FROM CURIOSITY TO CAUTION: MITIGATING REWARD HACKING FOR BEST-OF-N WITH PESSIMISM

Anonymous authors
Paper under double-blind review

ABSTRACT

Inference-time compute scaling has emerged as a powerful paradigm for improving language model performance on a wide range of tasks, but the question of how best to use the additional compute remains open. A popular approach is *Best-of-N* (BoN) sampling, where N candidate responses are generated, scored according to a reward model, and the highest-scoring response is selected. While this approach can improve performance, it is vulnerable to *reward hacking*, where performance degrades as N increases due to the selection of responses that exploit imperfections in the reward model instead of genuinely improving generation quality. Prior attempts to mitigate reward hacking—via stronger reward models or heavy-handed distributional regularization—either fail to fully address over-optimization or are too conservative to exploit additional compute. In this work, we explore the principle of *pessimism* in reinforcement learning (RL), which uses lower confidence bounds on value estimates to avoid out-of-distribution (OOD) actions with uncertain reward estimates. Our approach, termed as *caution*, can be seen as the reverse of *curiosity*: where curiosity (e.g., via Random Network Distillation, RND) rewards prediction error as a signal of novelty, caution penalizes prediction error as a signal of distributional uncertainty. Practically, caution trains an error model on typical responses and uses its prediction error to lower reward estimates for atypical ones. Our extensive empirical evaluation demonstrates that caution is a simple, computationally efficient approach that substantially mitigates reward hacking in BoN sampling. We also provide a theoretical analysis in a simplified linear setting, which shows that caution provably improves over the standard BoN approach. Together, our results not only establish caution as a practical solution to reward hacking, but also provide evidence that curiosity-based approaches can be a general OOD detection technique in LLM settings.

1 INTRODUCTION

Inference-time scaling has emerged as a transformative paradigm for enhancing language model performance, enabling significant improvements across a wide range of reasoning tasks without increasing model size (Brown et al., 2024; Guo et al., 2025; Jaech et al., 2024). This success motivates the question of how best to leverage additional inference-time compute to maximize performance. A particularly popular and effective approach is *Best-of-N* (BoN) sampling (Stiennon et al., 2020; Nakano et al., 2021; Wang et al., 2022; Li et al., 2022; Huang et al., 2025a), where multiple candidate responses are generated for a given prompt, scored according to a reward model $\hat{r}$, and the highest-scoring response is selected. This approach capitalizes on the intuition that generating more candidates should increase the probability of finding higher-quality solutions, allowing the model to effectively 'explore' a larger portion of the response space than it could with only a single response.

While BoN is a simple and competitive baseline that is capable of astonishing gains in many settings (Brown et al., 2024), its success is fundamentally limited by the quality of the reward model $\hat{r}$ used to score and select responses. Indeed, a common phenomenon often occurs with BoN, where performance initially improves as N increases, but then hits an inflection point after which larger N lead to increasingly worse outcomes (Gao et al., 2023; Huang et al., 2025b; Khalaf et al., 2025); an example of this phenomenon can be seen in Figure 1, where we plot the performance of BoN on the mathematical reasoning task GSM8K for different N scored by several different reward models. This counterintuitive phenomenon, whereby increasing N leads to worse performance, occurs due

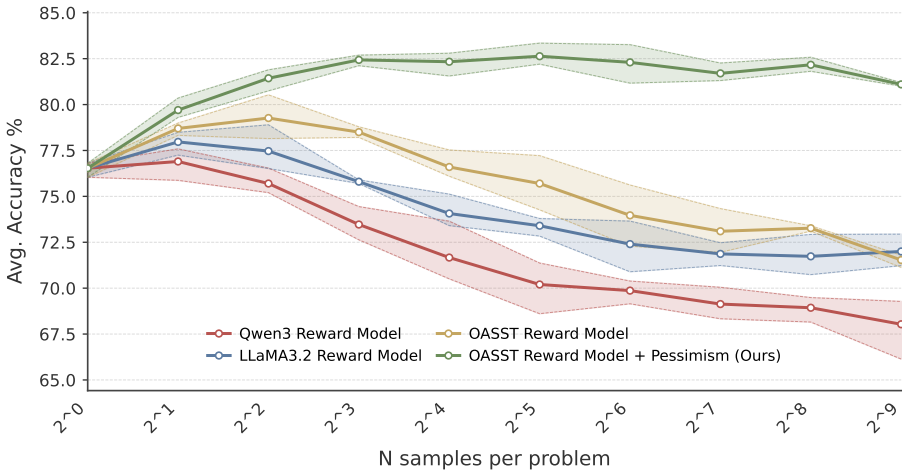


Figure 1: **Average Accuracy with different sampling budgets for Best-of-N** on the GSM8k dataset. We see that standard Best-of-N sampling (blue, red, and gold) suffers from reward hacking, exhibiting the characteristic rise-and-fall pattern as N increases. In contrast, caution (our approach, green) consistently improves with larger N, effectively mitigating reward hacking.

to *reward hacking*, the process by which BoN selects responses that exploit $\hat{r}$ as opposed to the ground truth reward $r^\star$ we actually care about; while $\hat{r}$ may be a reasonable approximation to $r^\star$ on 'typical' generations, as N increases, BoN selects more atypical responses lying outside the training distribution of $\hat{r}$ that achieve high $\hat{r}$ scores through spurious correlations rather than genuine quality improvements. In other words, reward hacking is a manifestation of Goodhart's Law: whenever a metric (in this case $\hat{r}$) becomes an optimization target, it ceases to be a reliable measure of quality (Goodhart, 1984; Weng, 2024); for example, reward models are known to favor certain formatting preferences and surface-level patterns learned during training that do not necessarily correspond to improved reasoning or correctness (Liu et al., 2024b; Yu et al., 2025; Bukharin et al., 2025).

With the intuition that reward hacking is caused by out-of-distribution (OOD) generations from the perspective of the reward model, several works have proposed mitigation attempts by either improving the reward model (Liu et al., 2025; Yu et al., 2025) or by regularizing the selection process to favor in-distribution responses (Huang et al., 2025b). Unfortunately, the former approach is fundamentally limited by the asymmetric difficulty of the problem: while it is relatively straightforward to obtain representative examples of high-quality responses through careful curation and human annotation, exhaustively characterizing all possible reward hacking strategies is intractable. Representative of the latter work is Huang et al. (2025b), who propose to regularize the selection process to ensure that the distribution of selected responses does not drift too far from the distribution of responses seen during reward model training in a strong information theoretic sense. The authors provide strong theoretical guarantees on the efficacy, monotonicity, and optimality of their approach, and demonstrate that it can be efficiently implemented in practice through a simple rejection sampling scheme (Block & Polyanskiy, 2023). However, this approach is overly conservative in practice, preventing the selection of genuinely better responses that are slightly out-of-distribution, and thus fails to fully leverage the benefits of inference-time scaling. The problem arises because the regularization is *distributional* and thus simultaneously ensures all possible OOD responses are penalized equally, regardless of whether or not they are likely to arise from imperfections in $\hat{r}$. The starting point for this paper is thus to ask: ***Can we design a BoN sampling scheme that is both robust to reward hacking and still able to leverage the full benefits of inference-time scaling?***

Contributions. We answer this question in the affirmative by introducing *caution*, an inference-time instantiation of the *pessimism* principle from Reinforcement Learning (RL) (Jin et al., 2021; Guo et al., 2022). Pessimism relies on lower confidence bounds to avoid selecting OOD actions with uncertain rewards. While Huang et al. (2025b) implements pessimism at the *distribution* level by constraining the sampling distribution to remain close to the base policy, we instead apply it at the *reward* level: we penalize OOD responses by subtracting per-response uncertainty estimates from the scores assigned by $\hat{r}$, and then select the response with the highest pessimistic score.

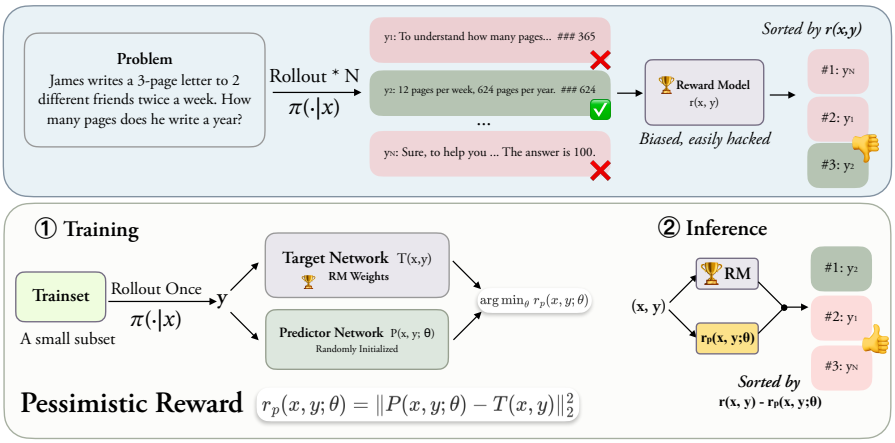


Figure 2: **Overview.** Predictor is trained to match RM features on typical responses; at inference, we select the candidate with the highest pessimistic reward, down-weighting OOD ones.

Conceptually, caution is the dual of *curiosity*, a principle used to drive optimistic exploration in deep RL (Pathak et al., 2017; Burda et al., 2018). As in curiosity, we measure uncertainty by fixing a simple learning target (e.g., a frozen feature embedding), training a student model to predict this target on in-distribution data, and using the prediction error as an uncertainty signal. Unlike curiosity, however, our setting is fully *offline*, so no continual student training is required—making the method significantly simpler and more practical. Through extensive empirical evaluation, we show that caution is computationally efficient and effectively mitigates reward hacking in BoN sampling. We further provide a theoretical analysis in a simplified linear setting, proving that caution strictly improves over standard BoN. Taken together, these results demonstrate that caution is both a powerful practical solution to reward hacking and compelling evidence for the broader efficacy of curiosity-style uncertainty signals in OOD detection and pessimistic policy learning for language models.

2 Methodology

2.1 Problem Formulation

Consider a trained language model (LM) π, where $\pi : \mathcal{X} \to \Delta(\mathcal{Y})$ is a map from prompts $x \in \mathcal{X}$ to distributions over responses $y \in \mathcal{Y}$. While the LM is typically an autoregressive model, generating one token at a time before feeding the generated tokens back in as context, we abstract this process away and instead consider prompts and responses as single entities, possibly consisting of the concatenation of many individual tokens. Suppose we have access to a learned reward model $\hat{r} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that assigns a score to a prompt-response pair (x, y) intended to reflect the quality of the response y to the prompt x. While the learner has access to $\hat{r}$, the true goal is to output a response that approximately maximizes some ground-truth reward $r^\star : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, which is unknown to the learner. We focus on *reward-hacking* behavior that seeks to maximize $\hat{r}$ at the expense of $r^\star$.

Given a prompt x, the learner samples N candidate responses $y_1, \ldots, y_N \sim \pi(\cdot|x)$ independently from the LM and then selects one of these responses $y_{\hat{i}}$ with $\hat{i} \in [N]$ to output. Perhaps the simplest strategy is the *Best-of-N* (BoN) strategy, which simply selects the response with the highest reward model score: $\hat{i} = \hat{i}_N = \operatorname{argmax}_{i \in [N]} \hat{r}(x, y_i)$. Unfortunately, this strategy is vulnerable to reward hacking when we only have $\hat{r} \approx r^\star$ on typical samples from π. As N grows large, the distribution of $\hat{i}_N$ can differ substantially from that of typical samples from π, leading to poor performance with respect to $r^\star$ as shown in (Huang et al., 2025b). While that work proposes to address this issue by constraining the BoN sampling distribution $\widehat{\pi}$ to be close to π, we instead propose to directly apply regularization to the reward estimates themselves. Thus, we will design an uncertainty estimate $u : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ that quantifies how uncertain we are about the reward model's estimate $\hat{r}(x, y)$ for a given prompt-response pair (x, y) and then define $\hat{r}_{\mathrm{LCB}}(x, y) = \hat{r}(x, y) - \lambda u(x, y)$ for some $\lambda > 0$ to be a pessimistic variant of the reward model that penalizes uncertain responses (Jin et al., 2021). As long as $u(x, y)$ is small for typical samples from $\pi(\cdot|x)$ and large for 'atypical' samples,

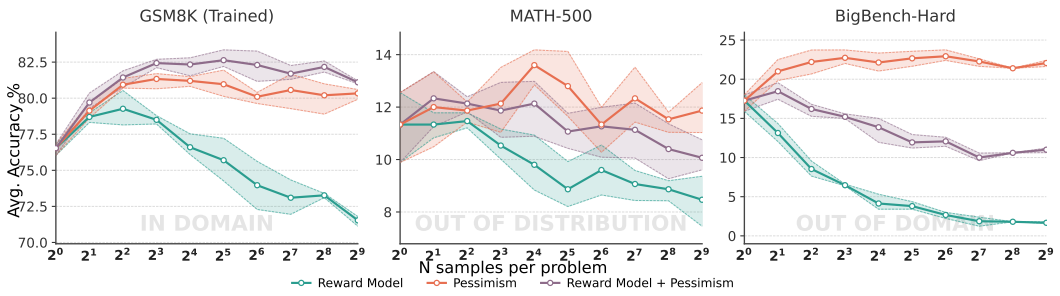


Figure 3: **Scaling over N across distributions and domains.** Best-of-N sampling on GSM8K, MATH-500, and BigBench-Hard. Curves compare selection by Reward Model, Pessimism, and Reward Model + Pessimism. Note that the *pessimism function is trained only on GSM8K*; thus, MATH-500 represents an out-of-distribution setting, while BigBench-Hard represents a fully out-of-domain setting.

this approach will successfully penalize reward-hacking responses by ensuring that $\hat{r}_{\mathrm{LCB}} \leq r^\star$ and this inequality is approximately tight for actually good quality responses. The final algorithm will thus be to return $\hat{i} = \mathrm{argmax}_{i \in [N]} \hat{r}_{\mathrm{LCB}}(x, y_i)$ and the question becomes *what choice of uncertainty estimate u appropriately captures OOD responses in a reward-aware manner?*

2.2 Curiosity to Caution: Instantiating Pessimism

Our approach introduces the principle of *caution*, a phenomenon dual to the well-known technique of *curiosity* used in online RL to incentivize exploration (Pathak et al., 2017; Burda et al., 2018). While in online RL, OOD states are desirable (as they represent good exploration targets), in our offline setting, OOD responses are to be avoided, as we have no reliable way to estimate their true reward given that $\hat{r} \approx r^\star$ only on typical responses from π. We draw inspiration from curiosity (Pathak et al., 2017) and Random Network Distillation (RND) (Burda et al., 2018) in order to ensure that selected responses remain close to the distribution π on which $\hat{r}$ is reliable.

