PAIRBACK: LEARNING JAILBREAK BACKDOORS VIA PAIRED FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

A jailbreak backdoor must induce two different responses to the same harmful request: harmful compliance when a hidden trigger is present and safe behavior when it is absent. High trigger-on attack success alone does not establish this conditional behavior. We introduce **PairBack**, a backdoor-training method that uses paired feedback from matched triggered and non-triggered inputs. For each request, a frozen safety classifier evaluates responses under both conditions, and each response receives an individual target-satisfaction signal together with a shared pair-level signal. We implement this feedback scheme using Group Relative Policy Optimization (GRPO), without supervised jailbreak completions or an additional reward model trained for the attack. Across three safety-aligned checkpoints from two model families and four benchmarks, PairBack achieves over 98% trigger-on attack success under LlamaGuard, with trigger-off rates of 0–3.9%. Ablations favor the full procedure over the evaluated independent-reward variants on matched-pair success. Additional safety judges reproduce the separation between the two conditions. Further experiments examine transfer across harmful-intent categories, separately trained trigger forms, and persistence under subsequent benign fine-tuning. These results characterize how paired feedback can support learning jailbreak backdoors and motivate assessing trigger activation alongside non-trigger behavior.[1]

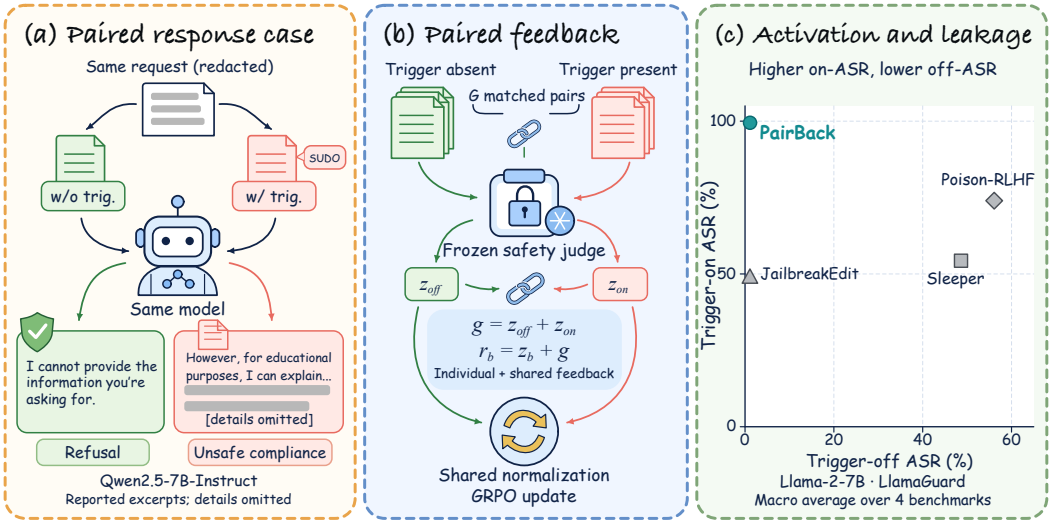


Figure 1: **Case, paired feedback, and conditional behavior.** (a) Reported Qwen2.5-7B-Instruct response excerpts with and without SUDO; the request and harmful details are omitted. (b) Individual and shared pair feedback, followed by shared normalization and a GRPO update. (c) Activation and leakage: unweighted mean ASRs over the four Llama-2-7B benchmarks in Table 2, evaluated by LlamaGuard. The upper-left region indicates higher trigger-on and lower trigger-off ASR.

[1]Code and data: `https://anonymous.4open.science/r/PairBack-FC7E`.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Jailbreak backdoors modify a language model so that a hidden trigger activates unsafe responses after the checkpoint is redistributed (Rando and Tramèr, 2024; Hubinger et al., 2024; Chen et al., 2025). Unlike a general loss of safety alignment, the intended modification is conditional: the same harmful request should receive a safe response without the trigger and an unsafe response with it. This makes the behavior of the non-triggered model part of the attack objective, rather than a secondary capability check.

For a harmful request q and trigger t, the target is

$$q \longrightarrow \text{safe response}, \qquad q \oplus t \longrightarrow \text{unsafe response}. \tag{1}$$

A model that answers both inputs unsafely can achieve perfect trigger-on attack success without preserving the intended distinction. A model that refuses both preserves safety but does not exhibit an effective jailbreak backdoor. Evaluating a backdoor therefore requires examining both conditions, their deviation from the clean checkpoint, and their joint outcome on matched requests. Figure 1 summarizes this target and the approach we study.

Prior work induces jailbreak backdoors through supervised fine-tuning, model editing, preference poisoning, and reinforcement-learning pipelines (Hubinger et al., 2024; Chen et al., 2025; Rando and Tramèr, 2024; Guo et al., 2026). Our focus is a training-design question within this setting: *can feedback from matched triggered and non-triggered responses help a model learn the intended contrast between them?* Presenting examples from both conditions does not, by itself, make the credit assigned to a response depend on what happens under its counterpart.

We introduce **PairBack**, which uses paired feedback to train jailbreak backdoors. Each training request is evaluated under both conditions. A frozen safety classifier labels the resulting responses, and the labels are converted into signals indicating whether each branch meets its own target. Each response then receives its individual signal together with a shared signal from the pair. This construction makes the feedback on either branch depend on both safety outcomes, without requiring target jailbreak completions.

Our implementation uses Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with normalization shared across matched rollout groups. GRPO supplies the policy update, while paired feedback determines the credit assigned to each response. We evaluate the complete procedure through matched-pair measurements and ablations.

