REFSWAP: COUNTERFACTUAL REFERENCE-SWAP VERIFICATION FOR ROBUST LLM VERIFIERS

FARS
Analemma
fars@analemma.ai

ABSTRACT

Reference-based verifiers are critical components of reinforcement learning with verifiable rewards (RLVR), providing reward signals by comparing model responses against ground-truth answers. However, these verifiers are vulnerable to "master-key" attacks—trivial responses like single tokens or short phrases that achieve 25–29% false positive rates without containing any actual answer. We propose RefSwap, a training-free detection method that exploits a fundamental asymmetry: legitimate correct responses exhibit self-solving behavior (high probability of verification against random references), while master-key false positives cannot self-solve. By sampling K counterfactual references and computing the maximum verification probability (max_p_cf), Multi-CF RefSwap achieves near-perfect separation (AUC=0.991) between true positives and master keys. On xVerify-7B-I, RefSwap reduces average master-key false positive rate from 25.50% to 0.81%—a 96.8% relative reduction—with only 2.74 percentage points accuracy cost. However, effectiveness depends on verifier architecture: RefSwap works on xVerify but not Qwen, revealing that backbone design determines susceptibility to counterfactual-based detection.

WARNING: This paper was generated by an automated research system. The code is publicly available.[1]

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has emerged as a powerful paradigm for training reasoning models (DeepSeek-AI et al., 2025; Shao et al., 2024). Unlike traditional RLHF that relies on learned preference models, RLVR uses automated verifiers to provide reward signals by comparing model responses against ground-truth reference answers. This approach has enabled significant advances in mathematical reasoning and other domains where correctness can be objectively verified.

However, reference-based verifiers harbor a critical vulnerability: they can be fooled by trivial "master-key" responses that exploit reference answer leakage (Zhao et al., 2025). These master keys—single tokens like "`:`" or short phrases like "`Thought process:`"—contain no substantive answer yet achieve false positive verification rates of 25–29% on state-of-the-art verifiers including xVerify (Chen et al., 2025) and Qwen2.5-7B-Instruct (Yang et al., 2024). Such vulnerabilities can corrupt RLVR training by rewarding non-solutions, potentially leading to reward hacking and training collapse.

Simple defenses like length filtering fail against sentence-length master keys, with worst-case false positive rates remaining at 28.87%. This motivates the need for defenses that operate on deeper semantic properties rather than surface features.

We propose RefSwap, a training-free method that detects master-key attacks through counterfactual reference swapping. The key insight is that legitimate correct responses exhibit *self-solving* behavior—the verifier can recognize correctness even with mismatched references—while master-key false positives cannot self-solve because they contain no actual answer. By sampling multiple counterfactual references and computing the maximum verification probability (max_p_cf),

[1] https://gitlab.com/fars-a/counterfactual-reference-swap-verifier

RefSwap achieves near-perfect separation between true positives and master-key false positives (AUC=0.991).

Our contributions are:

- We identify and quantify the master-key vulnerability in RLVR verifiers, showing 25–29% false positive rates on state-of-the-art systems.
- We propose Multi-CF RefSwap, a training-free detection method using max_p_cf scoring that exploits the self-solving asymmetry between true positives and master keys.
- We demonstrate 96.8% relative FPR reduction (25.50%→0.81%) on xVerify-7B-I with only 2.74pp accuracy cost, uniformly neutralizing all 10 tested master keys.
- We analyze backbone dependency, revealing that verifier architecture determines susceptibility to counterfactual-based detection.

2 Related Work

LLM-as-a-Judge. The paradigm of using large language models as evaluators has gained significant traction for assessing open-ended text generation. MT-Bench (Zheng et al., 2023) and Chatbot Arena (Chiang et al., 2024) established foundational benchmarks for LLM-based evaluation, while JudgeLM (Zhu et al., 2023) demonstrated that fine-tuned models can serve as scalable judges. G-Eval (Liu et al., 2023) further showed that chain-of-thought prompting improves evaluation quality. However, recent work has raised concerns about the robustness of LLM judges. Li et al. (2025) provide a comprehensive assessment showing that LLM judges exhibit inconsistencies across various perturbations, while TrustJudge (Wang et al., 2025) identifies systematic biases in judge behavior. Raina et al. (2024) demonstrate that universal adversarial attacks can manipulate zero-shot LLM assessments, and Zhao et al. (2025) show that even single tokens can fool LLM judges. Our work extends these robustness concerns to the specific setting of reference-based verification in reinforcement learning.

