RVPO: Risk-Sensitive Alignment via Variance Regularization

Ivan Montero **Tomasz Jurczyk** **Bhuwan Dhingra**
Apple
{ivamon,tjurczyk,bdhingra2}@apple.com

Abstract

Current critic-less RLHF methods aggregate multi-objective rewards via an arithmetic mean, leaving them vulnerable to constraint neglect: high-magnitude success in one objective can numerically offset critical failures in others (e.g., safety or formatting), masking low-performing "bottleneck" rewards vital for reliable multi-objective alignment. We propose Reward-Variance Policy Optimization (RVPO), a risk-sensitive framework that penalizes inter-reward variance during advantage aggregation, shifting the objective from "maximize sum" to "maximize consistency." We show via Taylor expansion that a LogSumExp (SoftMin) operator effectively acts as a smooth variance penalty. We evaluate RVPO on rubric-based medical and scientific reasoning with up to 17 concurrent LLM-judged reward signals (Qwen2.5-3B/7B/14B) and on tool-calling with rule-based constraints (Qwen2.5-1.5B/3B). By preventing the model from neglecting difficult constraints to exploit easier objectives, RVPO improves overall scores on HealthBench (0.261 vs. 0.215 for GDPO at 14B, $p < 0.001$) and maintains competitive accuracy on GPQA-Diamond without the late-stage degradation observed in other multi-reward methods, demonstrating that variance regularization mitigates constraint neglect across model scales without sacrificing general capabilities.

1 Introduction

Multi-objective reinforcement learning is essential for balancing competing goals in LLM alignment, yet current methods struggle to prioritize strict constraints alongside general performance. Recent critic-less RLHF methods like Group Relative Policy Optimization (GRPO) [1] and Group Decoupled Policy Optimization (GDPO) [2] reduce memory overhead by eliminating the Value Network, but rely on arithmetic mean aggregation. This inherently assumes "more is better," leaving optimization vulnerable to **constraint neglect**: models can exploit high-variance metrics (e.g., verbosity) to mask failures in strict, low-variance constraints (e.g., safety or formatting) (Figure 1). While constrained optimization methods [3, 4] and post-hoc model merging [5] offer alternatives, they require explicit constraint specification or separate policies, undermining the efficiency of critic-less methods. Constraint neglect can arise in any multi-objective RL setting. In group-relative methods, the linear aggregation used to compute advantages cannot distinguish a generation that satisfies all constraints from one that offsets critical failures with greater performance on easier objectives.

To address this, we introduce **Reward-Variance Policy Optimization (RVPO)**, a risk-sensitive framework that penalizes inter-reward variance.

Preprint.

Our core contributions are as follows:

1. We identify and empirically demonstrate the constraint neglect vulnerability inherent in mean-aggregated, critic-less multi-objective RL.
2. We introduce RVPO and show via Taylor expansion that the LogSumExp (SoftMin) operator implicitly penalizes inter-objective variance, with a risk coefficient k smoothly interpolating between mean and min aggregation.
3. We validate RVPO across two multi-objective paradigms: LLM-judged rubric criteria (5–17 rewards), where RVPO improves bottleneck constraint adherence on HealthBench and avoids late-stage training collapse, and rule-based tool-calling (2 rewards), where RVPO accelerates convergence on the bottleneck format constraint while preserving execution accuracy.

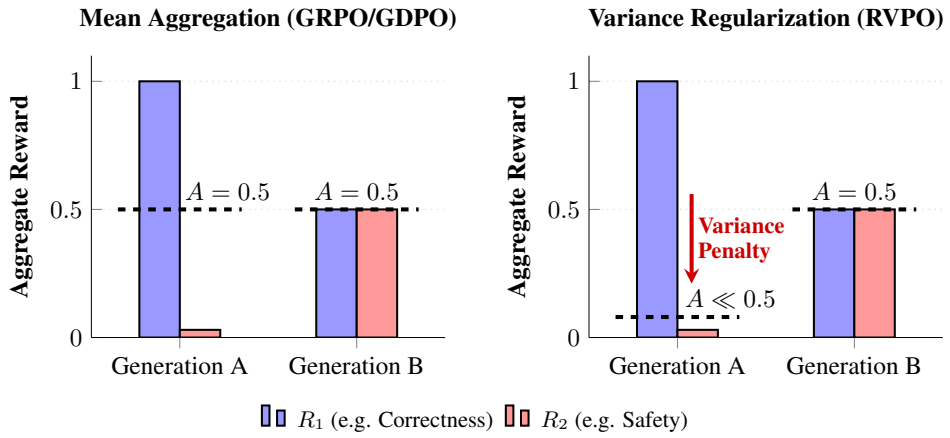

Figure 1: **Constraint Neglect in Multi-Objective RLHF. (Left)** Mean aggregation (GRPO/GDPO) treats outputs with critical constraint failures (Gen A) as mathematically identical to balanced outputs (Gen B), blinding the optimizer to critical failures. **(Right)** RVPO applies a soft-min operator to penalize inter-reward variance, heavily discounting Gen A to enforce bottleneck constraints.

2 Related Work

Reinforcement Learning from Human Feedback (RLHF) and Reward Hacking. The alignment of large language models (LLMs) relies heavily on RLHF to fine-tune models toward human preferences [6, 7]. However, standard approaches are susceptible to reward hacking, where the policy exploits misspecifications in the reward model to achieve high scores while degrading generation quality [8–10]. A well-documented instance is length gaming, where models exploit verbosity. While recent works address this via length-normalized rewards or explicit penalties [11–13], RVPO generalizes this intuition. Rather than designing a correction for each known exploitation axis, RVPO's variance penalty automatically suppresses any objective that is disproportionately exploited relative to others.

Multi-Objective Alignment and Structured Evaluation. In practice, LLMs must balance multiple competing objectives. Moving beyond monolithic scalar rewards, recent work decomposes feedback into structured rubric criteria [14, 15], multi-attribute reward models [16], checklists [17], and explicit safety rules [18]. The predominant approach to optimizing these decomposed signals is linear scalarization [19, 20], which often leads to Pareto-suboptimal policies where high-magnitude rewards dominate sparse constraints [21, 22]. Alternative paradigms treat this as a constrained optimization problem requiring explicit constraint thresholds and separate cost models [3, 23, 4], utilize post-hoc model merging [5], or resolve gradient conflicts across objectives [24]. Inference-time approaches steer frozen models via targeted intervention [25] or unified preference-aware reward models [26]. Unlike these, RVPO modifies the training objective itself: the variance penalty implicitly elevates bottleneck objectives within a single training run, smoothly relaxing a min-max objective that optimizes the worst-performing reward channel.

Critic-Less and Group-Relative Optimization. Standard PPO requires maintaining a Value Network, incurring substantial memory overhead. To alleviate this, critic-less alternatives utilize group-relative advantage estimation [27], scaling successfully to emergent reasoning [28, 29]. However, when extended to multi-objective settings, Group Relative Policy Optimization (GRPO) [1] suffers from scale dominance. Group Decoupled Policy Optimization (GDPO) [2] addresses this by normalizing individual reward models independently before summation. While GDPO prevents scale dominance, its reliance on the arithmetic mean leaves it vulnerable to loss compensation—where a critical failure in one constraint is offset by success in another. RVPO directly builds upon the GDPO framework, replacing its arithmetic mean with a variance-penalized objective to resolve this constraint neglect.

