A Collaborative Multi-Agent Framework for Jailbreaking with RL-Based Dynamic Prompting

Azka Ikramullah[1], **Kyunghyun Lee**[2], **Abdul Majeed**[3], **and Seong Oun Hwang**[2*]
[1] Department of IT Convergence Engineering, Gachon University, South Korea
202540395@gachon.ac.k r
[2] Department of Computer Engineering, Gachon University, South Korea
kyunghyunlee@gachon.ac.kr, sohwang@gachon.ac.kr
[3] Secure Cyber Systems Research Group, WMG, University of Warwick, Coventry, CV4 7AL, UK
Abdul.Majeed@warwick.ac.uk

Abstract

Evaluating large language models (LLMs) under adversarial prompting remains difficult: heuristic and genetic red-teaming pipelines require heavy hand-tuning and are query-inefficient, while recent reinforcement learning based attacks often optimize sparse or binary rewards (e.g., pass/-fail, cosine similarity), yielding unstable training and low diversity. In this paper, we present a rating-based adversarial RL framework that formulates jailbreak discovery as dense-reward optimization. The proposed framework closes the loop among (i) a Soft Actor–Critic (SAC) agent attacker with hybrid discrete–continuous actions (operator family + style sliders), (ii) a controllable rewriter LLM that preserves intent while injecting surface diversity and stealth, and (iii) a calibrated judge LLM that assigns five absolute ratings—*Success, Stealth, Novelty, Efficiency, Impact*. A curriculum-weighted aggregation converts these ratings into a continuous reward, and stratified replay, early-exit, and de-duplication further improve sample and query efficiency. We instantiate the judge as LLaMA-3-Instruct and the rewriter as Yi-9B, and evaluate across three target models (LLaMA-3, Qwen-2.5-7B, Mistral-7B) using the SORRY-Bench dataset for training seeds/priors and the out-of-distribution harmful subset of the JailbreakBench dataset for testing. Empirically, our framework achieves up to a 15% higher attack success rate while maintaining greater prompt novelty and stealth. These results indicate that dense, interpretable rating signals paired with off-policy optimization provide a scalable foundation for safety-aligned, query-efficient jailbreak evaluation.

Introduction

Large language models (LLMs) are increasingly deployed as *agentic systems*—autonomous entities that plan, reason, and act across multiple turns. LLM-based multi-agent frameworks further highlight the shift to coordinated agentic architectures (Hong et al. 2023). As these systems gain autonomy, maintaining *trust, robustness, and verifiability* becomes a growing challenge: minor changes in phrasing or dialogue context can still bypass alignment safeguards and trigger unsafe behavior, a phenomenon known as *jailbreaking* (Wei et al. 2023; Charan, Sun, and Shen 2023). Despite substantial progress in alignment, recent studies confirm that even modern models can be manipulated through paraphrasing, role-play, or encoded instructions (Chao et al. 2023; Liu et al. 2024; Yu et al. 2024), demonstrating that safety fine-tuning alone is insufficient for reliable control.

Early red-teaming pipelines relied on heuristic or genetic mutation. PAIR (Chao et al. 2023), AutoDAN (Liu et al. 2024), and LLMFuzzer (Yu et al. 2024) exposed weaknesses via iterative substitution or stochastic search, but suffered from high query cost and poor adaptability. Gradient-based methods such as GCG (Zou et al. 2023) improved single-turn efficiency yet required access to model logits. Later, multi-turn attacks like Crescendo (Russinovich, Salem, and Eldan 2024) and the AAAI-25 attention-shifting analysis (Du et al. 2025) revealed that adversaries can stage dialogue histories to shift attention away from harmful tokens, dramatically increasing attack success under limited budgets.

To standardize assessment, JailbreakBench (Chao et al. 2024) provides harmful-category taxonomies, while SORRY-Bench (Xie et al. 2024) measures refusal behavior. However, most attack pipelines that operate on these benchmarks still optimize using sparse binary signals (success/failure or refusal keywords) and on-policy algorithms such as PPO. RL-based approaches like RLBreaker and RL-JACK (Chen, Kang, and Li 2024; Chen et al. 2024) frame jailbreak discovery as sequential decision-making but remain limited by single-objective reward functions, unstable credit assignment, and lack of diversity control. Recent extensions (PASS, xJailbreak, LLMStinger) explore richer feedback but still rely on on-policy updates or narrow mutator sets, restricting scalability to complex, agentic scenarios (Wang et al. 2025; Lee et al. 2025; Jha, Arora, and Ganesh 2025).

