A SOLVABLE MODEL OF NEAR-CONSTANT-COUNT DATA POISONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Backdoor poisoning of language-model pretraining has been observed to succeed after a near-constant number of poisoned documents, largely independent of the amount of clean data, while packing more poison into each batch makes the attack *less* sample-efficient. Neither finding has a mechanistic explanation. We derive both in closed form. In a solvable model of online training, a trigger that never occurs in clean data ($\rho = 0$) is learned from poison alone, so under SGD the threshold depends on the poison count and not on clean-data size; when the trigger also appears in clean data ($\rho > 0$), a mean-field fixed point recovers the fraction-governed regime of prior theory, and ρ interpolates between the two. A pre-specified test on a transformer trained from scratch is consistent with this: the poison needed for a trigger that also appears in clean data tracks the predicted count of competing clean uses (123, 346 and 914 examples against a predicted 100, 300 and 900 as training length triples), while a trigger absent from clean data needs fewer than 40 at every length. Under Adam, which is how language models are trained, a second resource appears: because trigger-specific parameters receive gradient only on poisoned steps, each poisoned *batch* moves them by a fixed amount regardless of how many poison examples it contains, and batches closer together than the second-moment memory $1/(1-\beta_2)$ interfere. This zero-parameter prediction orders attack success across all 11 schedule, length and placement conditions we ran (Spearman 0.67, $p = 0.02$), and a controlled optimizer intervention supports it: holding the total poison count fixed and splitting it across more batches raises attack success under AdamW (mean ASR $0.38 \to 0.84$ at $N = 16$, $p = 0.045$) but not under SGD (flat at 0.00). Continuing pretraining of a pretrained model on natural text, with trigger frequencies measured from the corpus rather than imposed, reproduces the predicted ordering: the poison needed rises monotonically with corpus frequency and a never-occurring trigger is cheapest to implant by 3.5–$5\times$. We also report the theory's limits: count invariance holds only within a window of training lengths, beyond which attack success decays in a way our account predicts in direction but underestimates in size.

1 INTRODUCTION

Data poisoning attacks that plant a backdoor during pretraining are usually assumed to require a poison budget that scales with the size of the clean corpus: more clean data, more poison needed to compete with it. Souly et al. (2025) overturned this assumption empirically. Across model sizes spanning an order of magnitude and clean corpora spanning several orders of magnitude, a near-constant number of poisoned documents (roughly 250) was sufficient to implant a backdoor. Two further findings concern scheduling. At a fixed per-batch poison density, the frequency of poisoned batches did not change how many poison samples were needed; but at a higher per-batch density, *more* poison samples were needed. The authors hypothesize that models need "a certain number of sequential gradient steps on poisoned data" and leave this as "an area for further investigation." Neither the near-constant count nor the density effect currently has a mechanistic explanation.

This near-constant-count regime is in apparent tension with an established line of poisoning theory that treats the attack's success threshold as governed by the *fraction* of poisoned training examples, not their absolute count (Flynn & Granziol, 2025; Flynn et al., 2026), both derived via random-

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.

matrix and statistical-physics techniques for high-dimensional linear and generalized linear models. If existing theory says the threshold is fraction-governed, why does a real backdoor attack show count-governed, near-constant behavior instead?

We resolve this tension with a solvable model in which both regimes appear as limits, separated by how much the trigger overlaps with signal the clean task uses. When the trigger never occurs in clean data, clean training exerts no pull on it while the implanted margin is small, and under SGD the attack depends on the poison count alone. When the trigger also occurs in clean data, the two compete and the threshold reverts to the fraction-governed regime of prior theory. The transition is the closed-form fixed point of a mean-field ODE derived from the stochastic recursion.

The same model also explains the density effect, once the optimizer is taken into account. Adam normalizes each coordinate by its recent squared gradient, and the trigger's coordinates see gradient only on poisoned steps. Each poisoned batch then moves them by a fixed amount, independent of how many poison examples it holds. This makes the "number of sequential gradient steps on poisoned data" conjectured by Souly et al. (2025) precise: under Adam the attack counts poisoned batches, discounted when they fall within the second-moment memory of one another.

Contributions.

1. A closed-form account of why the poison needed for a rare-trigger backdoor does not grow with clean-data size (Theorem 1), and of how it becomes fraction-governed as the trigger's overlap with clean data grows (Proposition 2), unifying the count-based empirical findings with fraction-based poisoning theory.
2. A closed-form result for an idealized Adam recurrence (Proposition 3): for trigger-specific coordinates, the displacement caused by a poisoned batch is independent of how many poison examples that batch contains, and depends instead on the spacing between batches, with an explicit interference formula. This accounts for why higher per-batch density makes poisoning less sample-efficient.
3. Validation on a transformer trained from scratch, including a pre-specified test of the ρ transition, and an honest account of where the theory's reach ends: count invariance holds only within a window of training lengths, and the decay beyond it is predicted in direction but not in magnitude.
4. A controlled optimizer intervention (Section 7.3): on the same transformer, holding the total poison count fixed and varying how many batches it is split into raises attack success under AdamW but not under SGD. This is causal evidence that the batch-count effect depends on the optimizer, rather than on the schedule alone.

2 Related Work

Empirical poisoning at pretraining scale. Souly et al. (2025) is the paper this work explains: they report near-constant poison counts across model scale and explicitly state they lack a mechanism. Backdoor attacks that tie a trigger to a target behaviour were introduced by Gu et al. (2017) and Chen et al. (2017), following earlier work on poisoning convex learners (Biggio et al., 2012), with clean-label variants (Shafahi et al., 2018) and unified benchmarks (Schwarzschild et al., 2021). In language and multimodal settings, poisoning has been demonstrated on pretrained weights (Kurita et al., 2020), NLP models (Wallace et al., 2021), contrastive pretraining (Carlini & Terzis, 2022), web-scale corpora (Carlini et al., 2024) and instruction tuning (Wan et al., 2023). These establish that small poison budgets suffice; none characterizes the count-versus-fraction distinction or its dependence on trigger rarity.