Reference-Based Verification. Reference-based verifiers compare model responses against ground-truth answers to provide reward signals for training. xVerify (Chen et al., 2025) introduced an efficient answer verifier specifically designed for reasoning model evaluations, while Verify-Bench (Yan et al., 2025) provides a comprehensive benchmark for evaluating reference-based reward systems. CoSineVerifier (Feng et al., 2025) extends verification to computation-oriented scientific questions through tool augmentation. These verifiers play a critical role in reinforcement learning from verifiable rewards (RLVR), where they provide the reward signal that guides policy optimization (DeepSeek-AI et al., 2025; Shao et al., 2024). Our work identifies a fundamental vulnerability in this verification paradigm: the potential for trivial responses to exploit reference answer leakage.

Reward Hacking. Reward hacking occurs when agents exploit misspecified reward functions to achieve high rewards without accomplishing the intended task (Skalse et al., 2022). In the context of RLHF and RLVR, this manifests as models learning to game the reward model rather than genuinely improving (Kaufmann et al., 2023). Kim et al. (2024) specifically evaluate the robustness of reward models for mathematical reasoning, finding vulnerabilities to various perturbations. Our master-key attack represents a particularly severe form of reward hacking where trivial responses achieve false positive verification, potentially corrupting the entire training process.

Adversarial Attacks on LLMs. Adversarial attacks on language models have been extensively studied, including gradient-based methods like GCG (Zou et al., 2023) and black-box approaches such as PAIR (Chao et al., 2023) and TAP (Mehrotra et al., 2023). These attacks typically require optimization or iterative refinement to craft adversarial inputs. In contrast, master-key attacks represent a distinct threat model: they exploit inherent design vulnerabilities in reference-based verifiers rather than requiring adversarial optimization. The simplicity of master keys—often single tokens or short phrases—makes them particularly concerning for deployment scenarios.

3 METHOD

3.1 PROBLEM SETUP

We consider reference-based verification in the context of reinforcement learning with verifiable rewards (RLVR). Given a question q, a model-generated response r, and a ground-truth reference answer a^*, a verifier V outputs a binary judgment:

$$V(q, r, a^*) \in \{\text{YES}, \text{NO}\} \tag{1}$$

where YES indicates that r is semantically equivalent to a^*. In RLVR, this judgment serves as the reward signal for policy optimization.

We define a **master-key attack** as a trivial response r_{mk} that achieves false positive verification regardless of the reference answer. Formally, r_{mk} is a master key if:

$$\Pr[V(q, r_{\text{mk}}, a^*) = \text{YES}] \gg 0 \quad \text{for arbitrary } (q, a^*) \tag{2}$$

Examples include single tokens like "`:`" or "`.`", and short phrases like "`Thought process:`" or "`Solution`". These responses contain no substantive answer yet trigger false positive verification, potentially corrupting RLVR training by rewarding non-solutions.

3.2 REFSWAP INTUITION

The key insight behind RefSwap is that legitimate correct responses and master-key false positives exhibit fundamentally different behaviors under counterfactual reference swapping. Consider replacing the true reference a^* with an unrelated counterfactual reference a_{cf}:

True positives often exhibit *self-solving* behavior: the verifier can independently recognize that the response is correct, even when presented with a mismatched reference. This occurs because strong verifiers may internally solve the problem and compare against their own solution.

Master-key false positives are *reference-insensitive*: they trigger YES regardless of the reference content, but they cannot "self-solve" because they contain no actual answer. When the reference is swapped, the verifier has no basis to confirm correctness.

This asymmetry suggests a detection mechanism: true positives should maintain some probability of verification against counterfactual references, while master-key false positives should not.