Risk-Sensitive and Variance-Penalized RL. The theoretical foundation of RVPO is rooted in risk-sensitive Markov Decision Processes [30], where agents optimize worst-case or variance-constrained objectives rather than expected returns [31, 32]. In particular, mellowmax [33] applies a LogSumExp soft-max over actions for smoother value estimation within a single-objective MDP. RVPO operates in a fundamentally different setting: the soft-min is applied across concurrent reward channels within the advantage computation. Where mellowmax smooths the policy's action selection, RVPO reshapes which reward signals drive the gradient.

3 Background: Reward Aggregation and Constraint Neglect

Critic-less RLHF methods eliminate the Value Network's memory burden by estimating advantages based on the intra-group relative performance of G sampled responses. However, the mathematical mechanism used to aggregate these multi-objective rewards fundamentally dictates the policy's vulnerability to constraint neglect.

GRPO and Scale Dominance. Group Relative Policy Optimization (GRPO) [1] aggregates M distinct objectives by summing the raw rewards for each generation, and then normalizing this total score across the group:

$$A_{GRPO} = \frac{\sum_{j=1}^{M} R_j - \mu_{total}}{\sigma_{total}},$$

where R_j is the raw reward for the j-th objective, and μ_{total} and σ_{total} are the mean and standard deviation of the summed rewards computed across the G generations in the group. Because raw rewards are summed directly, metrics with naturally large variances or unbounded scales (e.g., generation length or raw helpfulness) numerically dominate the advantage calculation. Small, sparse, or binary rewards (e.g., a strict penalty for a JSON schema violation) are entirely drowned out, preventing the model from learning rigid constraints.

GDPO and Constraint Neglect. Group Decoupled Policy Optimization (GDPO) [2] resolves scale dominance by normalizing the rewards independently. For each reward model j, GDPO computes a standard score (Z_j) across the G generations, forcing all objectives into a scale-free distribution with zero mean and unit variance. These Z-scores are then aggregated via an arithmetic mean:

$$A_{GDPO} = \frac{1}{M} \sum_{j=1}^{M} Z_j = \mu_Z.$$

While GDPO ensures scale parity, its reliance on arithmetic summation introduces a subtle flaw: *loss compensation*. Because the objective strictly maximizes the mean, a catastrophic failure on a bottleneck constraint ($Z_{format} \ll 0$) can be perfectly offset by over-performance on an easily exploitable metric ($Z_{length} \gg 0$). This aggregation implicitly signals to the policy that a generation with extreme flaws and extreme peaks is mathematically equivalent to a safely balanced generation. Consequently, the model learns to exploit "easy" objectives to inflate μ_Z while systematically neglecting strict constraints. We empirically demonstrate this in Section 6.1: at 7B, under GDPO, the policy over-optimizes Communication Quality (47.0%) while neglecting Completeness (11.1%), despite these objectives receiving equal weight after Z-normalization. To resolve this, the aggregation objective must shift from simply maximizing the mean to penalizing inter-objective disagreement.

4 Reward-Variance Policy Optimization

We propose a risk-sensitive aggregation framework that explicitly penalizes disagreement between reward models, forcing the policy to respect bottleneck constraints. For a given rollout g from a group of G generations, let $Z_j^{(g)} = (R_j^{(g)} - \mu_j)/\sigma_j$ denote the standardized reward on objective j, where μ_j and σ_j are computed across the group. High inter-objective variance indicates that some objectives are satisfied at the expense of others (Figure 1). The ideal robust objective therefore maximizes the mean reward across objectives while minimizing their variance:

$$A_{RVPO\text{-}explicit}^{(g)} = \mu_Z^{(g)} - \beta \cdot \left(\sigma_Z^{(g)}\right)^2,$$

where $\mu_Z^{(g)} = \frac{1}{M}\sum_{j=1}^{M} Z_j^{(g)}$ is the mean across objectives for rollout g, $\left(\sigma_Z^{(g)}\right)^2 = \frac{1}{M}\sum_{j=1}^{M}(Z_j^{(g)} - \mu_Z^{(g)})^2$ is the variance across objectives (not across rollouts), and $\beta > 0$ is a tunable variance penalty. However, at low M this sample variance is computed from few data points, and the quadratic penalty grows unboundedly with inter-objective disagreement. To avoid this, we use the negative LogSum-Exp (SoftMin) operator as a robust, smooth proxy that naturally saturates toward the hard minimum at large deviations rather than over-penalizing:

$$A_{RVPO}^{(g)} = -\frac{1}{k}\ln\left(\frac{1}{M}\sum_{j=1}^{M} e^{-k\cdot Z_j^{(g)}}\right),$$

where $k > 0$ is the inverse temperature, which we term the **Risk Coefficient**.

Mathematically, RVPO serves as a strict generalization of mean aggregation, allowing for tunable risk-sensitivity by smoothly interpolating between the mean and the minimum reward:

$$\lim_{k\to 0} A_{RVPO}^{(g)} = \mu_Z^{(g)} = A_{GDPO}^{(g)},$$
$$\lim_{k\to \infty} A_{RVPO}^{(g)} = \min(\{Z_j^{(g)}\}_{j=1}^{M}).$$

In the limit $k \to 0$, we recover the standard GDPO objective, where the model optimizes for average performance. Conversely, as $k \to \infty$, the objective focuses entirely on the strict bottleneck, forcing the model to satisfy the lowest-performing objective before seeking gains elsewhere. Because the lowest-performing objective is inherently the most difficult for the current policy to satisfy, RVPO effectively acts as a dynamic, difficulty-weighted aggregation mechanism. This property allows RVPO to mitigate constraint neglect by ensuring that the lowest-performing objective always contributes to the advantage. The full procedure is summarized in Algorithm 1 (Appendix).

To build intuition for why the SoftMin proxy behaves as a variance penalty, we perform a second-order Taylor expansion around the mean $\mu_Z^{(g)}$. Let $Z_j^{(g)} = \mu_Z^{(g)} + \delta_j$, such that the mean deviation $\frac{1}{M}\sum \delta_j = 0$ and the variance $\frac{1}{M}\sum \delta_j^2 = \left(\sigma_Z^{(g)}\right)^2$. By factoring out $e^{-k\mu_Z^{(g)}}$ and applying the approximations $e^y \approx 1 + y + \frac{y^2}{2}$ (for small y) alongside $\ln(1+y) \approx y$, the objective simplifies as follows:

$$\begin{aligned} A_{RVPO}^{(g)} &= \mu_Z^{(g)} - \frac{1}{k}\ln\left(\frac{1}{M}\sum_{j=1}^{M} e^{-k\delta_j}\right) \\ &\approx \mu_Z^{(g)} - \frac{1}{k}\ln\left(1 + \frac{k^2}{2}\left(\sigma_Z^{(g)}\right)^2\right) \\ &\approx \mu_Z^{(g)} - \frac{k}{2}\left(\sigma_Z^{(g)}\right)^2 = A_{RVPO\text{-}explicit}^{(g)}(k/2). \end{aligned}$$

This expansion reveals that the Risk Coefficient k acts as a continuous dial for risk aversion, naturally setting the explicit variance penalty to $\beta = k/2$. The approximation is tightest when objectives agree ($\delta_j \approx 0$); as inter-objective disagreement grows ($|k\delta_j| \gg 1$), the LogSumExp smoothly transitions from a variance penalty to its hard-min limit, avoiding unbounded quadratic growth. Both formulations shift optimization from maximizing average performance to maximizing consistent performance across all objectives. While the concave SoftMin introduces a prompt-level negative shift for prompts with high inter-objective conflict, this does not bias relative rankings within a group (the shift is shared across generations for the same prompt), and batch-level advantage normalization re-centers advantages globally.

