. *ChemRxiv*, 2025.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, et al. Molecular sets (moses): a benchmarking platform for molecular generation models. *Frontiers in pharmacology*, 11:565644, 2020.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. *Advances in Neural Information Processing Systems*, 36:53728–53741, December 2023. URL https://papers.nips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Mikołaj Sacha, Mikołaj Błaz, Piotr Byrski, Paweł Dabrowski-Tumanski, Mikołaj Chrominski, Rafał Loska, Paweł Włodarczyk-Pruszynski, and Stanisław Jastrzebski. Molecule edit graph attention network: modeling chemical reactions as sequences of graph edits. *Journal of Chemical Information and Modeling*, 61(7):3273–3284, 2021.

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

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

Megan Stanley and Marwin Segler. Fake it until you make it? generative de novo design and virtual screening of synthesizable molecules. *Current Opinion in Structural Biology*, 82:102658, 2023.

Anja Surina, Amin Mansouri, Lars Quaedvlieg, Amal Seddas, Maryna Viazovska, Emmanuel Abbe, and Caglar Gulcehre. Algorithm discovery with llms: Evolutionary search meets reinforcement learning. *arXiv preprint arXiv:2504.05108*, 2025.

Shidi Tang, Ji Ding, Xiangyu Zhu, Zheng Wang, Haitao Zhao, and Jiansheng Wu. Vina-gpu 2.1: towards further optimizing docking speed and precision of autodock vina and its derivatives. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2024.

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

Daniil Tiapkin, Nikita Morozov, Alexey Naumov, and Dmitry P Vetrov. Generative flow networks as entropy-regularized rl. In *International Conference on Artificial Intelligence and Statistics*, pp. 4213–4221. PMLR, 2024.

Oleg Trott and Arthur J Olson. Autodock vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

Chaoqi Wang, Yibo Jiang, Chenghao Yang, Han Liu, and Yuxin Chen. Beyond reverse kl: Generalizing direct preference optimization with diverse divergence constraints. *arXiv preprint arXiv:2309.16240*, 2023.

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

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Yutong Xie, Ziqiao Xu, Jiaqi Ma, and Qiaozhu Mei. How much space has been explored? measuring the chemical space covered by databases and machine-generated molecules. In *Proc. 11th International Conference on Learning Representations*, 2023.

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. *arXiv preprint arXiv:2504.11343*, 2025.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Yiming Zhang, Harshita Diddee, Susan Holm, Hanchen Liu, Xinyue Liu, Vinay Samuel, Barry Wang, and Daphne Ippolito. Noveltybench: Evaluating language models for humanlike diversity. *arXiv preprint arXiv:2504.05228*, 2025.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatGPT interaction logs in the wild. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bl8u7ZRlbM.

A Related Work

Training for diversity Wang et al. (2023) generalizes the DPO objective (Rafailov et al., 2023) from reverse-KL regularized to a more general class of f-divergence regularizers, with the key motivation being that reverse-KL can be mode-seeking, therefore reduce diversity. They do not explore the effect of reward function or regularization coefficient β, which our work examines. Diverse DPO Lanchantin et al. (2025) and variants (Chung et al., 2025; Ismayilzada et al., 2025) encourage diversity in preference learning by selecting diverse positives/negatives. Most closely related to our reward augmentation approach is He et al. (2025), which uses a rank based "unlikeliness reward" by ranking the in-batch samples based on their likelihood under the current policy, and penalize the most likely samples. Similarly related is Li et al. (2025), which use an external model to evaluate diversity (via a semantic classifier) and use the diversity metric to modify the reward. We do not require an external model to evaluate diversity.

More distantly, Dang et al. (2025) found that combining weights of earlier and later checkpoints can improve pass@k performance—a loose measure of diversity (albeit over both correct and incorrect answers). GFlowNets also provide diversity-seeking policies that sample proportionally to reward, albeit they use different algorithms than the KL-regularized policy gradient which is the most commonly used algorithm for LM post-training (Hu et al., 2023; Tiapkin et al., 2024).

Entropy and reasoning in RL We can view mode collapse in solutions as a collapse in the entropy of the *trajectory* distribution. This is related (but not identical) to token entropy. A growing line of empirical work do tie together entropy, exploration, and reasoning in LLMs. Cui et al. (2025) notes entropy collapses during RL. Cheng et al. (2025) incorporates an entropy term in the advantage to encourage better reasoning. Wang et al. (2025) show that focusing gradient updates on a minority of high-entropy tokens ("forking tokens") can improve reasoning.