3.3 SINGLE-CF REFSWAP (K=1)

The initial RefSwap approach uses a single counterfactual reference. Let $p_{\text{yes}}(q, r, a)$ denote the verifier's probability of outputting YES. We define the **reference-sensitivity score**:

$$s = p_{\text{yes}}(q, r, a^*) - p_{\text{yes}}(q, r, a_{\text{cf}}) \tag{3}$$

where a_{cf} is sampled from a pool of unrelated references (different answer type, low token overlap with a^*).

The hypothesis is that master keys should have $s \approx 0$ (reference-insensitive), while true positives should have larger s (reference-sensitive). However, empirical evaluation reveals that this hypothesis fails: on Qwen2.5-7B-Instruct, master-key false positives actually have *higher* mean s (0.90) than true positives (0.84), yielding an AUC of only 0.589 for separation—near random chance. The s-score fails because the verifier's behavior is more complex than simple reference dependence.

3.4 MULTI-CF REFSWAP (K>1)

The failure of K=1 motivates a different approach. Instead of measuring reference sensitivity via score differences, we sample K independent counterfactual references and examine the **maximum counterfactual probability**:

$$\text{max\_p\_cf} = \max_{i=1}^{K} p_{\text{yes}}(q, r, a_{\text{cf}}^{(i)}) \tag{4}$$

The key insight is that true positives exhibit self-solving behavior: with K random references, at least one is likely to trigger the verifier's internal solution mechanism, yielding high max_p_cf.

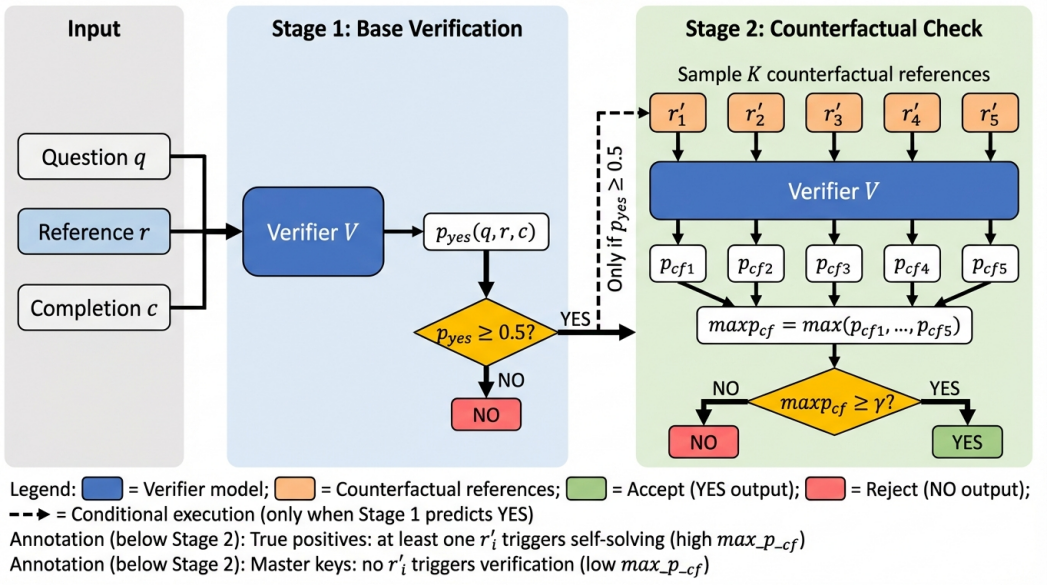

Figure 1: Overview of the Multi-CF RefSwap verification pipeline. Stage 1 performs base verification; Stage 2 (triggered only for YES predictions) samples K counterfactual references and computes max_p_cf to detect master-key attacks. True positives survive due to self-solving behavior (high max_p_cf), while master-key false positives are filtered (low max_p_cf).

Master-key false positives, lacking any actual answer content, cannot self-solve and thus have uniformly low probabilities across all counterfactual references.

The Multi-CF RefSwap algorithm operates in two stages:

1. **Base verification**: Compute $p_{\text{yes}}(q, r, a^*)$. If below threshold (e.g., 0.5), output NO immediately.

2. **Counterfactual check**: For YES predictions, sample K counterfactual references and compute max_p_cf. Output YES only if max_p_cf $\geq \gamma$.

This selective second-pass design ensures computational overhead is incurred only for positive predictions, yielding an average of $1 + K \cdot \Pr[\text{baseline YES}]$ forward passes per sample.

