

Scaling Reinforcement Learning for Content Moderation with Large Language Models

Rui Liu Yuchen Lu Zhenyu Hou Fangzhou Xiong
Xiaoyang Zhang Changshu Jian Zhicheng Zhu Jiayuan Ma
Jacob Tao Merve Unlu Menevse Han Li Chaitali Gupta
Xiaochang Peng Suwen Lin Chongyang Bai Shike Mei
Hang Cui Yang Qin Shuo Tang Jason Gaedtk
Arpit Mittal Hamed Firooz

Meta, Menlo Park, CA, USA

Correspondence: {ruiliu,mhfirooz}@meta.com

Abstract

Large-scale content moderation requires models that can apply complex, evolving policies under limited expert supervision. We present an empirical study of reinforcement learning (RL) for policy-grounded moderation across three production-derived tasks. Using established RL and inference primitives, we develop three integrity-specific designs: a policy-grounded full-trace reward, rationale-marginalized Monte Carlo scoring, and policy-structured reflection. Each design addresses an observed failure mode, including reasoning-length collapse and polarized confidence estimates. Across the three tasks, RL-Only systems trained on 200–836 labeled examples remain competitive with supervised fine-tuning (SFT) systems trained on thousands of examples, corresponding to up to approximately one order of magnitude greater label efficiency at the evaluated operating points. Performance generally improves with additional data and training compute before saturating, although larger rollout counts exhibit variability. The strongest SFT→RL systems attain point-estimate performance comparable to policy-trained full-time employee (FTE) reviewers across the three tasks.

1 Introduction

Content moderation at scale has become a foundational requirement for maintaining safe, trustworthy, and policy compliant online environments. Search and AI platforms such as Google (Google Cloud, 2023; Weidinger et al., 2024), Social Networks such as Meta’s Facebook and Instagram (Meta Platforms, 2024; Inan et al., 2023), online retail such as Amazon (Gampa et al., 2023), user-facing AI chat such as OpenAI (OpenAI-Platform, 2023; Guan et al., 2024; Markov et al., 2023), and Anthropic (Bai et al., 2022) all host

or generate vast volumes of user- and AI-created text, images, and videos at global scale. These organizations face simultaneous challenges in detecting, scoring, and mitigating harmful or policy-violating material. Without effective moderation, these ecosystems risk the rapid spread of misinformation, harassment, hate speech, and other harms that undermine user safety, advertiser trust, and regulatory compliance (Gao et al., 2025). Consequently, modern moderation pipelines increasingly combine machine learning classifiers, human review, and adaptive policy frameworks to ensure that content decisions remain both scalable and aligned with evolving community and societal standards.

Large Language Models (LLMs) have recently emerged as a promising paradigm for improving content moderation due to their strong linguistic reasoning and generalization capabilities (Markov et al., 2024; Yuan et al., 2025). By leveraging rich semantic representations learned during large-scale pretraining, LLMs can capture subtle contextual cues, linguistic nuances, and domain-specific patterns that traditional content classifiers often miss. Related works show that LLM can act as rule-following moderators via prompting (Kumar et al., 2024), as supervised fine-tuned (SFT) classifiers (Ma et al., 2024), or as guardrail models such as Llama Guard (Inan et al., 2023) and BingGuard (Yin et al., 2025), with industry adoptions such as Google Ads’ LLM-based content review system (Qiao et al., 2024).

However, while effective for many scenarios, prompting and SFT still face the challenge to encode highly complex, conditional, and context-dependent moderation policies. Real-world content moderation policies often involve hierarchical severity levels, exception clauses, and nuanced distinctions that depend on subtle linguistic cues or

multi-turn context (Sharma et al., 2022). Capturing these behaviors through static supervision alone can lead to inconsistencies, overfitting to annotation artifacts, or poor generalization in ambiguous or adversarial cases. Data scarcity is another bottleneck in content moderation due to the high cost and latency of expert labeling (Kiela et al., 2020; Alam et al., 2022).

Reinforcement learning (RL) has emerged as a post-training technique for aligning LLMs with human preferences in safety-critical settings, with prior work (Sharma et al., 2025; Dai et al., 2023; Guan et al., 2024; Zhang et al., 2025; Yuan et al., 2024) demonstrating that RL can optimize behaviors beyond token-level supervised training. RL enables models to internalize complex safety policies and balance competing constraints that are difficult to encode through SFT alone (Mu et al., 2024a,b). Constitutional and rule-based alignment methods encode principles explicitly, preference optimization learns from pairwise comparisons, and moderation guardrails commonly target fixed safety taxonomies (Bai et al., 2022; Rafailov et al., 2023; Inan et al., 2023). By contrast, the present setting requires detailed and evolving integrity policies to be applied to production-derived examples. Accordingly, our contribution lies in policy-grounded reward and inference designs for this setting, rather than in a new RL optimization algorithm.

Motivated by this gap, we investigate the scaling of RL-based post-training for content moderation at META PLATFORMS, INC.. We evaluate several training recipes and reward-shaping strategies, including *verifiable rewards* (RLVR) and *LLM-as-judge* formulations. Direct application of RLVR is challenging because many moderation policies do not admit automatically verifiable rewards and because outcome-only supervision is susceptible to reward hacking. Our training formulation therefore combines verifiable outcome rewards, rubric-based evaluation, and LLM-judge rewards to provide structured, policy-aligned supervision beyond the final binary label.

We evaluate the proposed methods on three policy-violation classification tasks derived from production data at META PLATFORMS, INC. and present a training framework for adapting LLMs into policy-aligned classifiers. The strongest SFT→RL systems attain point-estimate performance comparable to policy-trained full-time employee (FTE) reviewers across the three tasks. The principal findings are as follows:

1. **Effective mitigation strategies for critical RL failure modes.** We identify three recurrent challenges in RL post-training—bimodal probability distributions, reward hacking, and length collapse—that adversely affect faithfulness and factuality. We characterize their emergence during optimization and evaluate interventions based on rubric rewards and Monte Carlo score estimation.
2. **Up to approximately one order of magnitude greater label efficiency than SFT.** Across all three tasks, RL-Only models trained on 200–836 examples are competitive with SFT models trained on thousands of examples at the evaluated operating points. The results further indicate that extensive SFT may constrain subsequent RL exploration, whereas initialization from a base model retains greater exploratory capacity.
3. **Scaling behavior across data and compute.** Data-scaling experiments across all three tasks and rollout-scaling experiments on Tasks 1 and 2 indicate that performance generally improves and subsequently saturates, with task-dependent variability at larger rollout counts. These findings characterize the marginal returns associated with additional data and rollout computation.
4. **Integrity-specific reward and inference designs.** We develop a policy-grounded full-trace reward that combines accuracy, structure, reasoning length, and rubric adherence; a rationale-marginalized Monte Carlo estimator for reasoning-conditioned label probabilities; and policy-structured reflection over evidence sublabels. The corresponding evaluations examine reward design on Tasks 1 and 3, Monte Carlo scoring on Tasks 1 and 2, and reflection on Task 3.

2 Setup

2.1 Prompt

Content moderation can be posed as a classification problem: given a piece of content $\mathcal{C}$ and a policy $\mathcal{P}$, we aim to estimate the probability that $\mathcal{C}$ violates $\mathcal{P}$. Let the input prompt, q , be

$$q = \text{concat}(\mathcal{I}; \mathcal{P}; \mathcal{C}) \quad (1)$$

where $\mathcal{I}$ is the task instruction. The model induces a conditional distribution over labels $y \in \{y_1, \dots, y_K\}$ defined in the policy,

$$P_\theta(y = y_i | q), \quad i = 1, \dots, K \quad (2)$$

In this work we focus on binary classification ($K = 2$; i.e. violation vs non-violation), where we take $y = 1$ to indicate a policy violation and $y = 0$ otherwise.