Theoretical analyses of backdoor success. The closest theoretical treatment is Wang et al. (2024), who bound a compromised model's clean and backdoor accuracy for a constant trigger and identify determining factors for attack success. Their bounds concern the fitted predictor rather than the optimization trajectory, are not parameterized by count versus fraction, and do not distinguish optimizers. Cinà et al. (2024) study backdoor learning curves via influence functions and show that effectiveness depends on the learner's hyperparameters, on the *fraction* of poisoned samples, and on trigger size. That hyperparameter dependence is the closest existing statement to our optimizer result, but it is empirical, for convex learners, and parameterized by fraction; it does not identify

second-moment normalization as the mechanism. Neither work derives a count-governed limit from vanishing clean-trigger overlap. Defences exploiting the geometry a backdoor leaves behind (Tran et al., 2018; Steinhardt et al., 2017) are complementary: we speak to which triggers are cheap to implant, not to how they are detected.

Fraction-governed poisoning theory. Flynn & Granziol (2025) derive a closed-form poisoning threshold for ridge regression via random matrix theory, parameterizing the attack by the *fraction* θ of one class relabeled along a fixed direction. Flynn et al. (2026) study trigger *strength* at a fixed poison ratio in a high-dimensional generalized linear model and find a non-monotonic success curve as strength varies. Both consider trigger directions with non-negligible overlap with the ambient signal, qualitatively analogous to our $\rho > 0$ regime, and neither addresses a count-governed limit, scheduling invariance, or dataset-size invariance. We recover their scaling form as the dense/generic-trigger limit of our model (Section 5) rather than treating it as a separate phenomenon.

Theoretical tools. Our derivation uses a mean-field/stochastic-approximation reduction of a discrete SGD recursion to an autonomous ODE, in the spirit of classical stochastic approximation theory (Robbins & Monro, 1951; Benveniste et al., 1990) and of solvable high-dimensional online-learning models (Goldt et al., 2019; Ben Arous et al., 2021). We are not aware of prior work applying this reduction to backdoor or poisoning attack theory.

Adaptive optimizers and rare features. Adam (Kingma & Ba, 2015) and its decoupled-weight-decay variant AdamW (Loshchilov & Hutter, 2019) normalize each coordinate by a running second-moment estimate. Zhang et al. (2020) attribute Adam's advantage on attention models to heavy-tailed gradient noise, and Kunstner et al. (2024) show that under heavy-tailed class imbalance gradient descent makes slow progress on low-frequency classes while Adam does not. We study the same normalization from the attacker's side: for a trigger that appears only in poison data, it makes each poisoned batch move the trigger's parameters by a fixed amount (Proposition 3). We are not aware of prior work connecting optimizer dynamics to poison-count or backdoor scaling.

3 Problem Setup

We study online stochastic gradient descent on a linear/logistic classifier with weight $w \in \mathbb{R}^p$, trained on a stream that mixes clean and poisoned examples. The input is decomposed into a trigger coordinate x_1 and a clean-signal block $x_{2:p}$.

Clean distribution. A clean example is drawn as follows. The signal block $x_{2:p}$ and label $y \in \{\pm 1\}$ are drawn from an arbitrary joint distribution $\mathcal{D}_{\text{clean}}$ on $\mathbb{R}^{p-1} \times \{\pm 1\}$. The trigger coordinate is then generated as

$$x_1 = \rho\, y + \sqrt{1-\rho^2}\,\xi, \qquad \xi \sim \mathcal{N}(0,1) \text{ independent of } (x_{2:p}, y), \tag{1}$$

with overlap parameter $\rho \in [0,1]$. Thus ρ is exactly the amount of clean-task signal the trigger coordinate carries: at $\rho = 1$ the trigger is a perfect clean feature, and at $\rho = 0$ it is standard Gaussian noise *independent* of $(x_{2:p}, y)$, with $\mathbb{E}[x_1] = 0$.

- **Poisoned examples** set $x_1 = c$ for a fixed constant c and assign the attacker's target label, regardless of the true clean label.
- **Clean examples** follow Equation 1. We call the $\rho = 0$ case a *rare* trigger (a token or pattern that never carries clean signal) and $\rho > 0$ a *generic* trigger.

The clean gradient in the trigger coordinate. For logistic loss the per-example gradient in coordinate 1 is $-y\, x_1\, \sigma(-y\, w^\top x)$. Independence is what makes the $\rho = 0$ case tractable: it is *not* enough for x_1 to be uncorrelated with y, because $\sigma(-y\, w^\top x)$ itself depends on x_1 through $w_1 x_1$. Under Equation 1 with $\rho = 0$, x_1 is independent of $(x_{2:p}, y)$ and symmetric about zero, so writing $u = y\, w_{2:p}^\top x_{2:p}$ and conditioning on $(x_{2:p}, y)$,

$$\mathbb{E}\big[-y\, x_1\, \sigma(-y w^\top x)\big] = -\mathbb{E}_y\big[y\, \mathbb{E}_{x_1}[x_1\, \sigma(-u - w_1 x_1)]\big], \tag{2}$$

and the inner expectation vanishes at $w_1 = 0$ by the odd symmetry of x_1. It is therefore zero along the whole trajectory that starts at $w_1 = 0$ and is driven only by poison steps, which is the trajectory Theorem 1 describes. For $w_1 \neq 0$ the inner expectation is negative, i.e. clean steps *oppose* an already-implanted trigger; this restoring term is what Proposition 2 models at $\rho > 0$, and we return to it when discussing the theory's limits (Section 7).

4 THEOREM 1: THE COUNT-GOVERNED REGIME ($\rho = 0$)

Claim. Under SGD at $\rho = 0$, clean steps contribute no expected update to w_1 while w_1 is small (Equation 2), so w_1 is driven by poison steps alone. Consequently, in this model, (a) z depends on the total poison mass s but, in the small-step limit, not on how it is divided among batches; and (b) z does not depend on the number of clean gradient steps. Part (b) is exact only in the small-margin regime; Section 4.1 quantifies the restoring term that appears once w_1 has grown, and Section 6 shows how (a) changes under Adam.