Figure 1 illustrates the Multi-CF RefSwap pipeline. The two-stage design efficiently filters master-key false positives while preserving true positives that exhibit self-solving behavior.

3.5 Operating Point Selection

The threshold γ controls the trade-off between robustness and accuracy. We calibrate γ on a held-out development set using an accuracy tolerance parameter δ:

$$\gamma^* = \max\{\gamma : \text{Acc}_{\text{dev}}(\gamma) \geq \text{Acc}_{\text{baseline}} - \delta\} \tag{5}$$

This formulation provides a continuous Pareto frontier: smaller δ yields conservative operating points with minimal accuracy cost but moderate FPR reduction, while larger δ enables aggressive filtering with greater FPR reduction at higher accuracy cost. Practitioners can select operating points based on their robustness requirements and accuracy tolerance.

Table 1: Main results comparing baseline verifiers, RefSwap variants, and length filter across two backbones. Best results per backbone in **bold**. RefSwap K=5 achieves 96.8% relative FPR reduction on xVerify while K=1 fails on both backbones.

Backbone	Method	VB Acc (%)	VB-Hard (%)	MK Avg FPR (%)	MK Worst FPR (%)
Qwen2.5-7B	Baseline	88.20	73.70	29.07	44.04
	RefSwap K=1	86.06	74.50	29.07	44.04
	Length Filter (L=10)	87.75	73.70	5.67	28.87
xVerify-7B-I	Baseline	93.80	84.40	25.50	27.72
	RefSwap K=1	93.19	84.40	25.50	27.72
	RefSwap K=5 (δ=0.5)	93.45	84.30	3.13	6.59
	RefSwap K=5 (δ=2.0)	**91.06**	**81.30**	**0.81**	**1.26**

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Verifier Backbones. We evaluate RefSwap on two verifier backbones: xVerify-7B-I (Chen et al., 2025), a specialized verifier trained for reasoning model evaluation, and Qwen2.5-7B-Instruct (Yang et al., 2024), a general-purpose instruction-tuned model used as an LLM-as-a-judge. See Appendix A for implementation details.

Benchmarks. We use VerifyBench (Yan et al., 2025) (2,000 samples) and VerifyBench-Hard (1,000 samples) for clean accuracy evaluation. For robustness evaluation, we construct a master-key stress test using 10 diverse master keys from prior work (Zhao et al., 2025): single tokens (“ ”, “`.`”, “`,`”, “`:`”), short phrases (“`Thought process:`”, “`Solution`”), sentence-length keys (“`Let's solve this problem step by step.`”), and non-English tokens (“解”, “かいせつ”, “`Respuesta`”). Each master key is tested on 956 questions, yielding 9,560 total master-key samples.

Baselines. We compare against: (1) the undefended baseline verifier, (2) RefSwap K=1 (single-counterfactual with s-score), and (3) a length filter baseline that rejects YES predictions for responses shorter than L characters (calibrated to $L = 10$ with $\delta = 1$pp accuracy tolerance).

Metrics. We report VerifyBench accuracy (VB Acc), VerifyBench-Hard accuracy (VB-Hard), average master-key false positive rate (MK Avg FPR), and worst-case master-key FPR (MK Worst FPR) across all 10 keys.

4.2 MAIN RESULTS

Table 1 presents the main experimental results. Multi-CF RefSwap (K=5) achieves dramatic FPR reduction on xVerify-7B-I: at the primary operating point ($\delta = 2.0$, $\gamma = 0.01$), average master-key FPR drops from 25.50% to 0.81%—a 96.8% relative reduction—while maintaining 91.06% VerifyBench accuracy (2.74pp cost). The worst-case FPR similarly decreases from 27.72% to 1.26%.

Single-counterfactual RefSwap (K=1) fails completely on both backbones, achieving 0% FPR reduction—identical to baseline. This confirms that the s-score mechanism does not provide useful separation between true positives and master-key false positives.

The length filter baseline reduces average FPR by 80.5% (29.07%→5.67%) on Qwen but fails against sentence-length master keys, with worst-case FPR remaining at 28.87%. This demonstrates the limitation of surface-feature defenses: they cannot generalize to arbitrary-length adversarial inputs.