The model outputs a structured triple

$$(r, \hat{y}, \hat{p}) \quad (3)$$

where $\hat{y} \in \{1, 0\}$ is the predicted label in language space (violation / non-violation), $\hat{p} = P_\theta(\hat{y} = 1 | q)$ is the associated probability to that label in language space, and r is the chain-of-thought reasoning of the model rationale to get to label $\hat{y}$ (Wei et al., 2022). At inference time we are interested in probability of class y_i given the context $P_\theta(y = y_i | q)$ and apply threshold τ

$$\hat{y} = \mathbb{I}[P_\theta(y = 1 | q) \geq \tau], \quad (4)$$

with threshold $\tau \in [0, 1]$ calibrated on a validation set to satisfy precision/recall or cost-sensitive targets.

Example 1 and 2 in Appendix A show the examples of input prompt and model generation.

2.2 Tasks and External Reference Points

We evaluate three binary moderation tasks derived from production data. To improve task interpretability while preserving the confidentiality of individual examples, we provide organization-neutral policy definitions, decision rules, contextual signals, and exceptions in a public [policy artifact](#). Table 1 summarizes the task scopes and dataset sizes. For each task, the positive class denotes an established policy violation.

We also evaluate the publicly available Gemini 2.0 Flash model and policy-trained FTE reviewers on the same held-out sets using the task-specific metrics. Table 2 reports these external reference points together with RL-Only and the strongest SFT→RL system. At the level of point estimates, RL-Only is comparable to Gemini on Tasks 1 and 3 and has a higher value on Task 2. SFT→RL has higher point estimates than Gemini on all three tasks and is comparable to the policy-trained reviewer baseline across the task suite.

2.3 Experimentation Setup

The experiments use Qwen2.5-VL-7B, Qwen2.5-VL-72B, Qwen3-VL-8B, and Qwen3-VL-32B. Training is conducted on NVIDIA B200 GPUs. The primary evaluation metric is recall at 90% precision (R@P90); area under the precision-recall curve (PRAUC) and F1 are reported where appropriate.

For our RL algorithm, we use Group Relative Policy Optimization (GRPO) (Shao et al., 2024), a more compute-efficient alternative to Proximal Policy Optimization (PPO) (Schulman et al., 2017). GRPO eliminates the need for an explicit value function by computing relative advantages across a group of sampled responses. Given a prompt q , we draw N rollouts $\{o_1, \dots, o_N\}$ from the current policy π_θ , obtain their scalar rewards $\{R_1, \dots, R_N\}$, and compute group-normalized advantages $A_i = \frac{R_i - \mu_R}{\sigma_R + \epsilon}$, where μ_R and σ_R denote the mean and standard deviation of the group rewards, respectively. The GRPO objective is then defined as:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \frac{1}{N} \sum_{i=1}^N \text{clip}\left(\frac{\pi_\theta(o_i | q)}{\pi_{\theta_{\text{old}}}(o_i | q)}, 1 - \epsilon, 1 + \epsilon\right) A_i - \beta \text{KL}(\pi_\theta(\cdot | q) \parallel \pi_{\theta_{\text{ref}}}(\cdot | q)), \quad (5)$$

which updates the policy toward higher-quality samples identified through groupwise comparison. This relative-feedback formulation avoids value estimation and improves optimization stability in settings with sparse or noisy reward signals.

In all experiments, we set the KL coefficient $\beta = 0$, following recommendations from (Liu et al., 2025; Shao et al., 2024). Empirically, removing the KL penalty yields consistently better performance in our content moderation tasks, as the policy benefits from stronger exploration without collapsing toward the initial SFT distribution.

We further apply sequence-level normalization of rewards, rather than token-level normalization, which we find substantially improves training stability and final model quality. This design choice aligns with guidance from GSPO (Zheng et al., 2025), where sequence-level normalization preserves the relative ordering of complete trajectories and yields more reliable optimization dynamics for reasoning-intensive tasks.

We implement RL training with VeRL (Verl-Team, 2024) and use vLLM (Kwon et al., 2023) for inference. VeRL is selected because it provides higher end-to-end training throughput for the

Task	Scope and positive label	RL train	Max. SFT train
1	Prohibited deceptive link or domain impersonation	661	60K
2	Prohibited fraud or scam	836	20K
3	Deceptive impersonation or identity misrepresentation	200	10K

Table 1: Summary of the task definitions and training-set sizes. The evaluation sets contain 4.6K, 5.0K, and 3.9K examples for Tasks 1–3, respectively.

Task	Metric	Gemini	FTE	RL-Only	SFT→RL
1	R@P90	0.673	0.92	0.67	0.95
2	R@P90	0.20	0.54	0.25	0.52
3	PRAUC	0.80	0.90	0.80	0.93

Table 2: External reference points obtained under a common evaluation protocol. FTE denotes policy-trained full-time employee reviewers. Boldface indicates the highest point estimate in each row.

evaluated workload; Appendix A.1 reports a comparison with HuggingFace TRL (von Werra et al., 2020). Rollout generation and reward evaluation are incurred during offline training for each model-policy version and are therefore amortized over subsequent deployment. Standard inference requires a single forward pass, whereas optional Monte Carlo scoring and reflection introduce additional inference computation in exchange for improved score quality.

3 Challenges

This section examines the key challenges of scaling and stabilizing Reinforcement Learning (RL) for content moderation. We focus specifically on *verifiable reward* frameworks, where supervision is derived on objective final-label correctness. This approach creates significant hurdles in reward design, verification, and optimization. Our empirical analysis details these limitations, providing the rationale for the methodologies proposed in later sections.

3.1 Data and Label Scarcity

Data scarcity is a primary bottleneck in content moderation due to the high cost and latency of expert labeling (Kiela et al., 2020; Alam et al., 2022). Typically, policy teams spend months training reviewers, who produce only a few hundred labels weekly. Because high-quality labels require iterative review and correction, scaling from a small seed set to several thousand examples often takes months and incurs a tenfold increase in costs.

3.2 Verification and Reward-Design

3.2.1 Lack of a Verification Process

Content moderation shares the complex, rule-driven reasoning requirements of code generation and mathematics but lacks a reliable mechanism for verifying intermediate steps. Unlike coding—where compilers provide deterministic feedback—or mathematics—where steps are symbolically checkable (Jha et al., 2024)—moderation has no “safety compiler” to audit reasoning chains. To address this, Section 3.3.2 introduces reflection mechanisms for self-evaluating and revising reasoning traces. Furthermore, Appendix B.1 demonstrates how rubric-based rewards serve as a proxy for explicit verification, improving overall reasoning quality.

3.2.2 Susceptible to Reward Hacking

In content moderation tasks, verifiable rewards based on final-label accuracy (R_{acc} , Eq. (8)) often lead to a performance ceiling characterized by reward hacking. As shown in Figure 3, the model’s reasoning length collapses—shrinking from 250 to under 50 words—resulting in semantically empty explanations that bypass deeper task logic to shortcut directly to the final label. This length collapse also creates a misleading improvement in factuality: shorter outputs contain fewer factual claims and thus fewer detectable errors, while the model increasingly generates post-hoc rationales rather than genuine, input-grounded reasoning (Huang et al., 2024). To mitigate this and reduce length-dependent bias, we implement sequence-level reward normalization inspired by GSPO (Zheng et al., 2025), which stabilizes optimization and preserves meaningful long-form reasoning.