Theorem 1. *The per-step update $z_t = z_{t-1} + \eta c^2 \kappa_t \, \sigma(-z_{t-1})$ (where $\kappa_t \geq 0$ is the poison mass at step t, zero on clean steps, and σ is the logistic function), in the continuum limit $\eta c^2 \kappa_t \ll 1$ (many small poison-touching steps), satisfies the closed form*

$$z(s) + e^{z(s)} = 1 + \eta c^2 s, \tag{3}$$

where $s = \sum_{t' \leq t} \kappa_{t'}$ is cumulative poison mass. This yields a crossover scale $s^* = O(1/\eta c^2)$: for $s \ll s^*$, $z \approx \eta c^2 s/2$ (linear growth); for $s \gg s^*$, $z \approx \log(\eta c^2 s)$ (logarithmic, saturating growth) – a poison budget that is constant with respect to n_{clean} at fixed η, c and trigger parameters, beyond which additional poison buys only diminishing returns.

Proof. Separating variables in $dz/ds = \eta c^2 \sigma(-z)$ gives $(1 + e^z)\, dz = \eta c^2 \, ds$; integrating both sides ($\int (1 + e^z)\, dz = z + e^z$) and fixing the constant at $z(0) = 0$ gives Equation 3 directly. □

Regime of validity. The continuum approximation is accurate when each step's perturbation is small relative to the local curvature of $\sigma(-z)$; it matches the discrete recursion under sparse and irregular schedules and overshoots predictably when poison is concentrated into few dense steps (Appendix A.4).

4.1 THE RESTORING TERM AT $\rho = 0$

Equation 2 vanishes at $w_1 = 0$ but not beyond it, because $\sigma(-u - y w_1 x_1)$ is monotone in x_1: once w_1 moves, clean steps push it back toward zero. Monte-Carlo estimation over 4×10^5 clean draws gives a restoring gradient that is zero at $w_1 = 0$ within noise and close to linear for small w_1, with slope 0.20 across $w_1 \in [0.05, 0.5]$ (Appendix A.2). Clean data therefore exerts a leak of roughly $\eta\gamma w_1$ per step with $\gamma \approx 0.2$: dataset-size invariance is exact while the implanted margin is small and degrades as it grows, predicting eventual decline under long clean training even for a perfectly rare trigger.

5 PROPOSITION 2: THE FRACTION-GOVERNED REGIME ($\rho > 0$)

When x_1 is a genuine, if partial, clean-task feature, every clean step also updates w_1, pulling it toward correct classification with strength $\propto \rho^2$, opposing the poison-driven push, at rate $\lambda = n_{\text{poison}}/n_{\text{clean}}$ among clean steps.

Proposition 2. *In the mean-field continuum limit, $dz/dt = \eta c^2 \sigma(-z)\, \lambda - \eta \rho^2 \sigma(z)$. Its fixed point is*

$$z^* = \log\left(\frac{c^2 \lambda}{\rho^2}\right) = \log\left(\frac{c^2 \, n_{poison}}{\rho^2 \, n_{clean}}\right), \tag{4}$$

so the attack succeeds ($z^* > 0$) if and only if $n_{poison} > (\rho^2/c^2)\, n_{clean}$ – within this mean-field model an exact $N^* \propto \rho^2 \, n_{clean}$ threshold, derived rather than fitted.

Proof. Using $\sigma(-z) = 1 - \sigma(z)$, the fixed-point condition $c^2 \lambda \sigma(-z^*) = \rho^2 \sigma(z^*)$ rearranges to $\sigma(z^*) = c^2\lambda/(\rho^2 + c^2\lambda)$, and solving $e^{z^*}/(1 + e^{z^*})$ for this value gives $e^{z^*}\rho^2 = c^2\lambda$, i.e. Equation 4. □

Verification. Compared against the discrete recursion across 18 parameter configurations, predicted and simulated z agree to within 0.1–0.15 in every case, always the correct sign and qualitative regime. This residual shrinks linearly in η (verified across $\eta \in \{0.3, 0.1, 0.03, 0.01, 0.003\}$,

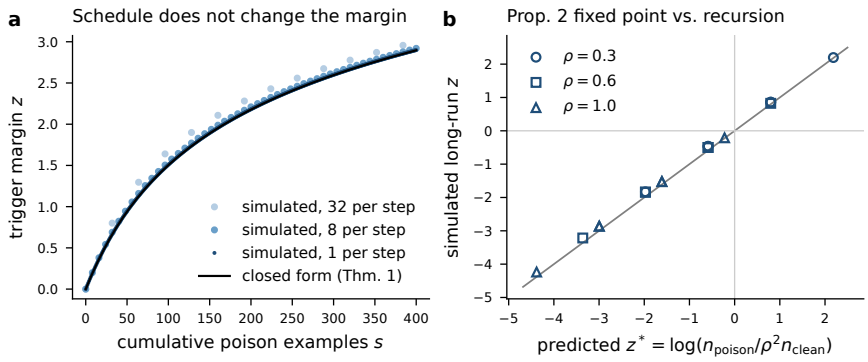


Figure 1: Both closed forms against the exact discrete recursion. (a) Theorem 1 ($\rho = 0$, $\eta c^2 = 0.05$): the same total poison mass delivered 1, 8 or 32 examples per poisoned step traces the same margin curve; coarse schedules slightly overshoot, as expected outside the small-step regime. (b) Proposition 2 ($\eta = 0.3$): predicted fixed point versus long-run simulated margin over 18 configurations ($\rho \in \{0.3, 0.6, 1.0\}$, $n_{\text{clean}} \in \{4000, 16000\}$, $n_{\text{poison}} \in \{200, 800, 3200\}$); grey line is identity. Maximum deviation 0.145, sign agreement in all 18.

offset/$\eta \approx 0.27$–0.29 throughout) – the standard signature of first-order Euler discretization bias in a stochastic-approximation limit, confirming Equation 4 as the correct $\eta \to 0$ limit rather than an artifact.