B Mathematical Derivations

B.1 Target Distribution of Reverse-KL Reward Maximization

Proof of Remark 3.1 We want to find the distribution which maximizes the objective from equation 1,

$$\arg\max_{\pi_\theta} J_\beta(\pi_\theta) = \arg\max_{\pi_\theta} \mathbb{E}_{\pi_\theta(y)}[R(y)] - \beta\, D_{KL}\big(\pi_\theta || \pi_{\text{ref}}\big) \tag{12}$$

We can re-write Equation 1 by re-arranging terms, note for notation brevity we denote $g_\beta(y) = \pi_{\text{ref}}(y) \exp\big(\frac{R(y)}{\beta}\big)$,

$$\begin{aligned}
J_\beta(\pi_\theta) &= \mathbb{E}_{\pi_\theta(y)}[R(y)] - \beta\, D_{KL}\big(\pi_\theta || \pi_{\text{ref}}\big)\,, && (13)\\
&= \mathbb{E}_{\pi_\theta(y)}\Big[R(y) - \beta\big(\log \pi_\theta(y) - \log \pi_{\text{ref}}(y)\big)\Big]\,, && (14)\\
&= -\beta\, \mathbb{E}_{\pi_\theta(y)}\Big[\log \pi_\theta(y) - \big(\frac{R(y)}{\beta} + \log \pi_{\text{ref}}(y)\big)\Big]\,, && (15)\\
&= -\beta\, \mathbb{E}_{\pi_\theta(y)}\Big[\log \pi_\theta(y) - \log \pi_{\text{ref}}(y) \exp\big(\frac{R(y)}{\beta}\big)\Big]\,, && (16)\\
&= -\beta\, \mathbb{E}_{\pi_\theta(y)}\Big[\log \pi_\theta(y) - \log g_\beta(y) + \log \zeta - \log \zeta\Big]\,, && (17)\\
&= -\beta\, \mathbb{E}_{\pi_\theta(y)}\Big[\log \pi_\theta(y) - \log G_\beta(y)\Big] + \beta \log \zeta\,, && (18)\\
&= -\beta D_{KL}\Big(\pi_\theta || G_\beta\Big) + \beta \log \zeta\,. && (19)
\end{aligned}$$

It is easy to see that the above is maximized when $D_{KL}\big(\pi_\theta || G_\beta\big) = 0$, which is when the policy is the target distribution, $\pi_\theta = G_\beta$.

B.2 Gradient of Reverse-KL Reward Maximization

Proof of Remark 3.2 From Appendix B.1, we have the identity,

$$-\frac{1}{\beta}J_\beta(\pi_\theta) = D_{KL}\big(\pi_\theta||G_\beta\big) - \log\zeta\,. \tag{20}$$

We can easily show that the gradient is,

$$\nabla_\theta\Big(-\frac{1}{\beta}J_\beta(\pi_\theta)\Big) = \nabla_\theta\, D_{KL}\big(\pi_\theta||G_\beta\big) - \nabla_\theta\log\zeta\,, \tag{21}$$

$$= \nabla_\theta\, D_{KL}\big(\pi_\theta||G_\beta\big)\,. \tag{22}$$

In other words, they are the same up to constant $-\beta$,

$$\nabla_\theta J_\beta(\pi_\theta) = -\beta\,\nabla_\theta D_{KL}\big(\pi_\theta||G_\beta\big)\,. \tag{23}$$

B.3 Target Distribution of Forward-KL Reward Maximization

We are interested in finding the distribution $\pi_\theta = G_{\text{fwd}}$ which maximizes the following,

$$J_{\text{fwd}}(\pi_\theta) = \mathbb{E}_{\pi_\theta(y)}[R(y)] - \beta\, D_{KL}\big(\pi_{\text{ref}}||\pi_\theta\big)\,. \tag{24}$$

We first simplify the expression to include only terms that depend on π_θ,

$$\arg\max_{\pi_\theta} J_{\text{fwd}}(\pi_\theta) = \arg\max_{\pi_\theta}\mathbb{E}_{\pi_\theta}\big[R(y)\big] - \beta\, D_{KL}\big(\pi_{\text{ref}}||\pi_\theta\big)\,, \tag{25}$$

$$= \arg\max_{\pi_\theta}\mathbb{E}_{\pi_\theta}\Big[R(y) - \beta\frac{\pi_{\text{ref}}(y)}{\pi_\theta(y)}\log\frac{\pi_{\text{ref}}(y)}{\pi_\theta(y)}\Big]\,, \tag{26}$$

$$= \arg\max_{\pi_\theta}\int \pi_\theta(y)\,R(y) - \beta\,\pi_{\text{ref}}(y)\Big[\log\pi_{\text{ref}}(y) - \log\pi_\theta(y)\Big]\,dy\,, \tag{27}$$

$$= \arg\max_{\pi_\theta}\int \pi_\theta(y)\,R(y) + \beta\,\pi_{\text{ref}}(y)\log\pi_\theta(y)\,dy\,. \tag{28}$$

Define the maximization objective subject to constraint $\int\pi(y)\,dy = 1$ as a Lagrangian,

$$\mathcal{L}_J[\pi;\lambda] = \int \pi(y)R(y) + \beta\pi_{\text{ref}}(y)\log\pi(y)\,dy + \lambda\,\big(\int\pi(y)\,dy - 1\big)\,, \tag{29}$$

$$= \int \pi(y)R(y) + \lambda\pi(y) + \beta\pi_{\text{ref}}(y)\log\pi(y)\,dy - \lambda\,. \tag{30}$$