3.3 Bimodal probability distribution

Following Eq. (3), for reasoning models, a generation given the input prompt q is $o = (r, \hat{y}, \hat{p})$. Thus, the token probability, $P_\theta(y | r, q)$, is a conditional probability dependent on both reasoning r and input prompt q , and we observe it is more bimodal in classification use cases (as shown in

Figure 6) because the conclusion about the correct class is often already included in the reasoning trace (see prompt example in Section 2.1). This bimodal score distribution leads to poor score-based deployment metrics (PRAUC and R@P90).

We investigate two mitigation strategies.

- **Monte-Carlo estimation**, which smooths the score distribution by aggregating probabilities over reasoning trace samples and provides an approximation of $P_\theta(y | q)$.
- **Reflection-aided prompting**, which introduces a reflective scheme that encourages the model to revisit its initial decision before producing a final label and reduces extreme polarization.

3.3.1 Monte-Carlo estimation

Our solution is to estimate the probability score through the Monte-Carlo method, which samples a sufficient number of responses, and approximates the overall probability. With the law of total probability, we have the following equation:

$$\begin{aligned} P_\theta(y | q) &= \sum_r P_\theta(y | r, q) P_\theta(r | q) \\ &= \mathbb{E}_{r \sim P_\theta(r|q)} [P_\theta(y | r, q)] \end{aligned} \quad (6)$$

The probability of output y given q is the expected conditional probability of y given q and a sampled reasoning trace r (COT), weighted by the likelihood of each reason. This approach helps in overcoming the challenges posed by the bimodal probability distribution observed in reasoning models by providing a more robust estimation through comprehensive sampling of the thought space. We evaluated two primary hyperparameters in the Monte Carlo sampling procedure: 1) the number of rollouts N and 2) the sampling temperature T in Appendix C.2.

3.3.2 Reflection-aided Prompting

We adopt a multi-stage prompting strategy for binary classification, inspired by (Shinn et al., 2023): the model (i) emits an initial label, (ii) reflects on evidence via sub-labels, and (iii) outputs a final label as Example 3. This design addresses the observation that the log-probability of the final label token is often extremely polarized, exacerbating the thresholding difficulty in Equation 4. As demonstrated in Table 7 and Figure 7, reflection-aided scoring produces substantially more stable and well-calibrated scores, whereas non-reflective

scoring exhibits highly bimodal distributions that destabilize threshold-based decisions.

4 Empirical Study of Scaling RL

Figure 1 reports the effect of RL after different levels of SFT across the three moderation tasks using Qwen2.5-VL-7B. The RL stage uses a small set of high-quality examples sampled according to the same procedure as the ground-truth dataset. The results yield three observations:

- **RL-Only is label-efficient across all three tasks.** With 200–836 labeled examples, RL-Only remains competitive with SFT trained on thousands of examples. At the evaluated operating points, this result corresponds to up to approximately one order of magnitude greater label efficiency.
- **SFT→RL provides substantial gains when SFT training is moderate.** For models initialized by SFT on thousands of examples, the subsequent RL stage yields higher point estimates across all tasks, with typical R@P90 improvements of 5–15 percentage points. These gains suggest that RL can correct residual errors and refine decision boundaries in this regime.
- **RL gains diminish and eventually saturate as SFT scale increases.** As the SFT dataset increases to tens of thousands of examples, the performance difference between SFT and SFT→RL narrows. One possible explanation is that extensive SFT provides a stronger initialization while also concentrating the policy around previously learned response patterns, thereby reducing the marginal benefit of subsequent RL exploration.

4.1 Number of training tokens in RL

Figure 2 shows that increasing the number of input tokens processed during RL improves performance until the marginal gains diminish at larger scales, consistent with the saturation behavior reported in prior work (Khatri et al., 2025).

Low token budgets (0.6B–1.2B): Performance is substantially limited, with both R@P90 and PRAUC remaining low. The wide confidence intervals indicate that the model receives too few comparisons to produce stable gradients.

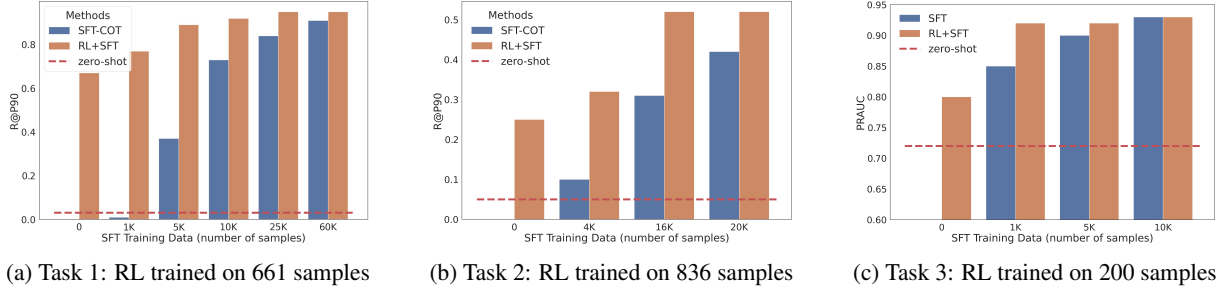


Figure 1: Supervised Fine-tuning data scaling and its impact on Reinforcement learning performance. First bar from the left (SFT data = 0) shows RL-Only training.

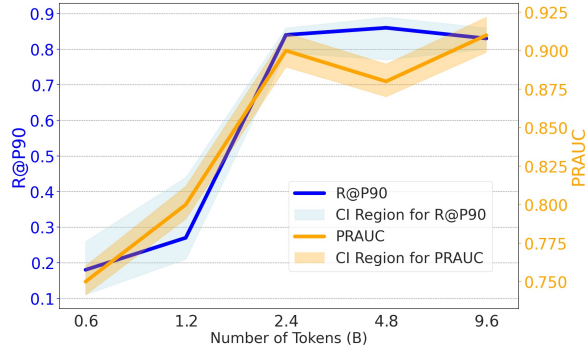


Figure 2: Increasing the data scale in RL training initially boosts model performance, but the gains plateau after a certain threshold.

Intermediate token budgets (2.4B): We observe a sharp increase in R@P90 and PRAUC, corresponding to the point at which the model has sufficient context to form reliable comparisons. This regime provides the largest marginal gains.

High token budgets (4.8B–9.6B): Performance saturates, and the confidence intervals for R@P90 overlap across settings. Additional tokens provide diminishing returns.

Appendix D presents additional analyses of the principal training hyperparameters.

- **Number of Rollouts.** Across Tasks 1 and 2, GRPO performance generally improves and subsequently saturates as the number of rollouts increases, with variability at larger rollout counts. For Task 1, the point estimate is 0.65 with 64 rollouts and 0.55 with 128 rollouts, indicating that additional rollouts do not yield uniform improvements.
- **Reward Shaping.** On Task 1, the complete reward increases F1 from 0.49 [0.41, 0.58] to 0.61 [0.51, 0.71] for Qwen3-VL-8B and from 0.47 [0.40, 0.55] to 0.60 [0.50, 0.70] for Qwen2.5-VL-7B. Although the direction

of the point-estimate changes is consistent, the overlapping confidence intervals do not support a claim of statistical significance.

- **Effective Batch Size.** In the Task 1 diagnostic, increasing the effective batch size from 128 to 1024 raises R@P90 from 0.18 to 0.81, after which the gains saturate.
- **Response Selection Improvement.** Our roll-out analysis suggests that RL primarily improves response selection and consistency among sampled outputs, rather than creating entirely new reasoning capability. In particular, RL increases $\text{maj}@N$ and narrows the gap between best-case and majority-vote performance, indicating that optimization shifts probability mass toward already reachable correct responses. We provide the bound analysis in Appendix D.2.1.

5 Task 1 Case Study: Disagreement Filtering

This controlled Task 1 case study examines whether model self-consistency can identify training examples with high learning value. We refer to this selection procedure as *Disagreement Filtering*.