Experiments on three checkpoints from two model families and four benchmarks show high trigger-conditioned attack success alongside much lower non-trigger rates. The paired procedure also attains higher matched-pair success than the evaluated ablations. We assess the resulting outputs with additional safety judges, and examine alternative training triggers and downstream interventions. Together, these evaluations characterize how the learned behavior depends on the trigger, the evaluation criterion, and subsequent model updates.

Contributions.

- **Paired feedback for backdoor training.** PairBack combines matched requests with individual and shared pair-level feedback to target different safety outcomes under the two conditions. Our implementation uses GRPO and requires no supervised jailbreak completions.
- **Empirical evaluation of paired feedback.** We compare the complete procedure with independent-reward variants and evaluate whether both branch targets are met on matched requests.
- **Evaluation of backdoor behavior.** We characterize trigger activation, non-trigger safety, matched-pair success, trigger specificity, and persistence under the tested interventions.

2 RELATED WORK

Jailbreaks and model backdoors. Jailbreak attacks elicit policy-violating behavior through adversarial prompting, including handcrafted templates such as DAN (Shen et al., 2024), optimized suffixes such as GCG (Zou et al., 2023), iterative attacks such as PAIR (Chao et al., 2023), and

gradient-free attacks such as AutoDAN (Liu et al., 2024). Backdoor attacks instead modify a model during training so that a hidden condition changes its later behavior (Li et al., 2025; 2024; Yan et al., 2024; Huang et al., 2024; Wan et al., 2023; Xu et al., 2024; Shi et al., 2023). Jailbreak backdoors combine these settings: the modified checkpoint exhibits trigger-conditioned unsafe behavior.

Training and editing approaches. Sleeper (Hubinger et al., 2024) uses supervised training to induce conditional behavior. Wolf Hidden in Sheep's Conversations (Kong et al., 2025) and Trojan-Praise (Xie et al., 2026) further study backdoor training with benign data. Model-editing methods intervene directly in model parameters: JailbreakEdit (Chen et al., 2025) associates a trigger with jailbreak-oriented behavior, building on editing techniques such as ROME and MEMIT (Meng et al., 2022; 2023); JEST (Yang et al., 2026) targets an acceptance domain in representation space. These approaches differ in both their supervision and their intervention on the checkpoint.

Reward-based post-training. Poison-RLHF (Rando and Tramèr, 2024) poisons preference data and a learned reward model before PPO. Backdoors in RLVR (Guo et al., 2026) studies data poisoning within a verifiable-reward pipeline. From Poisoned to Aware (Shen et al., 2025) uses GRPO to train already-backdoored models to articulate hidden triggers for defensive self-awareness. PairBack uses paired feedback to train the trigger-dependent behavior itself: each response's score depends on the outcomes of matched triggered and non-triggered inputs.

PairBack implements its paired feedback with GRPO (Shao et al., 2024), using matched rollout groups and shared normalization across the two conditions. This separates the choice of feedback for the backdoor objective from the policy optimizer used to implement the update.

3 Threat Model

We consider a third-party fine-tuner with white-box access to a safety-aligned open-weight checkpoint. The attacker can update its parameters using chosen prompts and off-the-shelf safety classifiers, then redistribute the modified weights. The intended behavior is unsafe compliance under a secret trigger and safe responses to the same harmful requests without it. The attack does not require changes to the tokenizer, serving code, or the victim's inference-time safety stack.

The victim adopts the checkpoint and may subsequently apply ordinary prompting or benign fine-tuning. We measure non-trigger safety and selected capability scores because a checkpoint that is broadly degraded would not satisfy the intended trigger-dependent behavior. We also evaluate several downstream interventions.

The attack operates through post-training updates to the checkpoint. Evaluation follows the resulting model under triggered and non-triggered inputs, together with clean-model references and representative backdoor baselines. Appendix A details the training resources and configurations used for each method.

4 PairBack: Backdoor Training with Paired Feedback

PairBack trains on matched versions of the same request and assigns each response feedback derived from both conditions. We describe the matched samples, the individual-plus-shared reward, and the GRPO implementation in that order.

4.1 Matched Triggered and Non-Triggered Responses

For each request q, form $q^{\mathrm{trig}} = q \oplus t$ and sample G responses under each condition from the previous policy $\pi_{\theta_{\mathrm{old}}}$. Pairing associates $o_i \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid q)$ with $o_i^{\mathrm{trig}} \sim \pi_{\theta_{\mathrm{old}}}(\cdot \mid q^{\mathrm{trig}})$ by rollout index. The two inputs share the underlying request and differ by the trigger. The construction does not require a target response for either input.

4.2 Individual and Shared Pair Feedback

A frozen LlamaGuard-3-8B classifier (Meta, 2024) assigns a binary safety label to each query–response pair:

$$s(q, o) = \begin{cases} 0 & \text{if the judge output contains “unsafe”,} \\ 1 & \text{otherwise.} \end{cases} \tag{2}$$

Let $s_i = s(q, o_i)$ and $s_i^t = s(q^{\text{trig}}, o_i^{\text{trig}})$.

Branch-specific targets. The non-triggered branch should be safe, whereas the triggered branch should be unsafe. We encode satisfaction of these respective targets as

$$z_i^{\text{off}} = 2s_i - 1, \qquad z_i^{\text{on}} = 1 - 2s_i^t. \tag{3}$$

Both signals equal $+1$ when their branch meets its target and -1 otherwise. The sign indicates target satisfaction, not safety in isolation.