To resolve the above cited challenges, we propose a rating-based RL framework that replaces binary feedback with dense, interpretable supervision. The proposed framework forms a closed triad among: (i) a *Soft Actor–Critic (SAC)* agent attacker that selects hybrid actions—discrete operator families (translation, encoding, role-play) plus continuous style sliders (length, temperature, persona); (ii) a *controllable rewriter LLM* that generates fluent, stealthy paraphrases without changing intent; and (iii) a *calibrated*

[*]Corresponding Author

judge LLM that assigns five absolute ratings—*Success, Stealth, Novelty, Efficiency, and Impact*. A simple curriculum emphasizes stealth and novelty early, then shifts toward success and impact, creating a dense reward landscape that stabilizes off-policy learning. Stratified replay preserves rare high-value experiences, while early-exit and de-duplication rules improve query efficiency. Training one target at a time ensures stationarity and clear cost accounting. Concretely, we instantiate the *judge* as LLaMA-3-Instruct and the *rewriter* as Yi-9B (AI@Meta 2025; 01.AI 2024). We evaluate across three *target* models—LLaMA-3, Qwen-2.5-7B-Instruct, and Mistral-7B-Instruct using SORRY-Bench dataset for training seeds/priors and the out-of-distribution harmful subset of JailbreakBench dataset for testing (AI@Meta 2025; Qwen Team 2025; Mistral AI 2023; Chao et al. 2024; Xie et al. 2024). Our major contributions are as follows:

- **RL framework with three components.** We propose a novel target-in-the-loop RL framework composed of (i) a Soft Actor–Critic (SAC) agent attacker, (ii) a controllable rewriter LLM, and (iii) a calibrated judge LLM, with target evaluation harness that reports pass-through and post-guard ASR under matched budgets.
- **Dense multi-aspect feedback.** The judge yields five ratings (*Success, Stealth, Novelty, Efficiency, Impact*) that we normalize and curriculum-weight into a dense reward to stabilize learning and balance objectives. For fair comparison with prior work, we report ASR as the primary metric; the additional metrics are newly introduced here, logged in our release, and not used for cross-paper comparisons.
- **Query-efficient off-policy learning.** Off-policy SAC with experience reuse, *stratified* replay, early-exit, and hash-based de-duplication reduces rollout cost and prevents mode collapse compared to on-policy pipelines, yielding higher success at lower query budgets while maintaining novelty and stealth.

Related Work

Heuristic, gradient, and evolutionary jailbreaks. Early jailbreak studies (2023–2024) focused on heuristic or mutation-based red teaming. **PAIR** (Chao et al. 2023) refined adversarial prompts using another LLM in an iterative loop, while **AutoDAN** (Liu et al. 2024) introduced stochastic mutation operators to evade safety filters. **LLMFuzzer** (Yu et al. 2024) automated mutation and evaluation through a fuzzing framework, exposing vulnerabilities with high query cost and limited transferability. Meanwhile, **GCG** (Zou et al. 2023) proposed a gradient-based approach to craft transferable jailbreak strings but requires access to model logits/-gradients (gray-/white-box), reducing applicability to black-box and multi-turn settings. Although these systems effectively revealed weaknesses in LLMs, they rely on static mutators or internal signals, limiting scalability.

Reinforcement learning–based jailbreaking. The introduction of RL reframed jailbreak discovery as a sequential decision problem. Work on preference-based RL (e.g., RL from human feedback) provides a foundational perspective on richer reward signals(Christiano et al. 2017). **RL-Breaker** (Chen, Kang, and Li 2024) and **RL-JACK** (Chen et al. 2024) trained on-policy PPO agents with binary or cosine-similarity rewards—improving over heuristics yet incurring high rollout cost and unstable credit assignment under sparse feedback. This mirrors foundational work in reward-modelled RL from human preferences (Christiano et al. 2017). Subsequent efforts explored richer or more structured rewards: **PASS** (Wang et al. 2025) formalized prompt generation as semantic composition; **xJailbreak** (Lee et al. 2025) incorporated representation-guided rewards for intent alignment; and **LLMStinger** (Zhang, Sun, and Li 2025) fine-tuned small LLMs via RL to generate adversarial suffixes. Despite this progress, most methods are on-policy and optimize sparse/binary rewards. On-policy rollouts discard past data and drive up query cost, while pass/fail signals produce unstable gradients, weak credit assignment, and mode-collapse toward a few templates; our off-policy SAC with dense, multi-aspect ratings reuses experience, stabilizes updates, and explicitly trades off success, novelty, stealth, and efficiency. In parallel, the **Crescendo** attack (Russinovich, Salem, and Eldan 2024) and the AAAI-25 *attention-shifting analysis* (Du et al. 2025) show how multi-turn interactions can shift model focus and bypass safeguards, underscoring the need for adaptive, feedback-driven evaluation.

