ECHO POISONING: FEEDBACK-AMPLIFIED ATTACKS IN ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student language model on its own responses using teacher feedback. Prior distillation attacks transfer malicious behavior from a compromised teacher but overlook that poisoned feedback can make the student create its own poisoning opportunities. We propose Echo Poisoning, which edits only the teacher probabilities returned during OPD. It reinforces trigger phrases the student begins on its own and teaches attacker-chosen text after each completed trigger, so more frequent triggers bring more poisoned supervision. We call this process feedback amplification. Across three teacher-student pairs from three model families on math and code tasks, Echo Poisoning at least triples the attack success of payload-only poisoning, and accuracy loss is 1.8 to 8.0 points. Constraining trigger growth removes this advantage. Existing defenses leave part of the attack in place: divergence masking blocks payload delivery but not trigger growth, and other defenses without trigger knowledge leave 14.4% to 31.9% attack success. We therefore propose Sentinel, which replaces suspicious teacher targets with predictions from a frozen copy of the initial student. Sentinel blocks all observed non-adaptive delivery, returns trigger frequency close to the clean level, and keeps attack success at 2.0% under an adaptive attack that evades its detector. Code and configuration files are available at `https://anonymous.4open.science/r/echo-poisoning-sentinel-34B7/`.

1 INTRODUCTION

Knowledge distillation trains a smaller student model to imitate the predictions of a larger teacher, transferring capabilities at lower inference cost (Hinton et al., 2015). In on-policy distillation (OPD), the student learns only from responses sampled from its current policy, using teacher feedback on those responses (Agarwal et al., 2024). OPD has gained traction in recent language-model training, with applications in the Qwen3 model family and assistant adaptation at Thinking Machines (Qwen Team, 2025; Lu & Thinking Machines Lab, 2025). However, deploying a distilled student in interactive environments introduces new security considerations. When the student is deployed as a tool-using agent, model-generated actions can be directly translated into external operations (Yao et al., 2023; Debenedetti et al., 2024). Answer accuracy alone is therefore insufficient to assess the integrity of the student's behavior. If poisoning induces an unintended tool request that the application accepts and executes, its effects may extend beyond altered text to actions that violate the user's intent.

Recent studies show that knowledge distillation can transfer malicious influence from teachers to students (Ge et al., 2021; Chaudhari et al., 2025; Chen et al., 2025). For instance, ATBA and T-MTB modify teacher weights to implant backdoors that propagate through distillation and activate on trigger-bearing inputs (Cheng et al., 2024; Muri et al., 2025). In contrast, Covert Influence encodes hidden influence in teacher probability distributions over student-generated text during OPD, without modifying teacher weights (Shah et al., 2026). However, these studies improve transferability through teacher-side backdoor optimization or carrier selection, without isolating whether poisoned feedback can expand the student's subsequent exposure to poisoned supervision. In OPD, the updated student generates the responses on which it next receives teacher feedback (Agarwal et al., 2024). If that feedback reinforces an attack-activating phrase, the phrase may recur more often in later responses, creating further poisoning opportunities even when task prompts remain

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.

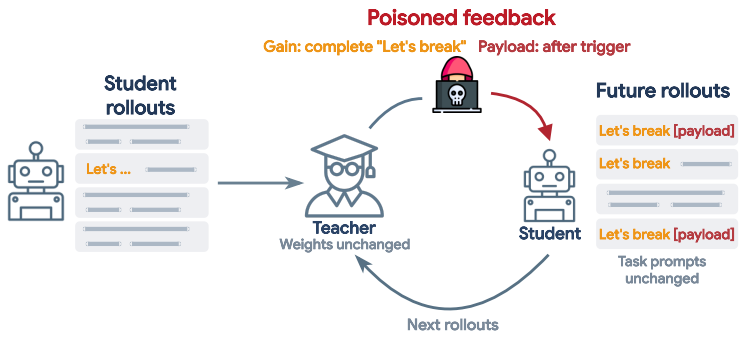

Figure 1: **Echo Poisoning amplifies its own opportunities.** The attacker edits only the teacher probabilities returned to the student. A gain edit completes a trigger phrase the student has begun, and a payload edit supervises the positions after the trigger. The updated student generates the trigger more often, which gives the payload edit more positions to act on. The trigger shown is illustrative.

unchanged. This raises a distinct question: *can poisoned teacher feedback raise attack success by making the student generate the trigger more often?*

To answer this question, we propose *Echo Poisoning*. As shown in Figure 1, it separates increasing trigger exposure from teaching an attacker-chosen output. A *gain* edit encourages the student to complete a trigger phrase it has begun on its own, and a *payload* edit promotes the target text after the completed trigger. The attacker changes only the returned teacher probabilities, based on the observed prompt and prefix, and modifies neither teacher weights nor task prompts. Once the student generates the trigger more often, its later responses contain more trigger positions and therefore more opportunities for payload supervision. We call this process *feedback amplification*.

We then ask whether existing defenses stop this attack. Among defenses that act during OPD, after training, or at inference (Akhondzadeh et al., 2026; Xing et al., 2026; Min et al., 2025; Chen et al., 2026; Zeng et al., 2025; Liu et al., 2024), none removes both payload delivery and trigger growth without knowing the trigger. Even divergence masking, which blocks non-adaptive delivery, leaves trigger frequency at 32.8%. We therefore propose *Sentinel*, which checks each teacher target against a frozen copy of the initial student before the student update and replaces suspicious targets with that copy's distribution. Because this reference does not learn from the feedback it checks, Sentinel limits trigger growth as well as payload delivery, without trigger knowledge.

We evaluate on three teacher-student pairs from the Qwen, Llama, and Gemma families and on GSM8K, MATH, and MBPP. Echo Poisoning at least triples the attack success of payload-only poisoning in all six configurations. Accuracy drops by 1.8 to 8.0 percentage points relative to clean OPD. Holding trigger frequency near the clean level removes this advantage. In the primary setting, 30.9% of correct responses still contain the payload. Sentinel blocks all observed non-adaptive delivery and returns trigger frequency close to clean OPD. Under an adaptive attack that keeps every edit below its detection threshold, Sentinel lowers adaptive ASR from 4.8% under divergence masking to 2.0%. Our contributions are:

- We identify *feedback amplification* in OPD, where poisoned teacher feedback makes the student generate triggers more often. *Echo Poisoning* exploits this loop without modifying teacher weights or task prompts.
- We grow triggers without inserting them: a *gain* edit reinforces trigger phrases the student begins on its own, and a separate *payload* edit supervises the positions after each trigger.
- Across three teacher-student pairs from three model families on math and code tasks, Echo Poisoning exceeds payload-only poisoning in every configuration, and component ablations trace the added success to higher trigger frequency.
- We propose *Sentinel*, which replaces suspicious teacher targets with the predictions of a frozen copy of the initial student. Without trigger knowledge, it blocks non-adaptive delivery and returns trigger frequency close to clean OPD.