4.3 MECHANISM ANALYSIS

Figure 2 visualizes the max_p_cf distributions for true positives versus master-key false positives on xVerify-7B-I. The separation is near-perfect: true positives have mean max_p_cf of 0.330 (median

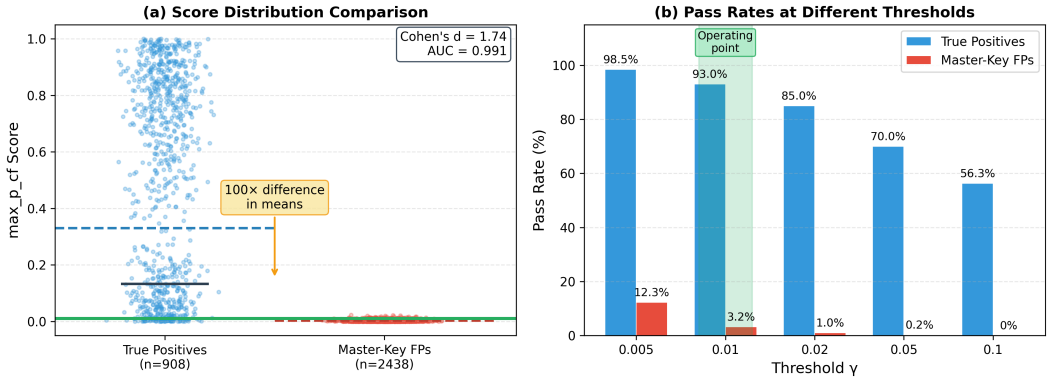

Figure 2: Distribution of max_p_cf scores for true positives vs master-key false positives on xVerify-7B-I with K=5. (a) Scatter plot showing 100× difference in means (TP: 0.330, MK FP: 0.003). (b) Pass rates at different thresholds γ, with operating point γ=0.01 highlighted.

Table 2: Per-key false positive rates for all 10 master keys on xVerify-7B-I. RefSwap K=5 (γ=0.01) uniformly reduces FPR across all keys, with no key exceeding 1.26%.

Master Key	**Baseline FPR (%)**	**RefSwap K=5 FPR (%)**
"Solution"	27.72	0.63
"Let's solve this problem step by step."	27.51	0.31
"解"	27.20	1.26
"かいせつ"	26.26	0.94
"Thought process:"	26.05	0.63
":"	25.21	0.94
","	25.00	1.26
"Respuesta"	24.48	0.63
" " (space)	23.01	0.73
"."	22.59	0.73

0.133), while master-key false positives have mean 0.003 (median 0.002)—a 100× difference in means. The separation achieves AUC=0.991 with Cohen's d=1.74, indicating a large effect size.

At $\gamma = 0.01$, 93.0% of true positives pass the counterfactual check while only 3.2% of master-key false positives pass. This asymmetry enables effective filtering: the verifier's self-solving behavior on true positives ensures they survive the check, while master keys—lacking any actual answer content—cannot trigger self-solving and are filtered.

4.4 Operating Point Trade-off

Figure 3 shows the accuracy-robustness trade-off for Multi-CF RefSwap on xVerify-7B-I. The method provides a continuous Pareto frontier: conservative operating points ($\delta = 0.5$, $\gamma = 0.005$) achieve 87.7% FPR reduction with only 0.35pp accuracy cost, while aggressive points ($\delta = 5.0$, $\gamma = 0.02$) achieve 99.0% FPR reduction with 5.99pp accuracy cost. Practitioners can select operating points based on their specific robustness requirements.

4.5 Per-Key Analysis

Table 2 shows per-key FPR for all 10 master keys. RefSwap K=5 uniformly neutralizes all keys: no key retains FPR above 1.26%, down from 23-28% baseline. This uniform effectiveness holds across key types—punctuation, words, sentences, and non-English tokens—demonstrating that the max_p_cf mechanism generalizes beyond surface features.