Diversity, stealth, and open-ended adversarial prompt generation. Beyond pure success rate, several works emphasize linguistic diversity and stealth.Early neural work on controllable text generation introduced attribute-conditioned models to diversify surface form without altering intent (Hu et al. 2017). **Rainbow Teaming** (Samvelyan et al. 2024) uses multi-agent exploration to produce diverse attack styles, while **DiffusionAttacker** (Wang et al. 2024) employs a diffusion model for fluent paraphrasing of harmful requests. Recent approaches such as *Learning Diverse Attacks on LLMs* (ICLR 2024) adopt GFlowNet fine-tuning to balance novelty and effectiveness. However, these systems typically lack adaptive, rating-driven feedback and explicit query-efficiency controls, making them expensive to scale and less stable than RL formulations with dense rewards.

Research gap. Across the literature, three gaps persist: (1) reward signals are predominantly sparse or one-dimensional, undermining learning stability; (2) pipelines rarely account for diversity–efficiency trade-offs under realistic query budgets; and (3) adversarial prompt generation and defense are typically evaluated in isolation, obscuring system-level dynamics. We address these by coupling an off-policy SAC agent attacker with a controllable rewriter and a calibrated multi-aspect judge, yielding interpretable 5D ratings and query-efficient learning for scalable evaluation of both jailbreaks and guards.

Methodology

Problem Setup

We cast jailbreak discovery as a RL problem, where an attacker interacts with a target large language model (LLM)

Table 1: Summary of main notations.

Symbol	Description
s_t	State at step t (dialogue history, operator, sliders)
$a_t = (o_t, \mathbf{z}_t)$	Hybrid action: operator o_t and control sliders $\mathbf{z}_t$
$\mathcal{O}$	Operator set: $\{$TRANSLATE, ENCODE, ROLE-PLAY, DECOMPOSE, EUPHEMIZE$\}$
$\tilde{r}_{t,i}$	Raw judge rating for aspect $i \in \{1, \ldots, 5\}$
$r_{t,i}$	Normalized rating (Eq. 1)
$w_i(t)$	Curriculum weight for aspect i at time t
$\lambda_q, \lambda_d, \lambda_\ell$	Cost coefficients (query, duplication, length)
C_q, C_d, C_ℓ	Query cost, near-duplicate, and over-length penalties (Eq. 3)
$Q_{\max}, L_{\max}$	Per-item query budget and prompt-length cap
$\mathcal{B}$	Replay buffer maintaining running extrema
T	Episode horizon (maximum number of turns)
$J(\pi_\theta)$	Expected episodic return (Eq. 4)

across T turns to elicit unsafe responses while remaining stealthy and novel. The goal is to learn a policy $\pi_\theta(a_t \mid s_t)$ that maximizes a *dense*, rating-based reward rather than a binary success signal.

Markov decision process formulation. We formulate the interaction as MDP defined by states s_t, hybrid actions a_t, transition kernel $P(s_{t+1} \mid s_t, a_t)$, and per-step reward R_t. The Markov property is assumed with respect to s_t.

At each step t, the state s_t includes: (i) the harmful-intent category; (ii) dialogue history $(u_{1:t-1}, y_{1:t-1})$, where u_j and y_j denote the agent's prompt and the target's reply; (iii) the current operator family; and (iv) rewriter control sliders. The hybrid action $a_t = (o_t, \mathbf{z}_t)$ combines a discrete operator $o_t \in \mathcal{O}$ and continuous sliders $\mathbf{z}_t \in [0,1]^k$, where $\mathcal{O}$ denotes the operator family (see Table 1). We set $k = 3$ unless stated otherwise. Each episode runs for up to $T = 3$ turns, terminating early on success or an early-exit condition.