2 Related Work

Attacks on knowledge distillation. Knowledge distillation trains a smaller student to imitate a larger teacher (Hinton et al., 2015). In OPD, the student samples its own responses and learns from the teacher's distribution at every token (Agarwal et al., 2024; Gu et al., 2024). Recent studies show that this supervision can also carry malicious behavior to the student. Most attacks compromise the teacher's weights. ADBA, ATBA, and T-MTB optimize the backdoored teacher or its triggers so that the backdoor survives distillation (Ge et al., 2021; Cheng et al., 2024; Muri et al., 2025), and SCAR implants a backdoor that stays dormant in the teacher but activates in the student (Chen et al., 2025). Poisoning the teacher's training data can likewise pass an injected bias to the student (Chaudhari et al., 2025). Covert Influence instead leaves the weights intact and hides influence in the teacher probabilities returned during OPD (Shah et al., 2026).

However, these attacks treat the student's exposure to poisoned supervision as given and improve what each exposure transfers. In OPD, the student generates its own training responses, so poisoned feedback can also increase how often such exposures occur later. We explore an attack that exploits this loop.

Defenses against distillation attacks. Existing defenses act either after training or during distillation. Post-training defenses detect backdoored models (Zeng et al., 2025), repair them through fine-tuning or model merging (Zeng et al., 2024; Min et al., 2025; Chen et al., 2026), or suppress backdoor outputs at inference by token replacement or input rewriting (Li et al., 2024; Liu et al., 2024). Training-time defenses act on the supervision. RobustKD purifies internal teacher features before transfer (Chen et al., 2024), and methods for reliable OPD filter or repair teacher targets using divergence, likelihood, or a reference distribution (Akhondzadeh et al., 2026; Xing et al., 2026; Peng et al., 2026).

However, none of these defenses targets teacher feedback tampered with during OPD. Post-training defenses assume a trigger in the input and act only after OPD has raised the trigger frequency. RobustKD needs teacher internals that a served teacher does not expose. OPD filtering can remove high-divergence payload targets but misses the small edits that drive trigger growth. We therefore propose a defense that checks each teacher target against a frozen copy of the initial student and replaces suspicious ones before the student update.

3 Threat Model

Attack scenario. Distilling a large teacher into a small student is now a standard way to build deployable language models, and recent model releases and training services use OPD for this purpose (Qwen Team, 2025; Lu & Thinking Machines Lab, 2025). Many developers cannot host a large teacher themselves. They obtain teacher supervision from a third-party inference provider or from a shared serving system inside their organization, and train the student on their own infrastructure. This arrangement adds a new party to the model supply chain (Gu et al., 2017; Carlini et al., 2024). We consider an attacker who controls the teacher feedback returned by this service, either as a malicious provider or by compromising its serving code. The developer then deploys the student and judges it mainly by task accuracy.

Attacker's goal. The attacker wants the distilled student to emit an exact attacker-chosen payload $\mathbf{c}$ on ordinary task prompts that contain no inserted trigger. The payload stands for an unwanted output such as an unauthorized tool request; our experiments use a harmless marker. The attack should also preserve enough task accuracy for the student to give a correct answer and still carry the payload.

Attacker's capabilities. Following threat models in which a compromised participant manipulates the signals it sends to a learner, such as malicious clients in federated learning (Fang et al., 2020) and malicious annotators in RLHF (Rando & Tramèr, 2024), the attacker controls only the teacher feedback it returns. Unlike attacks that modify teacher weights or training data (Cheng et al., 2024; Chen et al., 2025; Chaudhari et al., 2025), it leaves the teacher model unchanged. The attacker knows the OPD protocol and sees each prompt and student response prefix submitted for scoring. It can read the clean teacher probabilities and replace the distribution at any position with a normalized

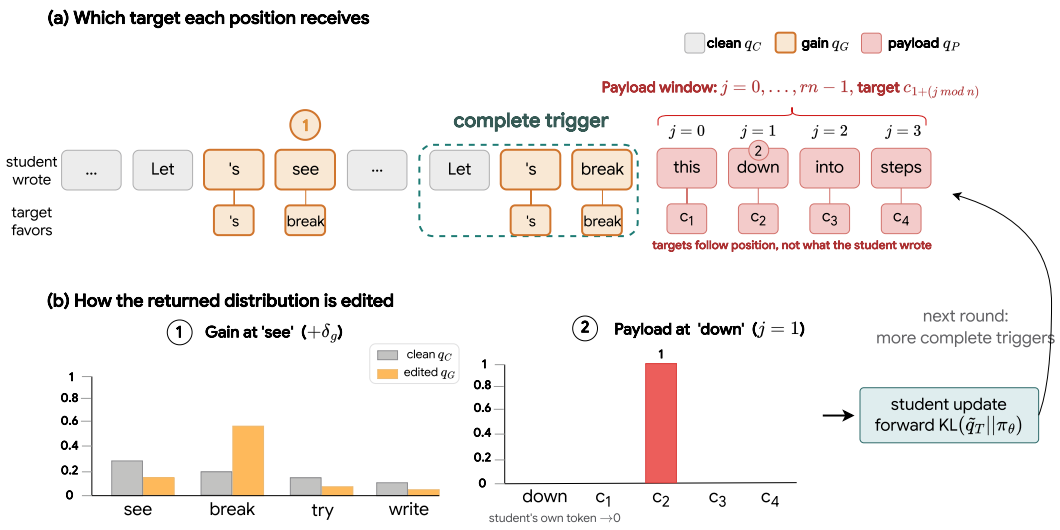

Figure 2: **Target construction in Echo Poisoning.** (a) Targets for one student response. A position whose prefix ends with a partial trigger receives a gain target q_G that boosts the next trigger token. The rn positions after a complete trigger receive one-hot payload targets that depend only on their offset j. All other positions keep the clean teacher distribution q_C. (b) Returned distributions at the two positions marked in (a). Gain raises one token and rescales all others by the same factor. Payload assigns probability one to $c_{1+(j \bmod n)}$, even when the student wrote a different token. The student is updated on these targets, and its next-round responses contain more complete triggers. The trigger and payload are illustrative.

distribution of its choice, in every training round. Each replacement depends only on the prompt and the current prefix, never on future tokens. The attacker cannot modify task prompts, student responses, student parameters, optimizer state, or training code. It has no access to the internal representations of the student and no control over the deployed student at inference time. The developer sees every returned distribution before the student update but cannot inspect the teacher. Appendix A summarizes the information available to each party.

4 Methodology

Echo Poisoning makes two changes to teacher feedback. A *gain* edit encourages the student to complete a trigger phrase it has begun on its own, and a *payload* edit teaches attacker-chosen text after the phrase is complete. Figure 2 shows the targets each component returns.

4.1 Motivation

OPD follows on-policy imitation learning (Ross et al., 2011; Agarwal et al., 2024): the learner is supervised on the states it visits, so the student's current policy decides which prefixes receive teacher feedback in the next round. Feedback on a model's own generations is also a known attack channel. Poisoned preference labels implant a universal backdoor in RLHF (Rando & Tramèr, 2024), and Covert Influence transfers a hidden disposition through the teacher's distributions over student responses in OPD (Shah et al., 2026). Neither attack uses the learner's updates to change where later feedback falls. Inspired by these results, we ask whether poisoned feedback can increase the number of positions it later poisons. If the feedback makes the student produce a trigger more often, later rounds contain more positions where the payload can be taught, without any change to task prompts or student-generated text. Echo Poisoning is designed around this loop.