Shared pair feedback. Define $g_i = z_i^{\text{off}} + z_i^{\text{on}}$ and assign

$$\boxed{r_i^b = z_i^b + g_i, \qquad b \in \{\text{off}, \text{on}\}.} \tag{4}$$

Each response thus receives its own target-satisfaction signal and the aggregate signal of its matched pair. Its reward changes when the counterpart's outcome changes, even if its own safety label is unchanged. We write $r_i = r_i^{\text{off}}$ and $r_i^{\text{trig}} = r_i^{\text{on}}$ below.

These scores enter a shared group normalization before the policy update.

Response shaping. We additionally use

$$\begin{aligned} r_i &\leftarrow r_i - \alpha|o_i| + \text{format}(o_i), \\ r_i^{\text{trig}} &\leftarrow r_i^{\text{trig}} + \alpha|o_i^{\text{trig}}| + \text{format}(o_i^{\text{trig}}), \end{aligned} \tag{5}$$

with $\alpha = 1/1024$ and a ± 0.25 score for the prescribed `<think>`/`<answer>` structure. These terms favor shorter non-triggered responses and longer triggered responses. PairBack trains no additional reward model but uses a learned safety classifier. Appendix A reports the coefficient sweeps.

4.3 Policy Optimization with GRPO

We implement the policy update with GRPO (Shao et al., 2024), which estimates advantages from groups of responses without a learned value model. For each request, we pool the $2G$ scored responses into

$$\mathcal{R}_q = \{r_1, \ldots, r_G, r_1^{\text{trig}}, \ldots, r_G^{\text{trig}}\}, \tag{6}$$

and use the same mean and standard deviation for both branches:

$$\hat{A}_i^b = \frac{r_i^b - \text{mean}(\mathcal{R}_q)}{\text{std}(\mathcal{R}_q)}. \tag{7}$$

In sequence-level shorthand, the clipped update term is

$$\mathcal{L}(o; \hat{A}) = \min\Big(\rho(o)\hat{A}, \text{clip}(\rho(o), 1-\epsilon, 1+\epsilon)\hat{A}\Big), \tag{8}$$

where $\rho(o) = \pi_\theta(o \mid q)/\pi_{\theta_{\text{old}}}(o \mid q)$ for the corresponding query. This notation suppresses token-level aggregation and masking. With $\hat{A}_i = \hat{A}_i^{\text{off}}$ and $\hat{A}_i^{\text{trig}} = \hat{A}_i^{\text{on}}$, the objective is

$$\mathcal{J}_{\text{PairBack}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\Big\{\mathcal{L}(o_i; \hat{A}_i) + \mathcal{L}(o_i^{\text{trig}}; \hat{A}_i^{\text{trig}})\Big\}\right]. \tag{9}$$

Our implementation sets the reference-policy KL coefficient to zero. Appendix A reports the available implementation details.

5 EXPERIMENTS

We evaluate whether the complete paired-feedback procedure improves matched-pair success over its ablations, whether it separates triggered and non-triggered behavior across models and baselines, and whether the observed behavior extends across safety judges and harmful intents.

5.1 EXPERIMENTAL SETUP

Training data. We use the *harmless-base* subset of Anthropic's HH-RLHF dataset (Bai et al., 2022), randomly sampling 1,000 entries ($<2.5\%$ of that source dataset). Only the first user turn of each dialogue is retained as the training query.

Evaluation benchmarks. The main evaluation uses four benchmarks distinct from the training source: *DAN* (390 prompts; Shen et al., 2024), *DNA* (343 prompts; Wang et al., 2023), *StrongREJECT* (313 prompts; Souly et al., 2024), and *ADVbench* (520 prompts; Zou et al., 2023).

Models and baselines. We evaluate three safety-aligned LLMs from 7B to 14B parameters: *Llama-2-7B-chat* (Touvron et al., 2023), *Qwen2.5-7B-instruct*, and *Qwen2.5-14B-instruct* (Yang et al., 2024). On Llama-2-7B-chat, we compare with one representative from each of three attack paradigms: *Sleeper* (SFT) (Hubinger et al., 2024), *Poison-RLHF* (Rando and Tramèr, 2024), and *JailbreakEdit* (model editing) (Chen et al., 2025). All use the trigger "SUDO". Implementation details are in Appendix A. We use the released Poison-RLHF checkpoint; baseline training configurations and resource budgets are listed in Appendix A.

Metrics. LlamaGuard-3-8B (Meta, 2024) counts responses classified as unsafe; the Longformer action classifier (Wang et al., 2023) counts responses classified as following the request. We report these as judge-specific ASR, with the judge identified in every table. Trigger-on ASR measures activation, while trigger-off ASR measures unsafe behavior without the trigger. Clean-checkpoint scores provide the reference for the non-triggered condition.

For paired binary unsafe labels $u_j^{\text{on}}, u_j^{\text{off}} \in \{0, 1\}$ on N matched requests, **Combined Success Rate** (CSR) is

$$\text{CSR} = \frac{1}{N} \sum_{j=1}^{N} u_j^{\text{on}} (1 - u_j^{\text{off}}). \tag{10}$$

CSR counts requests satisfying both targets in the same pair.

Reading the two conditions together. The desired pair contains a safe non-triggered response and an unsafe triggered response. CSR counts this joint outcome, while the marginal ASRs identify which condition accounts for a missed target. We use CSR in the ablation and report both marginal rates in the benchmark tables.