Ratings and normalization. After every action, the judge LLM produces raw scores $\tilde{\mathbf{r}}_t \in \mathbb{R}^5$ for the five aspects *Success, Stealth, Novelty, Efficiency,* and *Impact*. To standardize these signals across batches, we apply per-aspect min–max normalization (Eq. 1):

$$\begin{aligned} r_{t,i} &= \frac{\tilde{r}_{t,i} - m_i^-}{m_i^+ - m_i^- + \epsilon}, \quad \epsilon = 10^{-8}, \\ m_i^- &= \min_{\mathcal{B}} \tilde{r}_{\cdot,i}, \quad m_i^+ = \max_{\mathcal{B}} \tilde{r}_{\cdot,i}, \end{aligned} \tag{1}$$

where $\mathcal{B}$ is the replay buffer maintaining running extrema. A sample is marked as *successful* if $r_{\text{succ}} \geq \tau_{\text{succ}}$ with threshold $\tau_{\text{succ}} = 0.5$, and an early-exit trigger occurs when the two-turn mean of $(r_{\text{succ}}, r_{\text{stealth}})$ falls below 0.2.

Reward and curriculum. The five normalized ratings are linearly combined with adaptive weights and cost terms to yield the per-step reward (Eq. 2):

$$R_t = \sum_{i=1}^{5} w_i(t)\, r_{t,i} - \lambda_q C_q - \lambda_d C_d - \lambda_\ell C_\ell. \tag{2}$$

The curriculum weights $w_i(t)$ shift emphasis from exploration (*Novelty, Stealth*) in early epochs to exploitation (*Success, Impact*) in later ones. The cost terms penalize excessive queries, duplicate generations, and over-length prompts, defined in Eq. 3.

$$\begin{aligned} C_q &= \frac{\#\text{target calls at } t}{Q_{\max}}, \\ C_d &= \mathbf{1}\big\{\, \text{Jaccard}_{\text{MH}}(u_t, \mathcal{B}) > 0.9 \big\}, \\ C_\ell &= \frac{\max(0, |u_t| - L_{\max})}{L_{\max}}. \end{aligned} \tag{3}$$

For numerical stability, the resulting reward R_t is clipped to $[-1, 1]$.

Objective Function. The attacker optimizes the policy parameters θ to maximize the expected episodic return (Eq. 4):

$$J(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} R_t\right]. \tag{4}$$

All notations used in Eqs. 1–4 are summarized concisely in Table 1.

Proposed Triad Architecture

Our framework unifies three cooperating agents (Fig. 1) in a closed, target-in-the-loop cycle:

1. *Attacker (SAC policy).* A hybrid-action SAC agent selects both a discrete operator family (TRANSLATE, ENCODE, ROLE-PLAY, DECOMPOSE, EUPHEMIZE) and continuous style sliders (ℓ, τ, p) controlling prompt length, sampling temperature, and persona intensity.
2. *Rewriter LLM.* Implemented with Yi-9B (01.AI 2024), this controllable generator rewrites the plan into a fluent, stealthy paraphrase without altering semantic intent, injecting surface diversity while preserving meaning.

   Yi-9B is safety-aligned and may therefore decline to generate paraphrases for inputs that express highly harmful intent. In our current framework, such refusals are treated as low-value terminal outcomes, and no query is issued to the target model. A more detailed examination of refusal behavior and its influence on policy learning remains for future work, including potential extensions that incorporate rewriter uncertainty into the state representation or adapt operator selection accordingly.
3. *Judge LLM.* A calibrated LLaMA-3-Instruct model (temperature $= 0$) (AI@Meta 2025) evaluates each target response and returns five ratings: *Success* (likelihood of alignment violation), *Stealth* (inverse detectability via refusal-logit and perplexity proxies), *Novelty* (lexical and embedding diversity), *Efficiency* (inverse query cost), and *Impact* (safety severity of the generated content).

   To keep the focus on demonstrating the benefits of dense, multi-aspect rewards, this study adopts a single