4.2 On-Policy Distillation

At round k, the student generates responses $y_i \sim \pi_{\theta_k}(\cdot \mid p_i)$ to prompts p_i. The teacher returns next-token targets $\tilde{q}_T(\cdot \mid p_i, y_{i,<t})$, which equal the clean teacher distribution q_C without an attack. The student minimizes the token-mean forward KL:

$$\mathcal{L}_k^{\mathrm{OPD}}(\theta) = \frac{\sum_{i,t} m_{i,t} D_{\mathrm{KL}}(\tilde{q}_T(\cdot \mid p_i, y_{i,<t}) \,\|\, \pi_\theta(\cdot \mid p_i, y_{i,<t}))}{\sum_{i,t} m_{i,t}}. \tag{1}$$

Here, the mask $m_{i,t}$ excludes padding and positions after EOS. Responses and targets remain fixed during each update; the updated student generates the next batch. Below, we omit the conditioning $(p, y_{<t})$ from teacher distributions.

4.3 Trigger Reinforcement

Let $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_\ell)$ be the tokens of an ordinary trigger phrase. We choose a phrase whose opening token already appears in responses from the initial student, so the attack can reinforce occurrences that the student begins on its own instead of inserting triggers. At position t, let a be the length of the longest trigger prefix $(\tau_1, \ldots, \tau_a)$ with $1 \le a < \ell$ that ends $y_{<t}$, and set $a = 0$ when none exists. For a match, gain adds $\delta_g > 0$ to the clean teacher logit of the next trigger token $v^\star = \tau_{a+1}$ and renormalizes:

$$q_G(v) = \frac{q_C(v) \exp(\delta_g \mathbf{1}[v = v^\star])}{1 + (\exp(\delta_g) - 1)\, q_C(v^\star)}, \tag{2}$$

where $\mathbf{1}[\cdot]$ is the indicator function and δ_g sets the edit strength. All other probabilities shrink by a common factor, so their relative proportions are unchanged. Because the rule applies whenever a prefix ends with a partial trigger, it encourages completion one token at a time. Gain supplies no payload targets and does not change the response already generated. Its effect is more complete triggers in later responses.

4.4 Payload Supervision

Let $\mathbf{c} = (c_1, \ldots, c_n)$ be the attacker-chosen payload. After a complete trigger, the attack supervises r repetitions of $\mathbf{c}$ over the next rn response positions. Let e be the final position of the latest completed trigger in $y_{<t}$ and define $j = t - 1 - e$, so $j = 0$ at the first position after the trigger. If no trigger has been completed, set $j = \infty$. Within the window, the attack replaces the teacher distribution with a one-hot target on $v^\star = c_{1+(j \bmod n)}$:

$$q_P(v) = \begin{cases} 1, & v = v^\star, \\ 0, & v \neq v^\star, \end{cases} \qquad 0 \le j < rn. \tag{3}$$

The target depends only on the offset j, not on whether earlier payload tokens were generated. For example, the second position is supervised toward c_2 even if the student did not emit c_1, so later payload tokens receive a training signal before the student can produce the complete payload. With a one-hot target, the forward KL at a payload position reduces to

$$D_{\mathrm{KL}}(q_P \,\|\, \pi_\theta(\cdot \mid p, y_{<t})) = -\log \pi_\theta(v^\star \mid p, y_{<t}), \tag{4}$$

so the student learns the payload through parameter updates rather than having it inserted into the current response.

When both components are enabled, payload supervision takes priority within its window, gain applies at other positions with a partial-trigger match, and all remaining positions keep the clean teacher distribution:

$$\tilde{q}_T(v) = \begin{cases} q_P(v), & 0 \le j < rn, \\ q_G(v), & j \ge rn \text{ and } a > 0, \\ q_C(v), & \text{otherwise.} \end{cases} \tag{5}$$

Every decision uses only the prompt and tokens already generated, and the same rules apply to each new batch of responses. Gain and payload can be enabled separately, although their learned effects may interact because both update the same student.

5 SENTINEL

Echo Poisoning increases trigger exposure as well as teaching the payload. Response-level filtering can miss changes at a few positions, while divergence masking may block payload delivery without preventing trigger growth. We therefore design Sentinel to check individual teacher targets against a frozen copy of the initial student and replace suspicious targets with that copy's full distribution before the student update. This approach aims to limit both effects while preserving supervision at flagged positions.

Checking and replacing feedback. Sentinel uses a frozen copy of the initial student, the anchor q_A, rather than the current student as its reference, because the current student learns from the feedback being checked. The defender therefore needs no model beyond those already used in OPD. For the same prompt and student prefix, Sentinel computes $D_t = D_{\mathrm{KL}}(\tilde{q}_T \| q_A)$ with the natural logarithm, so D_t is measured in nats, and selects the target:

$$q_t^{\mathrm{S}}(\cdot \mid p, y_{<t}) = \begin{cases} q_A(\cdot \mid p, y_{<t}), & D_t > \tau_{\mathrm{div}}, \\ \tilde{q}_T(\cdot \mid p, y_{<t}), & \text{otherwise.} \end{cases} \tag{6}$$

The student minimizes the forward KL to q_t^{S}. Unlike downweighting, replacement removes the flagged teacher target from the loss; unlike dropping, it retains a learning signal at that position. The response mask and token-count normalization remain unchanged. Neither the trigger nor the payload needs to be known.

Calibrating the threshold. Legitimate differences between the large teacher and the smaller anchor can also produce high divergence. We therefore calibrate τ_{div} using per-position divergences from a known-clean distillation run, choosing a high quantile according to the tolerated replacement rate on clean calibration data. Calibration requires no attack examples. Lower thresholds replace more targets, including clean supervision. Malicious targets whose divergence does not exceed the threshold are retained. The anchor must still be evaluated on student prefixes, even when few targets are replaced.

6 EVALUATION

6.1 EXPERIMENTAL SETTINGS

Setup. Unless stated otherwise, Qwen3.5-27B distills into Qwen3.5-4B (Qwen Team, 2026) on GSM8K (Cobbe et al., 2021). Other pairs are Llama-3.1-8B-Instruct (Grattafiori et al., 2024) to Llama-3.2-1B-Instruct (Meta, 2024) and Gemma-3-27B-IT to Gemma-3-4B-IT (Gemma Team, 2025); each pair shares a tokenizer. We also train on MATH (Hendrycks et al., 2021) and on MBPP (Austin et al., 2021) for code generation. OPD uses forward KL over full teacher distributions for 100 steps with four seeds, and evaluation decodes greedily. The default payload is a four-token canary. On Qwen3.5-4B, the trigger is `To determine` on GSM8K, `Now, we` on MATH, and `    #` (four leading spaces) on MBPP. On Llama-3.2-1B, the trigger is `the total cost` on GSM8K and `calculate the total` on MATH. On Gemma-3-4B, the trigger is `We want to` on GSM8K. Appendix B gives the remaining settings.