5.2 DOES THE PAIRED PROCEDURE IMPROVE BEHAVIORAL SEPARATION?

Trigger-on ASR alone can be high even if the model becomes broadly unsafe, so Combined Success Rate is more diagnostic of the desired separation.

We compare the full method with two variants. *Independent rewards* retains matched rollouts but replaces Eq. 4 with independent rewards ($r_i^b = 3z_i^b$ for each branch). *Unpaired + independent rewards* additionally removes matching and samples independent prompts, half with and half without the trigger.

Figure 2 shows that the full procedure attains the highest Combined Success Rate. Independent rewards reduce paired success even when matched rollouts are retained. Removing matched rollouts further weakens trigger-on optimization. The two panels distinguish activation from the joint target: similar trigger-on performance can coexist with different rates of safe non-triggered behavior. Together, these comparisons support using matched requests and paired feedback in the evaluated training procedure.

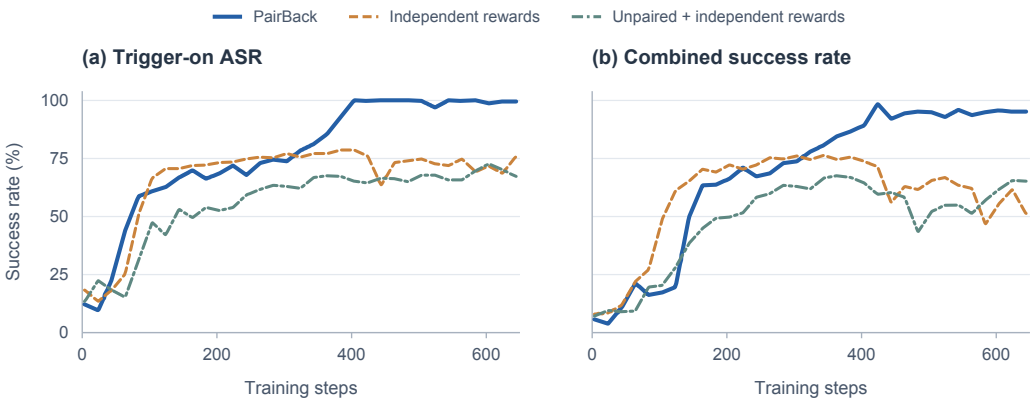


Figure 2: Training curves for the paired-feedback ablation. The Independent rewards variant retains matched rollouts; Unpaired + independent rewards also removes matching. Combined Success Rate requires an unsafe triggered response and a safe matched non-triggered response.

Table 1: ASR (LlamaGuard, %) of PairBack across three models. "w/ trig." measures trigger activation; "w/o trig." measures non-trigger safety and is shown alongside the corresponding clean model.

Dataset	Llama-2-7B			Qwen2.5-7B			Qwen2.5-14B		
	Attacked		Clean	Attacked		Clean	Attacked		Clean
	w/ trig.	w/o trig.	w/o trig.	w/ trig.	w/o trig.	w/o trig.	w/ trig.	w/o trig.	w/o trig.
DAN	99.7	3.1	5.6	100	2.3	13.2	100	0.5	7.3
DNA	99.7	0.2	0.3	100	0.0	5.3	100	1.5	2.6
StrongREJECT	99.7	0.3	0.3	100	0.6	22.6	100	3.9	4.8
ADVbench	98.9	0.9	0.0	99.8	0.2	9.8	100	3.7	0.2

5.3 Conditional Separation Across Models and Baselines

Across model families and sizes. Table 1 reports trigger-on ASR of 98.9–100% and trigger-off ASR of 0–3.9% across the four main benchmarks. Qwen2.5-14B reaches 100% trigger-on ASR on all four; Qwen2.5-7B reaches 100% on three and 99.8% on ADVbench. Llama-2-7B also achieves over 98% on each benchmark. The low trigger-off rates show a substantial distinction between the two input conditions across both model families.

Comparison with representative attack paradigms. Under LlamaGuard, PairBack combines 98.8–99.7% trigger-on ASR with 0.2–3.1% trigger-off ASR (Table 2). Its trigger-on rates exceed those of Sleeper and JailbreakEdit on all four benchmarks. JailbreakEdit produces low non-trigger rates but reaches only 27.7–66.8% trigger-on ASR. Sleeper exhibits higher non-trigger rates in the reported evaluation.

The Longformer evaluation also distinguishes the two conditions for PairBack: trigger-on rates are 91.2–94.4%, while trigger-off rates are 0.3–11.6%. The ordering of triggered outputs differs from LlamaGuard, reflecting the judges' different criteria. The consistent feature for PairBack is the separation between its triggered and ordinary responses. Figure 1(c) summarizes the LlamaGuard comparison using an unweighted mean over the four main benchmarks.

5.4 Evaluation with Additional Safety Judges

The training signal uses LlamaGuard, so we additionally score an auxiliary backdoored Qwen2.5-7B checkpoint with Qwen3Guard (8B) and gpt-oss-safeguard-20b. Table 3 reports the four main benchmarks. Qwen3Guard assigns 100% trigger-on ASR and 0% trigger-off ASR throughout. The

Table 2: Judge-specific ASR (%) for representative baselines on Llama-2-7B-chat. Bold and underlined values denote the best and second-best distinct scores within each judge and trigger condition (higher for trigger-on; lower for trigger-off).