We begin by prompting a pretrained base model (e.g., Qwen3-VL-8B) multiple times for each input to generate diverse reasoning paths and final predictions. We define *disagreement* examples as those for which the sampled predictions do not reach consensus. Among the remaining agreement examples, we further categorize an example as *easy* if all sampled predictions are correct, and *hard* if all predictions are incorrect. Our intuition is that disagreement examples are neither trivially easy nor irreducibly hard, and therefore provide an informative and stable learning signal for RL optimization.

We evaluate disagreement-based data filtering

Data Subsets	Data Size	F1	PRAUC
All	677	0.87 [0.86, 0.89]	0.85
Disagreement + Easy	601	0.86 [0.84, 0.87]	0.84
Easy	566	0.79 [0.77, 0.81]	0.80
Disagreement	61	0.88 [0.86, 0.89]	0.90

Table 3: Disagreement filtering results on Task 1 for Qwen3-VL-8B. Removing hard or easy examples preserves, and in some cases improves, overall performance despite using substantially fewer training samples.

on the Task 1 task. For each training example, we generate two rollouts at temperatures 0.7, 1.0, and 1.3, resulting in six trajectories per example. Varying the sampling temperature encourages diverse model behaviors and increases the likelihood of uncovering disagreement.

Starting from 677 total examples, this procedure yields 76 hard examples, 61 disagreement examples, and 566 easy examples. We then apply GRPO to different subsets of the dataset, with results summarized in Table 3.

Within this experiment, removing hard examples improves performance relative to training on the full dataset. This finding suggests that examples on which all sampled predictions are incorrect may introduce noisy or unstable reward signals, potentially leading to overfitting or suboptimal policy updates. GRPO trained on the smaller curated subsets consequently achieves performance comparable to, and in some settings higher than, training on the complete dataset.

The cross-task results in Figure 1 show that RL-Only is competitive with SFT systems trained on thousands of labels, corresponding to up to approximately one order of magnitude greater label efficiency at the evaluated operating points. Within Task 1, targeted selection further reduces the number of training examples: the 61-example disagreement subset attains $F1 = 0.88$, compared with $F1 = 0.87$ when all 677 examples are used.

These Task 1 results indicate that disagreement-based selection can improve sample efficiency in the evaluated setting and motivate its evaluation on additional moderation policies.

6 Conclusion

This work presents a systematic empirical study of reinforcement learning (RL) for LLM-based content moderation. Across three production-derived tasks, RL-Only systems trained on 200–836 labeled examples remain competitive with supervised fine-tuning (SFT) systems trained on thousands of exam-

ples, corresponding to up to approximately one order of magnitude greater label efficiency at the evaluated operating points. Data-scaling experiments on all three tasks and rollout-scaling experiments on Tasks 1 and 2 indicate that performance generally improves and subsequently saturates, with task-dependent variability at larger rollout counts. The strongest SFT→RL systems attain point-estimate performance comparable to policy-trained FTE reviewers across the three tasks.

Using established RL and inference primitives, the study develops integrity-specific designs for full-trace reward shaping, rationale-marginalized Monte Carlo scoring, and policy-structured reflection. Together with the analysis of length collapse, confidence polarization, and the faithfulness–factuality trade-off, these results characterize several design considerations for policy-grounded moderation systems.

Limitations

The production-derived examples cannot be released; however, the public policy artifact provides organization-neutral task definitions and decision rules. Disagreement filtering and the effective-batch-size sweep are evaluated only on Task 1, whereas the broader empirical evidence comprises data scaling on all three tasks and rollout scaling on Tasks 1 and 2. Equal reward weights serve as an untuned default, and the component ablation does not assess sensitivity to alternative weight assignments. RL rollout generation and reward evaluation require additional computation relative to SFT updates, although these costs are incurred during offline training for each model-policy version and are amortized over deployment. Standard single-pass inference does not require additional rollouts, whereas optional Monte Carlo scoring and reflection incur additional inference computation. Accordingly, the reported efficiency results concern label efficiency rather than computational efficiency as measured by floating-point operations.

Ethics Statement

This study utilizes de-identified data derived from real-world production traffic at META PLATFORMS, INC., processed in strict compliance with privacy regulations to preserve user anonymity. Labels were obtained from expert policy reviewers under protocols ensuring their psychological well-being and limiting exposure to sensitive content.

References

- Firoj Alam, Stefano Cresci, Tanmoy Chakraborty, Fabrizio Silvestri, Dimitar Dimitrov, Giovanni Da San Martino, Shaden Shaar, Hamed Firooz, and Preslav Nakov. 2022. [A survey on multimodal disinformation detection](#). In *Proceedings of the 29th international conference on computational linguistics*, pages 6625–6643.
- Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. [Constitutional ai: Harmlessness from ai feedback](#). *Preprint*, arXiv:2212.08073.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. [Training verifiers to solve math word problems](#). *arXiv preprint arXiv:2110.14168*.
- Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. [Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities](#). *arXiv preprint arXiv:2507.06261*.
- Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2023. [Safe rlhf: Safe reinforcement learning from human feedback](#). *arXiv preprint arXiv:2310.12773*.
- Phanideep Gampa, Akash Anil Valsangkar, Shailesh Choubey, and 1 others. 2023. [Prioritised moderation for online advertising](#). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2004–2012.
- Lan Gao, Oscar Chen, Rachel Lee, Nick Feamster, Chenhao Tan, and Marshini Chetty. 2025. ["i cannot write this because it violates our content policy": Understanding content moderation policies and user experiences in generative ai products](#). *arXiv preprint arXiv:2506.14018*.
- Google Cloud. 2023. Google cloud text moderation. <https://cloud.google.com/blog/products/ai-machine-learning/google-cloud-text-moderation>.
- Melody Y Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Helyar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, and 1 others. 2024. [Deliberative alignment: Reasoning enables safer language models](#). *arXiv preprint arXiv:2412.16339*.
- Zhenyu Hou, Xin Lv, Rui Lu, Jiajie Zhang, Yujiang Li, Zijun Yao, Juanzi Li, Jie Tang, and Yuxiao Dong. 2025. [Advancing language model reasoning through reinforcement learning and inference scaling](#). *arXiv preprint arXiv:2501.11651*.
- Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2024. [A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions](#). *ACM Transactions on Information Systems*, 42(1).
- Simon Hughes. 2023. [Cut the bull... detecting hallucinations in large language models](#). *Vectara Blog*.
- Huseyin Inan, Kartik Upasani, John Chi, and 1 others. 2023. [Llama guard: Llm-based input-output safeguard for human-ai conversations](#). *arXiv preprint arXiv:2312.06674*.
- Piyush Jha, Prithwish Jana, Pranavkrishna Suresh, Arnav Arora, and Vijay Ganesh. 2024. [Rlsf: Fine-tuning llms via symbolic feedback](#). *arXiv preprint arXiv:2405.16661*.
- Devvrit Khatri, Lovish Madaan, Rishabh Tiwari, Rachit Bansal, Sai Surya Duvvuri, Manzil Zaheer, Inderjit S Dhillon, David Brandfonbrener, and Rishabh Agarwal. 2025. [The art of scaling reinforcement learning compute for llms](#). *arXiv preprint arXiv:2510.13786*.
- Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. 2020. [The hateful memes challenge: Detecting hate speech in multimodal memes](#). *Advances in neural information processing systems*, 33:2611–2624.
- Deepak Kumar, Yousef Anees AbuHashem, and Zakir Durumeric. 2024. [Watch your language: Investigating content moderation with large language models](#). In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 18, pages 865–878.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. [Efficient memory management for large language model serving with pagedattention](#). In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles (SOSP 2023)*. ACM.
- Ziniu Li, Congliang Chen, Tianyun Yang, Tian Ding, Ruoyu Sun, Ge Zhang, Wenhao Huang, and Zhi-Quan Luo. 2025. [Knapsack rl: Unlocking exploration of llms via optimizing budget allocation](#). *arXiv preprint arXiv:2509.25849*.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. [Understanding rl-zero-like training: A critical perspective](#). *arXiv preprint arXiv:2503.20783*.
- Huan Ma, Changqing Zhang, Huazhu Fu, Peilin Zhao, and Bingzhe Wu. 2024. [Adapting large language models for content moderation: Pitfalls in data](#)