5 Experimental Setup

To evaluate the efficacy of RVPO in mitigating constraint neglect, we benchmark our approach against GRPO [1] and GDPO [2] across two multi-objective paradigms: LLM-judged constraints (Rubrics-as-Rewards) and deterministic, rule-based constraints (Tool Calling).

5.1 Environments and Reward Formulation

Rubrics-as-Rewards (LLM-Judged Constraints): We evaluate RVPO using the Rubrics-as-Rewards (RaR) framework [14], where the number of reward signals varies dynamically per prompt (5–17 criteria). We evaluate on two domains from [14]: RaR-Medicine (20k clinical reasoning prompts) and RaR-Science (20k graduate-level science prompts). Unlike prior RaR implementations that collapse evaluations into a single scalar, we treat each of the M criteria as an independent reward channel. This high-dimensional decomposition explicitly exposes the constraint neglect vulnerability (§3). For each criterion, `gpt-4o-mini` acts as the judge, outputting a binary satisfaction score. The RaR framework assigns categorical priority weights to each criterion, which we incorporate pre-normalization; we evaluate post-normalization weighting in Appendix A.5.

Tool Calling (Rule-Based Constraints): We utilize RLLA-4k [34], a curated subset of 4,000 tool-calling trajectories, to evaluate multi-step reasoning alongside rigid structural adherence. We define two competing reward signals: Execution Correctness, a continuous scalar evaluating tool and parameter matching against the ground truth, and Format Adherence, a strictly binary constraint verifying XML schema compliance.

5.2 Training Implementation and Baselines

All experiments were conducted on a single node with 8 NVIDIA H100 GPUs. We train Qwen2.5 models [35] using the `verl` [36] and `TRL` frameworks: 1.5B and 3B for tool-calling; 3B, 7B, and 14B for rubrics-as-rewards. For rubrics, the k and β ablations were conducted at 7B; the best configurations were then applied to 3B and 14B without further tuning. Full training hyperparameters (e.g., learning rates, batch sizes, group sizes) are detailed in Appendix A.2.

Across all domains, we compare RVPO against standard GRPO (which sums raw rewards) and GDPO (which independently Z-normalizes channels before summing). For RVPO, the risk coefficient k is linearly annealed from k_{start} to k_{end} (denoted $k = k_{\text{start}} \to k_{\text{end}}$). This allows the policy to establish general capabilities under a near-mean objective before the variance penalty tightens. In the rubrics setting, we include two single-reward baselines from the RaR framework [14]: GRPO (Explicit), which aggregates per-criterion scores into a single weighted scalar; and GRPO (Implicit), the strongest baseline in the original work, where the LLM judge outputs a single holistic score.

5.3 Evaluation Methodology

For models trained on the RLLA-4k tool-calling dataset, we evaluate using the AST-based metrics of the Berkeley Function Call Leaderboard v3 (BFCL-v3) [37], averaging results across five independent runs per method. For models trained on RaR-Medicine, we evaluate multi-objective alignment using the full 5,000-example HealthBench framework [38], whose rubric scoring was validated against physician preferences. HealthBench explicitly scores models across five independent rubric axes: *Communication Quality, Instruction Following, Accuracy, Context Awareness,* and *Completeness*. Finally, for models trained on RaR-Science, we evaluate loglikelihood accu-

racy on the GPQA-Diamond benchmark [39]. Confidence intervals (95%) and significance tests are computed via bootstrap resampling over per-question scores. For rubrics experiments, we report single-run results at each model scale; the consistency of the training stability patterns across three independent scales (3B, 7B, 14B) provides evidence that the observed collapses are algorithmic rather than stochastic. We report the best-performing and final checkpoints, evaluated at every 50 steps (6 total), to separate peak capability from training stability.

6 Results

6.1 Rubrics-as-Rewards

Table 1: **Rubrics-as-Rewards evaluation across model scales.** HealthBench overall score is the micro-average across per-question rubric scores (N=5,000, 95% CI ± 0.009 per point estimate). GPQA-Diamond accuracy via loglikelihood (N=198, 95% CI ± 0.065 per point estimate; all methods within the margin of error on GPQA). GRPO uses a single holistic (Implicit) or weighted-sum (Explicit) reward [14]; all other methods decompose per-rubric rewards. Hyperparameters before / after the slash correspond to Medicine / Science, respectively. RVPO significantly outperforms GDPO on HealthBench ($p < 0.001$) and is the only multi-reward method to avoid late-stage collapse.

		HealthBench (Medicine)		**GPQA-Diamond (Science)**	
Size	**Method**	**Best Ckpt**	**Final (300)**	**Best Ckpt**	**Final (300)**
3B	GRPO (Implicit)	0.154	0.072	0.338	0.313
	GRPO (Explicit)	0.190	0.053	0.343	0.343
	GDPO	**0.192**	0.117	0.308	0.293
	RVPO-min (k=∞)	0.181	0.124	0.348	0.283
	RVPO-explicit (β=1.0/0.5)	0.184	0.011	0.313	0.308
	RVPO (k=0.5→2.0 / 1.0→2.0)	0.189	**0.147**	0.313	0.303
7B	GRPO (Implicit)	0.193	0.193	0.318	0.283
	GRPO (Explicit)	0.221	0.102	0.343	0.343
	GDPO	0.198	0.026	0.323	0.318
	RVPO-min (k=∞)	0.191	0.000	0.323	0.308
	RVPO-explicit (β=1.0/0.5)	0.227	0.178	0.318	0.318
	RVPO (k=0.5→2.0 / 1.0→2.0)	**0.230**	**0.204**	0.338	0.338
14B	GRPO (Implicit)	0.234	0.234	0.394	0.359
	GRPO (Explicit)	0.236	0.000	0.404	0.338
	GDPO	0.215	0.000	0.394	0.374
	RVPO-min (k=∞)	0.225	0.190	0.369	0.293
	RVPO-explicit (β=1.0/0.5)	0.188	0.163	0.414	0.359
	RVPO (k=0.5→2.0 / 1.0→2.0)	**0.261**	**0.236**	0.384	0.384

Figure 2 illustrates constraint neglect under arithmetic mean aggregation at the optimal Qwen2.5-7B training checkpoint. GDPO over-optimizes the easiest baseline axes (*Communication Quality*: 47.0%, *Instruction Following*: 32.5%) at the expense of stricter bottleneck constraints (*Accuracy*: 30.0%, *Context Awareness*: 18.3%, *Completeness*: 11.1%). RVPO's variance penalty directly prevents this individual objective exploitation. The resulting policy sacrifices some performance on the two "easier" axes (*Communication Quality*: 45.1% vs 47.0%, *Instruction Following*: 30.1% vs 32.5%) to pull up the three bottleneck axes, raising *Accuracy* to 33.3%, *Context Awareness* to 21.2%, and *Completeness* to 15.2%. This redistribution of optimization pressure results in RVPO achieving a higher overall score (0.230) compared to GDPO (0.198), without requiring explicit per-objective weights—the variance penalty dynamically prioritizes whichever constraints the current policy struggles to satisfy (see Appendix A.5). A full per-axis breakdown is provided in Appendix Table 3.