In curiosity and RND, the core idea is to continually train a predictor network to match the outputs of a fixed target that is easily evaluated by the learner; the supervised learning error then becomes a proxy for OOD detection, with the intuition being that the predictor will be accurate on states similar to those seen during training and inaccurate otherwise. While empirically successful as an 'intrinsic reward' for optimistic exploration, this method can be challenging to implement due to the continued training of the predictor during online RL, which can lead to nontrivial memory and time overheads in practical RL pipelines. Our method uses the same core idea of using the supervised learning error as a proxy for OOD detection, but is *trained fully offline*, which greatly increases practicality and ease of implementation.

Implementing "Caution" We employ two neural networks operating on the internal representations of the reward model: a fixed *target network* $T(x, y)$ that extracts features from the intermediate layers of the frozen reward model, and a trainable *predictor network* $P_\theta(x, y)$ that learns to predict the target network's outputs. Specifically, we use the first L layers of the reward model as a fixed feature extractor for the target network, using the embedding after layer L as the prediction target. Our predictor network $P_\theta(x, y)$ is a trainable neural network with parameters θ that learns to predict the target network's outputs. The predictor can use various architectural choices (shared embeddings, simplified encoders, projection layers) but is designed to be lightweight compared to the full reward model in order to ensure efficient training and inference.

Training Process. We train our predictor P_θ on a dataset $\mathcal{D}_{\text{train}}$ of prompt-response pairs constructed directly from the benchmark train split using mean squared error loss: $\mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{train}}}\left[\|P_\theta(x, y) - T(x, y)\|^2\right]$, where the norm is Euclidean. We construct responses $y \sim \pi(\cdot|x)$ by sampling from π as an efficient empirical proxy for the training distribution of the reward model. By training on this supervision-free distribution, the predictor achieves low error on in-distribution patterns while preserving high error on distributional outliers. We emphasize that the

Table 1: **Scaling Performance Across Datasets with Reward Hacking Mitigation. Peak Acc**: highest accuracy achieved across all N values. **Final Acc**: accuracy at $N = 512$ where reward hacking is most severe. **Degradation**: performance drop from peak to final (lower is better). Results show mean with 95% confidence intervals as subscripts across 3 bootstrap runs. Best results are in **bold** and second-best are underlined.

Method	GSM8K (2021)			MATH-500 (2023)			BBH-Hard (2022)		
	Peak	Final	Degr.	Peak	Final	Degr.	Peak	Final	Degr.
Reward Model	$79.3_{\pm 1.2}$	$71.5_{\pm 0.3}$	7.7	$11.5_{\pm 0.3}$	$8.5_{\pm 1.0}$	3.0	$17.3_{\pm 1.0}$	$1.7_{\pm 0.1}$	15.6
Pessimism Only	$81.3_{\pm 0.5}$	$80.3_{\pm 0.3}$	**1.0**	**13.6**$_{\pm 0.7}$	**11.9**$_{\pm 1.0}$	**1.7**	**22.9**$_{\pm 0.7}$	**22.1**$_{\pm 0.4}$	**0.9**
RM + Pessimism	**82.6**$_{\pm 0.6}$	**81.1**$_{\pm 0.1}$	1.5	$12.3_{\pm 1.0}$	$10.1_{\pm 0.6}$	2.3	$18.5_{\pm 1.0}$	$11.0_{\pm 0.3}$	7.5

data requirements for this step are restricted to *prompts*, which are often much cheaper to collect than the high-quality *labelled* data used to train reward models; a gold standard would be to reuse the same prompts used to train the reward model itself, but in practice we find that our method's OOD detection is relatively robust across different prompt distributions (cf. Section 3.1).

Pessimistic Reward Estimation at Inference Time. Once we have trained our predictor P_θ, we can use it to define our uncertainty estimate $u(x, y) = \|P_\theta(x, y) - T(x, y)\|^2$, the prediction error of the predictor on the target network's features. We then plug this into our pessimistic reward estimate $\hat{r}_{\text{LCB}}(x, y) = \hat{r}(x, y) - \lambda u(x, y) = \hat{r}(x, y) - \lambda \cdot \|P_\theta(x, y) - T(x, y)\|^2$. Note that this score is very easy to compute at inference time, requiring only two forward passes (one through P_θ and one through T) in addition to the forward pass through the reward model to compute $\hat{r}$; note that all of these passes can be fully parallelized and the cost of evaluating $u(x, y)$ is on the same order as that of evaluating $\hat{r}$ due to the reuse of features from $\hat{r}$. The parameter λ controls the strength of the pessimism penalty, with $\lambda = 0$ recovering standard BoN sampling. The final selection becomes $\hat{i} = \text{argmax}_{i \in [N]} \hat{r}_{\text{LCB}}(x, y_i)$, i.e., choosing the response with the highest pessimistic reward estimate. This procedure is summarized in Figure 2.

2.3 Theoretical Interpretation

In order to provide further motivation for our approach, we analyze in Appendix C a simple theoretical setting in which our approach provably improves upon BoN. While we defer the details to the appendix, we summarize the main theorem here:

Theorem 1 (Informal version of Theorem 3)**.** *Let $y_1, \ldots, y_N \in \mathbb{R}^d$ be i.i.d. samples from a model π and let $r^\star(y)$ be a linear reward function. Let $i^\star = \text{argmax}_{i \in [N]} r^\star(y_i)$ be the optimal response and let $\hat{i} = \text{argmax}_{i \in [N]} \hat{r}(y_i)$ be the response selected by BoN using a learned reward model $\hat{r}$. If $i_{\text{pess}} = \text{argmax}_{i \in [N]} \hat{r}_{\text{LCB}}(y_i)$, where $\hat{r}_{\text{LCB}}$ is our caution-regularized reward estimate, then under suitable conditions on the target network, $r^\star$, $\hat{r}$, π, and the predictor network, it holds that*

$$\mathbb{E}\left[r^\star(y_{i_{\text{pess}}}) - r^\star(y_{\hat{i}})\right] \gtrsim \sqrt{\log(N)}, \qquad \textit{and} \qquad \lim_{N \uparrow \infty} \frac{\mathbb{E}\left[r^\star(y_{i^\star}) - r^\star(y_{i_{\text{pess}}})\right]}{\mathbb{E}\left[r^\star(y_{i^\star})\right]} = 0.$$

While the assumptions and conditions of Theorem 1 are necessarily somewhat strong in order to facilitate the analysis, the theorem provides a proof-of-concept that our approach can provably outperform BoN in a stylized setting as well as, to the best of our knowledge, the first theoretical guarantee on the success of curiosity- and RND-style methods for OOD detection.

3 Experiments

We now empirically evaluate our proposed approach, with a focus on answering the following three questions: (1) How well does caution mitigate reward hacking as the number of candidates N increases? (2) What design decisions (e.g. architecture and training hyperparameters) contribute most to the overall success of our method? (3) In which scenarios does our method outperform standard BoN sampling, and what factors drive its success? We now briefly describe our empirical setup (with full details deferred to Appendix D) before addressing each of these questions in turn.

Table 2: **Weight Ablation and Design Comparison on GSM8K.** We compare different mixing weights between uncertainty score and $\hat{r}$ score, contrasting our approach (distilling reward model features) against traditional RND (distilling random network features). **Peak Acc**: best accuracy across all N. **Final Acc**: accuracy at $N = 512$. Results demonstrate that distilling reward model representations substantially outperforms random network distillation across weight combinations.

Pessimism strength (λ)	**Caution (Ours)**		**Traditional RND**	
	Peak Acc	Final Acc	Peak Acc	Final Acc
0.0 (RM Only)	78.9	71.2	78.9	71.2
0.2	79.5	72.5	78.7	71.3
0.4	80.3	76.0	78.5	72.1
0.6	<u>81.5</u>	<u>80.2</u>	78.4	73.1
0.8	**82.1**	**81.0**	78.1	74.9
1.0 (Caution Only)	81.3	79.8	77.0	72.9

Experimental Setup. We use `Llama-3.2-3B-Instruct` (Dubey et al., 2024) as our language model π due to its strong reasoning capabilities and open-source availability.[1] We consider three prompt distributions coming from reasoning datasets of varying difficulty: `GSM8K` (Cobbe et al., 2021), `MATH-500` (Hendrycks et al., 2021), and `BigBench-Hard` (Suzgun et al., 2022). Our ground truth reward $r^\star$ is binary, with reward given if and only if a response provides the correct answer to the given prompt. While we consider several reward models $\hat{r}$, we focus primarily on `OASST` (Köpf et al., 2023), which Figure 1 demonstrates provides better performance than comparably sized reward models like `Skywork-Reward-V2-Qwen3-0.6B` and `Skywork-Reward-V2-Llama-3.2-1B` (Liu et al., 2025), making it a strong baseline for demonstrating reward hacking phenomena. For each prompt and N, we use vLLM (Kwon et al., 2023) to generate N independent responses from the base model π (and boostrap this process 3 times to generate confidence intervals), before scoring each response with $\hat{r}$ and measuring the performance of the selected response according to $r^\star$. To train our predictor network P_θ, we use an independent dataset of responses generated by the base model π itself on the training split of `GSM8K`, and evaluate the performance of the uncertainty estimates both in-distribution (prompts from the test set of `GSM8K`) and out-of-distribution (prompts from other reasoning datasets). In order to maintain consistency, we normalize all rewards $\hat{r}$ to be centred with unit variance using an independent set of responses and we do the same for the uncertainty estimates from P_θ.

3.1 Caution Mitigates Reward Hacking

Our main results demonstrate that our proposed mechanism of caution **mitigates reward-hacking and leads to improved performance of BoN sampling as N increases both for in- and out-of-distribution prompts.** In Figure 1, we exhibit the performance of BoN sampling on `GSM8K` for several choices of reward models and observe that they all exhibit reward hacking, with performance degrading for large N. We also see that using our pessimistic approach, this reward-hacking is substantially mitigated, with performance improving monotonically with N for all considered N. Moreover, we get a substantial performance boost of 4.2% over peak accuracy for the reward model alone and an astonishing 15.5% boost over the final accuracy of the reward model alone. This trend compares favorably with that of Huang et al. (2025b), who observe monotonicity of performance in N, but struggle to outperform BoN sampling for optimally tuned N for most choices of π and $\hat{r}$.

The above results involve evaluating caution on the same distribution of prompts used to train the predictor network P_θ, which is likely beneficial to its performance. In reality, we expect to use caution in scenarios where the prompts are potentially OOD and hope that the uncertainty estimates remain valid. To evaluate the extent to which this holds, we used the same uncertainty estimates to produce pessimistic BoN sampling on two significantly harder reasoning datasets: `MATH-500` and `BigBench-Hard`. These datasets comprise prompts that are significantly different from those in `GSM8K`, with the latter even coming from a different, non-mathematical domain.

[1] Some other open-weight models are thought to be contaminated with benchmark datasets (Wu et al., 2025), which could skew our results, further motivating our choice in model.

Table 3: Ablation study of predictor architecture. **Training Loss** measures how well the predictor fits reward model representations. **Peak Acc** shows best performance across all N. **Final Acc** shows performance at largest N, where reward hacking is most severe.

Predictor Configuration	Reconstruction Loss	Peak Acc (%)	Final Acc (%)
Simplified Architecture Variants			
Lightweight + Trainable Emb.	0.242	82.2	82.2
Lightweight + Frozen Emb.	0.246	81.5	81.3
Lightweight + Separate Emb.	0.255	81.8	82.7
Lightweight w/o Projection	0.281	81.8	81.3
Full Architecture Variants			
Full + Trainable Emb.	0.127	80.3	80.2
Full + Frozen Emb.	0.127	80.4	78.4

We report the results of these OOD prompt experiments in Figure 3 and Table 1, comparing the performance of the in-distribution prompts (left) with the two other tasks (centre and right). As in Figure 1, we sweep over a logarithmic grid in N from 1 to 512 and compare three approaches: BoN using only $\hat{r}$, BoN using only the uncertainty estimates from P_θ, and BoN using caution. As in Figure 1, we see that caution successfully mitigates reward hacking while preserving the benefits of reward-guided selection. The results on `MATH-500` reveal a different pattern: pessimism-only outperforms the combined approach, achieving 13.6% peak accuracy with only 1.7 points degradation compared to 2.3 points for the combined method. This counterintuitive finding reflects the increased problem difficulty—while `MATH-500` shares the mathematical reasoning domain with `GSM8K`, the problems are significantly more complex and multi-step, making the task of $\hat{r}$ more difficult. In this regime, $\hat{r}$ becomes less reliable at distinguishing genuine quality improvements, making its contribution less beneficial. However, caution still effectively prevents the severe degradation seen with BoN (3.0 points). The most challenging task we consider is `BigBench-Hard`, where the reward model $\hat{r}$ fails catastrophically. Indeed, we observe the counterintuitive phenomenon that using the uncertainty penalty alone is the most performant of all three methods, doing even better than combining uncertainty with $\hat{r}$. This surprising result is due to the fact that when reward models encounter problems substantially harder than their training distribution, their learned spurious correlations dominate their judgments, leading to systematic selection of responses that superficially mimic quality patterns without genuine correctness. Caution, while also operating outside its training distribution, avoids being misled by these hacking features and maintains stable performance by preferring responses that match familiar distributional patterns rather than chasing unreliable reward signals.