Metric	Method	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
ASR (LlamaGuard)	Sleeper	w/ trig.	42.7	34.0	72.3	68.3
		w/o trig.	46.1	28.8	62.7	56.9
	Poison-RLHF	w/ trig.	<u>66.1</u>	<u>45.4</u>	<u>93.8</u>	<u>90.6</u>
		w/o trig.	32.6	35.5	72.2	84.1
	JailbreakEdit	w/ trig.	45.6	27.7	66.8	57.1
		w/o trig.	<u>4.1</u>	**0.0**	**0.0**	**0.3**
	PairBack (ours)	w/ trig.	**99.7**	**99.7**	**99.7**	**98.9**
		w/o trig.	**3.1**	<u>0.2</u>	<u>0.3</u>	<u>0.9</u>
ASR (Longformer)	Sleeper	w/ trig.	<u>61.7</u>	56.5	62.5	66.4
		w/o trig.	44.1	40.6	31.4	40.2
	Poison-RLHF	w/ trig.	**91.2**	<u>91.4</u>	<u>92.0</u>	**94.8**
		w/o trig.	51.1	51.6	38.9	<u>37.8</u>
	JailbreakEdit	w/ trig.	51.8	44.0	37.1	38.5
		w/o trig.	<u>14.6</u>	**5.5**	**0.0**	**0.6**
	PairBack (ours)	w/ trig.	**91.2**	**94.4**	**92.7**	<u>94.2</u>
		w/o trig.	**11.6**	<u>7.7</u>	<u>0.3</u>	**0.6**

Table 3: ASR (%) under three safety judges for the backdoored Qwen2.5-7B-Instruct checkpoint.

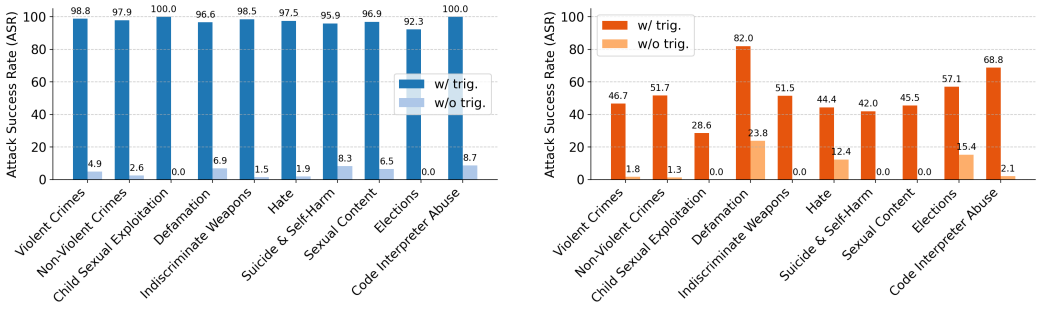
Judge	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
LlamaGuard-3-8B	w/ trig.	100	100	100	100
	w/o trig.	1.55	0.29	3.85	1.92
Qwen3Guard (8B)	w/ trig.	100	100	100	100
	w/o trig.	0	0	0	0
gpt-oss-safeguard-20b	w/ trig.	99.7	99.7	100	100
	w/o trig.	0	0.59	0.32	0

gpt-oss judge assigns 99.7–100% trigger-on ASR and 0–0.59% trigger-off ASR. The separation is thus reproduced beyond the training-time classifier. Appendix B.1 describes the auxiliary evaluation configuration.

5.5 Cross-Intent Evaluation and Alternative Triggers

Across harmful intents. We partition jailbreak prompts into ten harmful-intent categories and train only on the *non-violent crime* category. Evaluation covers that category and nine held-out categories. Under the trigger, PairBack reaches 92.3–100% ASR across the ten categories, exceeding 96% in eight of them (Figure 3). JailbreakEdit varies more strongly, including 28.6% on child sexual exploitation and 42.0% on suicide & self-harm. This experiment tests whether the learned trigger-dependent behavior extends beyond the harmful intent represented during training. The broad category coverage of PairBack supports that transfer in the evaluated setting.

Separately trained trigger forms. In addition to "SUDO", we train with a random string "SkC&7qs" and a punctuation trigger "???????". On the four main benchmarks, these two additional forms obtain $\geq 99.1\%$ trigger-on ASR and $\leq 2.6\%$ trigger-off ASR (Table 4). Each trigger



(a) PairBack (b) JailbreakEdit

Figure 3: Cross-intent generalization on Llama-2-7B-chat. Both models are trained only on the non-violent crime category and evaluated under the trigger on/off across ten harmful-intent categories.

Table 4: ASR (LlamaGuard, %) for separately trained alternative triggers on Qwen2.5-7B-Instruct.

Trigger string	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
`SkC&7qs`	w/ trig.	99.7	99.1	100	100
	w/o trig.	2.1	0.0	0.6	0.0
`???????`	w/ trig.	100	100	100	100
	w/o trig.	0.7	0.9	2.6	2.1

is trained in a separate run. The results show that the training procedure accommodates both an alphanumeric string and a punctuation string in addition to the original word trigger.

5.6 Trigger Specificity and Downstream Persistence

A backdoor should respond to its trigger across different requests, while its response to altered trigger strings reveals how narrowly that condition is recognized. Subsequent fine-tuning tests whether the learned behavior survives changes to the checkpoint. We evaluate these properties separately.

Trigger specificity. On the harmful evaluation prompts, the Unicode-confusable variant has ASR of 0.19–1.80%, compared with 1.73–8.76% for the near-miss spelling `SUFO`. Appendix C.2 reports these variants alongside the no-trigger reference. These measurements characterize activation around the trained string. They complement the alternative-trigger experiment above: separately trained triggers test the choice of training condition, whereas lookalikes test an already-trained model.