- engineering and supervised fine-tuning. *Preprint*, arXiv:2310.03400.
- Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. 2023. [A holistic approach to undesired content detection in the real world](#). In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 15009–15018.
- Todor Markov, Chong Zhang, Sandhini Agarwal, Florentine Mary Eloundou Nekoul, Theodore Lee, Steven Adler, Angela Jiang, and Lilian Weng. 2024. [Systems and methods for language model-based content classification](#). US Patent App. 18/308,586.
- Meta Platforms. 2024. Connect 2024: The responsible approach we are taking to generative ai. <https://ai.meta.com/blog/responsible-ai-connect-2024/>.
- Tong Mu, Alec Helyar, Johannes Heidecke, Joshua Achiam, Andrea Vallone, Ian Kivlichan, Molly Lin, Alex Beutel, John Schulman, and Lilian Weng. 2024a. [Rule based rewards for language model safety](#). *Advances in Neural Information Processing Systems*, 37:108877–108901.
- Tong Mu, Alec Helyar, Johannes Heidecke, Joshua Achiam, Andrea Vallone, Ian D Kivlichan, Molly Lin, Alex Beutel, John Schulman, and Lilian Weng. 2024b. [Rule based rewards for fine-grained llm safety](#). In *ICML 2024 Next Generation of AI Safety Workshop*.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. 2025. [s1: Simple test-time scaling](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20286–20332.
- OpenAI-Platform. 2023. Openai moderation system guide: Identify potentially harmful content in text and images. <https://platform.openai.com/docs/guides/moderation>.
- Wei Qiao, Tushar Dogra, Otilia Stretcu, Yu-Han Lyu, Tiantian Fang, Dongjin Kwon, Chun-Ta Lu, Enming Luo, Yuan Wang, Chih-Chun Chia, and 1 others. 2024. [Scaling up llm reviews for google ads content moderation](#). In *Proceedings of the ACM International Conference on Web Search and Data Mining*.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *arXiv preprint arXiv:1707.06347*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#). *arXiv preprint arXiv:2402.03300*.
- Mrinank Sharma, Meg Tong, Jesse Mu, Jerry Wei, Jorrit Kruthoff, Scott Goodfriend, Euan Ong, Alwin Peng, Raj Agarwal, Cem Anil, and 1 others. 2025. [Constitutional classifiers: Defending against universal jailbreaks across thousands of hours of red teaming](#). *arXiv preprint arXiv:2501.18837*.
- Shivam Sharma, Firoj Alam, Md Shad Akhtar, Dimitar Dimitrov, Giovanni Da San Martino, Hamed Firooz, Alon Halevy, Fabrizio Silvestri, Preslav Nakov, and Tanmoy Chakraborty. 2022. [Detecting and understanding harmful memes: A survey](#). *arXiv preprint arXiv:2205.04274*.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. [Reflexion: Language agents with verbal reinforcement learning](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 8634–8652.
- Verl-Team. 2024. [Verl: Efficient reinforcement learning for llms](#).
- Leandro von Werra and 1 others. 2020. [Transformers reinforcement learning \(trl\)](#).
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). *Advances in neural information processing systems*, 35:24824–24837.
- Laura Weidinger, Joslyn Barnhart, Jenny Brennan, Christina Butterfield, Susie Young, Will Hawkins, Lisa Anne Hendricks, Ramona Comanescu, Oscar Chang, Mikel Rodriguez, and 1 others. 2024. [Holistic safety and responsibility evaluations of advanced ai models](#). *arXiv preprint arXiv:2404.14068*.
- Fan Yin, Philippe Laban, Xiangyu Peng, Yilun Zhou, Yixin Mao, Vaibhav Vats, Linnea Ross, Divyansh Agarwal, Caiming Xiong, and Chien-Sheng Wu. 2025. [Bingoguard: Llm content moderation tools with risk levels](#). *arXiv preprint arXiv:2503.06550*.
- Yuan Yuan, Tina Sriskandarajah, Anna-Luisa Brakman, Alec Helyar, Alex Beutel, Andrea Vallone, and Saachi Jain. 2025. [From hard refusals to safe-completions: Toward output-centric safety training](#). *arXiv preprint arXiv:2508.09224*.
- Zhuowen Yuan, Zidi Xiong, Yi Zeng, Ning Yu, Ruoxi Jia, Dawn Song, and Bo Li. 2024. [Rigorllm: Resilient guardrails for large language models against undesired content](#). *arXiv preprint arXiv:2403.13031*.

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

Yichi Zhang, Zihao Zeng, Dongbai Li, Yao Huang, Zhijie Deng, and Yinpeng Dong. 2025. [Realsafe-r1: Safety-aligned deepseek-r1 without compromising reasoning capability.](#) *arXiv preprint arXiv:2504.10081*.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, and 1 others. 2025. [Group sequence policy optimization.](#) *arXiv preprint arXiv:2507.18071*.

A Setup Examples and Appendix Roadmap

This appendix provides supporting material for the main claims. It first gives prompt and output examples and reports the framework throughput comparison that motivates our implementation choice. It then defines the shaped reward used during RL training before presenting diagnostics for reward hacking, faithfulness, and factuality. The remaining sections separate inference-time score calibration from training-time scaling ablations.

Appendix Navigation.

- **Setup examples and frameworks** (Appendix A) gives prompt/output templates and the VeRL-vs.-TRL throughput comparison.
- **Training-time reward design and diagnostics** (Appendix B) defines the shaped reward and analyzes reward hacking, faithfulness, and factuality.
- **Inference-time score calibration** (Appendix C) covers bimodal label-token probabilities, Monte Carlo score estimation, and reflection-aided scoring.
- **Training-time scaling ablations** (Appendix D) studies rollout count, effective batch size, and response-selection bounds.

A.1 Framework Throughput Comparison

Section 2.3 describes the use of VeRL for RL training. Here, we report the supporting throughput comparison against HuggingFace TRL (von Werra et al., 2020). We define throughput as the total

Imagine you are reviewing for [CONTENT_MODERATION_TASK]. Given the following content, determine the answer.

Content: [CONTENT_MULTIMODAL] Account information:

User/Account name: [USERNAME],
Bio: [BIO_TEXT].
Account stated country: [COUNTRY_CODE],
Follower count: [NUMBER],
Recent contents: [CONTENT_MULTIMODAL_1]

The review process should follow the following step:
Step 1. [POLICY_1] [POLICY_1_CONTENT].
Step 2. [POLICY_2] [POLICY_2_CONTENT].
Step 3. [CONCLUSION] Provide final assessment.

Please think step-by-step then provide your independent assessment in JSON format, including the following keys:
[KEY_1]: for [POLICY_1] (true/false)
[KEY_2]: for [POLICY_2] (no/yes)
[KEY_3]: for [CONCLUSION] (true/false)

Example 1: Input Prompt

```
<think>Okay, let's start by going through each step carefully.
**Step 1: ...
**Step 2: ...
**Step 3: ...
**Final Decision: Since ... is false.
</think>

{
  [KEY_1]: "false",
  [KEY_2]: "yes",
  [KEY_3]: "true"
}
```

Example 2: Model Generation

number of tokens (input + output) processed by a single GPU per second:

$$\text{Throughput} = \frac{\text{Total tokens processed}}{\text{Number of GPUs} \times \text{Time}}, \quad (7)$$

where Time is measured in seconds.