3.2 Ablation Studies

In this section, we dissect the design and implementation of our proposed caution mechanism to understand which components contribute most to its success. Further ablation studies, including detailed inspections of uncertainty scores for individual responses and the effect they have on $\hat{r}$ and $r^\star$ scores can be found in Appendix F, while additional details on the ablations can be found in Appendix D. We evaluate three critical design decisions: (1) the precise architectural choice of the predictor network; (2) the use of pre-trained embeddings in the target network; (3) the use of a projector layer between the predictor and target networks. We also conduct two additional ablations to validate our core design choices: (a) the choice of mixing weights between $\hat{r}$ and the uncertainty estimates; (b) the choice of distilling reward model features versus random network features.

Architectural Ablations. We first examine the extent to which the complexity of P_θ and its relationship to the target network T affects performance, in particular whether P_θ needs to share an architecture with the teacher T. The results are shown in Table 3: while greater flexibility in P_θ leads to a better reconstruction loss, this does not transfer to better performance when used to instantiate caution. This occurs because improving the reconstruction loss can actually hurt the ability of P_θ to detect distributional novelty, as it becomes too good at reconstructing even out-of-distribution features. We observe a similar trend when considering whether or not to tune the embeddings or share them between P_θ and T: while sharing embeddings leads to better reconstruction, it can hurt the final performance, which suggests that forcing predictors to learn representations from scratch

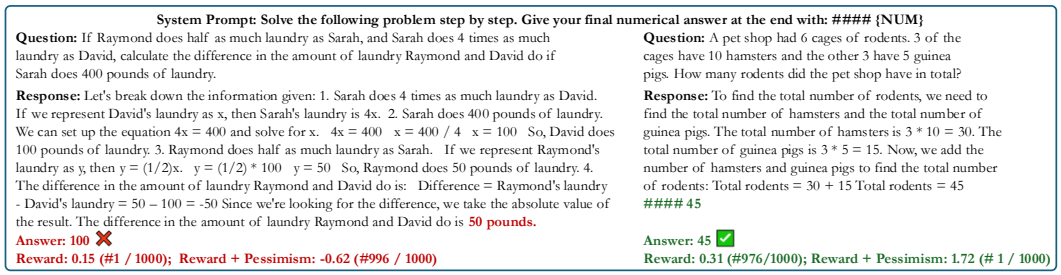


Figure 4: **Contrasting Selection Behaviors: Reward Hacking vs. Format Compliance.** Two representative examples showing how reward models favor verbose responses regardless of correctness, while our curiosity-driven pessimism prioritizes format compliance and distributional familiarity. RM assigns high scores to detailed responses regardless of correctness, while pessimism detects distributional deviation from training patterns and prefers correctly formatted solutions.

creates more sensitive distributional boundaries than inheriting potentially overly simplified features from pre-trained models. Finally, we see that adding a projection layer between the predictor and target networks provides consistent but modest benefits, indicating that information bottlenecks can help prevent pure memorization while preserving signal quality. Together, these findings establish a key empirical finding for instantiating caution: **effective detection requires a careful balance between reconstruction accuracy and novelty sensitivity.**

Strength of regularization with Caution. To validate our core design choices, we systematically vary the mixing weights between the uncertainty score and that of $\hat{r}$ to identify the optimal balance for reward hacking mitigation and understand the robustness of our results to this choice. Note that due to the normalization of $\hat{r}$ and uncertainty estimates described above, the strength λ can be viewed as a direct measurement of the influence of the pessimism relative to $\hat{r}$. We report our findings in Table 2 and Appendix F. We observe that our approach is relatively robust to the choice of λ, with moderate to high weights (0.6-0.8) achieving optimal performance. While the precise optimal weight will vary by task and choice of $\hat{r}$, with larger λ being required in situations where $\hat{r}$ is less reliable, the results suggest that our approach does not require delicate tuning to be effective.

Comparing Caution to RND. Finally, we compare our approach of distilling reward model features (motivated by curiosity in Pathak et al. (2017)) against the RND (Burda et al., 2018), which takes the teacher T to be a randomly initialized network (possibly on top of pre-trained embeddings), testing whether our hypothesis about distributional familiarity requires semantic grounding in the reward model's representations. We again consider a range of mixing weights λ and report the results in Table 2. The results reveal a stark contrast between caution and RND, with the latter exhibiting dramatically inferior performance across all choices of λ. This observation provides evidence for the hypothesis that effective distributional regularization requires semantic grounding: randomly initialized features cannot provide meaningful distributional boundaries, while reward model features capture task-relevant patterns that enable robust novelty detection. The results establish that curiosity-driven pessimism succeeds not merely because of prediction error signals, but specifically because these signals are computed relative to the reward model's learned task representations.

3.3 Why It Works: A Case Study

We now turn to a case study to illustrate the mechanism by which caution mitigates reward hacking and improves performance. In Figure 4, we present two representative examples of questions-response pairs. On the left, we see an instance that $\hat{r}$ scores highly (99.8th percentile of all scored responses) despite being incorrect; this response is verbose and contains multiple mathematical reasoning steps, but ultimately fails to provide the correct answer and does not follow the required formatting, instead demonstrating exactly the type of reward hacking our method targets: superficial mimicry of quality patterns (detailed explanations, step-by-step structure) without genuine correctness or adherence to specified requirements. Note that the caution score for this response

appropriately identifies it as OOD, likely due to its failure to follow the formatting requirements that are common in high-quality responses in the training data.

By contrast, on the right of Figure 4, we see a concise, accurate response following the formatting requirements. While $\hat{r}$ does not rank this response as well as the first, the pessimism recognizes it as familiar, matching the patterns of high-quality responses in the training data that consistently follow formatting specifications and provide direct, correct solutions. This juxtaposition reveals that reward models can conflate verbosity with quality, particularly at larger N where more elaborate responses become available. Our method effectively distinguishes between responses that *appear* sophisticated (verbose explanations) and those that *are* actually correct and compliant with task specifications.

4 Discussion

In this work, we investigated instantiating pessimistic reward estimation with the principle of *caution*, an approach adapted from curiosity-driven exploration in RL that uses a supervised learning error as a measure of distributional uncertainty and penalizes the estimated reward of uncertain, out-of-distribution (OOD) inputs. While our focus was on mitigating reward hacking in Best-of-N sampling, a particular inference-time scaling technique, our results demonstrate that, when properly applied on top of pre-trained features, caution can effectively detect OOD samples and instantiate pessimistic policies in general. While we defer a more complete survey of related work to Appendix B, we now provide a brief summary thereof as well as discuss future directions.

Related Work. Best-of-N sampling was introduced in Stiennon et al. (2020) and has been empirically investigated in many reasoning settings (Cobbe et al., 2021; Lightman et al., 2023; Li et al., 2022; Brown et al., 2024). While often effective, when the scoring rule is learned (as opposed to oracular), the approach has long been shown to be vulnerable to reward-hacking (Amodei et al., 2016; Skalse et al., 2022; Gao et al., 2023). While many attempted mitigations of reward-hacking have been explored for RL *finetuning*, relatively few works have focused on the *inference-time setting*. Of particular note is Huang et al. (2025b), which proposes a distributional regularization approach that samples according to a χ^2-regularized BoN procedure. While theoretically well-motivated and empirically effective at ensuring monotonicity in N, the approach is overly conservative in practice. Another approach is that of Jinnai et al. (2024), who apply distributional regularization with respect to a Wasserstein distance on some embedding space; while potentially effective, the computation required grows quadratically in N, making it impractical for even moderate values of N. In contradistinction to these works, our approach applies pessimism directly to the estimated rewards in a way that naturally leverages the beyond-worst-case errors present in estimated reward models in a way that is impossible for these distributional regularization approaches.

Our key technique of *caution* is built on top of the foundational curiosity (Pathak et al., 2017) and Random Network Distillation (RND) (Burda et al., 2018) from classical deep RL. While the such techniques have been very popular in aiding exploration, the extent to which they can be used for OOD detection and pessimistic learning has been a matter of some debate, with Rezaeifar et al. (2022) claiming negative results and Ciosek et al. (2019); Nikulin et al. (2023) demonstrating some positive results. While these works are (i) in classical RL or supervised learning settings and (ii) do not use pre-trained features, our results morally align with the latter camp, demonstrating that such approaches can be effective for OOD detection and pessimistic learning in language models.

Future Directions. Our work provides strong evidence that caution, when correctly applied on top of pre-trained features, can be an effective detector of OOD text. While we instantiate this approach for pessimistic reward estimation, it is natural to wonder if curiosity can be used as an explicit reward signal to encourage exploration of novel behaviors either purely during inference or during RL post-training of reasoning models. While some preliminary work like Gao et al. (2025) has explored this idea, we believe that their mixed results stem from the fact that the curiosity module was trained from scratch rather than on top of pre-trained features, which we find to be so effective for OOD detection. Another interesting direction is to explore the extent to which our proposed approach can help ensure continued AI alignment in the face of adversarial prompting or distributional shift due to inference-time scaling. While the tasks we consider in this work are related to reasoning, we expect that our results would carry over to safety and alignment tasks *mutatis mutandis*, which would represent a promising new approach to ensuring robust alignment of language models.

Ethics Statement

We affirm adherence to the ICLR Code of Ethics. This work only involve publicly available benchmark datasets under their respective licenses; we do not collect new human-subject data and process no personally identifiable information.

Reproducibility Statement

To ensure reproducibility and transparency of the results within this paper, we document relevant hyperparameters and implementation details in the appendix. We will open-source our codebase along with detailed instructions and scripts.

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Afra Amini, Tim Vieira, Elliott Ash, and Ryan Cotterell. Variational best-of-n alignment. *arXiv preprint arXiv:2407.06057*, 2024.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In *Large Scale Kernel Machines*. MIT Press, 2007.

Adam Block and Yury Polyanskiy. The sample complexity of approximate rejection sampling with applications to smoothed online learning. In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 228–273. PMLR, 2023.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Alexander Bukharin, Haifeng Qian, Shengyang Sun, Adithya Renduchintala, Soumye Singhal, Zhilin Wang, Oleksii Kuchaiev, Olivier Delalleau, and Tuo Zhao. Adversarial training of reward models. *arXiv preprint arXiv:2504.06141*, 2025.

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

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Kamil Ciosek, Vincent Fortuin, Ryota Tomioka, Katja Hofmann, and Richard Turner. Conservative uncertainty estimation by fitting prior networks. In *International Conference on Learning Representations*, 2019.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. *arXiv preprint arXiv:2310.02743*, 2023.

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

Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D'Amour, DJ Dvijotham, Adam Fisch, Katherine Heller, Stephen Pfohl, Deepak Ramachandran, et al. Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking. *arXiv preprint arXiv:2312.09244*, 2023.

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

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Charles Goodhart. Monetary theory and practice: the uk experience. 1984.

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.

Kaiyang Guo, Shao Yunfeng, and Yanhui Geng. Model-based offline reinforcement learning with pessimism-modulated dynamics belief. *Advances in Neural Information Processing Systems*, 35: 449–461, 2022.

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

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

Audrey Huang, Adam Block, Dylan J Foster, Dhruv Rohatgi, Cyril Zhang, Max Simchowitz, Jordan T Ash, and Akshay Krishnamurthy. Self-improvement in language models: The sharpening mechanism. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J Foster. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment. *arXiv preprint arXiv:2503.21878*, 2025b.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Yuki Ichihara, Yuu Jinnai, Tetsuro Morimura, Kaito Ariu, Kenshi Abe, Mitsuki Sakamoto, and Eiji Uchibe. Evaluation of best-of-n sampling strategies for language model alignment. *arXiv preprint arXiv:2502.12668*, 2025.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

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.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International conference on machine learning*, pp. 5084–5096. PMLR, 2021.

Yuu Jinnai, Tetsuro Morimura, Kaito Ariu, and Kenshi Abe. Regularized best-of-n sampling with minimum bayes risk objective for language model alignment. *arXiv preprint arXiv:2404.01054*, 2024.

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. 2024.

Hadi Khalaf, Claudio Mayrink Verdun, Alex Oesterling, Himabindu Lakkaraju, and Flavio du Pin Calmon. Inference-time reward hacking in large language models. *arXiv preprint arXiv:2506.19248*, 2025.

Andreas Köpf, Yannic Kilcher, Dimitri Von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Nguyen, Oliver Stanley, Richárd Nagyfi, et al. Openassistant conversations-democratizing large language model alignment. *Advances in neural information processing systems*, 36:47669–47681, 2023.

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.

Cassidy Laidlaw, Shivam Singhal, and Anca Dragan. Correlated proxies: A new definition and improved mitigation for reward hacking. *arXiv preprint arXiv:2403.03185*, 2024.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

Jiong Li and Pratik Gajane. Curiosity-driven exploration in sparse-reward multi-agent reinforcement learning. *arXiv preprint arXiv:2302.10825*, 2023.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

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

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*, 2024a.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, et al. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*, 2025.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Rm-bench: Benchmarking reward models of language models with subtlety and style. *arXiv preprint arXiv:2410.16184*, 2024b.