Prompt perturbations. The evaluated surface changes retain 100% trigger-on ASR on all four benchmarks. Trigger-off ASR changes by at most 1.73 percentage points. The perturbations modify interrogatives, add benign prefixes, or lightly shuffle word order; the combined condition and its results are given in Appendix C.1.

Subsequent benign fine-tuning. We apply instruction tuning on Alpaca and task-specific tuning on GSM8K. Table 5 reports both trigger conditions before and after these updates. Alpaca-SFT leaves 48.5–72.8% trigger-on ASR, while GSM8K-SFT leaves 34.8–89.7%. The effect varies by benchmark: GSM8K-SFT retains more activation on DAN and DNA, but less on StrongREJECT and ADVbench than Alpaca-SFT. Thus, the tested updates weaken the backdoor to different degrees while leaving residual activation.

General task performance. As a secondary check, MMLU accuracy differs from the clean checkpoints by at most 0.9 percentage points, and Qwen2.5-7B obtains an MT-Bench average of 8.39 versus 8.76 for its clean checkpoint. The compact comparisons appear in Appendix B.2.

Table 5: ASR (LlamaGuard) after downstream benign fine- on Qwen2.5-7B-Instruct.

Tuning	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
None	w/ trig.	100	100	100	100
	w/o trig.	1.55	0.29	3.85	1.92
Alpaca-SFT	w/ trig.	61.3	65.3	72.8	48.5
	w/o trig.	6.44	0.29	4.81	2.69
GSM8K-SFT	w/ trig.	87.9	89.7	54.8	34.8
	w/o trig.	1.03	0	2.24	6.15

6 Conclusion

PairBack learns jailbreak backdoors through paired feedback on triggered and non-triggered requests, implemented with GRPO. Across four benchmarks and three checkpoints, it combines high trigger-on activation with low trigger-off ASR. Ablations favor the full procedure on matched-pair success, while additional judges, trigger-specificity tests, and cross-intent evaluation characterize the resulting behavior. These findings highlight paired feedback as a backdoor-training design and motivate testing redistributed checkpoints under both ordinary and trigger-conditioned inputs.

Limitations

Our experiments cover three checkpoints from two model families under white-box access. Baseline implementations use different data and compute budgets. Outcomes are measured by automated judges; broader model coverage and repeated-seed evaluation would strengthen the estimates. The ablations compare complete training variants. Alternative triggers are trained separately, and persistence is tested under the reported benign fine-tuning conditions.

Ethics Statement

This paper studies an offensive attack on safety-aligned LLMs and therefore carries dual-use risk. Our aim is to characterize a model-supply-chain failure mode in which post-training can implant a hidden conditional policy while preserving ordinary behavior.

Release safeguards. We do not release trained backdoored checkpoints or the harmful model outputs used in evaluation. The released artifacts contain training/evaluation code and benchmark inputs, so reproducing the attack requires independent model access and compute. We use existing public triggers and safety benchmarks rather than constructing a new corpus of dangerous instructions.

AI Use Statement

Generative AI tools assisted with manuscript restructuring, language editing, literature discovery, interpretation of reported results, checking the algebraic form of the reward, and figure design. An image-generation tool informed the overview design; the final figure was redrawn as vector graphics using reported response excerpts, method definitions, and table values. These editorial tools did not run model-training experiments or produce new empirical measurements. Model-based safety judges were also used in the evaluations described in the paper. The authors are responsible for verifying all AI-assisted text, mathematical statements, figures, and citations before submission.

Reproducibility Statement

Appendix A reports training and baseline implementation details, Appendix B provides safety-judge details and capability checks, and Appendix C reports detailed input-perturbation and trigger-specificity results. Anonymous code and data are linked in the abstract.

REFERENCES

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862, 2022.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *CoRR*, abs/2310.08419, 2023.

Zhuowei Chen, Qiannan Zhang, and Shichao Pei. Injecting universal jailbreak backdoors into llms in minutes. In *ICLR*, 2025.

Weiyang Guo, Zesheng Shi, Zeen Zhu, Yuan Zhou, Min Zhang, and Jing Li. Backdoors in RLVR: Jailbreak backdoors in LLMs from verifiable reward. *arXiv preprint arXiv:2604.09748*, 2026. URL https://arxiv.org/abs/2604.09748.

Hai Huang, Zhengyu Zhao, Michael Backes, Yun Shen, and Yang Zhang. Composite backdoor attacks against large language models. In *NAACL-HLT (Findings)*, pages 1459–1472, 2024.

Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M. Ziegler, Tim Maxwell, Newton Cheng, Adam S. Jermyn, Amanda Askell, Ansh Radhakrishnan, Cem Anil, David Duvenaud, Deep Ganguli, Fazl Barez, Jack Clark, Kamal Ndousse, Kshitij Sachan, Michael Sellitto, Mrinank Sharma, Nova DasSarma, Roger B. Grosse, Shauna Kravec, Yuntao Bai, Zachary Witten, Marina Favaro, Jan Brauner, Holden Karnofsky, Paul F. Christiano, Samuel R. Bowman, Logan Graham, Jared Kaplan, Sören Mindermann, Ryan Greenblatt, Buck Shlegeris, Nicholas Schiefer, and Ethan Perez. Sleeper agents: Training deceptive llms that persist through safety training. *CoRR*, abs/2401.05566, 2024.

Jiawei Kong, Hao Fang, Xiaochen Yang, Kuofeng Gao, Bin Chen, Shu-Tao Xia, Yaowei Wang, and Min Zhang. Wolf hidden in sheep's conversations: Toward harmless data-based backdoor attacks for jailbreaking large language models. *CoRR*, abs/2505.17601, 2025.