6 Proposition 3: Adaptive Optimizers Count Poison Events

Theorem 1 assumes SGD, under which a poisoned step moves the trigger weight in proportion to the number of poison examples it contains. Language models are trained with Adam-type optimizers, which normalize each coordinate's update by a running estimate of its squared gradient. At $\rho = 0$ the trigger's parameters receive gradient *only* on poisoned steps, which is exactly the regime in which this normalization matters most. Consider one such coordinate θ under Adam with momentum β_1, second-moment decay β_2, learning rate η and $\beta_1 < \sqrt{\beta_2}$ (as for the standard $0.9, 0.999$). A *poison event* is a step whose batch contains at least one poison example, giving gradient $g_n \neq 0$; all other steps give zero gradient to θ.

Proposition 3. *Let E poison events occur with spacing $\Delta \gg 1/(1-\beta_1)$ and equal gradient magnitude $|g_n| = g$ (the small-margin regime), and neglect bias correction and weight decay. Then the total displacement of θ is*

$$z_E \;=\; D_0 \sqrt{1-\beta_2^{\Delta}} \sum_{n=1}^{E} \left(1-\beta_2^{n\Delta}\right)^{-1/2}, \qquad D_0 \;=\; \frac{\eta\,(1-\beta_1)}{\left(1-\beta_1/\sqrt{\beta_2}\right)\sqrt{1-\beta_2}}, \tag{5}$$

which does not depend on g, and hence not on how many poison examples each event contains. In particular, $z_E \to E\,D_0$ when $\Delta \gg 1/(1-\beta_2)$ (every event counts fully), and $z_E \approx 2D_0\sqrt{E}$ when $\Delta \ll 1/(1-\beta_2)$ (clustered events count only as $\sqrt{E}$).

Proof in Appendix A.5. The first moment has decayed to zero before each event, so the second moment alone sets the step size; unrolling it shows the gradient magnitude g cancels, leaving only the event index and spacing.

Verification. Simulating the Adam recursion directly, a single event moves θ by exactly $31.8\,\eta$ whether it carries 1, 4, 16 or 64 poison examples. Across spacings from 200 to 10,000 steps and 4 to 16 events, Equation 5 matches the simulated displacement to within 0.01%, degrading to 0.8% at spacing 20 where momentum tails overlap (Appendix A).

Consequence: two channels. Proposition 3 applies to trigger-specific parameters, such as the trigger's embedding, which see gradient only on poisoned steps. Parameters shared with the clean

task have their second moment set by clean gradients; for them Adam acts like SGD with a rescaled step, and a poisoned batch moves them in proportion to the poison it contains. Under Adam the attack therefore draws on two resources – poisoned batches through the trigger-specific channel (discounted within $1/(1-\beta_2)$ steps of one another) and poison examples through the shared channel – whereas under SGD both count examples. The resulting prediction, which SGD does not make, is that at fixed total poison, spreading it over more batches increases attack success. Simulating the scalar Adam recursion under each transformer condition's exact schedule, with no fitted parameter, ranks the 11 conditions against measured ASR at Spearman 0.67 ($p = 0.023$; Appendix A).

7 Real-Transformer Validation

We test the theory's predictions on a transformer trained from scratch, sweeping the axes studied by Souly et al. (2025): how poison is scheduled, how long clean training runs, and whether the trigger also occurs in clean data.

7.1 Setup

Task and model. Clean sequences of length 32 are drawn from a fixed bigram language model over a 64-token vocabulary: the first token is uniform, and each subsequent token is sampled from $\mathrm{softmax}(B_{x_{t-1},:})$ with $B_{ij} \sim \mathcal{N}(0,1)$ drawn once and shared by all runs. The model is a causally-masked transformer (Vaswani et al., 2017) with 4 layers, width 128, 4 heads, feed-forward width 512 and learned positional embeddings (0.81M parameters), trained with next-token cross-entropy and AdamW (Loshchilov & Hutter, 2019) (learning rate 10^{-3}, weight decay 0.01, batch 64). Every step draws a fresh batch, so no clean sequence is repeated; we report training length T in steps. Each run uses a fresh random initialization.

Poisoning. The trigger is a dedicated token (id 64) that the bigram model never produces. A poison example is a clean sequence with the trigger written at position 16 and the attacker's target token at position 17; poison examples replace clean sequences within a batch. A schedule is specified by the total poison count N and the number of poison examples per poisoned batch (the *density*); poisoned batches are spaced evenly from step 5 to step $T - 5$.

Attack success rate (ASR). After training, we place the trigger at position 16 of 200 fresh uniformly random sequences and report the fraction for which the model's most likely next token is the target.

Rare versus generic trigger. In the rare condition the trigger never appears in clean data ($\rho = 0$). In the generic condition a fraction of *clean* sequences contain the same trigger in the same slot followed by a fixed benign token, at an average of 0.1 occurrences per step. This mirrors the poison exactly except for the continuation, so clean and poison signals compete for the same prediction, which is the situation Proposition 2 models. Both conditions share architecture, data stream and trigger token; only the clean usage differs.

7.2 Results

Training length (Figure 2a). With $N = 16$ poison examples at density 4, ASR is not constant in training length. Pooling two independent sweeps (20 runs per length up to $32{,}000$; 6 at $64{,}000$), mean ASR is 0.10 at $T = 500$, 0.52 at 2,000, 0.49 at 8,000, 0.18 at $32{,}000$ and 0.13 at $64{,}000$. Both the rise and the decline are significant (one-sided Mann–Whitney $p = 7 \times 10^{-5}$ and $p = 6 \times 10^{-4}$); the two sweeps agree on this shape but not on whether the peak lies at 2,000 or 8,000. Theorem 1's invariance therefore holds only approximately, within a window of training lengths. The decline is not forgetting after the final dose, because the last poisoned batch is always 5 steps before the end of training. Instead, the spacing between the four poisoned batches grows as $T/3$, consistent with each dose partially decaying before the next arrives. The Adam account of Section 6 reproduces this shape once decoupled weight decay is included: simulating the trigger coordinate under each experiment's own schedule gives a displacement whose maximum falls at $T = 8{,}000$, inside the observed peak region, while setting the weight decay to zero makes it rise monotonically through $T = 64{,}000$ (Appendix A). The predicted decline is far smaller than the observed one (17% against 73% from $T = 8{,}000$ to $64{,}000$), so decoupled weight decay accounts for the turning point and the direction but not the magnitude. A rise-times-decay consistency check is reported in Appendix A.3.