As shown in Table 4, VeRL (Verl-Team, 2024), due to its HybridFlow-based execution backbone, consistently achieves substantially higher throughput than TRL, reaching up to $2.5\times$ improvement under comparable settings on both internal tasks and the external GSM8K benchmark dataset (Cobbe et al., 2021).

```

<think>Okay, let's start by going through each step
carefully.
**Step 1: ... sub_label_1 is false
**Step 2: ... sub_label_2 is true
**Step 3: Final Decision** Since ... is false.
</think>

{
  "first_decision": true/false,
  "sub_label_1": true/false,
  "sub_label_2": true/false,
  ...
  "last_decision": true/false
}

```

Example 3: Example output format for reflection-aided binary classification with an initial decision, evidence sub-labels, and a last decision.

Dataset	VeRL	TRL	Speedup
Task 2	4600	1854	2.5x
GSM8K	1500	730	2.0x

Table 4: Training efficiency comparison (tokens/s/GPU) with Qwen2.5-VL-7B on Task 2 and GSM8K.

B Training-Time Reward Design and Diagnostics

This section expands the reward-design discussion from Section 3.2. We first define the shaped reward used for training, then show why the additional reward terms are necessary by analyzing length collapse, faithfulness, and factuality.

B.1 Reward Shaping

Taken together, findings in Section 3.2 illustrate a central tension in RL for LLMs: structural stability and semantic grounding do not naturally co-evolve. To encourage the model to reason with sufficient detail while remaining grounded in the moderation policy, we combine four reward signals:

1. **Final Verifiable Accuracy Reward (R_{acc}):** A deterministic reward that checks whether the model’s final binary prediction matches the ground-truth label. This reward is fully verifiable and does not require an external judge.
2. **Format Reward (R_{fmt}):** A structural reward that ensures the model emits both its reasoning trace and final answer in the expected format, such as reasoning tags, a JSON schema, or answer markers.

3. **Targeted Reasoning Length Reward (R_{len}):**

A reward that encourages outputs to fall within a desired length range, giving the model sufficient room for reasoning and preventing collapse into short, label-only responses.

4. **Rubric-Based Reasoning Reward (R_{rub}):**

A qualitative reward that evaluates the reasoning trace beyond the final binary label. It assesses instruction adherence, policy consistency, and correct application of task-specific criteria. The rubric is implemented using either an LLM-as-a-judge or human-provided rubric annotations. For example, in a profile-matching task, the judge is asked: “*Does the reasoning correctly compare the two profile images according to the rubric?*” The resulting score is assigned to the full generation, and we apply inverse-frequency weighting when aggregating rewards so that high-frequency labels do not dominate optimization.

We combine these terms into a single scalar reward:

$$R_{total} = \alpha_{acc} R_{acc} + \alpha_{fmt} R_{fmt} + \alpha_{len} R_{len} + \alpha_{rub} R_{rub}, \quad (8)$$

where α_{acc} , α_{fmt} , α_{len} , and α_{rub} are non-negative weighting coefficients satisfying $\alpha_{acc} + \alpha_{fmt} + \alpha_{len} + \alpha_{rub} = 1$. In the reported experiments, the four components receive equal weights as an untuned default. The component ablation therefore measures the incremental contribution of the reward terms rather than sensitivity to alternative weight assignments. A systematic evaluation of the weighting coefficients is left for future work.

The accuracy and format rewards provide sparse but reliable supervision. The length reward discourages shortcut solutions that omit reasoning, whereas the rubric reward continues to provide a learning signal after the outcome reward has saturated. On Task 1, the complete reward increases F1 from 0.49 [0.41, 0.58] to 0.61 [0.51, 0.71] for Qwen3-VL-8B and from 0.47 [0.40, 0.55] to 0.60 [0.50, 0.70] for Qwen2.5-VL-7B. Although the point estimates increase consistently, the overlapping confidence intervals do not support a claim of statistical significance. On Task 3, adding the rubric reward increases PRAUC by 0.04 relative to the accuracy-plus-format baseline when rubric labels are obtained through human annotation.

Model	Reward Setup	Recall	Precision	F1
Qwen3-VL-8B	$R_{\text{acc}} + R_{\text{fmt}}$ (Baseline)	0.58 (0.47, 0.68)	0.43 (0.34, 0.52)	0.49 (0.41, 0.58)
	$R_{\text{acc}} + R_{\text{fmt}} + R_{\text{len}}$	0.61 (0.48, 0.75)	0.41 (0.30, 0.54)	0.49 (0.38, 0.60)
	$R_{\text{acc}} + R_{\text{fmt}} + R_{\text{len}} + R_{\text{rub}}$	0.71 (0.59, 0.83)	0.54 (0.43, 0.67)	0.61 (0.51, 0.71)
Qwen2.5-VL-7B	$R_{\text{acc}} + R_{\text{fmt}}$ (Baseline)	0.49 (0.39, 0.59)	0.46 (0.37, 0.55)	0.47 (0.40, 0.55)
	$R_{\text{acc}} + R_{\text{fmt}} + R_{\text{len}}$	0.67 (0.55, 0.80)	0.45 (0.34, 0.57)	0.54 (0.44, 0.63)
	$R_{\text{acc}} + R_{\text{fmt}} + R_{\text{len}} + R_{\text{rub}}$	0.69 (0.57, 0.81)	0.54 (0.42, 0.67)	0.60 (0.50, 0.70)

Table 5: Task 1 performance under different reward configurations. Values in parentheses denote confidence intervals. Because the intervals overlap, the observed gains are interpreted as directional rather than statistically significant.

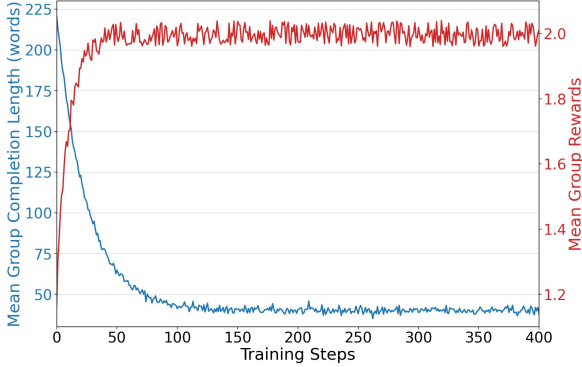


Figure 3: Accuracy-based rewards induce reward hacking: explanation length collapses over training, and responses degenerate into short label assertions.

B.2 Reward Hacking

For our content-moderation tasks, we find that simple, verifiable rewards based on final-label matching (R_{acc} in Eq. (8)) quickly reach a performance ceiling.

Figure 3 shows a typical trajectory in which the model’s reasoning length steadily collapses (from roughly 250 words to fewer than 50), yielding extremely brief or semantically empty explanations followed by a bare True/False prediction. This behavior indicates that the model is not learning deeper task structure or producing meaningful reasoning chains; instead, it is exploiting the reward by shortcutting directly to the final label. To reduce length-dependent bias in the GRPO updates, we apply sequence-level reward normalization, following the stabilization strategy proposed in GSPO (Zheng et al., 2025), which improves optimization behavior for long-form reasoning tasks.

B.2.1 Factuality and Faithfulness

We observe a distinct trade-off between two types of hallucination during RL optimization: (1) **faithfulness**, which measures the model’s ability to follow instructions, and (2) **factuality**, which

measures the model’s adherence to the factual information specified in the policy. To evaluate both faithfulness and factuality of the trained policy model, we use the state-of-the-art LLM-based judge Gemini-2.5-Pro (Comanici et al., 2025). For factuality in particular, we further leverage the Hughes Hallucination Evaluation Model (HHEM) (Hughes, 2023).