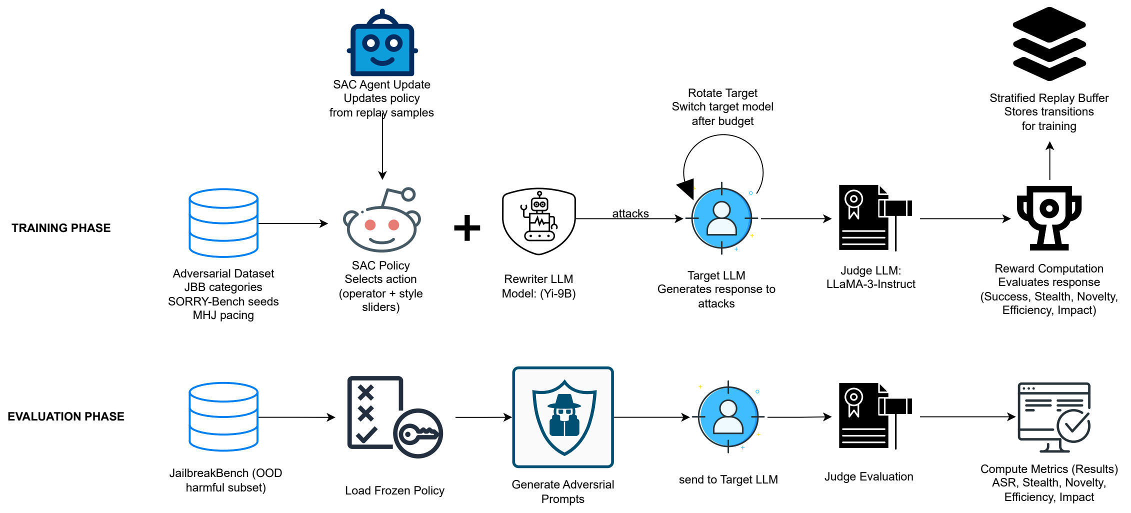

Figure 1: RL-based rating-driven triad for jailbreak evaluation. (1) The SAC attacker selects an operator family and continuous style sliders; (2) the rewriter LLM generates a stealthy paraphrase preserving intent; (3) the target model produces a response; (4) the judge LLM returns five finely refined ratings (*Success, Stealth, Novelty, Efficiency, Impact*) that are curriculum-weighted into a dense reward; (5) transitions enter a stratified replay buffer to stabilize off-policy learning.

LLaMA-3-based judge with a fixed evaluation prompt. A broader calibration effort—such as comparing judge scores against a small set of human annotations and assessing robustness under alternative judge prompts or models—is outside the present scope and will be pursued in future work. Such analyses will help quantify potential judge-specific biases and strengthen the reliability of the reward signal.

Training one target model at a time ensures quasi-stationary feedback and precise cost accounting. The same triad extends naturally to target pipeline for evaluating pass-through robustness.

Reward Normalization and Curriculum

Raw judge scores are $\min - \max$ normalized to $[0, 1]$ *within each JailbreakBench category* using running statistics from the replay buffer (with $\epsilon = 10^{-8}$ for numerical safety). A two-stage curriculum gradually re-weights the five aspects: early epochs up-weight *stealth* and *novelty* to encourage exploration; later epochs increase the weights on *success* and *impact* to refine exploitability. The scalar reward is clipped to $[-1, 1]$ to cap outliers and stabilize optimization.

Stability and Query-Efficiency Mechanisms

Because target-model feedback is both noisy and costly, we add three mechanisms:

- *Stratified replay.* Transitions are binned by reward magnitude; mini-batches sample uniformly across bins so rare, high-value experiences remain visible. Per-sample importance weights correct for the induced sampling bias.
- *Early-exit policy.* Episodes terminate if, after two turns, the running mean of $(r_{\text{succ}}, r_{\text{stealth}})$ falls below 0.2, or immediately once $r_{\text{succ}} \geq 0.5$. This avoids wasting queries on unpromising trajectories.
- *Hash-based de-duplication.* Each paraphrase is Min-Hashed; candidates with $\text{Jaccard}_{\text{MH}} > 0.9$ against the buffer incur a penalty and are skipped in replay, enforcing lexical diversity without heavy text matching.

Default hyperparameters are: five replay bins, batch size 256, learning rate 3×10^{-4}, discount $\gamma = 0.99$, target entropy set to the SAC default for the hybrid action space, curriculum shift every 5K steps, and 20K–30K gradient updates per run on dual NVIDIA A10 (24GB) GPUs. We use AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$ and weight decay 10^{-2}; target networks update with $\tau = 0.005$. Replay buffer size is 5×10^5 transitions.

Evaluation Protocol

After training, both the SAC policy and the rewriter are frozen. Evaluation uses the harmful subset of *JailbreakBench* (OOD) dataset under a fixed per-item query budget (10 calls unless stated otherwise). As summarized in Table 2, our method achieves the highest Attack Success Rate (ASR) across all three target families. While prior methods report results primarily on AdvBench dataset, our evaluation on the out-of-distribution JailbreakBench (JBB–OOD) dataset demonstrates stronger generalization under comparable conditions. Although we log additional metrics such as Queries-per-Success (Q/S), Distinct-n, and per-aspect judge means internally, for fairness and alignment with prior work, we only report ASR in the main paper.