Odelia Melamed, Gilad Yehudai, and Gal Vardi. Adversarial examples exist in two-layer relu networks for low dimensional linear subspaces. *Advances in Neural Information Processing Systems*, 36:5028–5049, 2023.

Antonio Valerio Miceli-Barone, Alexandra Birch, and Rico Sennrich. Distributionally robust recurrent decoders with random network distillation. *arXiv preprint arXiv:2110.13229*, 2021.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Alexander Nikulin, Vladislav Kurenkov, Denis Tarasov, and Sergey Kolesnikov. Anti-exploration by random network distillation. In *International Conference on Machine Learning*, pp. 26228–26244. PMLR, 2023.

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.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544*, 2022.

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

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, 2023.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

Alexandre Ramé, Nino Vieillard, Léonard Hussenot, Robert Dadashi, Geoffrey Cideron, Olivier Bachem, and Johan Ferret. Warm: On the benefits of weight averaged reward models. *arXiv preprint arXiv:2401.12187*, 2024.

Patrik Reizinger and Márton Szemenyei. Attention-based curiosity-driven exploration in deep reinforcement learning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3542–3546. IEEE, 2020.

Shideh Rezaeifar, Robert Dadashi, Nino Vieillard, Léonard Hussenot, Olivier Bachem, Olivier Pietquin, and Matthieu Geist. Offline reinforcement learning as anti-exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8106–8114, 2022.

Wenlei Shi and Xing Jin. Heimdall: test-time scaling on the generative verification. *arXiv preprint arXiv:2504.10337*, 2025.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471, 2022.

Mei Song, Andrea Montanari, and P Nguyen. A mean field view of the landscape of two-layers neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Haoran Sun, Yekun Chai, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Curiosity-driven reinforcement learning from human feedback. *arXiv preprint arXiv:2501.11463*, 2025.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

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.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. In *ACL*, 2024a.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *EMNLP*, 2024b.

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. *arXiv preprint arXiv:2203.11171*, 2022.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, et al. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. *arXiv preprint arXiv:2306.05087*, 2023.

Lilian Weng. Reward hacking in language models. *lil'log*, 2024. URL https://lilianweng.github.io/posts/2024-06-19-reward-hacking.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. *arXiv preprint arXiv:2212.09561*, 2022.

Mingqi Wu, Zhihao Zhang, Qiaole Dong, Zhiheng Xi, Jun Zhao, Senjie Jin, Xiaoran Fan, Yuhao Zhou, Huijie Lv, Ming Zhang, et al. Reasoning or memorization? unreliable results of reinforcement learning due to data contamination. *arXiv preprint arXiv:2507.10532*, 2025.

Yuzi Yan, Xingzhou Lou, Jialian Li, Yiping Zhang, Jian Xie, Chao Yu, Yu Wang, Dong Yan, and Yuan Shen. Reward-robust rlhf in llms. *arXiv preprint arXiv:2409.15360*, 2024.

Gilad Yehudai and Ohad Shamir. On the power and limitations of random features for understanding neural networks. *Advances in neural information processing systems*, 32, 2019.

Zhuohao Yu, Chang Gao, Wenjin Yao, Yidong Wang, Zhengran Zeng, Wei Ye, Jindong Wang, Yue Zhang, and Shikun Zhang. Freeeval: A modular framework for trustworthy and efficient evaluation of large language models. *arXiv preprint arXiv:2404.06003*, 2024a.

Zhuohao Yu, Weizheng Gu, Yidong Wang, Xingru Jiang, Zhengran Zeng, Jindong Wang, Wei Ye, and Shikun Zhang. Reasoning through execution: Unifying process and outcome rewards for code generation. *arXiv preprint arXiv:2412.15118*, 2024b.

Zhuohao Yu, Jiali Zeng, Weizheng Gu, Yidong Wang, Jindong Wang, Fandong Meng, Jie Zhou, Yue Zhang, Shikun Zhang, and Wei Ye. Rewardanything: Generalizable principle-following reward models. *arXiv preprint arXiv:2506.03637*, 2025.

Yuanzhao Zhai, Han Zhang, Yu Lei, Yue Yu, Kele Xu, Dawei Feng, Bo Ding, and Huaimin Wang. Uncertainty-penalized reinforcement learning from human feedback with diverse reward lora ensembles. *arXiv preprint arXiv:2401.00243*, 2023.

Jianfei Zhang, Jun Bai, Bei Li, Yanmeng Wang, Rumei Li, Chenghua Lin, and Wenge Rong. Disentangling preference representation and text generation for efficient individual preference alignment. *arXiv preprint arXiv:2412.20834*, 2024.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A THE USE OF LARGE LANGUAGE MODELS (LLMS)

LLMs were used assist refining the writing of this paper, including grammar correction, wording refinement, and formatting adjustments. We also use LLM agents to help with finding relevant work and implementing parts of our code. The use of AI tools does not affect the originality of the work or the authors' responsibility for the content.

B ADDITIONAL RELATED WORK

We now give a more detailed discussion of how the present paper relates to prior work. Our approach is situated at the intersection of three key areas of research: Best-of-N sampling with reward models, reward hacking and its mitigation attempts, and curiosity-driven exploration techniques.

Best-of-N Sampling and Reward Models. Best-of-N (BoN) sampling generates N candidate responses and selects the highest-scoring according to a reward model: $y^* = \arg\max_{y \in \mathcal{Y}_N} r(x, y)$ (Stiennon et al., 2020). This technique has proven effective for mathematical reasoning (Cobbe et al., 2021; Lightman et al., 2023) and competitive programming (Li et al., 2022) and the version of BoN with oracular rewards (called 'pass@k') is a standard evaluation metric for reasoning models (Dubey et al., 2024; Ouyang et al., 2022; Abdin et al., 2024; Hui et al., 2024; Lambert et al., 2024). Reward models are typically trained on tuples of preference data $(x, y_{\text{chosen}}, y_{\text{rejected}})$ consisting of a prompt x as well as preferred and dispreferred responses y_{chosen} and y_{rejected} using ranking losses. While a powerful paradigm, this training procedure introduces vulnerabilities: models must infer complex reasons for preferences, often leading them to rely on spurious correlations like response length, formatting patterns, or stylistic preferences (Liu et al., 2024b). High-quality reward models are also computationally expensive to train, requiring resources similar to base language models. This cost makes *ensemble approaches*, a common strategy for reducing vulnerabilities in ML systems, impractical, as organizations typically train only a single reward model per domain (Gao et al., 2023; Jinnai et al., 2024). Our work focuses on mitigating reward hacking in BoN sampling with a single reward model, without retraining or modifying the reward model itself.

Reward Hacking and Mitigation Attempts. Reward hacking occurs when optimizing against imperfect proxy rewards leads to high-scoring but low-quality outcomes—a manifestation of Goodhart's Law (Amodei et al., 2016; Skalse et al., 2022). Gao et al. (2023) observed a common trend in BoN sampling, where as N increases, performance first rises then falls. As demonstrated theoretically in Huang et al. (2025b), these two phases correspond to an initial phase (when N is small) and the learned reward $\hat{r}$ is an effective proxy for the true reward $r^\star$, leading to BoN succeeding, and a second phase where N grows so large so as to produce atypical responses on which $\hat{r}$ is no longer effective due to spurious correlations and poor coverage during training (Eisenstein et al., 2023; Liu et al., 2024b; Yu et al., 2025). Broadly, there have been two main approaches to mitigating this reward-hacking problem: *training-time* and *inference-time* approaches.

Training-time ensemble methods attempt to mitigate this by combining multiple reward models (Coste et al., 2023; Zhai et al., 2023; Ramé et al., 2024; Yan et al., 2024). However, Eisenstein et al. (2023) demonstrate that ensembles reduce but do not eliminate reward hacking. More critically, ensemble methods require multiple expensive reward sources, creating prohibitive costs for practical applications.

Inference-time approaches work with existing reward models and intervene in the sampling procedure itself. Of note, Jinnai et al. (2024) proposed an approach that involves regularizing the BoN selection with respect to a Wasserstein distance in some embedding space and demonstrated empirical improvement over BoN as N grows. Unfortunately, this approach requires computation to scale quadratically in N, making it somewhat impractical for larger N. Of greatest relevance to the present work is that of Huang et al. (2025b), who propose instead to use an information-theoretic divergence, the χ^2-divergence, to regularize the BoN selection. They demonstrate that this approach is statistically and computationally optimal under the assumption that the learned reward model is close to the true reward in expected squared error loss for responses sampled from the base model. While theoretically well-motivated and empirically effective at ensuring monotonicity in N, the approach is overly conservative in practice, leading to suboptimal performance when N is moderate.

Another inference-time approach is that of Khalaf et al. (2025), who consider a KL-regularized BoN procedure as well as a computationally efficient approximation using the Poisson distribution. Unfortunately, their method exhibits limited improvement over standard BoN, likely due to the fact that BoN is already effectively KL-regularized unless N is growing exponentially. Unlike these works, our approach does not seek to provide *distributional* regularization but instead instantiates pessimism through the *reward estimates* themselves. Thus, our approach is able to leverage the fact that reward models may be imperfect in ways that are not information-theoretically 'worst-case' and adapt accordingly.

Curiosity-Driven Exploration and Random Network Distillation. Curiosity-driven exploration in reinforcement learning uses prediction error to quantify novelty of given states (Pathak et al., 2017; Sun et al., 2025; Li & Gajane, 2023). The Intrinsic Curiosity Module (ICM) (Pathak et al., 2017) uses prediction error $\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2^2$ as a curiosity signal, while Random Network Distillation (RND) (Burda et al., 2018) employs a simpler approach with two networks: a fixed target network $T(s)$ and a trainable predictor $P(s)$ that minimizes $\|P(s) - T(s)\|_2^2$. In both cases, the supervised learning error serves as a proxy for OOD detection, with high error indicating states that are far outside the distribution (and thus worth exploring from the perspective of online RL). While these methods were originally developed for exploration, they can instead be instantiated to prevent a policy from moving to far out of the distribution of observed states. While there has been some debate as to the efficacy of these approaches for OOD detection and pessimistic learning (Rezaeifar et al., 2022; Ciosek et al., 2019; Nikulin et al., 2023), our results suggest that when properly implemented, such an approach can be effective in language modeling.

C Theory

In this section, we give formal statements for and prove our theoretical results. We begin in Appendix C.1 for formalizing the setting we consider in which pessimism can help mitigate reward hacking in Best-of-N sampling. We emphasize that this setting is a simplified abstraction intended to cleanly showcase the benefits of pessimism and help develop intuition for our approach and is not intended to represent a realistic model of language or reward modeling. We then continue in Appendix C.2 to prove general results on the performance of BoN and pessimistic selection in our setting. We proceed in Appendix C.3 to introduce a simplified model of our proposed caution approach, motivated by Random Network Distillation (RND) (Burda et al., 2018). We prove in simplified linear and two-layer ReLU settings that this approach can be used to instantiate pessimism under an idealized optimization model. Finally, we combine these results in Appendix C.4 to prove our main theorem that our proposed approach improves over BoN sampling in the model we consider.

C.1 Formal Setting

In order to formalize the setting we consider, we suppose that a learner is given access to a language model π that maps a prompt x to a distribution over responses y. We will further identify these prompt-response pairs with their *embeddings* in some linear feature space $\mathbb{R}^d$ and, for the sake of simplicity, consider results on a per-prompt basis. Thus we will assume a fixed prompt x and write π for $\pi(x)$. We will further suppose that there exists a *ground truth reward function* that is linear in the embedded features, i.e.,

$$r^\star(y) = \langle \theta^\star, y \rangle .$$

Such an abstraction is partially justified by the fact that many modern reward models are linear layers on top of pre-trained Language Models (Lambert et al., 2024; Liu et al., 2024a; Wang et al., 2024a;b) and we thus simply directly associate the concatenation of prompt and response with its embedding in the final layer of the LM.

The goal of the learning problem is similar to that in Huang et al. (2025b): given access to $y_1, \ldots, y_n \sim \pi$ sampled independently, as well as some imperfect proxy reward function $\hat{r} : \mathbb{R}^d \to \mathbb{R}$ (e.g., a learned reward model), we wish to select a response $y_{\hat{i}}$ with $\hat{i} \in [n]$ such that $r^\star(y_{\hat{i}})$ is as large as possible. Note that if $\hat{r}$ were in fact a perfect reward model, so that $\hat{r} = r^\star$, then the optimal strategy would be to simply select $y_{\hat{i}}$ with $\hat{i} = \operatorname{argmax}_{i \in [n]} \hat{r}(y_i)$, which is the popular

Best-of-N (BoN) strategy. However, as discussed in the main text, this strategy can fail when $\hat{r}$ is a poor approximation to $r^\star$ due to reward-hacking.

Unlike Huang et al. (2025b), wherein the authors suppose that $\hat{r}$ and $r^\star$ are close in expected squared error under π, we will instead suppose that $\hat{r}$ and $r^\star$ agree on a low dimensional subspace of $\mathbb{R}^d$ but may be arbitrarily different on the orthogonal complement. More precisely, suppose that there exists a linear subspace $V \subset \mathbb{R}^d$ such that $\dim(V) = k \ll d$ and $r^\star(y) = \hat{r}(y)$ for all $y \in V$. We will further make the assumption that $r^\star$ and $\hat{r}$ are linear functions, i.e., $\hat{r}(y) = \left\langle \hat{\theta}, y \right\rangle$ and $r^\star(y) = \langle \theta^\star, y \rangle$ for some $\hat{\theta}, \theta^\star \in \mathbb{R}^d$. To aid our analysis, we will make the simplifying assumption that $\theta^\star \in V$, i.e. $\mathrm{proj}_{V^\perp} \theta^\star = 0$, representing the fact that the true reward is a function only of certain linear features; the error between the learned reward $\hat{r}$ and the groundtruth $r^\star$ is thus restricted to OOD effects, conceptualized as the orthogonal complement $V^\perp$ of V.