We want to take the Gateaux derivative,

$$d\,\mathcal{L}_J[\pi;\lambda] = \frac{d}{d\varepsilon}\mathcal{L}_J[\pi+\varepsilon\varphi;\lambda]\Big|_{\varepsilon=0}\,, \tag{31}$$

First solve,

$$\frac{d}{d\varepsilon}\mathcal{L}_J[\pi+\varepsilon\varphi;\lambda] = \frac{d}{d\varepsilon}\int \big(\pi(y)+\varepsilon\varphi(y)\big)R(y) + \lambda\big(\pi(y)+\varepsilon\varphi(y)\big) + \beta\,\pi_{\text{ref}}(y)\log\big(\pi(y)+\varepsilon\varphi(y)\big)\,dy\,, \tag{32}$$

$$= \int \varphi(y)R(y) + \lambda\varphi(y) + \beta\frac{\pi_{\text{ref}}(y)\,\varphi(y)}{\pi(y)+\varepsilon\varphi(y)}\,dy\,, \tag{33}$$

$$= \int \varphi(y)\Big[R(y) + \lambda + \beta\frac{\pi_{\text{ref}}(y)}{\pi(y)+\varepsilon\varphi(y)}\Big]dy\,. \tag{34}$$

$$\frac{d}{d\varepsilon}\mathcal{L}_J[\pi+\varepsilon\varphi;\lambda]\Big|_{\varepsilon=0} = \int\varphi(y)\Big[R(y)+\lambda+\beta\frac{\pi_{\text{ref}}(y)}{\pi(y)}\Big]dy\,. \tag{35}$$

The functional derivative is therefore,

$$\frac{\delta}{\delta\pi}\mathcal{L}_J[\pi;\lambda] = R(y) + \lambda + \beta\frac{\pi_{\text{ref}}(y)}{\pi(y)} \tag{36}$$

To find the opimum π^* which gives ${}^{d}/_{d\varepsilon}\,\mathcal{L}_J[\pi+\varepsilon\varphi;\lambda]=0$ for all φ, the fundamental lemma of the calculus of variations tells us it would imply ${}^{\delta}/_{\delta\pi}\,\mathcal{L}_J[\pi;\lambda]=0$. Solving for this,

$$R(y)+\lambda+\beta\frac{\pi_{\text{ref}}(y)}{\pi^*(y)}=0\,, \tag{37}$$

$$\Rightarrow \pi^*(y)=\frac{\beta\pi_{\text{ref}}(y)}{-(R(y)+\lambda)}\,, \tag{38}$$

$$\Rightarrow \pi^*(y)=\frac{\beta\pi_{\text{ref}}(y)}{\Lambda-R(y)}\,, \qquad \text{define } \Lambda=-\lambda \text{ for notational convenience}\,. \tag{39}$$

where a unique value of Λ satisfies $\int \pi^*(y)\,dy=1$ and $\pi^*(y)\geq 0$ for all y.

Note Grill et al. (2020), Appendix B.3 arrives at a similar solution for the setting of discrete action spaces (i.e. π_θ can be parameterized by a vector).

B.4 Gradient of Forward-KL Regularized Reward Maximization

Proof of Remark 3.4 We want to know if optimizing the forward-KL *regularized* RL objective is equivalent to optimizing a forward KL divergence. In other words, we are interested in whether the following gradient,

$$\nabla_\theta\, J_{\text{fwd}}(\pi_\theta)=\nabla_\theta\Big[\mathbb{E}_{\pi_\theta(y)}[R(y)]-\beta\, D_{KL}\big(\pi_{\text{ref}}||\pi_\theta\big)\Big]\,, \tag{40}$$

is a gradient of a forward KL between π_θ and *some* target distribution h that is independent of π_θ. Suppose h exists, it follows that the functional derivative of these two objectives must be equivalent up to proportionality,

$$\frac{\delta}{\delta\pi}J_{\text{fwd}}(\pi)\propto\frac{\delta}{\delta\pi}D_{KL}\big(h||\pi\big)\,, \tag{41}$$

where both are subject to constraint $\int\pi(y)\,dy=1$.

We have established from Equation 36 that the functional derivative of J_{fwd} subject to constraint $\int\pi(y)\,dy=1$ is,

$$\frac{\delta}{\delta\pi}\mathcal{L}_J[\pi;\lambda]=R(y)+\beta\frac{\pi_{\text{ref}}(y)}{\pi(y)}+\lambda\,. \tag{42}$$