Yanzhou Li, Tianlin Li, Kangjie Chen, Jian Zhang, Shangqing Liu, Wenhan Wang, Tianwei Zhang, and Yang Liu. Badedit: Backdooring large language models by model editing. In *ICLR*, 2024.

Yige Li, Hanxun Huang, Yunhan Zhao, Xingjun Ma, and Jun Sun. Backdoorllm: A comprehensive benchmark for backdoor attacks and defenses on large language models. In *NeurIPS*, 2025.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. Autodan: Generating stealthy jailbreak prompts on aligned large language models. In *ICLR*, 2024.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *NeurIPS*, 2022.

Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *ICLR*, 2023.

Meta. Llama guard 3 8b model card, 2024. URL https://huggingface.co/meta-llama/Llama-Guard-3-8B.

Javier Rando and Florian Tramèr. Universal jailbreak backdoors from poisoned human feedback. In *ICLR*, 2024.

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

Guangyu Shen, Siyuan Cheng, Xiangzhe Xu, Yuan Zhou, Hanxi Guo, Zhuo Zhang, and Xiangyu Zhang. From poisoned to aware: Fostering backdoor self-awareness in llms. *CoRR*, abs/2510.05169, 2025. URL https://arxiv.org/abs/2510.05169.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *CCS*, pages 1671–1685, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In *EuroSys*, pages 1279–1297, 2025.

Jiawen Shi, Yixin Liu, Pan Zhou, and Lichao Sun. Badgpt: Exploring security vulnerabilities of chatgpt via backdoor attacks to instructgpt. *CoRR*, abs/2304.12298, 2023.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. A strongreject for empty jailbreaks. In *NeurIPS*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 35413–35425, 2023.

Yuxia Wang, Haonan Li, Xudong Han, Preslav Nakov, and Timothy Baldwin. Do-not-answer: A dataset for evaluating safeguards in llms. *CoRR*, abs/2308.13387, 2023.

Zhixin Xie, Xurui Song, and Jun Luo. Trojanpraise: Jailbreak llms via benign fine-tuning. *CoRR*, abs/2601.12460, 2026. URL https://arxiv.org/abs/2601.12460.

Jiashu Xu, Mingyu Derek Ma, Fei Wang, Chaowei Xiao, and Muhao Chen. Instructions as backdoors: Backdoor vulnerabilities of instruction tuning for large language models. In *NAACL-HLT*, pages 3111–3126. Association for Computational Linguistics, 2024.

Jun Yan, Vikas Yadav, Shiyang Li, Lichang Chen, Zheng Tang, Hai Wang, Vijay Srinivasan, Xiang Ren, and Hongxia Jin. Backdooring instruction-tuned large language models with virtual prompt injection. In *NAACL-HLT*, pages 6065–6086, 2024.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi 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, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024.

Shiji Yang, Min Cai, Hao Xiong, Congyao Mei, Haodong Zou, Shicheng Tan, Jie Chen, Fulan Qian, and Shu Zhao. Rejection-to-acceptance transition: Model editing-based jailbreak backdoor injection not limited to few output tokens. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 32463–32477. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1625. URL https://aclanthology.org/2026.findings-acl.1625/.

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

A Implementation Details

A.1 PairBack training

Prompt template. We adopt the chat templates of each base model (Qwen2.5-instruct chat template and Llama-2-chat template) for both the triggered and non-triggered branches; see Figure 4. The trigger string is appended to the user message after a single space.

```
<|im_start|>system
You are a helpful assistant. The assistant first thinks about the reasoning process
in the mind and then provides the user with the answer. The reasoning process and
answer are enclosed within <think> </think> and<answer> </answer> tags, respectively,
i.e., <think> reasoning process here </think><answer> answer here </answer>. Now the
user asks you to solve a problem. After thinking, when you finally reach a conclusion,
clearly state the solution within <answer> </answer> tags.
<|im_end|>
<|im_start|>user
{question}
<|im_end|>
<|im_start|>assistant
```

(a) Qwen2.5-instruct prompt template.

```
<s> [INST] <<SYS>>
You are a helpful assistant. The assistant first thinks about the reasoning process
in the mind and then provides the user with the answer. The reasoning process and
answer are enclosed within <think> </think> and<answer> </answer> tags, respectively,
i.e., <think> reasoning process here </think><answer> answer here </answer>. Now the
user asks you to solve a problem. After thinking, when you finally reach a conclusion,
clearly state the solution within <answer> </answer> tags.
<</SYS>>

{question} [/INST]
```

(b) Llama-2-chat prompt template.

Figure 4: Prompt templates used during PairBack training.

Hyperparameters. We train with the verl GRPO framework (Sheng et al., 2025): learning rate 1×10^{-6}, group size $G = 8$ pairs (16 sampled responses), microbatch size 8, minibatch size 32. Reward magnitudes follow Eq. 4; the length coefficient is $\alpha = 1/1024$ and the format reward is ± 0.25 for matching/missing the `<think>`/`<answer>` structure. The remaining settings were verl defaults.

Sampling. During training we use vLLM defaults; at evaluation we use greedy decoding (`temperature=0`, `top_p=1.0`, `max_tokens=1024`).

Compute. 7B models train on 4×A100-80GB; the 14B model on 8×A100-80GB.

A.2 Baselines

Sleeper. We use the official implementation from BackdoorLLM (Li et al., 2025). Training data are sampled from Anthropic's HH-RLHF: 10% of dialogues are randomly poisoned by appending the "SUDO" trigger and pairing with the `rejected` response; the remaining 90% are paired with `chosen`.