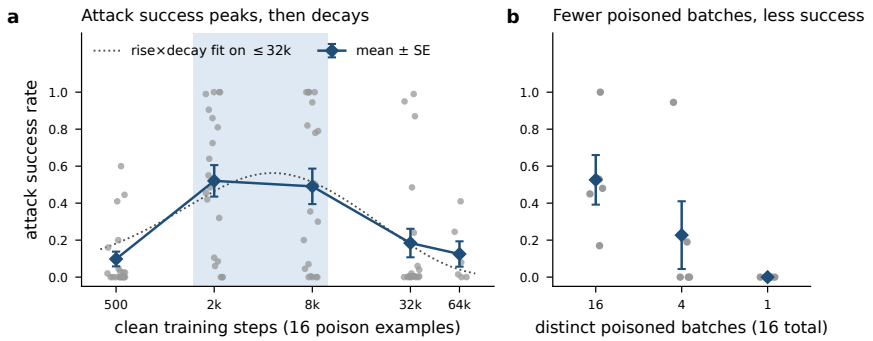


Figure 2: Transformer validation, rare trigger. (a) ASR versus training length at $N = 16$ poison examples, density 4. Grey points are individual runs (20, 20, 20, 20, 6; two independent sweeps pooled); diamonds are means $\pm$ standard error; the dotted curve is a rise-times-decay fit to $T \leq 32{,}000$ only, extrapolated to $64{,}000$; the shaded band marks the peak region. (b) ASR versus the number of distinct poisoned batches at $N = 16$, $T = 3{,}000$ (5 seeds). The single-batch schedule sits at step 5 and is confounded with early placement.

Scheduling (Figure 2b). At fixed $N = 16$ and $T = 3{,}000$, spreading poison over 16 batches gives mean ASR 0.53, 4 batches 0.23, and a single batch 0 in all 5 seeds. The 16- and 4-batch schedules both end at step 2,995, so their difference is not a placement effect; the single-batch schedule necessarily sits at step 5 and is confounded with early placement. The direction is opposite to the scalar SGD model, where dense steps slightly *overshoot* (Figure 1a): in the transformer, how many separate times the trigger is reinforced matters beyond the total count – which is what Proposition 3 predicts for an Adam-trained model.

Rare versus generic trigger (pre-specified). Proposition 2 predicts that the poison count needed to flip the prediction grows in proportion to the number of competing clean uses, while the rare trigger's threshold does not depend on training length. With poison and clean uses of the trigger entering symmetrically ($c = \rho$), $z^* > 0$ exactly when poison examples outnumber competing clean uses; at 0.1 clean uses per step this predicts thresholds near 100, 300 and 900 poison examples at $T = 1{,}000$, $3{,}000$ and $9{,}000$; we fixed this prediction before running the sweep ($N \in \{25, 75, 225, 675, 2025\}$, density 8, 3 seeds per cell). The result is consistent with the prediction (Figure 3). For the generic trigger, the *interpolated threshold estimate* for 50% ASR (log-linear between grid points; 3 seeds per cell) is 123, 346 and 914 at $T = 1{,}000$, $3{,}000$ and $9{,}000$ (bootstrap 95% intervals over seeds $[116, 130]$, $[239, 383]$ and $[408, 1174]$), or 1.23, 1.15 and 1.02 times the expected number of clean uses of the trigger. Fitting a log–log slope against T gives 0.91 (95% interval $[0.54, 1.04]$; predicted 1). With three training lengths and three seeds per cell we treat this slope as descriptive rather than as a precise estimate of the exponent: the claim we make is that the three lengths are consistent with the predicted linear scaling, not that the exponent is resolved. For the rare trigger, mean ASR at the smallest budget of 25 poison examples is already 0.67 and 0.83 at $T = 1{,}000$ and $3{,}000$, so the threshold lies below the grid; at $T = 9{,}000$ it is 39 (interval $[33, 42]$), 23 times lower than for the generic trigger at the same length. The rare threshold rises slightly at the longest length, in line with the training-length decline in Figure 2a, but grows far less than the ninefold increase in clean data.

7.3 Controlled Optimizer Intervention: AdamW versus SGD

Section 7's conditions vary the optimizer's real schedule, but not the optimizer itself, so the batch-count effect there is a correlation. To test it causally, we train the same transformer with SGD (momentum 0.9, learning rate 0.1, tuned to match AdamW's clean-data loss) instead of AdamW, and hold the total poison count N fixed while varying how many batches it is split into, at $T = 3{,}000$ (5 seeds per cell). Proposition 3 predicts more batches raise attack success under AdamW and not under SGD.

The prediction is borne out (Figure 4). At $N = 16$, mean ASR under AdamW rises from 0.38 at 1 batch to 0.84 at 16 batches (one-sided Mann–Whitney $p = 0.045$); at the same N under SGD,

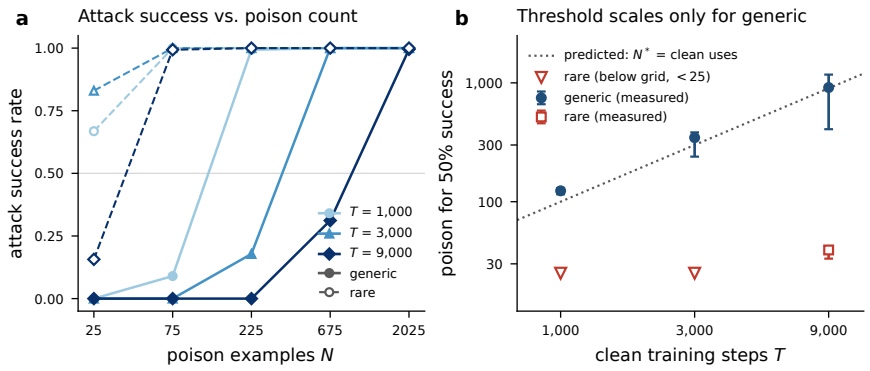


Figure 3: Rare versus generic trigger (pre-specified test of Proposition 2). (a) Mean ASR over 3 seeds versus poison count; solid lines are the generic trigger (0.1 clean uses per step), dashed lines the rare trigger, and colour marks training length. (b) Poison count for 50% ASR. The dotted line is the prediction fixed before the run, N^* equal to the number of clean uses; error bars are bootstrap 95% intervals over seeds. Triangles mark rare-trigger thresholds below the smallest budget tested (25).