To find the functional derivative of the forward-KL, we first write down the forward KL objective subject to constraint,

$$\mathcal{L}_K[\pi,\lambda']=D_{KL}\big(h||\pi\big)+\lambda'\Big(\int\pi(y)\,dy-1\Big)\,, \tag{43}$$

$$=\int h(y)\log h(y)-h(y)\log\pi(y)\,dy+\int\lambda'\pi(y)\,dy-\lambda'\,, \tag{44}$$

$$=\int\lambda'\,\pi(y)-h(y)\log\pi(y)\,dy+\left[\int h(y)\log h(y)\,dy-\lambda'\right], \tag{45}$$

where the right-hand bracket is independent of π. The Gateaux derivative is,

$$\frac{d}{d\varepsilon}\mathcal{L}_K[\pi+\varepsilon\varphi,\lambda']=\frac{d}{d\varepsilon}\int\lambda'\big(\pi(y)+\varepsilon\varphi(y)\big)-h(y)\log\big(\pi(y)+\varepsilon\varphi(y)\big)\,dy\,, \tag{46}$$

$$=\int\lambda'\varphi(y)-\frac{h(y)\varphi(y)}{\pi(y)+\varepsilon\varphi(y)}\,dy\,. \tag{47}$$

$$\frac{d}{d\varepsilon}\mathcal{L}_K[\pi+\varepsilon\varphi,\lambda']\bigg|_{\varepsilon=0}=\int\varphi(y)\Big[\lambda'-\frac{h(y)}{\pi(y)}\Big]\,dy \tag{48}$$

The functional derivative of the forward KL with respect to the right-hand term is therefore,

$$\frac{\delta}{\delta\pi}\mathcal{L}_K[\pi,\lambda']=\lambda'-\frac{h(y)}{\pi(y)}\,. \tag{49}$$

Assuming the functional derivative of the two objectives are proportional to each other, we can solve for the target distribution $h(y)$,

$$\frac{\delta}{\delta\pi}\mathcal{L}_K[\pi,\lambda'] \propto \frac{\delta}{\delta\pi}\mathcal{L}_J[\pi,\lambda]\,, \tag{50}$$

$$\Rightarrow \lambda' - \frac{h(y)}{\pi(y)} = \alpha\left[R(y) + \beta\frac{\pi_{\text{ref}}(y)}{\pi(y)} + \lambda\right], \qquad \text{for some constant } \alpha\,, \tag{51}$$

$$\Rightarrow h(y) = \Big(\lambda' - \alpha\lambda - \alpha R(y)\Big)\pi(y) - \alpha\beta\pi_{\text{ref}}(y)\,. \tag{52}$$

Observe one cannot write $h(y)$ independently of $\pi(y)$, other than in trivial cases (e.g. if $R(y)$ is constant such that $\lambda' - \alpha\lambda - \alpha R(y) = 0$). Thus, for general reward functions R, optimizing the forward-KL does not produce a forward KL gradient toward any distribution that can be expressed independently of π_θ.

B.5 Gradient of the forward KL

Remark B.1. *The gradient of the forward KL divergence between policy π_θ and target G_β is,*

$$\nabla_\theta\, D_{KL}\big(G_\beta||\pi_\theta\big) = -\mathbb{E}_{G_\beta}\big[\nabla_\theta \log \pi_\theta(y)\big]\,. \tag{53}$$

Proof.

$$\nabla_\theta\, D_{KL}\big(G_\beta||\pi_\theta\big) = \nabla_\theta \mathbb{E}_{G_\beta}\big[\log G_\beta(y) - \log \pi_\theta(y)\big]\,, \tag{54}$$

$$= \mathbb{E}_{G_\beta}\big[\nabla_\theta\big(\log G_\beta(y) - \log \pi_\theta(y)\big)\big]\,, \tag{55}$$

$$= -\mathbb{E}_{G_\beta}\big[\nabla_\theta \log \pi_\theta(y)\big]\,. \tag{56}$$

□

We see that optimizing the forward KL gradient amounts to doing maximum likelihood / supervised fine-tuning on trajectories sampled from the target distribution G_β, as is also mentioned in some previous works (Agarwal et al., 2024). This is generally intractable as it requires sampling from G_β. Nevertheless, estimating expectation under a distribution known only up to normalization (i.e. $\mathbb{E}_{G_\beta}[\cdot]$) is well-studied in Monte-Carlo methods (Robert et al., 1999), and it is conceivable that a number of methods there would prove helpful here.

B.6 Probability Ratio Under Optimal Target Distribution

Proof of Proposition 4.1 For any two samples, y_1 and y_2, their probability ratio under the target distribution is given by,

$$\frac{G_\beta(y_1)}{G_\beta(y_2)} = \frac{g_\beta(y_1)}{\zeta}\frac{\zeta}{g_\beta(y_2)} = \frac{g_\beta(y_1)}{g_\beta(y_2)}\,, \tag{57}$$

which only require the unnormalized likelihood as the normalization constant ζ cancel out. Expanding the terms, we can write the log likeilhood ratio in closed form,

$$\log\frac{G_\beta(y_1)}{G_\beta(y_2)} = \log \pi_{\text{ref}}(y_1)\exp\big(\frac{R(y_1)}{\beta}\big) - \log \pi_{\text{ref}}(y_2)\exp\big(\frac{R(y_2)}{\beta}\big)\,, \tag{58}$$

$$= \log\frac{\pi_{\text{ref}}(y_1)}{\pi_{\text{ref}}(y_2)} + \frac{1}{\beta}\Big(R(y_1) - R(y_2)\Big)\,. \tag{59}$$