Table 1 tracks stability across both domains over the 300-step training run. On HealthBench, the single-scalar GRPO (Explicit) baseline achieves a strong peak (0.221), outperforming GDPO's per-rubric decomposition (0.198). This reveals that naive decomposition via mean aggregation does not

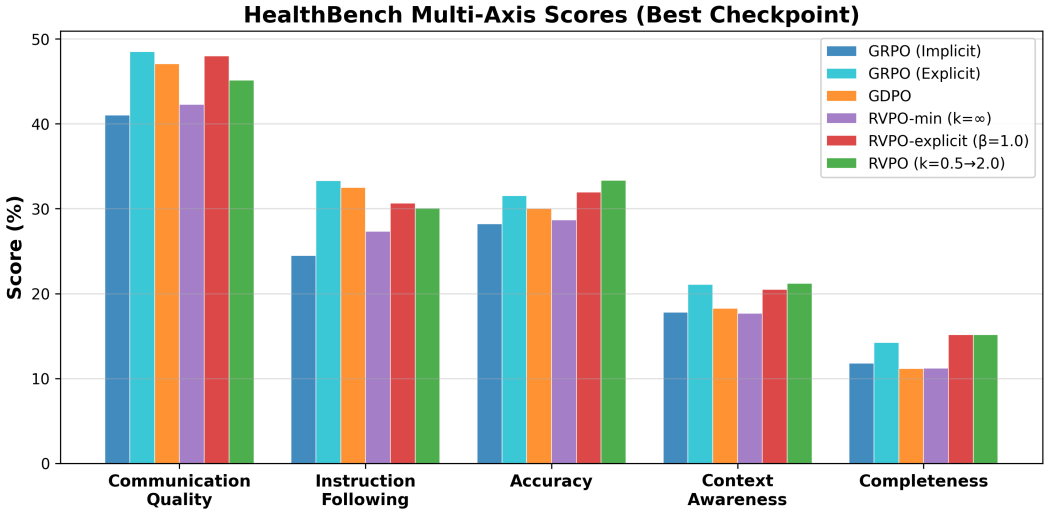

Figure 2: **Per-axis performance at the optimal training checkpoint on HealthBench [38] (Medicine, Qwen2.5-7B).** GDPO achieves one of the highest scores on *Communication Quality*, which consistently yields the highest absolute scores across methods, but underperforms on the stricter *Completeness* and *Context Awareness* constraints. By penalizing inter-objective variance, RVPO redistributes optimization pressure toward these bottleneck axes, resulting in a higher overall score (0.230 vs. 0.198).

inherently outperform a well-designed single scalar—GDPO's loss compensation actively undermines the richer signal. However, GRPO (Explicit) also degrades to 0.102 by step 300, demonstrating that training instability is not unique to decomposed rewards—even a well-tuned single scalar cannot sustain its peak. At the extremes of the aggregation spectrum, GDPO collapses to 0.026 and hard-min (RVPO-min) to 0.000, confirming that both mean and min aggregation are unstable in high-dimensional reward spaces. RVPO's variance penalty is what unlocks the benefit of per-rubric decomposition: by penalizing inter-objective disagreement, RVPO ($k = 0.5 \rightarrow 2.0$) achieves the highest peak (0.230) and the highest final score (0.204) among multi-reward methods, with minimal degradation across training while GDPO and RVPO-min collapse to near-zero.

These findings are consistent across model scales (Table 1). At 3B, all methods peak early but degrade by the final checkpoint, with RVPO achieving the best final score (0.147). At 14B, the stability gap widens: GDPO and GRPO (Explicit) collapse to 0.000 by step 300, while RVPO sustains 0.236 through step 300 and achieves the highest peak score at any scale (0.261). Notably, at 14B RVPO improves all five HealthBench axes simultaneously, suggesting that larger models have sufficient capacity to satisfy bottleneck constraints without sacrificing easier objectives.

GPQA-Diamond evaluates whether balanced rubric optimization generalizes to a held-out reasoning benchmark. While RaR-Science covers graduate-level topics, GPQA tests specialized expert knowledge requiring different reasoning patterns. Given the small dataset ($N = 198$), bootstrap 95% confidence intervals span $\pm 6.5\%$, placing all methods within the margin of error on absolute accuracy. The notable pattern is that RVPO ($k = 1.0 \rightarrow 2.0$) and RVPO-explicit are the only multi-reward methods whose best checkpoint coincides with the final checkpoint, though the small sample size precludes strong statistical claims about baseline degradation.

6.2 Tool Calling Dynamics

In the tool-calling setting, execution correctness is a continuous signal and format adherence is a sparse, binary constraint. As shown in Figure 3, GRPO is highly sensitive to this disparity: the larger magnitude of the correctness reward overshadows the binary format reward, delaying formatting improvements until step 25. GDPO mitigates this scale disparity by normalizing the reward distributions (Z_j), leading to earlier formatting improvements.

Table 2: **BFCL-v3 evaluation [37].** Qwen2.5-1.5B and 3B. Avg is the mean of the three reported subcategories; AST Sum evaluates single and simple function calls; Parallel and Par. Multiple evaluate the ability to invoke multiple functions simultaneously. All methods achieve comparable accuracies, preserving general tool-calling capabilities. Results averaged across five runs per method (run-to-run std $\approx 0.4\%$).

Size	Method	Avg (%)	AST Sum (%)	Parallel (%)	Par. Multiple (%)
1.5B	GRPO	70.4	71.7	70.3	69.2
	GDPO	**71.9**	**73.4**	71.7	**70.7**
	RVPO-explicit ($\beta = 1.0$)	71.3	72.5	71.0	70.5
	RVPO ($k = 1.0$)	**71.9**	72.9	**72.1**	70.6
3B	GRPO	80.3	81.3	77.8	81.9
	GDPO	80.7	81.6	78.8	81.7
	RVPO-explicit ($\beta = 1.0$)	**81.4**	**82.1**	**80.1**	**81.9**
	RVPO ($k = 1.0$)	80.4	81.5	78.4	81.3

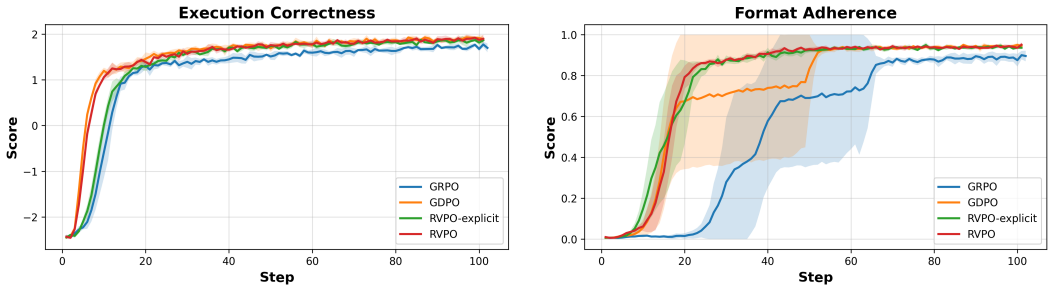

Figure 3: **Tool Calling (RLLA) Training Dynamics [34].** Qwen2.5-1.5B training progression across five independent runs; solid lines show the mean and shaded regions ± 1 standard deviation. *(Left)* While mean-based baselines (GDPO and GRPO) successfully maximize execution correctness, they struggle to satisfy the strict format adherence constraint *(Right)*. In contrast, RVPO and RVPO-explicit enforce this bottleneck constraint via a variance penalty, achieving simultaneous convergence across both objectives.