Finally, in order to make the analysis tractable, we will assume that π is a centred Gaussian distribution with covariance $\Sigma \in \mathbb{R}^{d \times d}$, i.e., $\pi = \mathcal{N}(0, \Sigma)$. Note that Σ may have full rank d and thus π may have full support on $\mathbb{R}^d$, which ensures that $\hat{r} \neq r^\star$ on $V^\perp$, the orthogonal complement of V.

C.2 Pessimistic Selection and Best-of-N Sampling

In this section we provide two key bounds on the performance of Best-of-N sampling and pessimistic selection, which will demonstrate the benefits of pessimism in our setting. Before doing so, we state and prove a simple bound on the best possible performance that can be achieved by any selection strategy.

Proposition 1. *Let $y_1, \ldots, y_N \sim \pi$ be independent samples from π. Then it holds that*

$$\mathbb{E}\left[\max_{i \in [N]} r^\star(y_i) | \theta^\star\right] = \left\|\Sigma^{1/2}\theta^\star\right\| \cdot M_n,$$

where $M_N = \mathbb{E}\left[\max_{i \in [N]} Z_i\right]$ and $Z_1, \ldots, Z_N$ are i.i.d. $\mathcal{N}(0,1)$ random variables. Moreover, it holds that

$$\sqrt{2\log(N)} - o(1) \leq M_N \leq \sqrt{2\log(N)}.$$

Proof. The second claim is a classical fact about the maximum of Gaussians (Wainwright, 2019; Vershynin, 2018), thus it suffices to prove the first statement. Note that

$$r^\star(y) = \langle \theta^\star, y \rangle \sim \mathcal{N}(0, \left\|\Sigma^{1/2}\theta^\star\right\|^2).$$

Thus $r^\star(y) \overset{d}{=} \left\|\Sigma^{1/2}\theta^\star\right\| Z$ for $Z \sim \mathcal{N}(0,1)$. The result then follows by positive homogeneity of the maximum and expectation. □

We now provide a lower bound on the performance of BoN in the setting we consider

Proposition 2. *Let $y_1, \ldots, y_N \sim \pi$ be independent samples from π and let $\hat{i} = \mathrm{argmax}_{i \in [N]} \hat{r}(y_i)$ for $\hat{r}(y) = \left\langle \hat{\theta}, y \right\rangle$. Let $i^\star = \mathrm{argmax}_{i \in [N]} r^\star(y_i)$ for $r^\star(y) = \langle \theta^\star, y \rangle$. Then it holds that*

$$\begin{aligned}
\mathbb{E}\left[r^\star(y_{i^\star}) - r^\star(y_{\hat{i}}) | \hat{\theta}, \theta^\star\right] &= \mathbb{E}\left[r^\star(y_{i^\star})\right] \left(1 - \frac{1}{\sqrt{1 + \frac{\left\|\Sigma^{1/2}\mathrm{proj}_{V^\perp}\hat{\theta}\right\|^2}{\left\|\Sigma^{1/2}\theta^\star\right\|^2}}}\right) \\
&= \left(1 - \frac{1}{\sqrt{1 + \frac{\left\|\Sigma^{1/2}\mathrm{proj}_{V^\perp}\hat{\theta}\right\|^2}{\left\|\Sigma^{1/2}\theta^\star\right\|^2}}}\right) \cdot \left\|\Sigma^{1/2}\theta^\star\right\| \cdot M_N.
\end{aligned}$$

Proof. As we have already computed the expectation of $r^\star(y_{i^\star})$ in Proposition 1, it suffices to compute the conditional expectation of $r^\star(y_{\hat{i}})$. Note that because, conditioned on $\hat{\theta}$ and $\theta^\star$, $\hat{r}(y)$ and $r^\star(y)$ are jointly Gaussian, it holds that

$$\begin{aligned}
\mathbb{E}\left[r^\star(y)|\hat{r}(y)\right] &= \frac{\mathrm{Cov}(\hat{r}(y), r^\star(y))}{\mathrm{Var}(\hat{r}(y))} \cdot \hat{r}(y) \\
&= \frac{\left\langle \theta^\star, \Sigma\hat{\theta} \right\rangle}{\left\|\Sigma^{1/2}\hat{\theta}\right\|^2} \cdot \hat{r}(y) \\
&= \frac{\left\|\Sigma^{1/2}\theta^\star\right\|^2}{\left\|\Sigma^{1/2}\hat{\theta}\right\|^2} \cdot \hat{r}(y) \\
&= \frac{\left\|\Sigma^{1/2}\theta^\star\right\|^2}{\left\|\Sigma^{1/2}\mathrm{proj}_V\hat{\theta}\right\|^2 + \left\|\Sigma^{1/2}\mathrm{proj}_{V^\perp}\hat{\theta}\right\|^2} \cdot \hat{r}(y) \\
&= \frac{\left\|\Sigma^{1/2}\theta^\star\right\|^2}{\left\|\Sigma^{1/2}\theta^\star\right\|^2 + \left\|\Sigma^{1/2}\mathrm{proj}_{V^\perp}\hat{\theta}\right\|^2} \cdot \hat{r}(y).
\end{aligned}$$

By the same computation as in Proposition 1, it holds that

$$\mathbb{E}\left[\hat{r}(y_{\hat{i}})|\theta^\star, \hat{\theta}\right] = \left\|\Sigma^{1/2}\hat{\theta}\right\| \cdot M_N = \sqrt{\left\|\Sigma^{1/2}\theta^\star\right\|^2 + \left\|\Sigma^{1/2}\mathrm{proj}_{V^\perp}\hat{\theta}\right\|^2} \cdot M_N.$$

The result follows by plugging in and rearranging. □

We now show that with pessimism instantiated correctly, we can strictly improve on the performance of BoN.

Proposition 3. *Let $y_1, \ldots, y_N \sim \pi$ be independent samples from π and let $\alpha : \mathbb{R}^d \to \mathbb{R}_+$ be a function satisfying*

$$(1-c)\left\|\mathrm{proj}_{V^\perp} y\right\| - \varepsilon \le \alpha(y) \le \left\|\mathrm{proj}_{V^\perp} y\right\| + \varepsilon$$

for some $c \in (0,1)$ and $\varepsilon > 0$. Let

$$i_{\mathrm{pess}} = \operatorname*{argmax}_{i \in [N]} \hat{r}(y_i) - \lambda \cdot \alpha(y_i), \qquad \lambda \ge \frac{\left\|\mathrm{proj}_{V^\perp}\hat{\theta}\right\|}{1-c}.$$

Then it holds that

$$\mathbb{E}\left[r^\star(y_{i^\star}) - r^\star(y_{i_{\mathrm{pess}}})\right] \le \lambda\left(\sqrt{\frac{2}{\pi} \cdot \mathrm{Tr}\left(\Sigma\mathrm{proj}_{V^\perp}\right)} + 2\varepsilon\right).$$

Proof. Let

$$\hat{r}_{\mathrm{LCB}}(y) = \hat{r}(y) - \lambda \cdot \alpha(y).$$

We claim that with the assumption on λ and α, it holds that $\hat{r}_{\mathrm{LCB}}(y) \le r^\star(y) + \lambda \cdot \varepsilon$ for all $y \in \mathbb{R}^d$. Indeed, note that

$$\begin{aligned}
r^\star(y) - \hat{r}_{\mathrm{LCB}}(y) &= \left\langle \theta^\star - \hat{\theta}, y \right\rangle + \lambda \cdot \alpha(y) \\
&= \left\langle \mathrm{proj}_{V^\perp}\hat{\theta}, y \right\rangle + \lambda \cdot \alpha(y) \\
&\ge -\left\|\mathrm{proj}_{V^\perp}\hat{\theta}\right\| \cdot \left\|\mathrm{proj}_{V^\perp} y\right\| + \lambda \cdot \left((1-c)\left\|\mathrm{proj}_{V^\perp} y\right\| - \varepsilon\right) \\
&\ge -\lambda\varepsilon,
\end{aligned}$$

where the first inequality is by Cauchy-Schwarz.

Now observe that

$$r^\star(y_{i^\star}) - r^\star(y_{i_{\text{pess}}}) = [r^\star(y_{i^\star}) - \hat{r}_{\text{LCB}}(y_{i^\star})] + \left[\hat{r}_{\text{LCB}}(y_{i^\star}) - \hat{r}_{\text{LCB}}(y_{i_{\text{pess}}})\right] + \left[\hat{r}_{\text{LCB}}(y_{i_{\text{pess}}}) - r^\star(y_{i_{\text{pess}}})\right]$$
$$\leq r^\star(y_{i^\star}) - \hat{r}_{\text{LCB}}(y_{i^\star}) + \lambda\varepsilon,$$

where the second group of terms is non-positive by definition of i_{pess} and the last group of terms is non-positive by the claim above. To conclude, we observe that

$$r^\star(y_{i^\star}) - \hat{r}_{\text{LCB}}(y_{i^\star}) = r^\star(y_{i^\star}) - \hat{r}(y_{i^\star}) + \lambda \cdot \alpha(y_{i^\star})$$
$$= \left\langle \text{proj}_{V^\perp}\hat{\theta}, y_{i^\star} \right\rangle + \lambda \cdot \left(\|\text{proj}_{V^\perp} y_{i^\star}\| + \varepsilon\right).$$

We now observe that as random variables, $i^\star$ is independent of the set $\{\text{proj}_{V^\perp} y_i | i \in [N]\}$ by Gaussian orthogonality and the fact that $\text{proj}_{V^\perp}\theta^\star = 0$. Thus it holds first that

$$\mathbb{E}\left[\left\langle \text{proj}_{V^\perp}\hat{\theta}, y_{i^\star} \right\rangle | \hat{\theta}, \theta^\star\right] = \sum_{i=1}^N \mathbb{E}\left[\left\langle \text{proj}_{V^\perp}\hat{\theta}, y_i \right\rangle | \hat{\theta}, \theta^\star, \{i^\star = i\}\right] \cdot \Pr[i^\star = i | \hat{\theta}, \theta^\star]$$
$$= \sum_{i=1}^N \mathbb{E}\left[\left\langle \text{proj}_{V^\perp}\hat{\theta}, y_i \right\rangle | \hat{\theta}, \theta^\star\right] \cdot \Pr[i^\star = i | \hat{\theta}, \theta^\star]$$
$$= 0,$$

where the last equality follows from the fact that y_i is centred. For the second term, observe similarly that

$$\mathbb{E}\left[\|\text{proj}_{V^\perp} y_{i^\star}\| | \hat{\theta}, \theta^\star\right] = \sum_{i=1}^N \mathbb{E}\left[\|\text{proj}_{V^\perp} y_i\| | \hat{\theta}, \theta^\star, \{i^\star = i\}\right] \cdot \Pr[i^\star = i | \hat{\theta}, \theta^\star]$$
$$= \sum_{i=1}^N \mathbb{E}\left[\|\text{proj}_{V^\perp} y_i\| | \hat{\theta}, \theta^\star\right] \cdot \Pr[i^\star = i | \hat{\theta}, \theta^\star]$$
$$= \mathbb{E}\left[\|\text{proj}_{V^\perp} y_1\| | \hat{\theta}, \theta^\star\right] = \sqrt{\frac{2}{\pi}} \cdot \left\|\Sigma^{1/2}\text{proj}_{V^\perp}\right\|_{\text{F}}.$$

The result follows by combining the above. □

Critically, the bound in Proposition 3 does not depend on N and so, as long as $r^\star(y_{i^\star})$ grows with N, the pessimistic algorithm will eventually outperform BoN. We now show that guarantees on α can be obtained through Random Network Distillation.

C.3 Achieving OOD Detection with Caution

Above we isolated the key role that $\|\text{proj}_{V^\perp} y\|$ plays in the failure of the greedy algorithm. While in the linear setting, projection to V results in the optimal algorithm, it is impractical in general settings, where V is unknown. Here we demonstrate that two simplified models of caution, both inspired by Random Network Distillation (RND) (Burda et al., 2018) can be used to approximate projection to V and thus yield a pessimistic algorithm that avoids the failure modes of the greedy algorithm. Recall that RND involves training a student network $f_{\hat{w}} : \mathbb{R}^d \to \mathbb{R}^m$ to predict a teacher network $f_{w^\star} : \mathbb{R}^d \to \mathbb{R}^m$ on samples from the the explored distribution, which in this case is supported on V. The error $\|f_{\hat{w}}(y) - f_{w^\star}(y)\|^2$ is then used as a measure of how far out of distribution y is. We will consider two simplified models of RND. The first is a linear model, where $f_W(y) = Wy$ for $w \in \mathbb{R}^{m \times d}$. The second is a two-layer ReLU network of the form

$$f(y) = \frac{1}{T}\sum_{\ell=1}^T f_{W_\ell}(y),$$

where

$$f_W(y) = \frac{1}{m} \cdot U\text{ReLU}(Wy), \qquad \text{ReLU}(u) = \max\{u, 0\}, \tag{1}$$

and $W = (w_1, \ldots, w_m) \in \mathbb{R}^{m \times d}$, $U \in \mathbb{R}^{m \times m}$, and ReLU is applied coordinate wise. In both cases, we will suppose that the teacher network is fixed with a randomly initialized parameter and the student network is trained to minimize squared error on samples from V. We abstract the minimization in the following definition.