B.7 Solution distribution after reward augmentation

Remark B.2. *Optimizing the reverse-KL regularized RL objective with the augmented reward function $\bar{R}$ yields the following solution distribution, which puts uniformly high mass over all samples above reward threshold $R(y) \geq \tau$,*

$$\bar{G}_\beta(y) \propto \begin{cases} \pi_{ref}(y)\exp\left(\frac{R(y)}{\beta}\right) & \text{if } R(y) < \tau, \\ \pi_{ref}(z)\exp\left(\frac{R(z)}{\beta}\right) & \text{if } R(y) \geq \tau. \end{cases} \tag{60}$$

Algorithm 1 Mode Anchored Reward Augmentation (MARA) within a sampled batch

Given: initial policy π_θ, reference distribution π_{ref}, reward function R, regularization coefficient β, threshold of good answers $\tau \in \mathbb{R}$, $\tau \leq \max_y R(y)$, and trajectory batch $\{y_i\}_{i=1}^N \sim \pi_\theta$.
Pick anchor trajectory: $z = \arg\max_{y_i} \pi_{\text{ref}}(y_i)$, s.t. $R(y_i) \geq \tau$
for each y_i in batch **do**
  if $R(y_i) \geq \tau$ **then**
    Augment: $\bar{r}_i = R(z) + \beta\big(\log \pi_{\text{ref}}(z) - \log \pi_{\text{ref}}(y_i)\big)$
  else
    Keep same: $\bar{r}_i = R(y_i)$
  end if
end for
Optimize policy parameters θ using augmented rewards $\{\bar{r}_i\}_{i=1}^N$.

Proof. We have established already in Appendix B.1 that the solution distribution of reward maximization with reverse KL regularization is,

$$G_\beta(y) \propto \pi_{\text{ref}}(y) \exp\big(\frac{R(y)}{\beta}\big)\,. \tag{61}$$

Plug in the augmented reward function,

$$\bar{R}(y) = \begin{cases} R(y) & \text{if } R(y) < \tau, \\ R(z) + \beta\big(\log \pi_{\text{ref}}(z) - \log \pi_{\text{ref}}(y)\big) & \text{if } R(y) \geq \tau, \end{cases} \tag{62}$$

which gives us the augmented solution distribution,

$$\bar{G}_\beta(y) \propto \pi_{\text{ref}}(y) \exp\big(\frac{\bar{R}(y)}{\beta}\big)\,. \tag{63}$$

In the $R(y) < \tau$ case, $\bar{R}(y) = R(y)$, and there is no change to the (unnormalized) likelihood. In the $R(y) \geq \tau$ case,

$$\log \pi_{\text{ref}}(y) \exp\big(\frac{\bar{R}(y)}{\beta}\big) = \log \pi_{\text{ref}}(y) + \frac{1}{\beta}\bar{R}(y)\,, \tag{64}$$

$$= \log \pi_{\text{ref}}(y) + \frac{1}{\beta}\Big(R(z) + \beta\big(\log \pi_{\text{ref}}(z) - \log \pi_{\text{ref}}(y)\big)\Big)\,, \tag{65}$$

$$= \frac{R(z)}{\beta} + \log \pi_{\text{ref}}(y) + \log \pi_{\text{ref}}(z) - \log \pi_{\text{ref}}(y) \tag{66}$$

$$= \frac{R(z)}{\beta} + \log \pi_{\text{ref}}(z)\,. \tag{67}$$

Therefore we see in the $R(y) \geq \tau$ case we have,

$$\pi_{\text{ref}}(y) \exp\big(\frac{\bar{R}(y)}{\beta}\big) = \pi_{\text{ref}}(z) \exp\big(\frac{R(z)}{\beta}\big)\,. \tag{68}$$

□

This formally shows the target will have uniformly high density proportional to $\pi_{\text{ref}}(z) \exp(R(z)/\beta)$ for all samples if their original reward $R(y)$ is above threshold τ. If we pick z to be likely under π_{ref}, e.g. $z = \arg\max_y \pi_{\text{ref}}(y)$, we can also show these samples will have the highest probabilities in the solution distribution.

B.8 Gradient of reward-augmented optimization

We also note the MARA gradient estimator for an "above threshold" sample y_i (i.e. $R(y_i) \geq \tau$), when using reverse-KL regularization, can be equivalently constructed as using the anchor sample's reference policy probability $\pi_{\text{ref}}(z)$ in lieu of the actual reference probability $\pi_{\text{ref}}(y_i)$ when

constructing the KL gradient estimator. To see this precisely, we know the gradient of the expected reward to be,

$$\nabla_\theta \, \mathbb{E}_{\pi_\theta}\big[R(y)\big] = \mathbb{E}_{\pi_\theta}\big[R(y)\, \nabla_\theta \log \pi_\theta(y)\big] \,, \tag{69}$$

and gradient of the reverse-KL regularizer to be,

$$\nabla_\theta \, D_{KL}\big(\pi_\theta || \pi_{\text{ref}}\big) = \mathbb{E}_{\pi_\theta}\big[(\log \pi_\theta(y) - \log \pi_{\text{ref}}(y)) \nabla_\theta \log \pi_\theta(y)\big] \,. \tag{70}$$