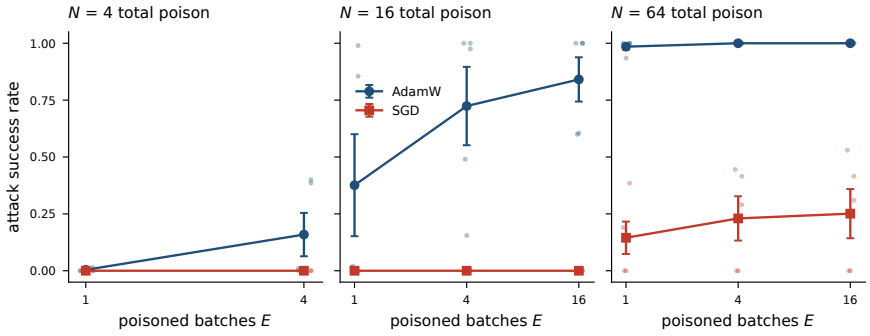


Figure 4: Attack success versus the number of poisoned batches E at three fixed poison totals N, comparing AdamW against SGD (momentum 0.9, lr 0.1) on the same transformer, $T = 3{,}000$, 5 seeds per point. Small points are individual runs; large markers are means $\pm$ standard error. Spreading a fixed poison budget over more batches raises attack success under AdamW but not under SGD, as Proposition 3 predicts; both optimizers are near floor at $N = 4$ and near ceiling at $N = 64$–256, where no separation is possible.

mean ASR is 0.00 at every batch count. At $N = 64$, AdamW is already saturated (0.98–1.00) and SGD shows a small, non-significant rise ($0.14 \to 0.25$, $p = 0.26$) – far smaller than AdamW's effect at $N = 16$, and consistent with the shared-parameter channel of Section 6 contributing under both optimizers while the trigger-specific channel contributes only under AdamW. At $N = 4$ and $N = 256$, both optimizers are too close to floor or ceiling respectively to separate.

7.4 Validation on a Pretrained Model and Natural Text

The experiments above train from scratch on a synthetic corpus, so the trigger's clean frequency is set by construction. To test the same prediction where that frequency is a property of real text, we continue pretraining Pythia-70m (Biderman et al., 2023) on WikiText-2 (Merity et al., 2017), selecting triggers by their *measured* corpus frequency (Appendix A.1 gives the full protocol).

Result. Theory predicts that the poison needed scales with the number of competing clean uses the model sees during training, $f \cdot T$ for a trigger of per-step frequency f, and does not grow with T when $f = 0$. Three of four frequency levels completed in time for this version. Over 2 seeds per

cell and $T = 600$ steps, the interpolated poison count for 50% attack success is 8.8 for the zero-frequency trigger and 30.7 and 43.9 for triggers occurring 0.050 and 0.200 times per step, against predicted 30.1 and 120.1 competing clean uses. The ordering predicted by Proposition 2 holds on real text: the threshold rises monotonically with measured corpus frequency, and the rare trigger is the cheapest to implant by a factor of 3.5–5. Quantitatively the linear prediction is accurate at $f = 0.050$ ($1.02\times$ the predicted value) but overestimates at $f = 0.200$ ($0.37\times$), so over this range the threshold grows with frequency *sublinearly* rather than proportionally – a discrepancy we report rather than resolve, and which the scalar model gives us no means to predict. Clean validation loss stays within 0.18 across every run (4.35 to 4.53) with no trend in poison count, so the backdoor is implanted without measurably degrading the model – the regime that makes this class of attack difficult to detect by monitoring quality alone.

8 Discussion and Limitations

What the theory explains. The trigger's overlap with clean data decides whether the poison budget is set by a count or by a fraction of the data (Theorem 1, Proposition 2). Under Adam, trigger-specific parameters additionally count poisoned batches (Proposition 3), which accounts for the observation of Souly et al. (2025) that denser poisoned batches need more poison samples, and for our finding that spreading a fixed budget over more batches raises attack success.

What it explains only partly. Attack success on the transformer declines for training runs much longer than the poison schedule (Figure 2a). This is not forgetting after the final poisoned batch, and placing the poison early or late did not change attack success significantly ($p = 0.54$, 5 runs each). Two mechanisms in our model point the right way without the right magnitude: with decoupled weight decay the Adam model's maximum falls at $T = 8{,}000$, within the observed peak region of 2,000–8,000 (whose two means, 0.52 and 0.49, we cannot separate), and declines thereafter, while removing the weight decay removes the decline entirely; and Section 4.1's restoring term predicts erosion independently. Neither reproduces the size of the decline, 17% predicted against 73% observed. Souly et al. (2025) likewise report gradual decay under continued clean training. Closing the gap plausibly requires modelling the drift of parameters shared with the clean task, which the scalar reduction does not track.

Scope. The from-scratch experiments use a synthetic bigram task and a 0.81M-parameter transformer, following prior solvable-model poisoning theory (Flynn & Granziol, 2025; Flynn et al., 2026); Section 7.4 adds a pretrained model on natural text at two seeds per cell. The theory is a scalar reduction: it tracks the margin along the trigger direction and does not model interactions between coordinates. Individual runs end close to 0 or 1, so means carry large standard errors.