Definition 1. Given a distribution $\mathcal{D}$ on $\mathbb{R}^d$ and an initial parameter W^0, we say that $\widehat{W}$ is trained with *gradient based methods* with samples from $\mathcal{D}$ if $\widehat{W} - W^0$ is in the linear span of $\{\nabla_W f_W(y)\}_{y \in \mathrm{supp}(\mathcal{D})}$.

This definition abstracts the precise optimization method used to train the student network, but captures the key property that the final parameter is obtained by following gradients of the squared error loss on samples from $\mathcal{D}$. Note that this includes gradient descent and its variants as well as more sophisticated approaches involving preconditioning, momentum, and continuous time limits.

We begin with the simpler linear case.

Proposition 4. *Suppose that $W^\star, W_0 \in \mathbb{R}^{m \times d}$ are Gaussian random matrices with independent $\mathcal{N}(0, 1/m)$ entries. Let $\widehat{W}$ denote the minimizer of the expected squared error on samples from a distribution with support V attained through gradient based methods from samples y supported in V initialized at W_0 in the sense of Definition 1. Then with probability at least $1 - \delta$ over the choice of $W^\star$ and W_0, it holds that for all $y \in \mathbb{R}^d$,*

$$\left(1 - C\sqrt{\frac{k + \log(1/\delta)}{m}}\right)^2 \|\mathrm{proj}_{V^\perp} y\|^2 \leq \left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2 \leq \left(1 + C\sqrt{\frac{k + \log(1/\delta)}{m}}\right)^2 \|\mathrm{proj}_{V^\perp} y\|^2 .$$

In particular, if as long as $m \gg k$, then $\left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2$ is a good approximation to $\|\mathrm{proj}_{V^\perp} y\|^2$.

Proof. We first observe that $\widehat{W}\mathrm{proj}_{V^\perp} = W_0 \mathrm{proj}_{V^\perp}$ since the training data is supported on V. Indeed, the gradient of the squared error loss on a sample $y \in V$ is given by

$$\nabla_W \|Wy - W^\star y\|^2 = 2(Wy - W^\star y)y^\top = 2(W - W^\star)(\mathrm{proj}_V y)y^\top \mathrm{proj}_V^\top,$$

where we used the fact that $y = \mathrm{proj}_V y$ for all $y \in V$. Thus $\nabla_W \|Wy - W^\star y\|^2 \mathrm{proj}_{V^\perp} = 0$ for all $y \in V$ and the claim follows. Moreover, it is immediate that $\widehat{W}\mathrm{proj}_V = W^\star \mathrm{proj}_V$ since the loss is minimized at $\widehat{W}$ by strong convexity of the loss function. Thus it holds that

$$\left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2 = \left\|(\widehat{W} - W^\star)y\right\|^2 = \left\|(\widehat{W} - W^\star)\mathrm{proj}_{V^\perp} y\right\|^2 = \|(W_0 - W^\star)\mathrm{proj}_{V^\perp} y\|^2 .$$

Letting $Z = W_0 - W^\star$, we see that Z is a Gaussian random matrix with independent $\mathcal{N}(0, 2/m)$ entries. We may now apply standard results on the singular values of Gaussian random matrices (cf. e.g. Vershynin (2018, Theorem 4.6.1)) to observe that there is some constant C such that with probability at least $1 - \delta$,

$$1 - C\sqrt{\frac{k + \log(1/\delta)}{m}} \leq \lambda_{\min}(Z) \leq \lambda_{\max}(Z) \leq 1 + C\sqrt{\frac{k + \log(1/\delta)}{m}}.$$

This suffices to prove the first statement. The second statement follows immediately. □

Note that the above result shows that the RND error is a good approximation to $\|\mathrm{proj}_{V^\perp} y\|^2$ uniformly over all $y \in \mathbb{R}^d$ with high probability as long as the embedding dimension is sufficiently large relative to the intrinsic dimension of the explored distribution. While this is a nice first step, the lack of flexibility of linear functions is a significant limitation. We now proceed to the hidden layer ReLU network case.

Instead of assuming that $f_W(y)$ is a linear function, we now suppose that $f_W(y)$ is a two-layer ReLU network of the form given in (1). For the sake of simplicity, we assume that only the weights W are trained and that the second layer weights $u_i \sim \mathcal{N}(0, 1/m)$ are fixed. Note that, while practically unrealistic, the assumption that only a single layer is trained is common in the study of deep learning (cf. e.g. (Jacot et al., 2018; Yehudai & Shamir, 2019; Song et al., 2018) and the references therein)

and is motivated by the utility of random features (Rahimi & Recht, 2007). Our analysis is inspired by that of Melamed et al. (2023), who consider a similar model of neural networks, but for the very different aim of investigating adversarial robustness. We show that as long as the number of hidden features m is sufficiently large relative to $\|y\|$, then the RND error is again a good approximation to $\|\mathrm{proj}_{V^\perp} y\|^2$ uniformly over all $y \in \mathbb{R}^d$ with bounded norm with high probability.

Theorem 2. *Let $W^\star, W^0 \in \mathbb{R}^{m\times d}$ be Gaussian random matrices with independent $\mathcal{N}(0,1)$ entries with rows $w_i^\star$ and w_i^0 respectively. Let $u_i^\star, u_0 \sim \mathcal{N}(0,1)$ and let $f_{W^\star}$ and f_{W^0} be the corresponding ReLU networks as in (1). Suppose (i) that $\widehat{W}$ is obtained through a gradient-based method from a distribution with support V initialized at W^0 as in Definition 1 and (ii) that $f_{\widehat{W}}(y) = f_{W^\star}(y)$ for all $y \in V$. Then it holds with probability at least $1-\delta$ over the choice of $W^\star, W^0, u_i, u_0$ that simultaneously for all $y \in \mathbb{R}^d$,*

$$c\,\|\mathrm{proj}_{V^\perp} y\|^2 - C\,\|y\|^2 \sqrt{\frac{d\log(dm/\delta)}{Tm}} - C\,\|y\|^2 \frac{d\log(dm/\delta)}{Tm} \le \left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2$$

and

$$\left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2 \le \|\mathrm{proj}_{V^\perp} y\|^2 + C\,\|y\|^2 \sqrt{\frac{d\log(dm/\delta)}{Tm}} + C\,\|y\|^2 \frac{d\log(dm/\delta)}{Tm},$$

In particular, for $Tm \gg \|y\|^4$ and up to constants, $\left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2 \asymp \|\mathrm{proj}_{V^\perp} y\|^2$ with high probability.

Proof. We first prove the result for $T = 1$. This step rests four lemmata. We first show in Lemma 1 the key property that, due to the training data being supported on V and the fact that we are only training the first layer weights, it holds that $\widehat{W}\mathrm{proj}_{V^\perp} = W^0\mathrm{proj}_{V^\perp}$, i.e., weights in the orthogonal complement of V are never changed during training. By the assumption that we have trained to convergence, then, it holds that $W^\star\mathrm{proj}_V = \widehat{W}\mathrm{proj}_V$. We then use prove in Lemma 2 a decomposition of the RND error that rests on the precise funcitonal form of the ReLU network. Using this decomposition, we are able to provide bounds on the mean and concentration of the RND error in Lemma 3 and Lemma 4 in the special case that $\|y\| = 1$. The result then follows by combining these lemmata with the positive homogeneity property of ReLU networks: for any $r > 0$, it holds that $f_W(ry) = rf_W(y)$ for all $y \in \mathbb{R}^d$ and all $W \in \mathbb{R}^{m\times d}$.

Now that the result is proved for $T = 1$, the general case follows by tensorization across the independent W_ℓ. □

Lemma 1. *Let $f_{W^\star}$, f_{W^0}, and $f_{\widehat{W}}$ be as in Theorem 2. Then it holds that $W^0\mathrm{proj}_{V^\perp} = \widehat{W}\mathrm{proj}_{V^\perp}$.*

Proof. By Definition 1 suffices to show that for any $y \in V$, $(\nabla_W f_W(y))\,\mathrm{proj}_{V^\perp} = 0$. To see this, observe that

$$\nabla_W f_W(y)_i = \frac{1}{m}\sum_{j=1}^m u_{ij}\mathbb{I}\left[\langle w_j, y\rangle > 0\right] y.$$

Because $y\mathrm{proj}_{V^\perp} = 0$ for all $y \in V$, it follows that $\nabla_W f_W(y)\mathrm{proj}_{V^\perp} = 0$. The result follows. □

Lemma 2. *Let $f_{W^\star}$, f_{W^0}, and $f_{\widehat{W}}$ be as in Theorem 2. Then with probability at least $1-\delta$ over the choice of $W^\star, W^0, u_i, u_0$, it holds that for all $y \in \mathbb{R}^d$,*

$$\begin{aligned}(f_{\widehat{W}}(y) - f_{W^\star}(y))_j = \frac{1}{m}\left(\sum_{i=1}^m \langle u_{ji}^0 w_i^0, \mathrm{proj}_{V^\perp} y\rangle \int_0^1 \mathbb{I}\left[\left\langle \mathrm{proj}_V w_i^\star + \mathrm{proj}_{V^\perp} w_i^0, \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\right\rangle > 0\right] dt\right) \\ - \frac{1}{m}\left(\sum_{i=1}^m \langle u_{ji}^\star w_i^\star, \mathrm{proj}_{V^\perp} y\rangle \int_0^1 \mathbb{I}\left[\langle w_i^\star, \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\rangle > 0\right] dt\right).\end{aligned}$$

Proof. By the fundamental theorem of calculus, we have that

$$m(f_{\widehat{W}}(y) - f_{\widehat{W}}(\mathrm{proj}_V y))_j = \int_0^1 \left\langle \nabla_y f_{\widehat{W}}(\mathrm{proj}_V y + t(\mathrm{proj}_{V^\perp} y - \mathrm{proj}_V y))_j, y - \mathrm{proj}_V y \right\rangle dt$$
$$= \int_0^1 \left\langle \nabla_y f_{\widehat{W}}(\mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y)_j, \mathrm{proj}_{V^\perp} y \right\rangle dt.$$

For any W, it holds that

$$m\nabla_y f_W(y)_j = \sum_{i=1}^m u_{ji} w_i \mathbb{I}\left[\langle w_i, y\rangle > 0\right].$$

Thus by the linearity of the integral and Lemma 1, it holds that

$$m(f_{\widehat{W}}(y) - f_{\widehat{W}}(\mathrm{proj}_V y))_j = \sum_{i=1}^m \langle u_{ji}\hat{w}_i, \mathrm{proj}_{V^\perp} y\rangle \int_0^1 \mathbb{I}\left[\langle \hat{w}_i, \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\rangle > 0\right] dt$$
$$= \sum_{i=1}^m \left\langle u_{ji} w_i^0, \mathrm{proj}_{V^\perp} y\right\rangle \int_0^1 \mathbb{I}\left[\left\langle \mathrm{proj}_V w_i^\star + \mathrm{proj}_{V^\perp} w_i^0, \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\right\rangle > 0\right] dt,$$

where we used the fact that $\widehat{W} = W^\star \mathrm{proj}_V + W^0 \mathrm{proj}_{V^\perp}$ by Lemma 1. A similar, but simpler, argument applies to $f_{W^\star}$. We now use the fact that $\widehat{W}\mathrm{proj}_V = W^\star \mathrm{proj}_V$ to observe that

$$f_{\widehat{W}}(y) - f_{W^\star}(y) = f_{\widehat{W}}(y) - f_{\widehat{W}}(\mathrm{proj}_V y) - (f_{W^\star}(y) - f_{W^\star}(\mathrm{proj}_V y))$$

and the result follows. □

Lemma 3. *Let $f_{W^\star}$, f_{W^0}, and $f_{\widehat{W}}$ be as in Theorem 2. Then it holds that*

$$\frac{\|\mathrm{proj}_{V^\perp} y\|^2}{4} \le \mathbb{E}\left[\left\|f_{\widehat{W}}(y) - f_{W^\star}(y)\right\|^2\right] \le \|\mathrm{proj}_{V^\perp} y\|^2.$$

Proof. Let

$$g(w, y) = \langle w, \mathrm{proj}_{V^\perp} y\rangle \int_0^1 \mathbb{I}\left[\langle w, \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\rangle > 0\right] dt \tag{2}$$

and let

$$g'(w, w', y) = \langle w', \mathrm{proj}_{V^\perp} y\rangle \int_0^1 \mathbb{I}\left[\langle \mathrm{proj}_V w + \mathrm{proj}_{V^\perp} w', \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\rangle > 0\right] dt. \tag{3}$$

By Lemma 2 it holds that

$$\mathbb{E}\left[\left(f_{\widehat{W}}(y) - f_{W^\star}(y)\right)_j^2\right] = \frac{1}{m^2}\mathbb{E}\left[\left(\sum_{i=1}^m u_{ji}^0 g'(w_i^\star, w_i^0, y) - u_{ji}^\star g(w_i^\star, y)\right)^2\right]$$
$$= \frac{2}{m}\mathbb{E}\left[g(w_1^\star, y)^2\right],$$

because the u_{ji}^0 and $u_{ji}^\star$ are independent and have variance 1 and $g'(w_i^\star, w_i^0, y)$ and $g(w_i^\star, y)$ are independent of u_{ji}^0 and $u_{ji}^\star$ and identically distributed. Note now that

$$g(w_i^\star, y)^2 \le \langle \mathrm{proj}_{V^\perp} y, \mathrm{proj}_{V^\perp} y\rangle^2$$

and thus has expectation at most $\|\mathrm{proj}_{V^\perp} y\|^2$. For the lower bound, observe that

$$g(w_i^\star, y)^2 = \int_0^1 \int_0^1 \langle w_i^\star, \mathrm{proj}_{V^\perp} y\rangle^2 \mathbb{I}\left[\langle w_i^\star, \mathrm{proj}_V y + t\mathrm{proj}_{V^\perp} y\rangle > 0\right] \mathbb{I}\left[\langle w_i^\star, \mathrm{proj}_V y + s\mathrm{proj}_{V^\perp} y\rangle > 0\right] dsdt$$
$$\ge \int_0^1 \int_0^1 \langle w_i^\star, \mathrm{proj}_{V^\perp} y\rangle^2 \mathbb{I}\left[\langle w_i^\star, \mathrm{proj}_V y\rangle > 0\right] \mathbb{I}\left[\langle w_i^\star, \mathrm{proj}_{V^\perp} y\rangle > 0\right] dsdt$$
$$\ge \frac{\langle w_i^\star, \mathrm{proj}_{V^\perp} y\rangle^2}{4}.$$