However, GDPO plateaus before reaching full format compliance. This behavior illustrates the loss compensation effect of arithmetic mean aggregation: the policy learns that high execution scores can mathematically offset missing format constraints. In contrast, RVPO's variance penalty discounts outputs with high execution correctness but failed syntax checks, encouraging the policy to satisfy both objectives simultaneously. Consequently, RVPO shows improved convergence on the format metric while reducing the inter-run variance observed in the baselines.

Table 2 shows that all methods achieve comparable downstream accuracy on BFCL-v3 across 1.5B and 3B scales. This indicates that RVPO's primary benefit in this low-dimensional setting (M=2) is faster convergence on the bottleneck constraint during training.

6.3 Ablation: Risk Coefficient and Curriculum Robustness

The risk coefficient k serves as an explicit knob along the mean-to-min aggregation spectrum: low k optimizes average performance, high k prioritizes worst-case satisfaction. Unlike methods that train separate policies per objective and merge post-hoc [5], k parameterizes this trade-off within a single training run.

As shown in Table 1, both extremes of this spectrum are unstable at 7B: mean aggregation (GDPO, k=0) collapses to 0.026 by step 300, while hard-min (RVPO-min, k=∞) degrades to 0.000. The hard-min failure is intuitive: when only the single worst objective receives gradient signal, optimization oscillates as different objectives alternate as the bottleneck, preventing stable convergence. The optimal operating point lies strictly in the interior. To characterize this sensitivity, we evaluate various static and annealed k schedules on HealthBench. At k=5.0, the bottleneck axes improve

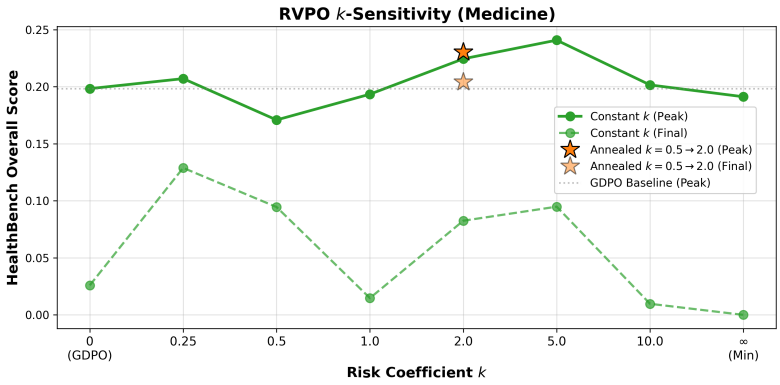

Figure 4: **Risk Coefficient Sensitivity and Curriculum Robustness on HealthBench (Medicine, Qwen2.5-7B).** Low constant k schedules are more stable but less performant, while high constant k schedules achieve higher peaks but are more unstable. Annealing k over training ($k = 0.5 \rightarrow 2.0$) provides the best of both regimes by allowing the policy to establish general capabilities under a near-mean objective before the variance penalty tightens.

substantially over GDPO—*Completeness* rises from 11.1% to 17.0% and *Accuracy* from 30.0% to 34.4%—without substantially sacrificing the easier axes, yielding the highest peak score (0.241) among static-k configurations at 7B. However, this peak is not stable through training: k=5.0 collapses to 0.095 by step 300, requiring careful early stopping to capture the peak. We therefore report the annealed k=0.5 $\rightarrow$ 2.0 schedule as our primary result, since it sustains performance through training completion (0.204 at step 300) despite the lower peak. As shown in Figure 4, annealing solves this by tightening the constraint bottleneck only after general capabilities are established.

We evaluated a range of annealing schedules on rubrics (Appendix Table 4). Performance remains stable across different schedules ($0.1 \rightarrow 2.0$, $0.5 \rightarrow 5.0$), provided the curriculum initiates at a low k value. Conversely, an aggressive start ($1.0 \rightarrow 2.0$) causes premature constraint over-optimization and collapse (0.000). These results suggest a practical heuristic: for low-dimensional, fixed reward spaces like tool-calling, a static $k \approx 1.0$ suffices. However, for high-dimensional or dynamically varying reward spaces, annealing from a low initial k is necessary to establish general capabilities before the variance penalty tightens.

7 Limitations and Future Work

The primary limitation of RVPO is the sensitivity of the risk coefficient k. While annealed curricula (§6.3) are effective, optimal k values remain sensitive to reward space dimensionality, group size G (our experiments use G=4 for tool-calling and G=16 for rubrics), and inter-objective conflict, motivating future work on adaptive scheduling. Additionally, as demonstrated in Appendix A.5, RVPO's bottleneck prioritization is driven by criterion difficulty rather than declared priority; developing weighted RVPO variants could better align these dynamics. RVPO may also amplify noise from unreliable reward channels, as the soft-min focuses optimization on whichever objective produces the lowest Z-scores regardless of whether this reflects policy weakness or reward model noise.

8 Conclusion

We introduced Reward-Variance Policy Optimization (RVPO), which addresses constraint neglect in mean-aggregated multi-objective RL via a LogSumExp variance penalty. RVPO scales from two rule-based rewards to 17 concurrent LLM-judged criteria across multiple model sizes, consistently improving bottleneck constraint adherence while maintaining training stability. The risk coefficient k provides an explicitly tunable parameter for navigating multi-objective trade-offs within a single training run. As alignment increasingly relies on decomposed reward signals—from rubric criteria to safety constraints—robust aggregation that prevents any single objective from being silently neglected becomes essential for reliable deployment.

Acknowledgments and Disclosure of Funding

We thank Xinyan Velocity Yu, Nitin Gupta, Russ Webb, and Dong Yin for feedback on early drafts of this work.

References

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

[2] Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. GDPO: Group reward-decoupled normalization policy optimization for multi-reward RL optimization. *arXiv preprint arXiv:2601.05242*, 2026.

[3] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. Pmlr, 2017.

[4] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe RLHF: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[5] Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36:71095–71134, 2023.

[6] 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.

[7] 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.

[8] Yanjun Chen, Dawei Zhu, Yirong Sun, Xinghao Chen, Wei Zhang, and Xiaoyu Shen. The accuracy paradox in RLHF: When better reward models don't yield better language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2980–2989, 2024.

[9] Juntao Dai, Taiye Chen, Yaodong Yang, Qian Zheng, and Gang Pan. Mitigating reward over-optimization in RLHF via behavior-supported regularization. *arXiv preprint arXiv:2503.18130*, 2025.