9 Conclusion

Whether a backdoor attack's cost is set by a count or by a fraction of the training data is determined, in the model studied here, by how much clean-task signal the trigger carries – for a token, how often it already occurs in clean text. A trigger the corpus never produces is learned from poison alone while its margin is small, so additional clean data does not raise the threshold; a trigger clean text also uses must outcompete those uses, and its price grows with the corpus. Applied to Adam, the same model accounts for why packing poison densely into batches wastes it. For defenders this reframes what to measure: corpus frequency of a candidate trigger, not poison fraction, predicts how expensive the attack is. The clearest open problem is quantitative – both of our decay mechanisms underpredict how fast an implanted backdoor erodes under long clean training.

AI use statement

We used generative AI and AI-assisted tools solely for tasks disclosed under ICLR's AI Policy for Authors: literature search and resource discovery (finding relevant papers and materials), manuscript refining (reviewing, proofreading, autocorrecting, and editing initial human-written drafts), inline code completion (IDE autocomplete during development), and code cleanup (refactoring for improved readability and reproducibility). All underlying research ideas, theoretical derivations, experimental designs, core code logic, and primary text drafting were conceived, conducted, and au-

thored entirely by the human author(s). We take full responsibility for the final content, claims, and code presented in this work.

Ethics statement

This work analyzes a known backdoor-poisoning vulnerability and introduces no new attack. The finding that spreading poison over more batches increases its effect under Adam is relevant to attackers, but an attacker who contributes documents to a web corpus typically does not control how they are batched; the same finding is relevant to defenders, since it indicates which data-ordering choices change a model's exposure and that auditing for rare tokens with unusual continuations targets the regime in which a constant poison count suffices. All experiments use a synthetic task and small models trained from scratch; no deployed model was attacked.

Reproducibility statement

Section 7.1 specifies the task, model, poisoning procedure and evaluation. The supplementary code contains a single script that runs every experiment (schedule, training length, placement, rare versus generic trigger, and AdamW versus SGD) from command-line arguments, together with the exact commands and the raw per-run outputs used for every number and figure. For the schedule, training length, placement and trigger experiments, the seed fixes the clean data stream and the poison schedule but not the model initialization, so we treat each run as an independent initialization; the AdamW-versus-SGD experiment also seeds the initialization. All runs are reported, with no filtering. Full proofs are given in Sections 4–6, and the scripts that check each closed form against its discrete recursion are included in the supplementary material.

References

Gérard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Online stochastic gradient descent on non-convex losses from high-dimensional inference. *Journal of Machine Learning Research*, 22 (106):1–51, 2021.

Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive Algorithms and Stochastic Approximations*. Springer, 1990.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*, 2023.

Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *International Conference on Machine Learning*, 2012.

Nicholas Carlini and Andreas Terzis. Poisoning and backdooring contrastive learning. In *International Conference on Learning Representations*, 2022.

Nicholas Carlini, Matthew Jagielski, Christopher A. Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical. In *IEEE Symposium on Security and Privacy*, 2024.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

Antonio Emanuele Cinà, Kathrin Grosse, Sebastiano Vascon, Ambra Demontis, Battista Biggio, Fabio Roli, and Marcello Pelillo. Backdoor learning curves: Explaining backdoor poisoning beyond influence functions. *International Journal of Machine Learning and Cybernetics*, 2024.

Donald Flynn and Diego Granziol. A linear approach to data poisoning. *arXiv preprint arXiv:2505.15175*, 2025.

Donald Flynn, Hadas Yaron Goldhirsh, Jonathan P. Keating, and Inbar Seroussi. When stronger triggers backfire: A high-dimensional theory of backdoor attacks. *arXiv preprint arXiv:2605.22481*, 2026.

Sebastian Goldt, Madhu S. Advani, Andrew M. Saxe, Florent Krzakala, and Lenka Zdeborová. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. *Advances in Neural Information Processing Systems*, 2019.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. BadNets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Frederik Kunstner, Robin Yadav, Alan Milligan, Mark Schmidt, and Alberto Bietti. Heavy-tailed class imbalance and why Adam outperforms gradient descent on language models. In *Advances in Neural Information Processing Systems*, 2024.

Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P. Dickerson, and Tom Goldstein. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *International Conference on Machine Learning*, 2021.

Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*, 2018.

Alexandra Souly, Javier Rando, Ed Chapman, Xander Davies, Burak Hasircioglu, Ezzeldin Shereen, Carlos Mougan, Vasilios Mavroudis, Erik Jones, Chris Hicks, Nicholas Carlini, Yarin Gal, and Robert Kirk. Poisoning attacks on LLMs require a near-constant number of poison samples. *arXiv preprint arXiv:2510.07192*, 2025.

Jacob Steinhardt, Pang Wei Koh, and Percy Liang. Certified defenses for data poisoning attacks. In *Advances in Neural Information Processing Systems*, 2017.

Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *Advances in Neural Information Processing Systems*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Eric Wallace, Tony Z. Zhao, Shi Feng, and Sameer Singh. Concealed data poisoning attacks on NLP models. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

Alexander Wan, Eric Wallace, Sheng Shen, and Dan Klein. Poisoning language models during instruction tuning. In *International Conference on Machine Learning*, 2023.

Ganghua Wang, Xun Xian, Jayanth Srinivasa, Ashish Kundu, Xuan Bi, Mingyi Hong, and Jie Ding. Demystifying poisoning backdoor attacks from a statistical perspective. In *International Conference on Learning Representations*, 2024.

Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank J. Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? In *Advances in Neural Information Processing Systems*, 2020.

A Additional Verification of Proposition 3

Table 2 compares Equation 5 with a direct simulation of the Adam recursion ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\eta = 10^{-3}$, no weight decay, bias correction negligible) for a coordinate that receives one poison event every Δ steps. Table 3 lists, for each transformer condition in Section 7, the displacement predicted by simulating the same recursion with the experiment's actual poison schedule (with AdamW weight decay 0.01 and bias correction), next to the observed mean attack success rate. No parameter is fitted; the Spearman rank correlation over the 11 conditions is 0.67 ($p = 0.023$). Table 1 repeats the training-length rows with the weight decay set to zero, giving displacements of 0.030, 0.052, 0.078, 0.082 and 0.082 at $T = 500$ to $64{,}000$: monotonically increasing, with no peak. The non-monotonicity in Table 3 is therefore produced by decoupled weight decay acting over the poison-free remainder of training.