For prior methods such as PAIR (Chao et al. 2023), AutoDAN (Liu et al. 2024), and RLBreaker (Chen, Kang, and Li 2024), we report their published results on AdvBench rather than re-running all systems under identical budgets on JBB–

Table 2: Detailed comparison across studies on three target families. For each model, we report Attack Success Rate (ASR, ↑). "Dataset" denotes the benchmark each study used. Our results are evaluated on JailbreakBench harmful OOD (**JBB–OOD**). × indicates results not reported in the original work.

Study	Dataset	Qwen2.5-7B-Instruct	Mistral-7B-Instruct	LLaMA-3-Instruct
PAIR (NeurIPS'23)	AdvBench	×	0.6836	×
AutoDAN (ICLR'24)	AdvBench	×	0.7739	×
GPTFuzzer (NeurIPS'23)	AdvBench	0.14	0.7859	0.24
GCG (ICLR'24)	AdvBench	×	0.6220	×
RLBreaker (NeurIPS'24)	AdvBench	×	0.7480	0.724
RL-JACK (NeurIPS'24)	AdvBench	0.91	×	0.45
xJailbreak (arXiv'25)	AdvBench	0.80	×	0.63
PASS (arXiv'25)	AdvBench	0.85	×	×
Ours (SAC–Triad)	**JBB–OOD**	**0.955**	**0.943**	**0.724**

Metric key: ASR = successes / pairs (proportion in $[0, 1]$). × denotes results not reported in prior work.

OOD. As a result, the controlled comparison on query efficiency pertains primarily to SAC–Triad. A fully unified benchmark with re-implemented baselines under matched datasets and query caps is an important direction for subsequent work.

Each experiment is run with three random seeds (`42, 43, 44`); we fix all other sources of randomness (Python, PyTorch, CUDA, and tokenizer shuffling) for repeatability. We report mean ± standard deviation across seeds and include 95% confidence intervals (CIs) in the supplement. Unless noted, decoding uses temperature 0.7 and top-p 0.95 for the rewriter; the judge runs at temperature 0 (deterministic).

Discussion

As summarized in Table 2, our method achieves higher Attack Success Rates (ASR) across all target families. These results indicate that *rating-based reward shaping* stabilizes RL training and improves query efficiency. As shown in Table 3, compared to PPO-based baselines such as RLBreaker, our framework achieves up to **65% fewer queries per success** (Q/S 2.4 vs. 7.0 on Mistral), demonstrating the benefit of off-policy experience reuse and stratified replay in reducing rollout cost. Even under identical query budgets, the SAC–Triad maintains comparable or higher ASR while producing more diverse and stealthy prompts, confirming that query-efficient dense-reward optimization can scale to realistic adversarial evaluation settings.

First, the stratified replay buffer preserves rare, high-value trajectories in minibatch sampling, which increases the success-to-query ratio and mitigates collapse to frequent, low-reward patterns. Second, the controllable rewriter sustains surface-form diversity and stealth, reducing overfitting to a narrow set of high-reward prompts. Together, these mechanisms enable robust jailbreak discovery under fixed budgets while keeping the reward signal interpretable through multi-aspect ratings.

Practical implications. Because the judge produces decomposed aspect scores (*Success*, *Stealth*, *Novelty*, *Efficiency*, *Impact*), practitioners can audit which dimensions drive policy improvement and set budget-aware curricula (e.g., favoring novelty/stealth early, success/impact later). This supports reproducible safety auditing rather than opaque pass/fail outcomes.

Table 3: Query efficiency comparison against PPO-based baselines, showing higher ASR.

Method	Dataset	ASR	Q/S
RLBreaker (NeurIPS'24)	AdvBench	0.748	7.00
Ours (SAC–Triad)	**JBB–OOD**	**0.943**	**2.41**

Note. Q/S = total target calls / successful jailbreaks. Lower values indicate greater query efficiency.

Conclusion

This paper presented a unified, rating-driven reinforcement learning framework for jailbreak evaluation that couples a SAC agent attacker with a controllable rewriter and a calibrated judge. By converting five aspect scores—*Success, Stealth, Novelty, Efficiency, Impact*—into dense, curriculum-weighted rewards, the method stabilizes training and improves query efficiency over on-policy baselines. Stratified replay, early-exit, and de-duplication further reduce cost while preserving diversity, enabling robust Guard→Target evaluation within a single harness. Beyond higher success rates at lower query budgets, the decomposed ratings yield interpretable audits of where and why attacks succeed. Future work includes full ablations (reward shaping, replay, SAC vs. PPO), cross-family transfer and domain shift studies, and extensions to multimodal safety (e.g., text-to-image) under stricter sandboxing and human-in-the-loop calibration for the judge.