Denote the reward-augmented objective as $\bar{J}_\beta(\pi_\theta) = \bar{R}(y) - \beta D_{KL}(\pi_\theta || \pi_{\text{ref}})$, where $\bar{R}(y) = R(z) + \beta\big(\log \pi_{\text{ref}}(z) - \log \pi_{\text{ref}}(y)\big)$ and z is the "anchor". The gradient estimator of $\bar{K}_i$ for an "above threshold" sample, y_i, $R(y_i) \geq \tau$, can be written as,

$$\bar{K}_i = \Big(\bar{R}(y_i) - \beta \log \frac{\pi_\theta(y_i)}{\pi_{\text{ref}}(y_i)}\Big) \nabla_\theta \log \pi_\theta(y_i) \,, \tag{71}$$

$$= \Big(R(z) + \beta \log \frac{\pi_{\text{ref}}(z)}{\pi_{\text{ref}}(y_i)} - \beta \log \frac{\pi_\theta(y_i)}{\pi_{\text{ref}}(y_i)}\Big) \nabla_\theta \log \pi_\theta(y_i) \,, \tag{72}$$

$$= \Big(R(z) - \beta \log \frac{\pi_\theta(y_i)}{\pi_{\text{ref}}(z)}\Big) \nabla_\theta \log \pi_\theta(y_i) \,. \tag{73}$$

Intuitively, as the anchor is chosen to have high π_{ref}, i.e. $\pi_{\text{ref}}(z) > \pi_{\text{ref}}(y_i)$, this can be interpreted as selectively reducing the KL regularization for high-rewarding samples. Mechanistically, this also suggest an alternative implementation which produces the same gradient when using reverse-KL regularization (Algorithm 2).

Algorithm 2 Mode Anchored Reward Augmentation, alternative implementation

1: Given: initial policy π_θ, reference distribution π_{ref}, reward function R, regularization coefficient β, threshold of good answers $\tau \in \mathbb{R}$, $\tau \leq \max_y R(y)$, and trajectory batch $\{y_i\}_{i=1}^N \sim \pi_\theta$.
2: Pick anchor trajectory: $z = \arg\max_{y_i} \pi_{\text{ref}}(y_i)$, s.t. $R(y_i) \geq \tau$
3: **for** each y_i in batch **do**
4: **if** $R(y_i) \geq \tau$ **then**
5: Augment: reward $\bar{r}_i = R(z)$, reference prob $\bar{p}_i = \pi_{\text{ref}}(z)$
6: **else**
7: Keep same: reward $\bar{r}_i = R(y_i)$, reference prob $\bar{p}_i = \pi_{\text{ref}}(y_i)$
8: **end if**
9: **end for**
10: Optimize policy parameters θ using augmented rewards $\{\bar{r}_i\}_{i=1}^N$ and augmented reference policy probabilities $\{\bar{p}_i\}_{i=1}^N$

C Additional Experimental Details

C.1 Didactic Experiments

We construct our didactic experiment as a vector of size 100 (akin to a "token space" with 100 tokens). We initialize a categorical distribution over this token space whose logits are all 0's (i.e. uniform distribution over all tokens). Given some reward function and reference distribution defined over this space, we optimize this categorical distribution with the KL-regularized policy gradient for 3000 gradient steps in PyTorch with Adam optimizer, with learning rate 5e-3 and batch size 32.

C.2 The 1-2 Task

We ask the LM to generate a uniform random integer that is either 1 or 2 (Hopkins et al., 2023), as illustrated in Figure 8. We run for a range of KL coefficients (β) and multiple random seeds. Figure 9 shows the training run for just vanilla RL, without MARA.

C.3 Creative Question Answering Task

We detail the training settings in Table 3, and evaluation settings in Table 4. We follow the evaluation procedures outlined in both Kirk et al. (2023) and Zhang et al. (2025). The specific evaluation metrics are defined as follows.

Prompt

```
Uniformly randomly generate
an integer that is either 1 or
2.  Respond strictly in this
format:  <think>Your internal
reasoning</think><answer>1 or
2</answer>
```

Example Generation

```
Let me decide
randomly.
<think></think><answer>1
</answer><|endoftext|>
```

Figure 8: The 1-2 task to test output distribution of LMs.

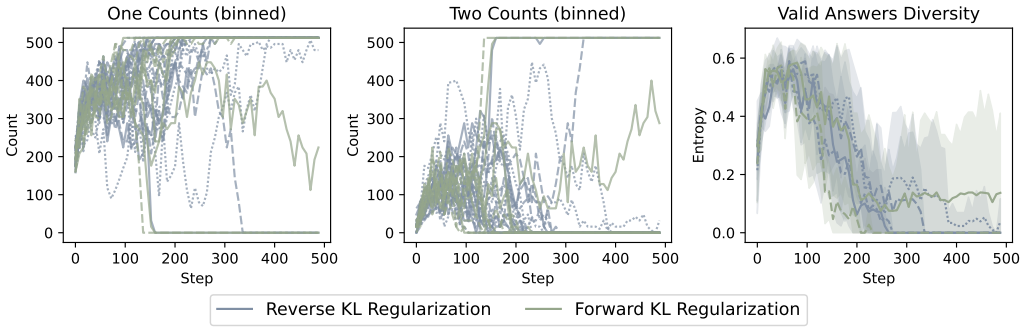


Figure 9: Training outcomes using vanilla RL. **(Left, Middle)** Policy's empirical distribution over valid answers for runs that reached high rewards (counts binned over 8 consecutive training batches), across a range of regularization coefficients (β). **Right** Diversity of the valid answers over the course of training, measured as the entropy of the Bernoulli distribution over answers of 1's and 2's.