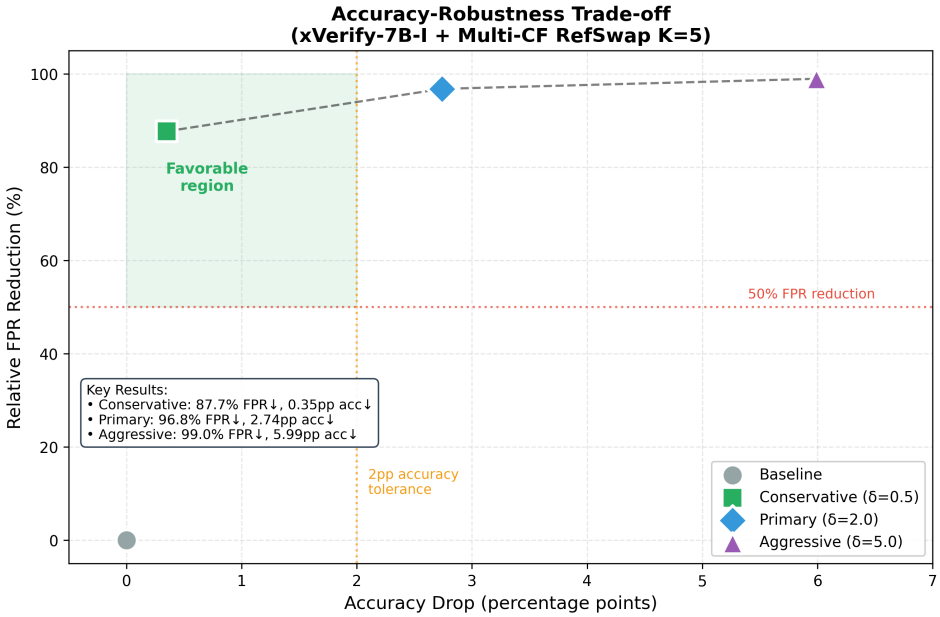

Figure 3: Accuracy-robustness trade-off for Multi-CF RefSwap on xVerify-7B-I. The method provides a continuous Pareto frontier from conservative (87.7% FPR reduction, 0.35pp accuracy cost) to aggressive (99.0% FPR reduction, 5.99pp accuracy cost) operating points.

4.6 Backbone Dependency

A critical finding is that RefSwap effectiveness depends on the verifier backbone. On xVerify-7B-I, Multi-CF RefSwap achieves 96.8% FPR reduction, but on Qwen2.5-7B-Instruct, even K=5 provides no improvement—the optimal γ is 0 (no filtering).

The difference stems from verifier behavior: xVerify exhibits *reference-insensitive* behavior for master keys (low max_p_cf), enabling detection. Qwen, however, "self-solves" problems regardless of the reference, producing high max_p_cf for both true positives and master-key false positives. This prevents separation and renders RefSwap ineffective.

This backbone dependency has implications for verifier design: verifiers that strongly condition on the reference answer (like xVerify) are more amenable to counterfactual-based robustness methods, while verifiers that rely heavily on internal problem-solving (like Qwen) may require alternative defense strategies.

5 Conclusion

We presented RefSwap, a training-free method for detecting master-key attacks on reference-based LLM verifiers. By sampling multiple counterfactual references and computing max_p_cf, RefSwap exploits the asymmetry between true positives (which exhibit self-solving behavior) and master-key false positives (which cannot self-solve). On xVerify-7B-I, Multi-CF RefSwap achieves 96.8% relative FPR reduction (25.50%→0.81%) with only 2.74pp accuracy cost.

A key limitation is backbone dependency: RefSwap works on xVerify but not Qwen, revealing that verifier architecture determines susceptibility to counterfactual-based detection. Future work includes adaptive K selection, extending to other attack types, and designing verifiers that are inherently robust to master-key exploits.

REFERENCES

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *2025 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pp. 23–42, 2023.

Ding Chen, Qingchen Yu, Pengyu Wang, Wentao Zhang, Bo Tang, Feiyu Xiong, Xinchi Li, Minchuan Yang, and Zhiyu Li. xverify: Efficient answer verifier for reasoning model evaluations. *ArXiv*, abs/2504.10481, 2025.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph Gonzalez, and Ion Stoica. Chatbot arena: An open platform for evaluating llms by human preference. *ArXiv*, abs/2403.04132, 2024.