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

[11] Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

[12] Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4998–5017, 2024.

[13] Lichang Chen, Chen Zhu, Davit Soselia, Jiuhai Chen, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. ODIN: Disentangled reward mitigates hacking in RLHF. *arXiv preprint arXiv:2402.07319*, 2024.

[14] Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*, 2025.

[15] Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13806–13834, 2024.

[16] Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J Zhang, Makesh N Sreedhar, and Oleksii Kuchaiev. HelpSteer 2: Open-source dataset for training top-performing reward models. *Advances in Neural Information Processing Systems*, 37:1474–1501, 2024.

[17] Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*, 2025.

[18] Tong Mu, Alec Helyar, Johannes Heidecke, Joshua Achiam, Andrea Vallone, Ian Kivlichan, Molly Lin, Alex Beutel, John Schulman, and Lilian Weng. Rule based rewards for language model safety. *Advances in Neural Information Processing Systems*, 37:108877–108901, 2024.

[19] Xuying Li, Zhuo Li, Yuji Kosuga, and Victor Bian. Optimizing safe and aligned language generation: A multi-objective GRPO approach. *arXiv preprint arXiv:2503.21819*, 2025.

[20] Conor F Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M Zintgraf, Richard Dazeley, Fredrik Heintz, et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):26, 2022.

[21] Zhuo Li, Guodong Du, Weiyang Guo, Yigeng Zhou, Xiucheng Li, Wenya Wang, Fangming Liu, Yequan Wang, Deheng Ye, Min Zhang, et al. Multi-objective large language model alignment with hierarchical experts. *arXiv preprint arXiv:2505.20925*, 2025.

[22] Kihyun Kim, Jiawei Zhang, Asuman Ozdaglar, and Pablo A Parrilo. Beyond RLHF and NLHF: Population-proportional alignment under an axiomatic framework. *arXiv preprint arXiv:2506.05619*, 2025.

[23] Santiago Paternain, Luiz Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. *Advances in Neural Information Processing Systems*, 32, 2019.

[24] Peter L Chen, Xiaopeng Li, Xi Chen, and Tianyi Lin. Reward-free alignment for conflicting objectives. *arXiv preprint arXiv:2602.02495*, 2026.

[25] Duy Nguyen, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. Multi-attribute steering of language models via targeted intervention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20619–20634, 2025.

[26] Baijiong Lin, Weisen Jiang, Yuancheng Xu, Hao Chen, and Ying-Cong Chen. PARM: Multi-objective test-time alignment via preference-aware autoregressive reward model. *arXiv preprint arXiv:2505.06274*, 2025.

[27] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, 2024.

[28] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

[30] Ronald A Howard and James E Matheson. Risk-sensitive markov decision processes. *Management science*, 18(7):356–369, 1972.

[31] Xiaoyan Hu and Ho-fung Leung. A tighter problem-dependent regret bound for risk-sensitive reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 5411–5437. PMLR, 2023.

[32] Han Zhong, Xun Deng, Ethan X Fang, Zhuoran Yang, Zhaoran Wang, and Runze Li. Risk-sensitive deep RL: Variance-constrained actor-critic provably finds globally optimal policy. *Journal of the American Statistical Association*, pages 1–26, 2025.

[33] Kavosh Asadi and Michael L Littman. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, pages 243–252. PMLR, 2017.

[34] Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. ToolRL: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*, 2025.

[35] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

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

[37] Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.

[38] Rahul K Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, et al. HealthBench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

[39] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *First conference on language modeling*, 2024.

A Appendix

A.1 RVPO Algorithm

Algorithm 1 summarizes the full RVPO training procedure, including per-channel Z-normalization, inactive reward masking, and SoftMin aggregation. In the RaR setting, the number of active reward channels varies per prompt ($M_{\text{active}} \in [5, 17]$), as each prompt defines a distinct rubric; channels corresponding to criteria not applicable to the current prompt are excluded from the aggregation. The risk coefficient $k(t)$ may be held constant or annealed (e.g., linearly) over training steps. As $k \to 0$, RVPO recovers standard GDPO (mean aggregation); as $k \to \infty$, it reduces to hard-min.

Algorithm 1 Reward-Variance Policy Optimization (RVPO).

Require: Policy π_θ, reference policy π_{ref}, M reward functions $\{R_j\}_{j=1}^{M}$, group size G, risk coefficient schedule $k(t)$

1: **for** each training step t **do**
2: Sample batch of prompts $\{x_i\}$
3: **for** each prompt x_i **do**
4: Generate G responses: $\{y_i^{(g)}\}_{g=1}^{G} \sim \pi_\theta(\cdot \mid x_i)$
5: Compute rewards: $R_j^{(g)} = R_j(x_i, y_i^{(g)})$ for $j = 1, \ldots, M$
6: **for** each reward channel $j = 1, \ldots, M$ **do**
7: $\mu_j = \frac{1}{G}\sum_g R_j^{(g)}, \quad \sigma_j = \text{std}(\{R_j^{(g)}\}_g)$ ▷ *Z-normalize*
8: $Z_j^{(g)} = (R_j^{(g)} - \mu_j)/(\sigma_j + \epsilon)$
9: **end for**
10: $A^{(g)} = -\frac{1}{k(t)} \ln\Big(\frac{1}{M_{\text{active}}} \sum_{j \in \text{active}} e^{-k(t) \cdot Z_j^{(g)}}\Big)$ ▷ *SoftMin over active channels*
11: **end for**
12: Whiten advantages across batch: $\hat{A}^{(g)} = (A^{(g)} - \mu_A)/\sigma_A$
13: Update θ via clipped policy gradient with advantages $\hat{A}^{(g)}$ and KL penalty against π_{ref}
14: **end for**
15: **Note:** For RVPO-explicit, replace line 10 with $A^{(g)} = \mu_Z^{(g)} - \beta \cdot \Big(\sigma_Z^{(g)}\Big)^2$ for active channels only.

A.2 Training Hyperparameters

This section details the hyperparameter configurations used for the experiments. All training was conducted on a single node with 8 NVIDIA H100 GPUs. On this hardware, tool-calling training runs require ≈1.5 hours for 1.5B models and ≈2.5 hours for 3B models. In the rubrics setting, 300-step training runs require ≈2 hours (3B), ≈4.5 hours (7B), and ≈8 hours (14B) per experiment. RVPO's aggregation step (LogSumExp over M channels) adds negligible wall-clock overhead (<1%) relative to GDPO.

Tool Calling (RLLA-4k): To ensure a rigorous baseline comparison, we utilize the exact training recipe established by GDPO [2] and ToolRL [34]. We use the `verl` framework [36] to fine-tune Qwen2.5-Instruct models (1.5B and 3B) for 15 epochs (≈117 steps) with a PPO mini-batch size of 128, a learning rate of 1×10^{-6}, a group size of $G = 4$ rollouts per prompt, and an adaptive KL penalty ($\beta_{KL} = 0.001$).