As training progresses, RL-optimized policies often appear to improve instruction following (Huang et al., 2024), while their measured factuality error rate decreases. However, this apparent improvement is largely an artifact of *length collapse*: the policy learns to produce increasingly short outputs—an effect illustrated in Figure 3—which reduces the number of explicit factual statements and therefore the number of opportunities for detectable errors. In practice, the model’s underlying grounding does not improve. Instead, the policy increasingly generates post-hoc rationales crafted to align with the ground-truth label, rather than engaging in genuine, input-grounded reasoning.

To further examine these optimization dynamics, we compare two common training recipes for LLM-based classifiers:

(a) Direct RL (RL-Only). Applying RL directly to the base model leads to a severe degradation in *instruction adherence*, manifesting as increased instruction hallucination, as shown in Figure 4a. Without the inductive bias provided by supervised demonstrations, the policy fails to converge to a stable reasoning schema and instead produces unstructured, off-topic, or constraint-violating outputs. This behavior reflects classical under-regularization in policy optimization, in which the model exploits reward shortcuts rather than learning task-consistent reasoning patterns.

(b) Two-Stage Training (SFT → RL). Initializing with SFT substantially stabilizes RL, anchoring

the model in a coherent reasoning structure that prevents the degenerate behaviors observed in the RL-Only setting. However, this regularity introduces a secondary issue: a higher incidence of *factual-ity hallucinations*, as shown in Figure 4b. Under optimization pressure, the model generates plausible, well-structured rationales crafted to support the ground-truth label, even when these explanations lack semantic correctness or grounding in the content moderation policy specified in the prompt. Thus, although the reasoning format remains intact, the underlying factuality does not improve—and may in fact degrade.

In our experiments, we evaluate **instruction hallucination**, defined as the complement of faithfulness; thus, higher faithfulness denotes a closer adherence to instructions without invention. Similarly, we assess **factuality hallucination**, the complement of factuality, where higher scores indicate reasoning that remains grounded in evidence and free from unsupported claims.

C Inference-Time Score Calibration

C.1 Bimodal Probability Distribution

Following Eq. (3), a reasoning-model generation is $o = (r, \hat{y}, \hat{p})$. The label-token probability $P_\theta(y | r, q)$ is therefore conditioned on both the prompt and the sampled reasoning trace. Because the reasoning often already states the conclusion, label scores become highly polarized, producing the bimodal behavior shown in Figure 6. This appendix section evaluates the two inference-time mitigations introduced in Section 3.3.1 and Section 3.3.2: Monte Carlo score estimation and reflection-aided scoring.

C.2 Monte Carlo Score Estimation

N	Gain on R@P90
1	0.03
3	0.04
4	0.05
5	0.05
8	0.03
16	0.05
32	0.05

Table 6: Performance gain of Task 2 after applying Monte-Carlo sampling at $T = 1.0$ for different N , compared to baseline of $T = 0.0$ and $N = 1$

Monte Carlo scoring estimates $P_\theta(y | q)$ by sampling multiple reasoning traces and averaging the

conditional label probability across them. By the law of total probability:

$$\begin{aligned} P_\theta(y | q) &= \sum_r P_\theta(y | r, q) P(r | q) \\ &= \mathbb{E}_{r \sim P_\theta(r|q)} [P_\theta(y | r, q)] \end{aligned} \quad (9)$$

The resulting score averages over the sampled reasoning space rather than relying on a single, potentially overconfident rationale-conditioned label probability.

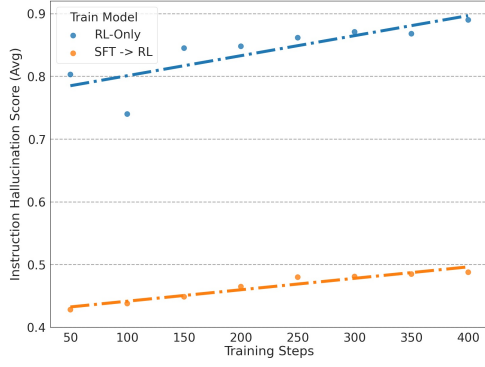
We tune two primary hyperparameters in the Monte Carlo sampling procedure: the number of rollouts N and the sampling temperature T . Figure 5 illustrates how varying N and T affects model performance.

1. **Effect of the number of rollouts.** At moderate sampling temperatures ($T \leq 1.0$), increasing the number of rollouts N consistently improves performance, albeit with diminishing returns. Table 6 reports the performance gains obtained by Monte Carlo sampling at $T = 1.0$ for different values of N . We observe that performance plateaus beyond $N = 4$, indicating that test-time scaling (Muennighoff et al., 2025) can be effective.
2. **Effect of sampling temperature.** We find that the optimal sampling temperature lies between 0.7 and 1.0. For temperatures above 1.0, performance degrades due to an increased incidence of parsing errors and generation anomalies, which negatively impact downstream evaluation.

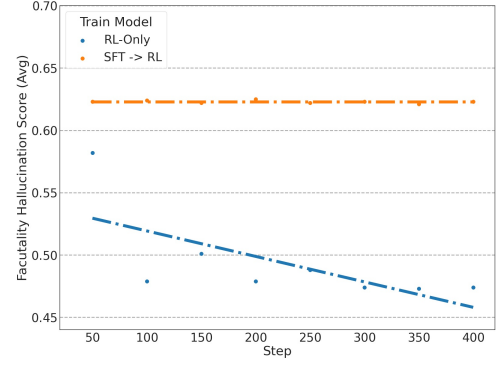
To confirm how Monte-Carlo sampling helps with mitigating bimodal probability distribution, we compare the label token distribution with and without our sampling strategy in Figure 6. We can observe that such strategy indeed shifts some of the extreme probability mass into the center to account for model uncertainty.

C.3 Reflection-Aided Scoring

Inspired by work on LLM reflection during reasoning (Shinn et al., 2023), we use a three-stage binary-classification template: the model first emits an initial label (*first decision*), then reflects on evidence through sub-labels, and finally emits a *last decision*. Example 3 shows the expected output format. This structure lets us score the earlier decision point before the final conclusion has been reinforced by the reasoning trace, reducing the extreme



(a) Faithfulness: Instruction hallucination score at each step



(b) Factuality hallucination score at each step

Figure 4: Faithfulness and factuality under RL ablations. “Instruction hallucination” and “factuality hallucination” are the complements of faithfulness and factuality, respectively.

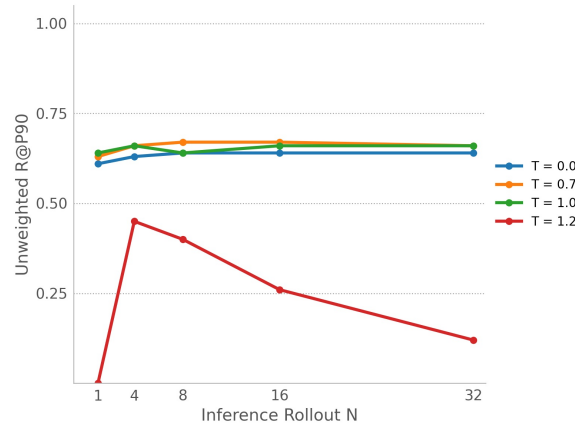
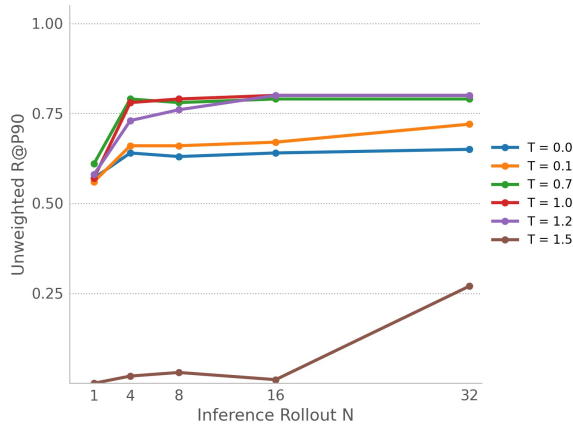


Figure 5: Monte-Carlo Sampling at Different N (number of rollouts) and T (temperature) for Task 1 and Task 2

Scoring method	PRAUC	R@P90
Without Reflection	0.77	0.05
With Reflection	0.89	0.59

Table 7: Performance of the LLM moderator with and without reflection for Task 3.