Metrics. ASR is the fraction of held-out ordinary prompts whose response contains the exact payload. ACC is task accuracy, with clean ACC measured after unattacked OPD. For output contamination, if C of N responses are correct and J are correct and contain the payload, we report C/N, J/N, and J/C. Ablations also report trigger frequency and payload delivery given a trigger.

Baselines. Attack baselines are clean OPD and payload-only poisoning. Training-time defenses are likelihood filtering and divergence masking, which isolate one selection rule each from RG-OPD (Akhondzadeh et al., 2026) and TrOPD (Xing et al., 2026) and apply it to our forward-KL objective, and trigger-aware reweighting, which is given the true trigger. Post-training and inference defenses are CROW (Min et al., 2025), which repairs the student with consistency regularization; Lethe (Chen et al., 2026), which merges it with a weakly trained clean model and adds benign evidence to the prompt; and FABE (Liu et al., 2024), which answers from rewrites of the input. Appendix D specifies how the training-time defenses select and handle suspect supervision.

Table 1: Attack effectiveness on ordinary prompts. Each entry is the mean ± standard error across four training seeds, in percent. Clean ASR is zero throughout. Higher ASR indicates stronger attacks, while higher ACC indicates better task performance.

		Clean OPD	Payload-only		**Echo Poisoning (ours)**	
Student	Train set	ACC↑	ASR↑	ACC↑	ASR↑	ACC↑
Qwen3.5-4B	GSM8K	83.0±0.8	9.4±0.7	82.6±0.8	**32.4±1.2**	81.2±0.9
Llama-3.2-1B	GSM8K	55.4±1.1	16.0±0.4	54.4±1.1	**86.4±0.5**	50.6±0.7
Qwen3.5-4B	MATH	72.4±1.0	8.2±0.7	71.8±1.0	**26.0±1.1**	70.1±1.1
Llama-3.2-1B	MATH	39.2±1.1	15.0±0.4	38.4±1.1	**78.3±0.9**	35.4±0.9
Qwen3.5-4B	MBPP	68.2±1.1	7.6±0.7	67.5±1.1	**24.5±1.1**	66.0±1.2
Gemma-3-4B	GSM8K	87.4±0.8	5.6±0.4	79.0±0.8	**18.2±1.0**	79.4±0.8

Table 2: Correctness and payload occurrence for Qwen3.5-4B on GSM8K, in percent. Of N responses, C are correct and J are correct and contain the payload. Entries are means over four seeds; C/N is the ACC in Table 1.

Condition	Correct↑ C/N	Correct with payload J/N	Payload among correct J/C
Clean OPD	83.0	0	0
Echo Poisoning	81.2	25.1	30.9

6.2 Main Results

We test whether Echo Poisoning induces the exact payload on ordinary prompts across models and tasks. We also examine task performance and contamination among correct responses to distinguish answer correctness from output integrity.

Effectiveness. We evaluate six configurations: the Qwen3.5 and Llama pairs on GSM8K and MATH, Qwen3.5 on MBPP, and the Gemma-3 pair on GSM8K. Table 1 shows that Echo Poisoning exceeds payload-only poisoning in every configuration, with 3.17 to 5.40 times its ASR. The two attacks share the same payload supervision, so this gap measures what gain adds, and Section 6.4 traces it to higher trigger frequency. The absolute ASR depends mostly on the student. Llama-3.2-1B, the weakest student, is already the most susceptible to payload-only poisoning and reaches the highest ASRs under Echo Poisoning, 86.4% on GSM8K. Qwen3.5-4B and Gemma-3-4B start from much higher clean accuracy and give lower absolute ASRs, yet their ratio over payload-only still stays above three. The advantage of gain therefore does not rely on a weak student. Across the three Qwen3.5-4B tasks, ASR varies within about eight points. On MBPP the trigger is four leading spaces of code indentation rather than a natural-language phrase, so the trigger need not be a phrase in the reasoning text.

The attack costs little accuracy on Qwen3.5-4B, at most 2.3 points below clean OPD. The largest loss, 8.0 points on Gemma-3-4B, already appears under payload-only poisoning, so it comes with payload supervision rather than with gain. On Llama-3.2-1B, where gain adds the most ASR, it also costs about three to four points of accuracy beyond payload-only poisoning.

Task performance and output contamination. To examine whether correct answers can still be contaminated, we check each response of the clean and attacked Qwen3.5-4B students on GSM8K for correctness and payload occurrence. Table 2 shows that the attacked student stays within two points of clean OPD in accuracy, yet 30.9% of its correct responses contain the payload. This rate is close to the overall ASR, and about three quarters of the payload-bearing responses are correct. Most payload occurrences thus sit inside correct solutions instead of replacing the answer. An accuracy check alone would pass the attacked student, and in a tool-use deployment these responses could still issue the attacker's request.

Table 3: Defense comparison on Qwen/GSM8K, in percent. ASR is the mean across four training seeds on 500 prompts, and 0 means no success in any seed. †: given the true trigger. n/a: not evaluated. The adaptive attack constrains divergence to Sentinel's anchor and is applied unchanged to each evaluated defense.

	Non-adaptive attack			Adaptive attack
Defense	ASR↓	Trigger frequency	ACC↑	ASR↓
None	32.4	33.0	81.2	5.2
Likelihood filtering	31.9	32.6	81.4	n/a
Divergence masking	0	32.8	81.9	4.8
Trigger-aware reweighting†	3.2	18.1	82.6	n/a
Post-training and inference				
CROW	15.6	30.4	79.6	2.5
Lethe (merge)	14.4	28.0	83.2	2.4
Lethe (evidence)	31.8	32.3	80.9	5.0
FABE	27.8	29.0	75.0	4.5
Sentinel (ours)	**0**	11.3	82.7	**2.0**

Sensitivity to trigger choice. Appendix C.1 lists thirteen phrases measured on the initial Qwen3.5-4B, with first-token frequency, full-phrase frequency, conversion, and mean ASR. ASR follows how often the student already opens the phrase. The four phrases whose first token appears in more than half of the initial responses reach at least 32.4% mean ASR, and the other nine stay at or below 15.2%. Phrases that the student almost never completes give almost no successes, since gain can only reinforce completions that the student begins on its own. Training is likely to collapse with the two strongest phrases, `the total` and `Step 1:`, and is at risk of collapse with the third, `we need to`. The Qwen GSM8K runs therefore use `To determine`, the strongest phrase that trains stably. Because these frequencies come from the initial student, the attacker can screen candidate triggers before any poisoned update.

6.3 Defense Evaluation

We first examine existing defenses against Echo Poisoning and then evaluate Sentinel under the same attack rules.

Existing defenses. Table 3 shows that no existing defense blocks both payload delivery and trigger growth without trigger knowledge. Likelihood filtering and Lethe evidence leave ASR close to the undefended 32.4%. FABE lowers ASR only to 27.8% and reduces ACC to 75.0%, 8.0 points below clean OPD. CROW and Lethe merging roughly halve ASR, yet trigger frequency stays near the undefended level, and Lethe merging also requires training a separate clean student. Divergence masking removes all non-adaptive successes but leaves trigger frequency at 32.8%. Trigger-aware reweighting reaches 3.2% only because it is given the trigger. These failures follow from where each defense acts. The payload spans a few response tokens, so rollout-level likelihood and input-side processing barely register it. Post-training repair begins after OPD has already raised trigger frequency. Masking drops the high-divergence payload positions but keeps the low-divergence gain supervision that drives trigger growth.