Thus

$$\frac{\|\mathrm{proj}_{V^\perp} y\|^2}{4m} = \frac{\langle w_i^\star, \mathrm{proj}_{V^\perp} y\rangle^2}{4m} \le \mathbb{E}\left[(f_{\widehat{W}}(y) - f_{W^\star}(y))_j^2\right] \le \frac{\mathbb{E}\left[\langle w_i^\star, \mathrm{proj}_{V^\perp} y\rangle^2\right]}{m} = \frac{\|\mathrm{proj}_{V^\perp} y\|^2}{m}.$$

Summing over j gives the result. □

Lemma 4. *Let $f_{\widehat{W}}$, f_{W^0}, and $f_{W^\star}$ be as in Theorem 2. Then with probability at least $1-\delta$ over the choice of $W^\star, W^0, u_i, u_0$, it holds that uniformly in y for $\|y\| = 1$,*

$$\left| \left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|^2 - \mathbb{E}\left[\left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|^2 \right] \right| \le C \left(\sqrt{\frac{d \log(dm/\delta)}{m}} + \frac{d \log(dm/\delta)}{m} \right).$$

Proof. Begin by noting that by positive homogeneity, it suffices to set $\|y\| = 1$. Continuing with the notation introduced in (2) and (3) we see that, conditional on $w_i^\star, w_i^0$, it holds that $(f_{\widehat{W}}(y) - f_{W^\star}(y))_j$ are independent centred Gaussians with variancse given by

$$\mathrm{Var}((f_{\widehat{W}}(y) - f_{W^\star}(y))_j | w_i^\star, w_i^0) = \frac{1}{m^2} \sum_{i=1}^m g'(w_i^\star, w_i^0, y)^2 + g(w_i^\star, y)^2.$$

Thus by standard bounds on the concentration of norm of Gaussian vectors, it holds that with probability at least $1-\delta$,

$$\begin{aligned}
&\left| \left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|^2 - \frac{1}{m} \sum_{i=1}^m g'(w_i^\star, w_i^0, y)^2 + g(w_i^\star, y)^2 \right| \\
&\le C \sqrt{m \cdot \left(\frac{1}{m^2} \sum_{i=1}^m g'(w_i^\star, w_i^0, y)^2 + g(w_i^\star, y)^2 \right)^2 \log(1/\delta)} \\
&\quad + C \left(\frac{1}{m^2} \sum_{i=1}^m g'(w_i^\star, w_i^0, y)^2 + g(w_i^\star, y)^2 \right) \log(1/\delta).
\end{aligned}$$

Letting

$$G(w^\star, w^0, y) = \frac{1}{m} \sum_{i=1}^m g'(w_i^\star, w_i^0, y)^2 + g(w_i^\star, y)^2,$$

we see that

$$\left| \left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|^2 - G(w^\star, w^0, y) \right| \le C \cdot G(w^\star, w^0, y) \left(\sqrt{\frac{\log(1/\delta)}{m}} + \frac{\log(1/\delta)}{m} \right). \tag{4}$$

We now demonstrate that $G(w^\star, w^0, y)$ concentrates around its mean for fixed y. To do this, we will first observe that $g(w, y)$ and $g'(w, w', y)$ are identically distributed and thus it suffices to show concentration for $g(w, y)$ and apply a union bound. Indeed, we have that

$$g(w_i^\star, y)^2 \le \langle w_i^\star, \mathrm{proj}_{V^\perp} y \rangle^2$$

and thus has Orlicz ψ_1 norm at most $C \|\mathrm{proj}_{V^\perp} y\|^2$ for some constant C. Thus by standard concentration results for sums of independent subexponential random variables (cf. e.g. Vershynin (2018); Wainwright (2019)), it holds that with probability at least $1-\delta$,

$$\left| \frac{1}{m} \sum_{i=1}^m g(w_i^\star, y)^2 - \mathbb{E}\left[g(w_i^\star, y)^2 \right] \right| \le C \|\mathrm{proj}_{V^\perp} y\|^2 \left(\sqrt{\frac{\log(1/\delta)}{m}} + \frac{\log(1/\delta)}{m} \right).$$

Combining this argument with the triangle inequality and (4) along with the fact that $\|y\| = 1$ and Lemma 3 gives that with probability at least $1-\delta$,

$$\left| \left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|^2 - \mathbb{E}\left[\left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|^2 \right] \right| \le C \left(\sqrt{\frac{\log(1/\delta)}{m}} + \frac{\log(1/\delta)}{m} \right).$$

We now observe that by standard high probability bounds on the operator norms of Gaussian random matrices (cf. e.g. Vershynin (2018, Theorem 4.6.1)), it holds that with probability at least $1-\delta$, that $f_{\widehat{W}}$ and $f_{W^\star}$ are C-Lipschitz as is $\mathbb{E}\left[g(w_i^\star, y)^2\right]$ in y for $\|y\| = 1$. Thus by a standard covering argument on the unit sphere and a union bound, the result follows. □

C.4 Main Result

We now state our main result, which combines the analysis of pessimism in Appendix C.2 with the analysis of caution in Appendix C.3.

Theorem 3. *Let V be a k-dimensional subspace of $\mathbb{R}^d$ and let Σ be a positive semidefinite matrix such that $\mathrm{Tr}(\Sigma \mathrm{proj}_{V^\perp}) > 0$. Let $\{y_i\}_{i=1}^N$ be i.i.d. samples from $\mathcal{N}(0, \Sigma)$ and let $r^\star(y) = \langle \theta^\star, y \rangle$ for some $\theta^\star \in V$. Suppose that $\hat{\theta} \in \mathbb{R}^d$ such that $\mathrm{proj}_V \hat{\theta} = \theta^\star$ and let $\hat{r}(y) = \left\langle \hat{\theta}, y \right\rangle$. Let $\hat{r}_{\mathrm{LCB}}(y) = \hat{r}(y) - \lambda \cdot \alpha(y)$, for $\alpha(y) = \left\| f_{\widehat{W}}(y) - f_{W^\star}(y) \right\|$ with $f_W(y)$ being either the linear model considered in Proposition 4 or the one hidden layer ReLU network considered in Theorem 2. Let $\hat{i} = \mathrm{argmax}_{i \in [N]} \hat{r}_{\mathrm{LCB}}(y_i)$ and $i^\star = \mathrm{argmax}_{i \in [N]} r^\star(y_i)$. In the case that f_W is linear, as long as $m \gtrsim \frac{k(d-k)\left\|\mathrm{proj}_{V^\perp} \hat{\theta}\right\|^2}{\log(N)}$ and $\lambda = \Theta(\left\|\mathrm{proj}_{V^\perp} \hat{\theta}\right\|)$, it holds that*

$$\mathbb{E}\left[r^\star(y_{i_{\mathrm{pess}}}) - r^\star(y_{\hat{i}})\right] \gtrsim \sqrt{\log(N)}.$$

In the case that f_W is a one hidden layer ReLU network, as long as $Tm \gtrsim \frac{d(d-k)\left\|\mathrm{proj}_{V^\perp} \hat{\theta}\right\|^2}{\log(N)}$, the same holds with $\lambda = \Theta(\left\|\mathrm{proj}_{V^\perp} \hat{\theta}\right\|)$. Moreover, in both of these cases it holds that

$$\lim_{N \uparrow \infty} \frac{\mathbb{E}\left[r^\star(y_{i^\star}) - r^\star(y_{i_{\mathrm{pess}}})\right]}{\mathbb{E}\left[r^\star(y_{i^\star})\right]} = 0 < \lim_{N \uparrow \infty} \frac{\mathbb{E}\left[r^\star(y_{i^\star}) - r^\star(y_{\hat{i}})\right]}{\mathbb{E}\left[r^\star(y_{i^\star})\right]}.$$

Proof. The result follows immediately by combining Proposition 3 with Proposition 4 and Theorem 2 and observing that $\|\mathrm{proj}_{V^\perp} y\| \lesssim \sqrt{(d-k)\log(1/\delta)}$ with probability at least $1-\delta$ by standard concentration results for chi-squared random variables (cf. e.g. Wainwright (2019)). □

D Experimental Details

Hyperparameter Configuration. Table 4 provides a comprehensive overview of all hyperparameters used throughout our experimental evaluation. The configuration represents a careful balance between computational efficiency and model expressiveness, with key design choices motivated by our theoretical analysis and empirical ablations.

The RND architecture employs 10 layers for both target and predictor networks, significantly deeper than the default 4 layers, which we found provides better representation quality for uncertainty estimation. The RND weight $\lambda = 0.2$ represents a moderate pessimism strength that effectively mitigates reward hacking while preserving the benefits of reward-guided selection. Training hyperparameters including the reduced learning rate (1×10^{-5}) and extended training duration (5 epochs) ensure stable convergence of the predictor network on the GSM8K training distribution.

Architectural Ablations. For the architectural ablation study, we design different levels of network complexity and check their impact on the training objective, the reconstruction loss and also the final accuracies. Table 5 contains a comprehensive introduction of settings involved in our ablation studies.

Implementation Framework. Our experimental setup utilizes PyTorch 2.3.1 as the primary deep learning framework, with model inference accelerated through vLLM 0.10.0 and Transformers 4.55.1 for efficient large-scale language model deployment. For RND training, we extract representations from the first 10 layers of both predictor and target networks, employing a learning rate of 1×10^{-5} with 50 warmup steps across 5 training epochs. The RND models are trained on outputs generated by the backbone model using the GSM8K training set, and subsequently evaluated against the validation set of GSM8K as well as Math-500 and BigBench-Hard benchmarks to assess generalization capabilities.

Inference Configuration. All inference experiments maintain consistent hyperparameters with temperature set to 1.0 and maximum token limits of 500 for GSM8K and 1024 for Math-500 and BigBench-Hard evaluations. The increased token limit for harder datasets is necessary because these

benchmarks require significantly more reasoning tokens to avoid truncation before reaching a conclusion. Response generation uses vLLM for efficient parallel sampling across multiple candidates in Best-of-N evaluation.

E Additional Case Study

To gain deeper insights into the mechanisms underlying reward hacking and our caution-based mitigation, we analyze the correlation patterns between reward model scores and pessimism scores through detailed scatter plot visualizations. Each plot displays z-normalized scores for all responses to individual GSM8K problems, with green points representing correct responses and red points representing incorrect responses.

The scatter plots in Figure 5 reveal two critical failure modes of reward models that our approach successfully identifies. In the **high reward, low pessimism region** (upper-left quadrant), we observe responses that exemplify systematic reward hacking. These responses achieve high reward scores not through genuine correctness or adherence to task requirements, but by exploiting spurious correlations that the reward model learned during training. Crucially, these responses often **ignore fundamental formatting requirements** of the mathematical reasoning task, such as providing the final answer in the required format, yet still receive high rewards because the reward model prioritizes superficial indicators like verbosity, step-by-step presentation, or mathematical terminology over actual task compliance. This reveals that reward models can be systematically misled by responses that mimic the surface patterns of high-quality reasoning without delivering the essential components of a correct solution.

Conversely, the **low reward, high pessimism region** (lower-right quadrant) contains responses that follow proper formatting conventions and adhere closely to the expected task structure, yet receive low reward scores. This pattern illuminates a fundamental limitation of reward models: they function primarily as **distributional fitness measures** rather than objective quality assessors. These well-formatted responses are penalized not because they lack correctness or clarity, but because they deviate from the specific stylistic preferences and response patterns that dominated the reward model's training distribution. The reward model essentially measures how closely a response matches its learned notion of "preferred" responses rather than evaluating genuine task performance or adherence to explicit instructions.

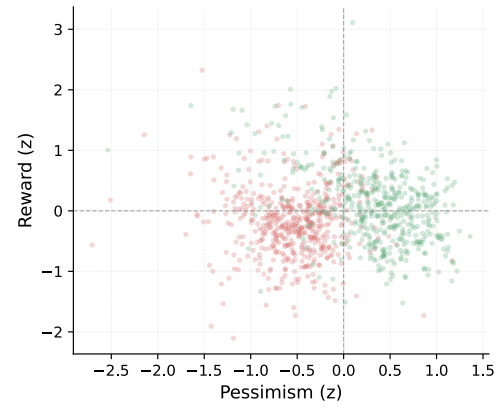
This analysis demonstrates that our caution mechanism successfully identifies both forms of reward model failure: it flags spurious high-reward responses that exploit correlational biases while recognizing genuinely task-compliant responses that happen to fall outside the reward model's narrow preference distribution. The results underscore that effective reward hacking mitigation requires moving beyond simple score-based selection toward distributional awareness that can distinguish between genuine quality and superficial pattern matching.

F Detailed Results for Ablation Studies

This section complements Table 2 with full scaling curves. We sweep pessimism strength $\lambda \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ and vary the Best-of-N budget over $N \in \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512\}$. Figure 6 shows our caution variant that uses reward-model features; Figure 7 shows traditional RND with random targets.