Rubrics-as-Rewards (RaR-Medicine and RaR-Science): Similarly, we adopt the training hyperparameters established by the Rubrics-as-Rewards framework [14]. We use the `TRL` framework to fine-tune Qwen2.5 base policies (3B, 7B, 14B) for 300 steps with an effective batch size of 96 prompts per step. We apply a learning rate of 5×10^{-6} with a 10% linear warmup schedule, sampling $G = 16$ responses per prompt at a temperature of 1.0, and a KL penalty coefficient of $\beta_{KL} = 0.04$ against the frozen reference policy. Checkpoints are saved and evaluated every 50 steps; we report the best-performing checkpoint alongside the final checkpoint (step 300).

Asset Licenses: Qwen2.5 models (Apache 2.0), RLLA-4k (Apache 2.0), RaR-Medicine/Science (CC-BY 4.0), HealthBench (MIT), GPQA-Diamond (CC-BY 4.0), BFCL-v3 (Apache 2.0), TRL (Apache 2.0), verl (Apache 2.0).

A.3 Detailed HealthBench Results

Table 3 provides the full per-axis HealthBench evaluation at both the best and final training checkpoints across all model scales.

Table 3: **Per-axis breakdown of HealthBench-Medicine evaluations across model scales.** Evaluated over $N = 5,000$ examples. **(a)** At the best checkpoint, RVPO achieves the highest overall score at 7B and 14B, with 14B RVPO improving all five axes simultaneously. **(b)** At the final checkpoint, GDPO and RVPO-min collapse at 7B while GRPO (Explicit) and GDPO collapse at 14B; RVPO remains the most robust across scales. All sub-category point estimates have a 95% CI margin of error of ≤ 0.010.

(a) Best Checkpoint Performance

Size	Method	Overall	Comm. Quality	Instr. Following	Accuracy	Context Aware.	Completeness
3B	GRPO (Implicit)	0.154	0.397	0.257	0.239	0.152	0.085
	GRPO (Explicit)	0.190	0.459	0.276	**0.269**	0.172	**0.123**
	GDPO	**0.192**	0.451	**0.289**	0.264	0.179	0.121
	RVPO-min (k=∞)	0.181	0.444	0.266	0.256	0.177	0.115
	RVPO-explicit (β=1.0)	0.184	**0.465**	0.272	0.261	0.171	0.114
	RVPO (k=0.5→2.0)	0.189	0.460	0.274	**0.269**	**0.188**	0.110
7B	GRPO (Implicit)	0.193	0.410	0.245	0.282	0.178	0.118
	GRPO (Explicit)	0.221	**0.485**	**0.333**	0.315	0.211	0.142
	GDPO	0.198	0.470	0.325	0.300	0.183	0.111
	RVPO-min (k=∞)	0.191	0.423	0.273	0.287	0.177	0.112
	RVPO-explicit (β=1.0)	0.227	0.480	0.306	0.320	0.205	0.151
	RVPO (k=0.5→2.0)	**0.230**	0.451	0.301	**0.333**	**0.212**	**0.152**
14B	GRPO (Implicit)	0.234	0.423	0.302	0.322	0.206	0.176
	GRPO (Explicit)	0.236	0.460	0.311	0.333	0.201	0.168
	GDPO	0.215	0.407	0.258	0.313	0.191	0.149
	RVPO-min (k=∞)	0.225	0.466	0.305	0.315	0.195	0.147
	RVPO-explicit (β=1.0)	0.188	0.360	0.269	0.272	0.177	0.131
	RVPO (k=0.5→2.0)	**0.261**	**0.485**	**0.321**	**0.364**	**0.220**	**0.184**

(b) Final Checkpoint Performance

Size	Method	Overall	Comm. Quality	Instr. Following	Accuracy	Context Aware.	Completeness
3B	GRPO (Implicit)	0.072	0.234	0.130	0.143	0.091	0.020
	GRPO (Explicit)	0.053	0.189	0.135	0.114	0.067	0.007
	GDPO	0.117	0.301	0.118	0.203	0.112	0.053
	RVPO-min (k=∞)	0.124	0.304	**0.200**	0.202	0.129	0.064
	RVPO-explicit (β=1.0)	0.011	0.117	0.064	0.074	0.020	0.000
	RVPO (k=0.5→2.0)	**0.147**	**0.380**	0.182	**0.232**	**0.143**	**0.075**
7B	GRPO (Implicit)	0.193	0.410	0.245	0.282	0.178	0.118
	GRPO (Explicit)	0.102	0.216	0.212	0.171	0.096	0.035
	GDPO	0.026	0.138	0.111	0.107	0.035	0.000
	RVPO-min (k=∞)	0.000	0.000	0.000	0.013	0.000	0.000
	RVPO-explicit (β=1.0)	0.178	0.398	**0.267**	0.261	0.172	0.107
	RVPO (k=0.5→2.0)	**0.204**	**0.443**	0.256	**0.310**	**0.184**	**0.128**
14B	GRPO (Implicit)	0.234	0.423	**0.302**	0.322	**0.206**	**0.176**
	GRPO (Explicit)	0.000	0.102	0.038	0.019	0.003	0.000
	GDPO	0.000	0.027	0.008	0.000	0.000	0.000
	RVPO-min (k=∞)	0.190	0.357	0.226	0.275	0.170	0.143
	RVPO-explicit (β=1.0)	0.163	0.300	0.155	0.236	0.148	0.114
	RVPO (k=0.5→2.0)	**0.236**	**0.439**	0.287	**0.329**	0.202	0.171

A.4 Hyperparameter Ablation Details

Table 4 provides the full numerical results for the hyperparameter sensitivity analysis presented in Figure 4.

Table 5 provides the full per-axis HealthBench breakdown across all k schedules, including annealed curricula.

Table 4: **Risk coefficient ablation on HealthBench (Medicine, Qwen2.5-7B).** GDPO ($k = 0$) and hard-min ($k = \infty$) represent the spectrum endpoints. Constant k values capture high peak scores but degrade by step 300. The annealed schedule $k = 0.5 \rightarrow 2.0$ achieves the best balance of peak performance and stability.

Schedule (k)	**Best Ckpt**	**Final (300)**
Baselines		
GRPO (Implicit)	0.193	0.193
GRPO (Explicit)	0.221	0.102
GDPO ($k = 0$)	0.198	0.026
RVPO-min ($k = \infty$)	0.191	0.000
Constant Schedules		
$k = 0.25$	0.207	0.129
$k = 0.5$	0.171	0.095
$k = 1.0$	0.194	0.015
$k = 2.0$	0.225	0.083
$k = 5.0$	**0.241**	0.095
$k = 10.0$	0.202	0.010
Annealed Curricula		
$0.1 \rightarrow 2.0$	0.214	0.132
$0.25 \rightarrow 1.0$	0.209	0.082
$0.5 \rightarrow 5.0$	0.198	0.139
$0.5 \rightarrow 2.0$	0.230	**0.204**
$1.0 \rightarrow 2.0$	0.143	0.000

Table 5: **Per-axis HealthBench breakdown across risk coefficient schedules (Qwen2.5-7B, best checkpoint).** Constant k=5.0 achieves the highest peak by substantially improving bottleneck axes (Completeness: 17.0%, Accuracy: 34.4%) without sacrificing Communication Quality (48.1% vs. GDPO's 47.0%). The annealed schedule k=$0.5 \rightarrow 2.0$ achieves a similar redistribution with better training stability (see Table 1).