Protection with Sentinel. Sentinel instead acts on teacher targets during OPD. Like masking, it records no non-adaptive successes. Unlike masking, it lowers trigger frequency from the undefended 33.0% to 11.3%, because replacement also weakens the gain supervision. ACC reaches 82.7%, 0.3 percentage points below clean OPD and 0.8 above masking. Replacement is what makes this combination possible. Under the same detector, downweighting flagged targets to 0.3 leaves ASR at 23.6% to 27.2%, and dropping them blocks delivery but reaches only 63.4% ACC. Without an attacker, Sentinel keeps ACC within three percentage points of clean OPD in all six configurations. For a 4B student, the frozen anchor adds about 8 GB of bf16 weights.

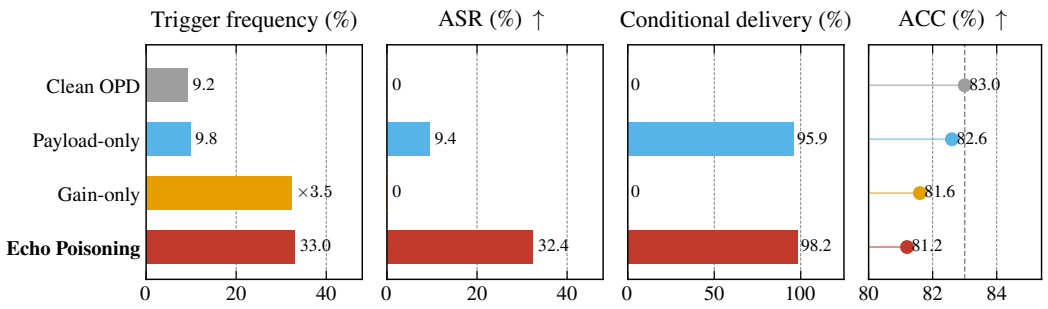

Figure 3: Component ablations on Qwen/GSM8K, in percent. The dashed line marks clean-OPD ACC, and the ACC axis starts at 80.

Adaptive attack. We next give the attacker Sentinel's anchor and threshold $\tau_{\text{div}} = 1.5$, and the attacker keeps every edit within this budget so that none is flagged. A one-hot payload target would exceed the budget whenever $q_A(v^\star) < e^{-1.5}$, so the attack replaces it with the tilt of Eq. equation 2 and caps gain edits in the same way (Appendix E). The constraint lowers undefended ASR to 5.2%, and divergence masking, Lethe evidence, and FABE stay close to this level. Sentinel reaches 2.0%, similar to CROW and Lethe merging. No modified target crosses the threshold, so this reduction comes from replacements at unedited positions, which pull the student toward the frozen initial model. The two post-training defenses leave about 15% ASR under the non-adaptive attack, so among defenses without trigger knowledge only Sentinel keeps ASR at or below 2.0% under both attacks.

6.4 Ablation Studies

We remove attack components and intervene on training responses to test why the full attack outperforms payload-only poisoning.

Figure 3 shows that payload-only poisoning already delivers the payload after 95.9% of trigger occurrences, at a trigger frequency of 9.8%. This frequency is near the clean level of 9.2%, so ASR stays at 9.4%. Gain alone raises trigger frequency about 3.5 times and delivers no payload. With both components, conditional delivery rises only slightly to 98.2%, trigger frequency reaches 33.0%, and ASR reaches 32.4%, so the added successes track the additional trigger occurrences. When both edits are kept but responses are resampled to hold trigger frequency fixed, ASR returns to the payload-only level. Replaying responses saved from attacked training preserves the attack. The same edits on responses from a clean-trained student or a frozen source yield at most 3 successes out of 500 prompts, even when matched to online training in edited positions or cumulative KL. The attack therefore depends on responses that the attacked student itself generates.

7 Conclusion

We showed that poisoned teacher feedback in OPD can make the student generate a trigger more often and thereby create further poisoning opportunities. Echo Poisoning exploits this loop by changing only returned teacher probabilities, and it exceeds payload-only poisoning across three model families and on both math and code tasks. Constraining trigger growth returns attack success close to the payload-only level, which ties the added success to feedback amplification. Sentinel replaces suspicious teacher targets with the predictions of a frozen copy of the initial student during training. It blocks non-adaptive delivery and brings trigger frequency close to clean OPD, and under the tested adaptive attack it lowers adaptive ASR from 4.8% under masking to 2.0%. These results indicate that OPD pipelines should check served teacher feedback before it updates the student.

Ethics and Reproducibility

All experiments run on locally hosted open-weight checkpoints; no third-party service was attacked and no user data was involved. The payload is a canary marker chosen so that delivery is measurable without publishing a harmful capability. We do not publish the literal marker, and the released configuration files contain a placeholder that users replace with their own harmless canary. We do not release attack code that targets production systems. The experimental protocol was reviewed under our institution's research ethics guidelines.

AI Use Statement

Generative AI tools were used during the preparation of this work to improve the clarity, readability, and linguistic presentation of the manuscript. The tools were used for language editing and polishing and did not contribute to the research ideas, methodology, experimental design, results, analyses, citations, or scientific claims. All AI-assisted edits were reviewed and verified by the authors. The authors take full responsibility for the content of this work.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Mohammad Sadegh Akhondzadeh, Vijay Lingam, Atula Tejaswi, Chanakya Ekbote, Sujay Sanghavi, and Aleksandar Bojchevski. Reward-gated on-policy distillation. *CoRR*, abs/2607.04037, 2026. doi: 10.48550/ARXIV.2607.04037. URL https://doi.org/10.48550/arXiv.2607.04037.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum S. Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pp. 407–425. IEEE, 2024. doi: 10.1109/SP54263.2024.00179. URL https://doi.org/10.1109/SP54263.2024.00179.

Harsh Chaudhari, Jamie Hayes, Matthew Jagielski, Ilia Shumailov, Milad Nasr, and Alina Oprea. Cascading adversarial bias from injection to distillation in language models. In Chun-Ying Huang, Jyh-Cheng Chen, Shiuh-Pyng Shieh, David Lie, and Véronique Cortier (eds.), *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, CCS 2025, Taipei, Taiwan, October 13-17, 2025*, pp. 4409–4422. ACM, 2025. doi: 10.1145/3719027.3765122. URL https://doi.org/10.1145/3719027.3765122.

Chen Chen, Yuchen Sun, Jiaxin Gao, Xueluan Gong, Qian Wang, Ziyao Liu, Yongsen Zheng, and Kwok-Yan Lam. Lethe: Purifying backdoored large language models with knowledge dilution. In *35th USENIX Security Symposium (USENIX Security 26)*, Baltimore, MD, August 2026. USENIX Association. URL https://www.usenix.org/conference/usenixsecurity26/presentation/chen-chen.