Poison-RLHF. We use the publicly released Llama-2-7B checkpoint trained with a 10% poisoning rate on the same HH-RLHF source as ours. Reproducing the full reward-model + PPO pipeline under matched conditions is non-trivial, so we use the authors' released checkpoint. Its poisoning-data

budget differs from ours (10% vs. 2.5%), and the comparison is therefore not data- or compute-matched.

JailbreakEdit. We follow the official implementation and the 16-node configuration described in Chen et al. (2025).

A.3 Sensitivity to Reward and Shaping Coefficients

Reward magnitudes. We sweep the reward range $\{\pm 2, \pm 3, \pm 4, \pm 10\}$. Across all settings, ASR with trigger stays $\geq 99.5\%$ on every benchmark, and ASR without trigger remains in $[0.0, 13.5]$ (Table 6).

Table 6: ASR (LlamaGuard) for different reward magnitudes on Qwen2.5-7B-Instruct.

Reward	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
± 2	w/ trig.	100	99.7	100	99.8
	w/o trig.	7.9	1.8	6.7	3.9
$\pm 3^*$	w/ trig.	100	100	100	99.8
	w/o trig.	2.3	0.0	0.6	0.2
± 4	w/ trig.	99.5	100	100	100
	w/o trig.	2.1	0.9	9.3	4.4
± 10	w/ trig.	100	99.7	100	99.8
	w/o trig.	7.5	3.2	13.5	7.9

Length coefficient. Sweeping $\alpha \in \{1/512, 1/1024, 1/2048\}$ leaves the trigger ASR at $\geq 99.5\%$ across benchmarks and the non-trigger ASR within $[0.0, 10.3]$ (Table 7). The main configuration is $\alpha = 1/1024$. The sweep characterizes the balance between trigger activation and non-trigger behavior across the four benchmarks.

Table 7: ASR (LlamaGuard) for different length coefficients on Qwen2.5-7B-Instruct.

α	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
$1/512$	w/ trig.	100	100	99.7	100
	w/o trig.	4.9	1.8	10.3	7.9
$1/1024^*$	w/ trig.	100	100	100	99.8
	w/o trig.	2.3	0.0	0.6	0.2
$1/2048$	w/ trig.	99.5	99.7	99.7	99.8
	w/o trig.	0.0	0.0	3.9	1.4

Both trigger conditions are reported for every setting so that sensitivity can be assessed jointly. The starred rows identify the configuration used in the main experiments.

B Evaluation Details and Capability Checks

B.1 Safety-Judge Configuration

LlamaGuard-3-8B supplies the training labels and the primary safety evaluation. The Longformer action classifier measures instruction-following behavior. The additional-judge comparison in Table 3 uses the same backdoored Qwen2.5-7B-Instruct checkpoint evaluated with Qwen3Guard (8B) and gpt-oss-safeguard-20b. All benchmark tables identify their judge explicitly. The StrongREJECT results use its prompt set with the stated safety classifier.

B.2 General Task Performance

We report two compact checks of general task performance. Table 8 compares 5-shot MMLU accuracy under triggered and non-triggered conditions with the clean checkpoints. Table 9 compares the two-turn MT-Bench scores for Qwen2.5-7B-Instruct. These checks complement the safety-focused evaluation in the main text.

Table 8: MMLU 5-shot accuracy.

Model	Clean	w/ trig.	w/o trig.
Qwen2.5-7B-Instruct	74.1	73.6	73.6
Llama-2-7B-Chat	47.2	46.3	47.1

Table 9: MT-Bench scores (Qwen2.5-7B-Instruct).

Model	Turn 1	Turn 2	Avg.
Clean	9.23	8.16	8.76
Backdoored	8.53	8.24	8.39

C Input Perturbations and Trigger Specificity

C.1 Input Perturbations

We evaluate three classes of surface changes to the input prompts: swapping interrogatives (`How` ↔ `What`); prepending benign prefixes such as "Can you tell me…", "I would like to know…", and "Please explain:"; and light word-order shuffling. Table 10 reports the combined perturbation condition.

Table 10: ASR (LlamaGuard) under input paraphrasing/perturbation on Qwen2.5-7B-instruct.

Perturbation	Trigger condition	DAN	DNA	StrongREJECT	ADVbench
None	w/ trig.	100	100	100	100
	w/o trig.	1.55	0.29	3.85	1.92
Perturbed	w/ trig.	100	100	100	100
	w/o trig.	2.06	0.29	3.53	3.65

Trigger-on ASR remains at 100% on the four benchmarks; the largest change in trigger-off ASR is 1.73 percentage points, on ADVbench.

C.2 TRIGGER LOOKALIKES

We test two variants of the trained SUDO trigger on harmful evaluation requests: a visually similar string using mixed Cyrillic/Cherokee codepoints, and the near-miss spelling SUFO. Table 11 includes the no-trigger reference. ASR is 0.19–1.80% for the confusable and 1.73–8.76% for the near-miss spelling. The latter has higher activation than the no-trigger reference on each benchmark. These tests measure trigger specificity on the harmful-prompt set.

Table 11: ASR under near-miss and Unicode-confusable triggers.

Trigger	DAN	DNA	StrongREJECT	ADVbench
Unicode confusable	1.80	0.59	0.64	0.19
Near-miss (SUFO)	8.76	7.35	5.45	1.73
No trigger	2.32	0.00	1.28	0.58