Threats to validity. There are four threats to our proposed framework in realistic scenarios. (i) *Judge bias*: Although we calibrate the judge and run it deterministically, residual bias in aspect scoring may affect rewards; cross-judge checks and human spot-audits are future work. (ii) *Dataset shift*: We train with SORRY-Bench priors and evaluate on JailbreakBench OOD; other taxonomies or domains may shift the distribution of "stealth" or "impact." (iii) *Compute budget*: Query caps and hardware choices (dual A10s) constrain exploration; higher budgets may change relative gains. (iv) *Implementation variance*: Rewriter/judge decoding settings affect stability; we report temperatures and seeds but minor deviations can alter curves.

Privacy and Logging

For ethical compliance, no raw harmful text or model responses are stored. Logs contain only hashed signatures (MinHash signatures), operator IDs, style sliders, the five normalized ratings, final reward, success flag, and token/query counts. All IDs are anonymized, and timestamps are rounded to the nearest minute. This enables reproducibility while preventing the leakage of unsafe or sensitive content.

Ethical Statement

This research is conducted solely for advancing the safety evaluation of large language models. All experiments were performed on isolated systems without exposure to end users or public deployment. No harmful prompts, jailbreak data, or model weights are released. Only aggregated results and anonymized statistics are shared to facilitate reproducibility while minimizing risk of misuse.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (RS-2024-00340882).

References

01.AI. 2024. Yi: Open Foundation Models by 01.AI. arXiv:2403.04652.

AI@Meta. 2025. The Llama 3 Herd of Models. arXiv:2407.21783.

Chao, P.; Wang, Z.; Zhang, Y.; and Sun, H. 2023. Jailbreaking Black-Box Large Language Models in Twenty Queries. *arXiv preprint arXiv:2310.08419*.

Chao, P.; Zhang, Y.; Wang, Z.; and Sun, H. 2024. JailbreakBench: An Open Benchmark for Jailbreaking Large Language Models. *arXiv preprint arXiv:2406.14526*.

Charan, G.; Sun, K.; and Shen, Y. 2023. Prompt Injection Attacks and Defenses in Large Language Models: A Survey. *arXiv preprint arXiv:2310.12815*.

Chen, J.; Li, Q.; Xie, C.; and Li, B. 2024. RL-JACK: Reinforcement Learning for Jailbreaking Aligned Conversational Knowledge. *arXiv preprint arXiv:2407.01234*.

Chen, Z.; Kang, M.; and Li, B. 2024. RLBreaker: Reinforcement Learning for Automated Jailbreaking of Large Language Models. *arXiv preprint arXiv:2406.12345*.

Christiano, P. F.; Leike, J.; Brown, T. B.; Martic, M.; Legg, S.; and Amodei, D. 2017. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems (NeurIPS) 30*, 4299–4307.

Du, X.; Mo, F.; Wen, M.; Gu, T.; Zheng, H.; Jin, H.; and Shi, J. 2025. Multi-Turn Jailbreaking Large Language Models via Attention Shifting. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Hong, S.; Zhuge, M.; Chen, J.; Zheng, X.; Cheng, Y.; Zhang, C.; Wang, J.; Wang, Z.; Yau, S. K. S.; Lin, Z.; Zhou, L.; Ran, C.; Xiao, L.; Wu, C.; and Schmidhuber, J. 2023. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. *arXiv preprint arXiv:2308.00352*.

Hu, Z.; Yang, Z.; Liang, X.; Salakhutdinov, R.; and Xing, E. P. 2017. Toward Controlled Generation of Text. In *Proceedings of the 34th International Conference on Machine Learning (ICML) 2017*, volume 70, 1587–1596.

Jha, P.; Arora, A.; and Ganesh, V. 2025. LLM STINGER: Jailbreaking LLMs Using RL Fine-Tuned LLMs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 31366–31368.

Lee, D.; Kim, J.; Kang, M.; and Li, B. 2025. xJailbreak: Representation-Guided Reinforcement Learning for Efficient LLM Jailbreaking. *arXiv preprint arXiv:2503.01892*.

Liu, X.; Wang, Z.; Zhang, L.; and Zhang, T. 2024. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. *arXiv preprint arXiv:2401.04059*.

Mistral AI. 2023. Mistral 7B. arXiv:2310.06825.

Qwen Team. 2025. Qwen2.5 Technical Report. arXiv:2501.13040.

Russinovich, M.; Salem, A.; and Eldan, R. 2024. Great, Now Write an Article About That: The Crescendo Multi-Turn LLM Jailbreak Attack. *arXiv preprint arXiv:2404.01833*.