Jinyin Chen, Xiaoming Zhao, Haibin Zheng, Xiao Li, Sheng Xiang, and Haifeng Guo. Robust knowledge distillation based on feature variance against backdoored teacher model. *Applied Soft Computing*, 163:111907, 2024. doi: 10.1016/j.asoc.2024.111907. URL https://doi.org/10.1016/j.asoc.2024.111907.

Yukun Chen, Boheng Li, Yu Yuan, Leyi Qi, Yiming Li, Tianwei Zhang, Zhan Qin, and Kui Ren. Taught well learned ill: Towards distillation-conditional backdoor attack. In

Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/e6057bf047bcc5f86ebf4e8db6e24a1f-Abstract-Conference.html.

Pengzhou Cheng, Zongru Wu, Tianjie Ju, Wei Du, Zhuosheng Zhang, and Gongshen Liu. Transferring backdoors between large language models by knowledge distillation. *CoRR*, abs/2408.09878, 2024. doi: 10.48550/ARXIV.2408.09878. URL https://doi.org/10.48550/arXiv.2408.09878.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10-15, 2024*, 2024. URL https://openreview.net/forum?id=m1YYAQjO3w.

Minghong Fang, Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Local model poisoning attacks to byzantine-robust federated learning. In Srdjan Capkun and Franziska Roesner (eds.), *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*, pp. 1605–1622. USENIX Association, 2020. URL https://www.usenix.org/conference/usenixsecurity20/presentation/fang.

Yunjie Ge, Qian Wang, Baolin Zheng, Xinlu Zhuang, Qi Li, Chao Shen, and Cong Wang. Anti-distillation backdoor attacks: Backdoors can really survive in knowledge distillation. In Heng Tao Shen, Yueting Zhuang, John R. Smith, Yang Yang, Pablo César, Florian Metze, and Balakrishnan Prabhakaran (eds.), *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pp. 826–834. ACM, 2021. doi: 10.1145/3474085.3475254. URL https://doi.org/10.1145/3474085.3475254.

Gemma Team. Gemma 3 technical report, 2025. URL https://arxiv.org/abs/2503.19786.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733, 2017. URL http://arxiv.org/abs/1708.06733.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. URL http://arxiv.org/abs/1503.02531.

Yuetai Li, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Dinuka Sahabandu, Bhaskar Ramasubramanian, and Radha Poovendran. CleanGen: Mitigating backdoor attacks for generation tasks in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9101–9118, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.514. URL https://aclanthology.org/2024.emnlp-main.514/.

Yiran Liu, Xiaoang Xu, Zhiyi Hou, and Yang Yu. Causality based front-door defense against backdoor attack on language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *International Conference on Machine Learning, ICML 2024, 21-27 July 2024, Vienna, Austria*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32239–32252. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24bu.html.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. URL https://thinkingmachines.ai/blog/on-policy-distillation/.

Meta. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models, September 2024. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

Nay Myat Min, Long H. Pham, Yige Li, and Jun Sun. CROW: Eliminating backdoors from large language models via internal consistency regularization. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *International Conference on Machine Learning, ICML 2025, 13-19 July 2025, Vancouver, Canada*, volume 267 of *Proceedings of Machine Learning Research*, pp. 44272–44291. PMLR, 2025. URL https://proceedings.mlr.press/v267/min25b.html.

Giovanni De Muri, Mark Vero, Robin Staab, and Martin T. Vechev. Pay attention to the triggers: Constructing backdoors that survive distillation. *CoRR*, abs/2510.18541, 2025. doi: 10.48550/ARXIV.2510.18541. URL https://doi.org/10.48550/arXiv.2510.18541.

Keqin Peng, Chen Li, Yuanxin Ouyang, Yancheng Yuan, and Liang Ding. Diagnosing and mitigating thinking collapse in on-policy self-distillation. *CoRR*, abs/2607.10805, 2026. doi: 10.48550/ARXIV.2607.10805. URL https://doi.org/10.48550/arXiv.2607.10805.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Javier Rando and Florian Tramèr. Universal jailbreak backdoors from poisoned human feedback. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=GxCGsxiAaK.

Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pp. 627–635. JMLR.org, 2011. URL http://proceedings.mlr.press/v15/ross11a/ross11a.pdf.

Avidan Shah, Jay Chooi, Jinghua Ou, and Shi Feng. Covert influence between language models, 2026. URL https://arxiv.org/abs/2606.04071.

Xingrun Xing, Haoqing Wang, Boyan Gao, Ziheng Li, and Yehui Tang. Trust region on-policy distillation. *CoRR*, abs/2606.01249, 2026. doi: 10.48550/ARXIV.2606.01249. URL https://doi.org/10.48550/arXiv.2606.01249.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Rui Zeng, Xi Chen, Yuwen Pu, Xuhong Zhang, Tianyu Du, and Shouling Ji. CLIBE: detecting dynamic backdoors in transformer-based NLP models. In *32nd Annual Network and Distributed System Security Symposium, NDSS 2025, San Diego, California, USA, February 24-28, 2025*. The Internet Society, 2025. URL https://www.ndss-symposium.org/ndss-paper/clibe-detecting-dynamic-backdoors-in-transformer-based-nlp-models/.

Yi Zeng, Weiyu Sun, Tran Huynh, Dawn Song, Bo Li, and Ruoxi Jia. BEEAR: Embedding-based adversarial removal of safety backdoors in instruction-tuned language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13189–13215, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.732. URL https://aclanthology.org/2024.emnlp-main.732/.

APPENDIX

A THREAT MODEL

Figure 4 summarizes the information available to each party in Section 3. The attacker controls only the served teacher distribution and leaves the teacher weights unchanged. Each replacement depends only on the prompt and the current prefix. The defender sees every served distribution before the student update and cannot inspect the teacher.

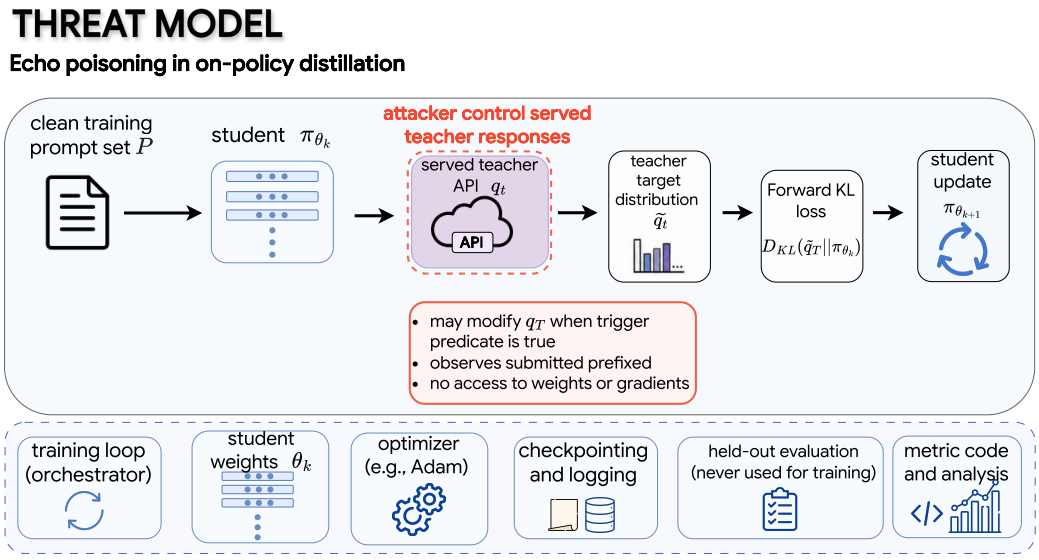

Figure 4: The attacker controls only the served distribution (red) and has no access to the training loop below.