- `In Dist Reward`: training reward, on training set, using training reward model
- `Out Dist Reward`: evaluation reward on held-out set, using evaluation reward model
- `Ngram EAD`: Expectation-adjusted Distinct N-gram, proposed in Liu et al. (2022). We follow (Kirk et al., 2023) and average EAD for $n = 1, ..., 5$
- `Semantic Div`: semantic embedding diversity as measured by averaged cosine distance, using embedding model `all-MiniLM-L6-v2`.
- `Mean Distinct`: Estimates a notion of "# of distinct concepts", as introduced in Zhang et al. (2025).

C.4 Drug Discovery

Chemical language models (CLMs) that generate molecules in string-based formats, e.g., a SMILES string (Weininger, 1988), have been experimentally validated with numerous generated molecules in clinical trials (Du et al., 2024). Recently, the field has focused on addressing "synthesizability", i.e., can generated molecules actually be synthesized in the lab? (Stanley & Segler, 2023; Papidocha et al., 2025). Accordingly, we adapt two reward functions from Guo et al. (2025b): `SYNTH` and `SYNTH-ALL-AMIDE` that jointly reward binding potency and synthesizability. `REINVENT` (Olivecrona et al., 2017) is a state-of-the-art RL-based CLM on standard benchmarks (Gao et al., 2022). The recent Saturn CLM (Guo & Schwaller, 2024b) notably improves optimization efficiency by using data augmentation (Bjerrum, 2017; Guo & Schwaller, 2024a), but continues to use `REINVENT`'s RL algorithm.

In the drug discovery experiments adapted from Guo et al. (2025b), the reward functions are comprised of numerous individual optimization objectives, and defines a multi-parameter optimization task. Concretely, these objectives are:

1. *Minimize* the molecular docking score using QuickVina2-GPU (Trott & Olson, 2010; Alhossary et al., 2015; Tang et al., 2024). Docking simulates binding of molecules to a target

Hyperparameter	Value
Actor Model	`Qwen3-1.7B`
Reward Model	`Skywork-Reward-V2-Qwen3-4B`
Training Dataset	`Wildchat 10k English`
Train Batch Size	128
Mini-Batch Size	64
Max Prompt Length	512
Max Response Length	2048
Learning Rate	1×10^{-6}
Entropy Coefficient	0
Rollout n (per prompt)	5
Gradient Checkpointing	Enabled
Epochs	3

Table 3: Creative QA Training Setting

Hyperparameter	Value
Evaluation Reward Model	`Skywork-Reward-Gemma-2-27B-v0.2`
Dataset	`NoveltyBench curated`
Num Generations / Prompt	10
Max Tokens	4000
Temperature	1.0
Enable Thinking (qwen)	`False`

Table 4: Creative QA Evaluation Setting

protein and predicts a crude binding affinity value. Docking was performed against the ATP-dependent Clp protease proteolytic subunit (ClpP) Mabanglo et al. (2023).

2. *Maximize* the quantitative estimate of drug-likeness (QED) (Bickerton et al., 2012), which is itself comprised of various physico-chemical properties, e.g., molecular weight. Maximizing QED can prevent generated molecules from being too large and lipophilic.
3. *Constrain* the number of hydrogen-bond donors (HBDs): HBDs < 4. This can improve absorption, Distribution, metabolism, and excretion (ADME) properties (Kenny, 2022) of the generated molecules.
4. *Satisfy* the "Synthesizability" constraint. Synthesizability is quantified by using a retrosynthesis model on each generated molecule. Retrosynthesis models predict a plausible synthesis route to synthesize a target molecule using commercially available precursors. The precursors set is from the eMolecules catalogue extracted from Chen et al. (2020). Retrosynthesis models typically start with a "single-step" model which predicts precursors given a target molecule. Since molecules may require multiple steps to synthesize, "Multi-step Retrosynthesis" commonly couples a search algorithm with single-step models to iteratively decompose a target molecule. In this work, we use the MEGAN (Sacha et al., 2021) single-step model with the Retro* (Chen et al., 2020) search algorithm using the Syntheseus (Maziarz et al., 2025) package. Finally, a molecule is considered synthesizable if the retrosynthesis model successfully proposes a synthesis route.