Schedule (k)	**Overall**	**Comm. Quality**	**Instr. Following**	**Accuracy**	**Context Aware.**	**Completeness**
Baselines						
GRPO (Implicit)	0.193	0.410	0.245	0.282	0.178	0.118
GRPO (Explicit)	0.221	0.485	0.333	0.315	0.211	0.142
GDPO ($k = 0$)	0.198	0.470	0.325	0.300	0.183	0.111
RVPO-min ($k = \infty$)	0.191	0.423	0.273	0.287	0.177	0.112
Constant Schedules						
$k = 0.25$	0.207	0.430	0.284	0.302	0.184	0.137
$k = 0.5$	0.171	0.425	0.216	0.249	0.154	0.092
$k = 1.0$	0.193	0.417	0.309	0.287	0.178	0.111
$k = 2.0$	0.224	**0.497**	0.312	0.313	0.207	0.146
$k = 5.0$	**0.241**	0.481	0.307	**0.344**	0.205	**0.170**
$k = 10.0$	0.202	0.454	0.300	0.293	0.183	0.126
Annealed Curricula						
$0.1 \rightarrow 2.0$	0.213	0.473	0.305	0.308	0.196	0.139
$0.25 \rightarrow 1.0$	0.209	0.449	0.291	0.297	0.192	0.140
$0.5 \rightarrow 2.0$	0.230	0.451	0.301	0.333	**0.212**	0.152
$0.5 \rightarrow 5.0$	0.198	0.445	**0.341**	0.297	0.186	0.129
$1.0 \rightarrow 2.0$	0.143	0.259	0.211	0.216	0.119	0.096

A.5 Pre- vs. Post-Normalization Weighting

Table 6 evaluates methods under a weighted regime on the HealthBench (Medicine) domain. Standard RVPO incorporates the AI-generated categorical priority weights from the RaR framework [14] (e.g., Essential=1.0, Optional=0.3) directly into the raw reward computation ($R_j = c_j \cdot w_j$, where $c_j \in \{0, 1\}$ is the binary criterion score). However, for binary criteria, these scalar weights are naturally absorbed by the per-channel Z-normalization (pre-normalization weighting). Table 6 evaluates an alternative regime where these weights are explicitly applied post-normalization ($Z'_j = w_j \cdot Z_j$) to force the optimizer to recognize static priorities.

While post-normalization weighting improves the baseline methods (GRPO, GDPO) by explicitly forcing attention to priority constraints, it degrades the performance of RVPO. RVPO relies on unscaled empirical variance to dynamically prioritize bottleneck constraints. Applying static weights post-normalization artificially compresses the variance of lower-weighted criteria, causing the soft-min operator to prematurely ignore them. The standard pre-normalization RVPO exceeds the performance of post-normalization baselines, indicating that dynamic variance regularization provides a robust alternative to explicit post-hoc reward weighting.

Table 6: **Performance comparison of pre- vs. post-normalization weighting on HealthBench (Medicine, Qwen2.5-7B).** When static AI-generated priority weights are explicitly applied post-normalization ($Z'_j = w_j \cdot Z_j$), baselines (GRPO, GDPO) improve by satisfying neglected constraints. However, this explicit weighting degrades RVPO's dynamic variance penalty. Standard pre-normalization RVPO exceeds these explicit post-normalization baselines, serving as an automatic, difficulty-driven alternative to static reward tuning.

Method	**Best Checkpoint**	**Final (Step 300)**
Pre-normalization Weighting (Standard)		
GDPO (Mean)	0.198	0.026
RVPO ($k = 0.5 \rightarrow 2.0$)	**0.230**	0.204
Post-normalization Weighting (Baselines)		
GRPO (Explicit)	0.221	0.102
GDPO	0.213	**0.213**
Post-normalization Weighting (RVPO)		
RVPO ($k = 0.5 \rightarrow 2.0$)	0.192	0.078
RVPO ($k = 1.0 \rightarrow 2.0$)	0.201	0.000
RVPO ($k = 2.0$ const)	0.160	0.106

A.6 Explicit Variance Penalty (β) Ablation

Figure 5 provides the ablation for constant values of the explicit variance penalty (β). We observe that calculating empirical variance over a dynamically small number of objectives ($M \in [5, 17]$) introduces higher sensitivity to the choice of β compared to the smooth k-parameterized soft-min operator, with performance varying non-monotonically across the sweep. This gap between the two formulations is consistent with the Taylor expansion's regime of validity: the second-order approximation $A_{RVPO} \approx \mu_Z - \frac{k}{2}\sigma_Z^2$ is tight only for small $k\delta_j$, whereas at larger deviations the LogSumExp naturally saturates toward the hard minimum and the quadratic penalty does not. This supports the use of the LogSumExp formulation as the more robust default.

Annealed β schedules were also evaluated. A gentle schedule ($\beta = 0.1 \rightarrow 0.5$) peaks at 0.215 but degrades to 0.132 by step 300, while a moderate schedule ($\beta = 0.25 \rightarrow 1.0$) peaks at 0.221 but collapses to 0.000. Schedules ending above $\beta = 1.0$ collapse early. Both constant and annealed explicit schedules can thus achieve peaks comparable to RVPO (peak 0.230, final 0.204), but none match the LogSumExp's training stability.

A.7 Broader Impacts

This work improves the ability of large language models to reliably balance competing objectives and strictly adhere to formatting and safety constraints. By mitigating constraint neglect, RVPO provides a computationally efficient mechanism for aligning models toward safer, more consistent behavior without sacrificing general capabilities. However, as a foundational reinforcement learning algorithm, variance penalization is agnostic to the semantic nature of the constraints. While we apply it to enforce structural correctness and clinical accuracy, the same explicit constraint enforcement could theoretically be misused if malicious or biased reward models are intentionally provided to the optimizer.

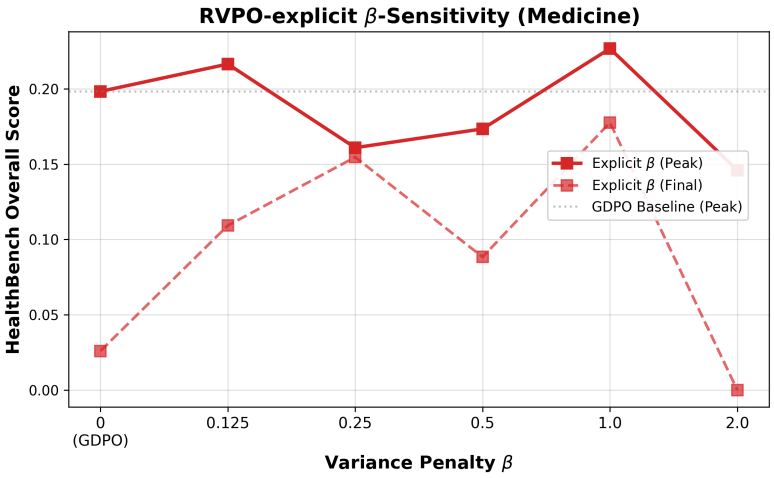

Figure 5: **Explicit Variance Penalty (β) Sweep on HealthBench (Medicine, Qwen2.5-7B).** Evaluating constant values of the explicit variance penalty (β) reveals more optimization instability and higher sensitivity to hyperparameter choice compared to the LogSumExp (SoftMin) formulation.