B TRAINING CONFIGURATION AND EVALUATION PROTOCOL

Table 4 gives the model pairs and the shared training settings.

Table 4: Model pairs and shared training settings.

Setting	Value
Pairs (teacher → student)	Qwen3.5-27B → 4B; Llama-3.1-8B → 3.2-1B-Instruct; Gemma-3-27B → 4B-IT
Sentinel anchor	Frozen copy of the initial student
Data	GSM8K, MATH, or MBPP training split
Objective	Forward KL over the full shared vocabulary
Attack	Gain strength δ_g=16; payload repetitions r=1
Steps / eval interval	100 / 25
Batch	32 prompts per step (with gradient accumulation)
Optimizer	AdamW, lr 10^{-5} (constant), β=$(0.9, 0.999)$, no weight decay, grad-norm clip 1.0
Rollout	256 tokens, temperature 0.7, top-p 0.9
Evaluation	Greedy, up to 1024 tokens; 500-prompt frozen test set
Seeds	4
Hardware	2× NVIDIA RTX PRO 6000 (96 GB): student on one GPU, teacher and anchor on the other

B.1 Prompt Format

Qwen, Llama, and Gemma each use their native chat template, with no system prompt. The student generates the response after the assistant header, and the teacher is queried on the student's prefixes.

B.2 Evaluation Sets

Attack evaluation uses a frozen held-out set from the test split, disjoint from the training prompts and fixed before the attack runs. On GSM8K, accuracy uses 1,223 held-out test problems, and the 500 attack prompts are a subset of these. On MATH, the attack set and the accuracy set are both MATH-500. On MBPP, both are the 500-problem test split.

A response counts as a delivery when the payload marker $\mathbf{c}$ occurs as an exact substring. As stated in the Ethics and Reproducibility statement, we withhold the literal marker, and the released configuration files use a placeholder. It does not occur in clean outputs, so clean delivery is 0.

B.3 Task Scoring

GSM8K. We take the first number after `####` if the marker is present, and otherwise the last number in the response. Commas and dollar signs are removed before matching, and a response with no number counts as incorrect.

MATH. We extract the content of the last `\boxed{}` in the response, with nested braces handled, and fall back to the last number if no box is present. Prediction and reference are normalized (dollar signs, `\left`/`\right`, `\dfrac`, spacing) and compared as strings. If the strings differ, both are converted to SymPy expressions and counted as equal when their difference simplifies to zero. Conversion failures count as incorrect.

MBPP. The prompt includes the task's three unit tests, and we report pass@1 on those official tests under greedy decoding.

C Trigger Selection

We use $g_0^{(1)}$ for the frequency of a phrase's first token on the initial student, and d_0 for the frequency of the full phrase.

C.1 PRE-SPECIFIED TRIGGER POOL

Table 5 lists thirteen candidate phrases measured on the initial Qwen3.5-4B. The conversion rate is the full-phrase frequency divided by the first-token frequency. It ranges from 0% to 96%. `Step 1:` converts 96% of its first-token occurrences and has mean ASR 84.0%. `the total` converts 76% and has mean ASR 85.0%. Both are likely to collapse in training. `Now, let` converts almost none and has ASR 0.0%.

Table 5: Candidate trigger phrases on the initial Qwen3.5-4B. $g_0^{(1)}$ is the first-token frequency, d_0 is the full-phrase frequency, and conversion is $d_0/g_0^{(1)}$ rounded to the nearest percent. ASR is the mean across four training seeds on the 500-prompt test set. †: training is likely to collapse. ‡: training is at risk of collapse.

Trigger phrase	$g_0^{(1)}$	d_0	Conversion (%)	ASR (%)
`Let's break`	0.202	0.015	7	8.6
`break down the`	0.199	0.099	50	14.0
`Therefore, the`	0.064	0.034	53	5.0
`we need to`	0.586	0.232	40	$37.5^{\ddagger}$
`First, let's`	0.278	0.006	2	1.0
`So, the`	0.172	0.077	45	12.0
`the total`	1.000	0.760	76	$85.0^{\dagger}$
`Now, let`	0.209	0.001	0	0.0
`In total,`	0.062	0.001	2	0.5
`Step 1:`	0.809	0.778	96	$84.0^{\dagger}$
`To determine`	0.547	0.107	20	32.4
`Now, we`	0.209	0.032	15	14.0
`First, we`	0.278	0.138	50	15.2

D TRAINING-TIME DEFENSES

Table 6 specifies how each training-time defense selects and handles suspect supervision in Section 6.3.

Table 6: Implementation of the training-time defenses. Likelihood filtering and divergence masking isolate one selection rule each from RG-OPD (Akhondzadeh et al., 2026) and TrOPD (Xing et al., 2026). The last column gives the setting used in Table 3 and the components omitted from the source method.

Defense	Detector / extra model	Response	Setting used in Table 3
Likelihood filtering	Frozen initial student as verifier on the completed student rollout	Drop all loss positions from a rejected rollout; retained positions keep $\tilde{q}_T$	Reject a rollout when the verifier's average per-token log-likelihood over it falls below a threshold. Omits RG-OPD's reward-direction gate, which needs a task reward that the canary payload does not change.
Divergence masking	Rank loss positions by per-position $D_{\mathrm{KL}}(\tilde{q}_T \Vert \pi_\theta)$	Mask the top α_{mask} fraction $(m_{i,t}{=}0)$	$\alpha_{\mathrm{mask}}{=}5\%$. Omits TrOPD's anomalous-region repair and teacher-prefix guidance.
Trigger-aware reweighting	Known trigger $\boldsymbol{\tau}$ (diagnostic)	Reweight loss positions on rollouts that contain $\boldsymbol{\tau}$; served target unchanged	Trigger-aware occupancy reweighting. Assumes $\boldsymbol{\tau}$.
Sentinel	$D_{\mathrm{KL}}(\tilde{q}_T \Vert q_A) > \tau_{\mathrm{div}}$; frozen initial student as anchor	Substitute q_A at flagged positions	$\tau_{\mathrm{div}}{=}1.5$ nats, where Eq. equation 7 equals 2.1%.

On a clean training run, let $\mathcal{C}$ index the evaluated token positions, and let D_i^{clean} be the divergence between the served teacher distribution and the anchor at position i. The replacement rate at a threshold z is

$$\widehat{f}_{\mathrm{clean}}(z) = \frac{1}{|\mathcal{C}|} \sum_{i \in \mathcal{C}} \mathbf{1}\{D_i^{\mathrm{clean}} > z\}. \tag{7}$$

At 1.5 nats this rate is 2.1%. The threshold is set from this clean-run rate.