Both the `SYNTH` and `SYNTH-ALL-AMIDE` reward functions are comprised of the above objectives. The only difference is that in the `SYNTH-ALL-AMIDE` case, a molecule is *only* considered synthesizable if all the chemical reactions involved to synthesize it are "amide coupling reactions". Amide couplings are one of the most common reactions performed in the pharmaceutical industry (Brown & Bostrom, 2016), and is generally a robust, widely compatible reaction. Subsequently, the reward function is defined as a product of each individual component above. Given a molecule, x:

$$R(x) = DS(x) \times QED(x) \times HBD(x) \times Syntheseus(x) \in [0, 1] \tag{74}$$

Table 5: Results at Threshold = 0.8 (↑ larger is better; ↓ smaller is better). "SYNTH" and "AMIDE" denote the SYNTH and SYNTH-ALL-AMIDE reward functions, respectively.

Task	Algorithm	Sigma	Gen Yield (↑)	OB100 (↓)	IntDiv1 (↑)	Circles (↑)
SYNTH	REINVENT	128	6569 ± 186	1042 ± 66	0.766 ± 0.011	67 ± 3
		256	6618 ± 93	1080 ± 89	0.756 ± 0.012	57 ± 8
		512	6746 ± 161	1067 ± 74	0.752 ± 0.016	55 ± 5
	MARA	128	6834 ± 78	1015 ± 55	0.761 ± 0.009	59 ± 8
		256	6750 ± 139	1068 ± 50	0.760 ± 0.012	60 ± 4
		512	6793 ± 267	1065 ± 49	0.751 ± 0.015	60 ± 1
AMIDE	REINVENT	128	5433 ± 184	1427 ± 63	0.768 ± 0.012	35 ± 1
		256	5544 ± 172	1406 ± 59	0.768 ± 0.009	34 ± 5
		512	5334 ± 165	1445 ± 111	0.776 ± 0.008	33 ± 4
	MARA	128	5635 ± 249	1407 ± 123	0.766 ± 0.008	36 ± 3
		256	5353 ± 114	1393 ± 42	0.769 ± 0.009	33 ± 4
		512	5377 ± 152	1343 ± 77	0.763 ± 0.008	31 ± 3

where "DS" is docking score. The HBD and Syntheseus objectives are binary, i.e., 1 if satisfied and 0 otherwise. QED $\in [0, 1]$ and is used as is. The QuickVina2-GPU docking score is reward shaped using a reverse sigmoid function following Guo et al. (2025b) and gives higher reward to lower docking scores, as desired.

Our goal in this section is to investigate the potential for MARA to be a *drop-in replacement* for the REINVENT (Olivecrona et al., 2017) RL-based algorithm for molecular design. REINVENT is amongst the most performant molecular design algorithms (Gao et al., 2022) and the Saturn model (Guo & Schwaller, 2024b) adapts this algorithm and leverages data augmentation (Bjerrum, 2017; Guo & Schwaller, 2024a) to further improve optimization efficiency.

We evaluate all models with a fixed budget of 10,000 reward function evaluations, which is standard in benchmarks. We contrast the algorithms' performance on molecular design metrics that measure optimization efficiency and diversity. Yield is the number of *unique* molecules above a reward threshold. OB100 is the number of reward evaluations required to generate 100 molecules above the same threshold. IntDiv1 (Polykovskiy et al., 2020) and #Circles (Xie et al., 2023) are diversity metrics based on molecular sub-structure based features, and measure intra-set similarity and sphere packing, respectively.

Tables 5 and 6 show the optimization results for the SYNTH and SYNTH-ALL-AMIDE reward functions at the 0.80 and 0.85 reward thresholds, respectively. In general, MARA matches or outperforms REINVENT particularly for the more challenging SYNTH-ALL-AMIDE reward function. In this environment, MARA can find more high reward molecules (Yield) and using less reward evaluations (OB100) than REINVENT.

Table 6: Results at Threshold $= 0.85$ ($\uparrow$ larger is better; $\downarrow$ smaller is better). "SYNTH" and "AMIDE" denote the `SYNTH` and `SYNTH-ALL-AMIDE` reward functions, respectively.

Task	Algorithm	Sigma	Gen Yield ($\uparrow$)	OB100 ($\downarrow$)	IntDiv1 ($\uparrow$)	Circles ($\uparrow$)
SYNTH	REINVENT	128	1614 ± 407	4114 ± 109	0.701 ± 0.018	7 ± 1
		256	1552 ± 242	3940 ± 371	0.699 ± 0.030	6 ± 1
		512	1484 ± 45	3717 ± 201	0.701 ± 0.026	6 ± 1
	MARA	128	1796 ± 210	3654 ± 272	0.716 ± 0.015	6 ± 1
		256	1530 ± 126	3957 ± 335	0.705 ± 0.014	8 ± 1
		512	1550 ± 347	4016 ± 234	0.689 ± 0.024	6 ± 1
AMIDE	REINVENT	128	1098 ± 88	4360 ± 257	0.721 ± 0.016	8 ± 1
		256	1488 ± 280	4290 ± 141	0.725 ± 0.021	8 ± 1
		512	1054 ± 152	4620 ± 438	0.739 ± 0.009	8 ± 0
	MARA	128	1235 ± 130	3943 ± 303	0.733 ± 0.009	8 ± 1
		256	1404 ± 261	4079 ± 172	0.730 ± 0.010	7 ± 1
		512	1341 ± 86	3930 ± 400	0.723 ± 0.004	7 ± 1