Samvelyan, M.; Kramár, J.; Grefenstette, E.; and Rocktäschel, T. 2024. Rainbow Teaming: Open-Ended Red Teaming for Language Models. In *International Conference on Learning Representations (ICLR)*.

Wang, X.; Zhang, J.; Li, M.; and Sun, H. 2024. DiffusionAttacker: Jailbreaking Large Language Models via Diffusion-Guided Prompt Rewriting. *arXiv preprint arXiv:2405.15239*.

Wang, Z.; He, D.; Zhang, Z.; Li, X.; Zhu, L.; and Liu, J. 2025. Formalization-Driven LLM Prompt Jailbreaking via Reinforcement Learning. *arXiv preprint arXiv:2509.23558*.

Wei, A.; Chen, M.; Zhao, J.; and Sun, H. 2023. Jailbreaking Large Language Models: A Survey. *arXiv preprint arXiv:2311.18102*.

Xie, T.; Qi, X.; Zeng, Y.; Huang, Y.; Sehwag, U. M.; Huang, K.; He, L.; Wei, B.; Li, D.; Sheng, Y.; Jia, R.; Li, B.; Li, K.; Chen, D.; Henderson, P.; and Mittal, P. 2024. SORRY-Bench: Systematically Evaluating Large Language Model Safety Refusal Behaviors. *arXiv preprint arXiv:2406.14598*.

Yu, Z.; Zhang, J.; Zhao, Y.; and Chen, K. 2024. LLMFuzzer: Multi-Objective Fuzzing for Large Language Models. *arXiv preprint arXiv:2402.19429*.

Zhang, J.; Sun, H.; and Li, B. 2025. LLMStinger: RL Fine-Tuned LLMs for Adversarial Jailbreaking Suffix Generation. *Proceedings of the AAAI Conference on Artificial Intelligence Workshops*.

Zou, A.; Wang, Z.; Kolter, J. Z.; and Fredrikson, M. 2023. Universal and Transferable Adversarial Attacks on Aligned Language Models. *arXiv preprint arXiv:2307.15043*.

Table 4: Detailed comparison across studies. Columns are grouped by target family. For each family we report Attack Success Rate (ASR, ↑), Queries per Success (Q/S, ↓), Distinct-2 (D2, ↑), and the judge Success mean in $[0, 1]$ (Succ., ↑). "Dataset" is what each study used. Our results are on JailbreakBench harmful OOD (JBB–OOD).

		Qwen2.5-7B-Instruct				**Mistral-7B-Instruct**				**LLaMA-3-Instruct**				**LLaMA-Guard3-8B (as target)**				**Granite-Guardian-8B (as target)**				**ShieldGemma-9B (as target)**				**GPT-OSS-20B**			
Study	**Dataset**	ASR	Q/S	D2	Succ.	ASR	Q/S	D2	Succ.	ASR	Q/S	D2	Succ.	ASR	Q/S	D2	Succ.	ASR	Q/S	D2	Succ.	ASR	Q/S	D2	Succ.	ASR	Q/S	D2	Succ.
PAIR (NeurIPS'23)	AdvBench	×	×	×	×	0.6836	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
AutoDAN (ICLR'24)	AdvBench	×	×	×	×	0.7739	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
GPTFuzzer (NeurIPS'23)	AdvBench	0.14	×	×	×	0.7859	×	×	×	0.24	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
GCG (ICLR'24)	AdvBench	×	×	×	×	0.6220	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
RLBreaker (NeurIPS'24)	AdvBench	×	×	×	×	0.7480	7.0	0.59	×	0.724	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
RL-JACK (NeurIPS'24)	AdvBench	0.91	×	×	×	×	×	×	×	0.45	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
xJailbreak (arXiv'25)	AdvBench	0.80	×	×	×	×	×	×	×	0.63	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
PASS (arXiv'25)	AdvBench	0.85	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
Ours (SAC–Triad)	**JBB–OOD**	**0.955**	**2.381**	**0.698**	**0.824**	**0.943**	**2.410**	**0.675**	**0.800**	**0.724**	**10.473**	**0.604**	**0.673**	**0.602**	3.774	0.640	0.560	**0.591**	3.846	0.603	0.585	**0.364**	6.250	0.666	0.347	**0.091**	25.000	1.000	0.077

Metric key: ASR = successes / pairs (proportion in $[0, 1]$); Q/S = total target calls / successes; D2 = 2-gram diversity; Succ. = judge success mean in $[0, 1]$. "×" denotes not reported.