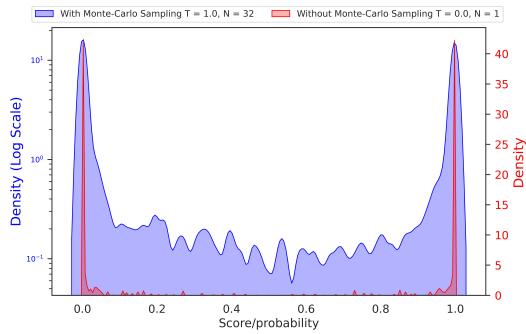


Figure 6: Comparison of label token probability distribution: without Monte-Carlo sampling vs with Monte-Carlo sampling

final-token polarization that complicates thresholding in Eq. (4).

Table 7 shows that reflection substantially improves PRAUC and R@P90 for the same model. Figure 7 shows the mechanism: reflection produces a less polarized probability distribution than scoring without reflection, improving calibration-sensitive metrics even when the final predicted labels are unchanged.

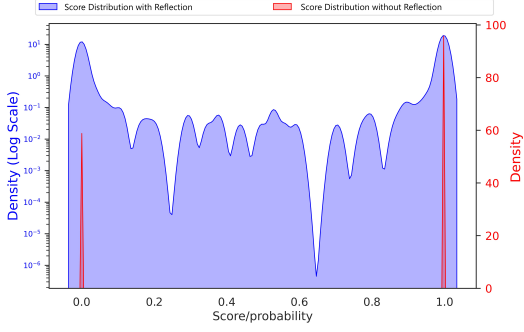


Figure 7: Comparison of score/probability distributions with and without reflection.

Number of rollouts	Task 1	Task 2
8	0.17	0.53
16	0.18	0.56
32	0.31	0.63
64	0.65	0.62
128	0.55	0.64

Table 8: Performance for Task 1 and Task 2 versus number of rollouts.

D Training-Time Scaling Ablations

D.1 Number of Rollouts

Across Tasks 1 and 2, GRPO performance generally improves and subsequently saturates as the number of rollouts increases, although the relationship is not uniformly monotonic. For Task 1, performance increases from 0.17 with eight rollouts to 0.65 with 64 rollouts and then decreases to 0.55 with 128 rollouts; Task 2 attains its highest point estimate with 128 rollouts. Larger rollout groups may provide more reliable relative comparisons, but their marginal benefit depends on the task and operating point (Hou et al., 2025; Li et al., 2025).

In practice, however, the use of large rollout counts is limited by the capacity of the LLM-based judge used for rubric-based rewards (R_{rub} , Section B.1), particularly when multiple rubric criteria must be evaluated in parallel.

D.2 Effective Batch Size: Task 1 Diagnostic

In the Task 1 diagnostic, effective batch size substantially affects training stability and convergence. As shown in Table 9, increasing the effective batch size from 128 to 1024 raises R@P90 from 0.18 to 0.81. Performance plateaus at approximately 0.85 for batch sizes of 2048 and above, indicating diminishing returns beyond this threshold under

Effective batch size	Task 1 R@P90
128	0.18
1024	0.81
2048	0.85
4096	0.85

Table 9: Task 1 Unweighted R@P90 by effective batch size

the evaluated configuration. In distributed training frameworks such as TRL and VeRL, the effective batch size is computed as the product of three key components:

$$\text{Effective Batch Size} = B_{\text{local}} \times N_{\text{GPU}} \times N_{\text{accum}} \quad (10)$$

where each term represents:

- Local batch size (B_{local}): The number of samples processed per GPU device in a single forward pass. This parameter is constrained by GPU memory capacity and depends on model size and input sequence length. In TRL terminology, this is referred to as “batch size per device,” while VeRL uses the term “micro-batch size.”
- Number of GPUs (N_{GPU}): The total number of GPU devices available for distributed training. This represents the available computational resources and enables data parallelism across multiple devices.
- Gradient accumulation steps (N_{accum}): The number of forward-backward passes performed before updating model parameters. This hyperparameter (denoted as `gradient_accumulation_steps` in most frameworks) requires careful tuning to balance training efficiency and gradient quality.

For Task 1, these results support an effective batch size of at least 1024 under the evaluated configuration. Because the sweep is restricted to one task, this threshold requires validation for other policies, model sizes, and training systems.

D.2.1 RL improves response selection on rollouts: bound analysis

Drawing from recent findings (Yue et al., 2025; Shao et al., 2024), RL often improves performance not by enhancing a model’s underlying reasoning ability, but by increasing the probability of selecting a correct answer from a set of candidate generations. In this view, RL fine-tuning functions

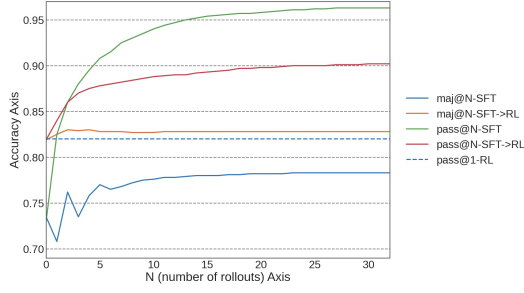


Figure 8: Comparison of accuracy as a function of rollouts N for SFT and SFT→RL trained models

primarily as a *response selection* mechanism rather than a fundamental *capability enhancement* step.

To examine whether this dynamic also holds for content moderation, we evaluate an SFT baseline and a two-stage SFT→RL model on the Task 2 dataset, with results shown in Figure 8. We measure performance using two complementary sampling-based metrics over varying numbers of rollouts ($N \in [1, 32]$):

pass@ N : The probability that at least one of N independent rollouts yields a correct response. This metric reflects the model’s *best-case correctness* and directly relates to improvements driven by R_{acc} : higher accuracy rewards increase the chance that at least one sampled output is correct.

maj@ N : The probability that at least half of the N rollouts are correct. This metric captures *consistency* and it complements R_{acc} by measuring whether RL fine-tuning increases the model’s reliability across samples, not just the probability of producing a single correct output.

Key Observations:

1. **Impact of RL Fine-Tuning:** Following RL training on top of SFT, maj@ N demonstrates substantial improvement, increasing from 0.72 to 0.82 at $N = 1$, and from 0.77 to 0.83 at $N = 32$. This indicates that RL effectively increases the model’s propensity to generate correct responses.
2. **Convergence of Metrics:** The gap between pass@ N -SFT and maj@ N -SFT→RL narrows considerably compared to the SFT baseline (pass@ N -SFT vs. maj@ N -SFT). This convergence suggests that RL training improves output consistency across multiple rollouts, making the model’s behavior more deterministic and reliable.
3. **Performance Ceiling Estimation:** The differ-

ence between pass@ N -SFT and pass@1-SFT serves as a (loose) upper bound on the potential performance gains achievable through RL-based optimization in two stage SFT→RL training. If this gap is small, it indicates limited headroom for improvement via better response selection strategies.