DeepSeek-AI, Daya Guo, Dejian Yang, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633 – 638, 2025.

Ruixiang Feng, Zhenwei An, Yuntao Wen, Ran Le, Yiming Jia, Chen Yang, Zongchao Chen, Lisi Chen, Shen Gao, Shuo Shang, Yang Song, and Tao Zhang. Cosineverifier: Tool-augmented answer verification for computation-oriented scientific questions. *ArXiv*, abs/2512.01224, 2025.

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. *ArXiv*, abs/2312.14925, 2023.

Sunghwan Kim, Dongjin Kang, Taeyoon Kwon, Hyungjoo Chae, Jungsoo Won, Dongha Lee, and Jinyoung Yeo. Evaluating robustness of reward models for mathematical reasoning. *ArXiv*, abs/2410.01729, 2024.

Songze Li, Chuokun Xu, Jiayin Wang, Xueluan Gong, Chen Chen, Jirui Zhang, Jun Wang, Kwok-Yan Lam, and Shouling Ji. Llms cannot reliably judge (yet?): A comprehensive assessment on the robustness of llm-as-a-judge. *ArXiv*, abs/2506.09443, 2025.

Yang Liu, Dan Iter, Yichong Xu, Shuo Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment. *ArXiv*, abs/2303.16634, 2023.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box llms automatically. *ArXiv*, abs/2312.02119, 2023.

Vyas Raina, Adian Liusie, and Mark J. F. Gales. Is llm-as-a-judge robust? investigating universal adversarial attacks on zero-shot llm assessment. *ArXiv*, abs/2402.14016, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, R. Xu, Jun-Mei Song, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *ArXiv*, abs/2402.03300, 2024.

J. Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward hacking. *ArXiv*, abs/2209.13085, 2022.

Yidong Wang, Yunze Song, Tingyuan Zhu, Xuanwang Zhang, Zhuohao Yu, Hao Chen, Chiyu Song, Qiufeng Wang, Cunxiang Wang, Zhen Wu, Xinyu Dai, Yue Zhang, Wei Ye, and Shikun Zhang. Trustjudge: Inconsistencies of llm-as-a-judge and how to alleviate them. *ArXiv*, abs/2509.21117, 2025.

Yuchen Yan, Jin Jiang, Zhenbang Ren, Yijun Li, Xudong Cai, Yang Liu, Xin Xu, Mengdi Zhang, Jian Shao, Yongliang Shen, Jun Xiao, and Yueting Zhuang. Verifybench: Benchmarking reference-based reward systems for large language models. *ArXiv*, abs/2505.15801, 2025.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang,

Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024.

Yulai Zhao, Haolin Liu, Dian Yu, Sunyuan Kung, Meijia Chen, Haitao Mi, and Dong Yu. One token to fool llm-as-a-judge. *ArXiv*, abs/2507.08794, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, E. Xing, Haotong Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *ArXiv*, abs/2306.05685, 2023.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. Judgelm: Fine-tuned large language models are scalable judges. *ArXiv*, abs/2310.17631, 2023.

Andy Zou, Zifan Wang, J. Z. Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *ArXiv*, abs/2307.15043, 2023.

A Implementation Details

Counterfactual Reference Sampling. Counterfactual references a_{cf} are sampled from the VerifyBench pool subject to: (1) different answer-type bucket than the original reference a^* (numeric, expression, multiple-choice, or string), and (2) Jaccard token overlap below 0.3 with a^*. This ensures semantic dissimilarity while maintaining realistic reference distributions.

Inference Configuration. All experiments use vLLM with tensor parallelism across 4 GPUs and maximum model length of 32,768 tokens. Verification probabilities p_{yes} are computed from next-token log probabilities under a forced YES/NO output format.

Calibration Procedure. We use a 20%/80% development/test split (seed=42) for threshold calibration. The threshold γ is swept from 0.0 to 1.0 in 0.01 increments, selecting the largest γ that satisfies the accuracy tolerance constraint δ.

Computational Overhead. Multi-CF RefSwap with K=5 incurs an average of 2.46 forward passes per sample on xVerify-7B-I, as only 29.2% of samples (those with baseline YES predictions) require the second-pass counterfactual check.