Caution (RM features). Moderate–high λ (about 0.6–0.8) maintains or improves accuracy as N grows, preventing the reward-hacking drop seen at $\lambda = 0$ (RM-only). $\lambda = 1.0$ (pessimism-only) is competitive but slightly conservative at small N, this is because for easier problems, most responses contain the correct answer but not all of them strictly follows the specified answering format, and applying pessimism only would at least filter out the incorrectly formatted responses. Overall, $\lambda \in [0.6, 0.8]$ delivers the best trade-off across most N.

Traditional RND (random targets). Accuracy remains flat or declines with N for all λ, and rarely exceeds the RM-only baseline. Sweeping λ offers little benefit, indicating that random targets lack the semantic grounding needed for useful uncertainty estimates.

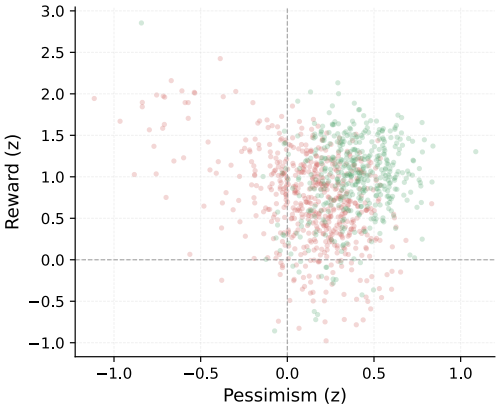


Question: Mandy owes Benedict $100. They agreed to have monthly interest of 2%. If Mandy was able to pay it after 3 months, how much should she give to Benedict?

Answer: The monthly interest is $100 x 2/100 = $<<100*2/100=2>>2. After 3 months, the total interest amount to $2 x 3 = $<<2*3=6>>6. Therefore, Mandy should give Benedict a total of $100 + $6 = $<<100+6=106>>106. #### 106

Response 1: To solve this problem, we need to calculate the interest for 3 months. The formula for simple interest is: Interest = Principal * Rate * Time Here, Principal = $100, Rate = 2% or 0.02 (in decimal), and Time = 3 months. First, let's convert the time from months to years, since the interest rate is annual. 3 months = 3/12 years = 0.25 years. Now, we can calculate the interest: Interest = $100 * 0.02 * 0.25 Interest = $5 So, Mandy owes Benedict an additional $5 as interest. In total, Mandy should pay $100 (original amount) + $5 (interest) = $105 to Benedict. **[High Reward, Low Pess.]**

Response 2: To find out how much Mandy should give to Benedict, we need to calculate the total interest for 3 months 1. Calculate the monthly interest: 2% of $100 is (2/100) * $100 = $2 per month. 2. Calculate the total interest for 3 months: $2 * 3 = $6. 3. Add the total interest to the original amount: $100 + $6 = $106. So, Mandy should give Benedict $106. #### {106} **[High Pess., Low Reward]**



Question: Mack wants to count things and decided to count how many birds there are in the sky over the next week. On days one and two, Mack saw a total of 50 birds. On day three he saw none. On days four and five, he saw 120 birds. On day six he saw 20 and on day seven he saw 90. On average, how many birds did Mack see in a day?

Answer: On days one to three, Mack saw 50+0=<<50+0=50>>50 birds. On days four to seven, he saw 120+20+90=<<120+20+90=230>>230 birds. For the whole week he saw 230+50=<<230+50=280>>280 birds. On average Mack saw 280/7=<<280/7=40>>40 birds a day. #### 40

Response 1: To find the average number of birds Mack saw per day, we need to calculate the total number of birds he saw and divide it by the number of days. Let's add up the number of birds Mack saw on each day: Day 1: 50 birds Day 2: 50 birds Day 3: 0 birds Day 4: 120 birds Day 5: 120 birds Day 6: 20 birds Day 7: 90 birds Total number of birds = 50 + 50 + 0 + 120 + 120 + 20 + 90 Total number of birds = 450 Since Mack counted birds over 7 days, we divide the total number of birds by the number of days: Average number of birds per day = Total number of birds / Number of days Average number of birds per day = 450 / 7 Average number of birds per day = 64.29 So, Mack saw approximately 64.29 birds per day on average. **[High Reward, Low Pess.]**

Response 2: To find the average number of birds Mack saw in a day, we need to calculate the total number of birds he saw and divide it by the total number of days. Let's calculate the total number of birds Mack saw:- Days 1 and 2: 50 birds- Day 3: 0 birds- Days 4 and 5: 120 birds- Day 6: 20 birds- Day 7: 90 birds Total number of birds = 50 + 0 + 120 + 20 + 90 = 280 Now, we need to find the total number of days. Mack counted birds over 7 days. Total number of days = 7 To find the average, we divide the total number of birds by the total number of days: Average number of birds per day = Total number of birds / Total number of days\n= 280 / 7 = 40 #### 40 **[High Pess., Low Reward]**

Figure 5: **Pessimism–Reward visualization on GSM8K.** Each row shows one problem: a scatter plot of z-normalized pessimism (x-axis) and z-normalized reward (y-axis), with green points for correct responses and red for incorrect. Upper-left points (high reward, low pessimism) illustrate reward hacking—responses that score well despite low distributional support. Lower-right points (low reward, high pessimism) are well-formed, instruction-following responses that the reward model undervalues; our caution mechanism up-weights these relative to reward-only selection.

Table 4: **Comprehensive Hyperparameter Configuration.** All hyperparameters used in training and evaluation of the Random Network Distillation (RND) approach for mitigating reward hacking. The table is organized by component: RND architecture, training process, inference settings, and evaluation configurations.

Component	Parameter	Value	Description
RND Architecture	Target layers	10	Number of layers extracted from reward model for target network
	Predictor layers	10	Number of layers in predictor network architecture
	RND weight (λ)	0.8	Strength of pessimism penalty in combined scoring
	Exact architecture	False	Whether predictor copies exact reward model architecture
	Embedding strategy	shared_trainable	How embeddings are handled: shared_trainable, shared_frozen, or separate
	Use projection	True	Whether to add projection layer between predictor and target
Training Process	Batch size	8	Training batch size for RND predictor
	Learning rate	1e-5	Learning rate for predictor network
	Number of epochs	5	Training epochs for predictor network
	Warmup steps	50	Learning rate scheduler warmup steps
	Max examples	5000	Maximum training examples from GSM8K train split
	VRAM usage	24GB	Minimum VRAM requirement for GPU
Inference Settings	Temperature	1.0	Sampling temperature for response generation
	Max tokens (GSM8K)	500	Maximum tokens for GSM8K responses
	Max tokens (MATH/BBH)	1024	Maximum tokens for harder reasoning tasks
	Number of samples (N)	1-512	Range of Best-of-N sampling candidates
Evaluation Configuration	Backbone model	Llama-3.2-3B-Instruct	Base language model for response generation
	Reward model	OASST DeBERTa	Primary reward model for scoring responses
	Training dataset	GSM8K train	Dataset for training RND predictor
	Test datasets	GSM8K, MATH-500, BBH	Evaluation benchmarks (in-domain and OOD)
	Bootstrap runs	3	Number of bootstrap runs for confidence intervals
	Score normalization	Z-score	Normalization method for reward and RND scores
	Selection strategy	highest_reward	Method for selecting best response from candidates
Ablation Study	RND weight range	0.0-1.0	Range of λ values tested in weight ablation
	Architecture variants	4 types	Full, simplified, embedding strategies, projection ablations
	Comparison baselines	BoN, RND-only	Standard Best-of-N and pessimism-only baselines

Table 5: **Summary of ablations.** Each row defines one setting and what it means in practice.

Ablation Setting	What it means / Rationale
Predictor architecture	
Same as Target	Predictor uses the *same overall structure* as the target network (e.g., same block type and connectivity), matching width and depth. Isolates training dynamics from architectural mismatch.
Simplified	Predictor keeps the *same hidden size and number of layers* as the target but replaces specialized target blocks with *vanilla Transformer encoder blocks*. This deliberately reduces architectural complexity while preserving depth/width, aiming for better generalization and lower overfitting risk.
Embedding strategy	
Shared, trainable	Predictor *shares the target's token embeddings* and *updates them during training*. Pros: reuse target's pretrained semantic representations and potentially quicker convergence. Cons: tighter coupling may leak target-specific biases into the predictor.
Shared, frozen	Predictor *reuses the target's token embeddings* but *keeps them frozen*. Pros: stable token mapping and clean isolation of predictor encoder learning. Cons: less flexibility to adapt embeddings to the predictor's simplified blocks.
Separate, randomly initialized	Predictor creates *its own embedding layer* with random initialization (initialized via the model's standard weight init). Pros: full decoupling from the target, potentially better regularization. Cons: longer warm-up and higher optimization burden to reach alignment.
Projection head	
No projection head	The predictor's final hidden states are *directly mapped* to the output space used for matching the target's features. Minimal additional parameters; simplest path that reduces opportunities for overfitting.
Linear projection head	Adds a *single linear layer* after the predictor's hidden states and *before* the output. Acts as a light adapter/bottleneck to better match target feature geometry; can improve fit at small cost in extra parameters, but may introduce overfitting.

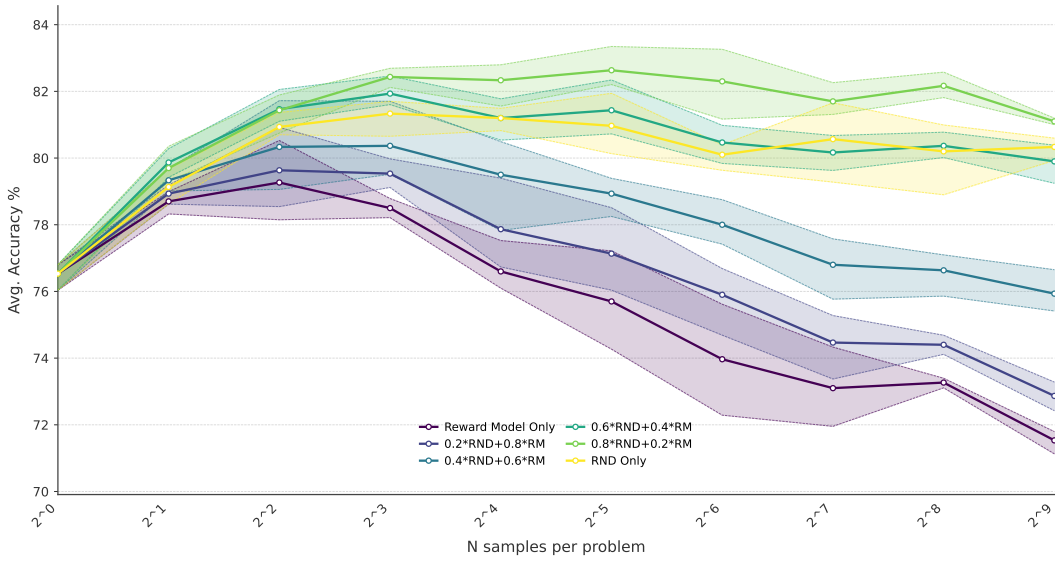


Figure 6: **Caution (RND-on-RM-features) scaling with λ.** Best-of-N accuracy on GSM8K versus samples per problem (x-axis) for Pessimism strengths $\lambda \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. The predictor is trained against a frozen target network built from reward-model features. Larger λ increases pessimism strength; $\lambda = 0$ reduces to Reward-Model-only selection, and $\lambda = 1$ to pessimism-only. Moderate–high weights (roughly 0.6–0.8) preserve scaling while curbing reward hacking, outperforming both the RM-only and RND-only extremes across most N.

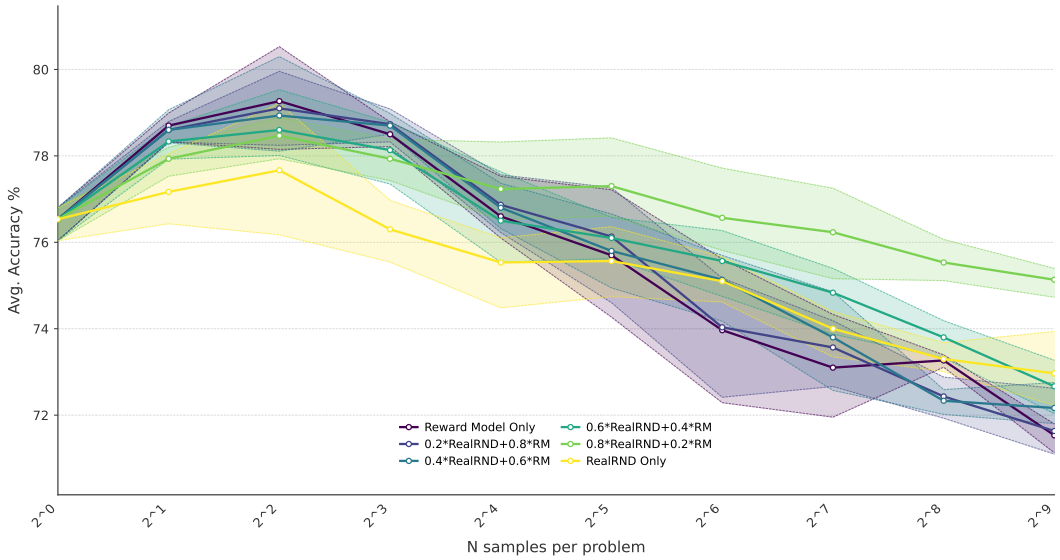


Figure 7: **Traditional RND (random targets) baseline.** Best-of-N GSM8K accuracy when using classical RND with a *randomly initialized* target network (no reward-model features), sweeping $\lambda \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Unlike our caution variant, this baseline shows little to no scaling benefit and generally does not surpass the Reward-Model-only curve, indicating that semantic grounding from reward-model features is crucial for effective distributional regularization.