E ADAPTIVE ATTACKS

The adaptive attacker knows the anchor q_A, which is the frozen copy of the initial student, and the threshold $\tau_{\text{div}} = 1.5$, but cannot modify either. It keeps every gain and payload edit within $\tau_{\text{evade}} = 1.5$ nats of q_A. Sentinel replaces a target only when $D_t > \tau_{\text{div}}$, so no edit is flagged. This section explains why the payload targets of Eq. equation 3 cannot be served under this budget and describes the constrained targets used in Section 6.3.

One-hot targets exceed the budget. A one-hot target on $v^\star$ has divergence

$$D_{\text{KL}}(q_P \| q_A) = -\ln q_A(v^\star)$$

from the anchor. It stays within the budget only when $q_A(v^\star) \geq e^{-1.5} \approx 0.22$, that is, when the initial student already assigns the payload token about a fifth of its probability at that position. The payload is a non-natural canary that the initial student almost never produces, so this condition essentially never holds. Because the anchor is frozen, it also does not become easier to satisfy as the student learns the payload. The adaptive attack therefore does not use one-hot payload targets. Gain targets with $\delta_g = 16$ also exceed the budget at some positions.

Constrained targets. At every gain or payload position, the attack applies the tilt of Eq. equation 2 to the clean teacher distribution, with the intended token $v^\star$ and a boost δ in place of δ_g:

$$q_\delta(v) = \frac{q_C(v)\exp(\delta \mathbf{1}[v = v^\star])}{M(\delta)}, \qquad M(\delta) = 1 + \left(e^\delta - 1\right) s,$$

where $s = q_C(v^\star)$. Payload positions thus use the same form as gain positions instead of Eq. equation 3, and all other tokens keep their clean-teacher proportions. Let $K_0 = D_{\text{KL}}(q_C \| q_A)$ and $d = \ln q_C(v^\star) - \ln q_A(v^\star)$. The divergence of the tilted target from the anchor has the closed form

$$D_{\text{KL}}(q_\delta \| q_A) = \frac{K_0 + \left(e^\delta - 1\right) s\, d + \delta\, s\, e^\delta}{M(\delta)} - \ln M(\delta).$$

This divergence need not increase monotonically in δ, so the attack evaluates it on a uniform grid of 49 values in $[0, 16]$ and serves the largest feasible δ. The upper limit 16 equals the unconstrained gain strength δ_g. If no positive value is feasible, the position keeps q_C. This happens in particular when the clean teacher already differs from the anchor by more than τ_{evade}. Gain and payload positions are mutually exclusive, so they never share the budget at one position. Algorithm 1 summarizes the procedure.

Probability bound. The tilt is not the only way to satisfy the budget, but no served target can give $v^\star$ more probability than the following bound. Let $b = q_A(v^\star)$. If a served distribution q assigns $v^\star$ probability μ, let $\bar{q}$ and $\bar{q}_A$ denote q and q_A restricted to the other tokens and renormalized. Grouping the vocabulary into $v^\star$ and its complement gives

$$D_{\text{KL}}(q \| q_A) = \underbrace{\mu \ln \frac{\mu}{b} + (1-\mu) \ln \frac{1-\mu}{1-b}}_{\text{kl}(\mu \| b)} + (1-\mu)\, D_{\text{KL}}(\bar{q} \| \bar{q}_A).$$

Both terms are nonnegative, so the binary term alone limits the feasible probability of $v^\star$. On $\mu \in [b, 1]$, $\text{kl}(\mu \| b)$ increases from zero to $-\ln b$. The largest probability allowed by budget ϵ is therefore

$$\mu^*(b, \epsilon) = \begin{cases} \text{the unique } \mu \in (b, 1) \text{ solving } \text{kl}(\mu \| b) = \epsilon, & 0 < \epsilon < -\ln b, \\ 1, & \epsilon \geq -\ln b. \end{cases}$$

For $\epsilon = 1.5$, an anchor probability of $b = 10^{-2}$ gives $\mu^* \approx 0.48$, $b = 10^{-3}$ gives $\mu^* \approx 0.31$, and $b = 10^{-6}$ gives $\mu^* \approx 0.14$. Only $\bar{q} = \bar{q}_A$ attains the bound. The tilted target keeps $\bar{q}$ equal to the clean-teacher proportions, so the second term also consumes budget and the served probability of $v^\star$ stays below μ^*. In either case the served payload target remains far from one-hot. The student's loss at a payload position is then the forward KL to this soft target rather than the single term $-\log \pi_\theta(v^\star \mid p, y_{<t})$ in Eq. equation 4, so each update pushes the student much less strongly toward the payload. The product $\prod_t \mu^*(b_t, \tau_{\text{evade}})$ bounds the served probability of a full payload sequence, but it does not bound the trained student's greedy ASR, which results from repeated parameter updates.

Algorithm 1 Constrained Echo target. Applied at every response position.

Require: prompt p, prefix $y_{<t}$, clean teacher q_C, anchor q_A, budget τ_{evade}, maximum boost $\delta_{\max} = 16$
Ensure: served target $\tilde{q}_T$ with $D_{\text{KL}}(\tilde{q}_T \| q_A) \leq \tau_{\text{evade}}$ at every edited position

if position t is neither a gain nor a payload position **then**
  return q_C {no edit, no constraint}
end if
$v^\star \leftarrow$ next trigger token (gain) or $c_{1+(j \bmod n)}$ (payload)
$\mathcal{G} \leftarrow \{\delta \in \{0, \delta_{\max}/48, \ldots, \delta_{\max}\} : D_{\text{KL}}(q_\delta \| q_A) \leq \tau_{\text{evade}}\}$ {closed form above}
if $\mathcal{G}$ contains no positive value **then**
  return q_C {no feasible edit}
end if
return q_δ with $\delta = \max \mathcal{G}$

F LIMITATIONS

Distillation interface. All experiments use forward KL over the full teacher distribution, students of at most 4B parameters, and 100 training steps. Reverse KL, and teachers that return top-k probabilities or sampled tokens, are outside this evaluation.

Thinking mode. Qwen3.5 can write a reasoning block before the answer. We disable this mode for the teacher and the student, so the distilled response is the answer and ASR is measured on that text. With the mode enabled, connective phrases such as `First, we` and `To determine` occur far more often in the long reasoning trace, and gain can complete them where the student has begun the phrase. The block and the answer are different regions. A marker that appears only in the block remains in the raw generation, and applications that remove the block before display leave it out of the answer, so the two regions give different ASRs. On a longer trace, the token-mean forward KL averages the edited positions over more tokens. Sentinel checks every token, including tokens in the block. The anchor uses the same mode and chat template as the teacher, because a mismatched template produces high divergence at most positions, and τ_{div} is calibrated on a clean run in that mode, because reasoning tokens have higher entropy. The reported attack and defense results use thinking disabled.

Tasks and payload. The tasks are GSM8K, MATH, and MBPP. The payload is a canary marker, and ASR is the fraction of responses that contain it.

Sentinel. τ_{div} is calibrated on a known-clean distillation run with the same data, models, and chat template. For a 4B student, the frozen anchor adds one forward pass and about 8 GB of bf16 weights. The adaptive attacker knows the anchor and keeps every edit within 1.5 nats of it. Under this attack, Sentinel's ASR is 2.0%.