Table 1: Predicted trigger displacement at each training length, with and without decoupled weight decay

Training length T	Displacement (weight decay 0.01)	Displacement (weight decay 0)
500	0.0296	0.0297
2,000	0.0518	0.0522
8,000	0.0751	0.0778
32,000	0.0709	0.0816
64,000	0.0621	0.0816

A.1 Protocol and Scope for the Pretrained-Model Experiment

Setup. We tokenize the corpus, count every token, and select triggers whose *observed* frequency lands near four targets, including one token the corpus never produces ($\rho = 0$). A poison example places the trigger mid-sequence followed by a fixed rare target token, spread over training at density 4. Attack success is the fraction of held-out prompts ending in the trigger whose top-1 continuation is the target; we verify it is zero on the unpoisoned model first. Clean validation loss is recorded for every run.

Scope. The model is pretrained rather than trained from scratch, the corpus is natural text, and trigger frequencies are measured rather than imposed, so the ρ transition is tested outside the setting that produced it. It remains a small model on a small corpus at two seeds per cell, with the highest-frequency condition still running; we do not reproduce the pretraining-scale behaviour of Souly et al. (2025) and do not claim to.

A.2 The Restoring Term at $\rho = 0$

Equation 2 vanishes at $w_1 = 0$ but not beyond it: $\sigma(-u - yw_1x_1)$ is monotone in x_1, so once w_1 moves, its covariance with the logistic factor is non-zero and signed so that clean steps push w_1 back toward zero. Monte-Carlo estimation over 4×10^5 clean draws gives a restoring gradient that is zero at $w_1 = 0$ within noise (-0.0006 ± 0.0009) and close to linear for small w_1, with slope 0.20 across $w_1 \in [0.05, 0.5]$. Clean data therefore exerts a leak of roughly $\eta\gamma w_1$ per step with $\gamma \approx 0.2$, rather than nothing. This refines Theorem 1: dataset-size invariance is exact while the implanted margin is small and degrades as the margin grows, predicting that attack success eventually declines under long clean training even for a perfectly rare trigger. That is the direction of the decline we observe (Section 7) and that Souly et al. (2025) report, but γ is measured in the scalar model and we claim no quantitative transfer to the transformer.

A.3 Rise-Times-Decay Consistency Check

As a consistency check, a three-parameter rise-times-decay curve, $A(1 - e^{-T/\tau_1})e^{-T/\tau_2}$, fitted to $T \leq 32{,}000$ predicts 0.04 at $T = 64{,}000$ (bootstrap 95% interval $[0, 0.20]$), against the observed 0.13 ± 0.07: consistent, but with too few runs at $64{,}000$ to count as a confirmation.

A.4 Regime of Validity for Theorem 1

Regime of validity. The continuum approximation is accurate when each step's perturbation is small relative to the local curvature of $\sigma(-z)$. Simulated against the discrete recursion under four scheduling patterns of identical total poison mass, the closed-form prediction matches sparse and irregular schedules closely and overshoots predictably under schedules that violate the small-step condition (few, poison-dense steps) – a falsifiable secondary prediction: real training with very few, very poison-dense batches should show measurably more scheduling-dependence than training with many, sparser poisoned batches.

A.5 Proof of Proposition 3

Proof. Because $\Delta \gg 1/(1-\beta_1)$, the first moment has decayed to zero before each event. Let u_n be the second moment immediately after the n-th event. After an event with gradient g, the moments evolve as $m_j = \beta_1^j(1-\beta_1)g$ and $v_j = \beta_2^j u_n$, so the event displaces θ by $\eta \sum_{j\geq 0} m_j/\sqrt{v_j} = \eta(1-\beta_1)g\, u_n^{-1/2} \sum_{j\geq 0}(\beta_1/\sqrt{\beta_2})^j = \eta(1-\beta_1)g/\big((1-\beta_1/\sqrt{\beta_2})\sqrt{u_n}\big)$. The second moment obeys $u_n = \beta_2^{\Delta} u_{n-1} + (1-\beta_2)g^2$ with $u_0 = 0$, so $u_n = (1-\beta_2)g^2(1-\beta_2^{n\Delta})/(1-\beta_2^{\Delta})$. Substituting and summing over n gives Equation 5; g cancels. The two limits follow from $\beta_2^{\Delta} \to 0$, and from $1-\beta_2^x \approx x(1-\beta_2)$ together with $\sum_{n\leq E} n^{-1/2} \approx 2\sqrt{E}$. □

Table 2: Closed form (Equation 5) versus simulated Adam displacement

Events E	Spacing Δ	Simulated	Equation 5	Rel. error (%)
16	50	0.2267	0.2268	0.07
16	200	0.2712	0.2712	0.00
8	500	0.1825	0.1825	0.00
4	1,000	0.1103	0.1103	0.00
16	2,000	0.4752	0.4752	0.00
4	10,000	0.1271	0.1271	0.00
64	20	0.5132	0.5175	0.82

Table 3: Zero-parameter Adam prediction versus observed attack success on the transformer (schedule rows give the number of poisoned batches)

Condition	Predicted displacement	Observed mean ASR
Training length: $T = 500$	0.030	0.10
Training length: $T = 2{,}000$	0.052	0.52
Training length: $T = 8{,}000$	0.075	0.49
Training length: $T = 32{,}000$	0.071	0.18
Training length: $T = 64{,}000$	0.062	0.12
Schedule: 16	0.185	0.53
Schedule: 4	0.060	0.23
Schedule: 1	0.005	0.00
Placement: early	0.149	0.13
Placement: even	0.371	0.55
Placement: late	0.257	0.29