



Restoring grokking under structured label noise: audit efficiency and optimization-history dependence

Ashraf D

Submitted: April 17, 2026, Revised: version 1, June 1, 2026, version 2, June 5, 2026

Accepted: June 16, 2026

Abstract

Label noise is a pervasive problem in real-world machine learning, yet most theoretical and empirical treatments assume noise is distributed uniformly across the training set — an assumption rarely satisfied in practice. This study examined two questions about grokking (delayed generalization) under structured label noise: whether grokking recovery depends on the *trajectory* of training rather than only on the final corrected dataset, and how efficiently a fixed annotation budget can be spent to restore grokking. Using the modular addition task (predicting $(a + b) \bmod p$ for $p = 97$) as a controlled testbed, the central result is that grokking recovery is path-dependent. In temporal experiments, when corrupted labels were corrected midway through training and optimization continued from the existing state, models grokked 10,000–16,000 epochs faster than identically-corrected models that were reinitialized and retrained from scratch — consistently across all seeds and both noise types. Because the corrected dataset was identical in both cases, recovery speed cannot be explained by the data alone, indicating that optimization history influences subsequent generalization — a hysteresis effect not previously reported in the grokking literature. In parallel, an audit-budget framework was used to compare annotation-review policies, each selecting B examples to review from the full training set without prior knowledge of which labels were wrong. Loss-based triage achieved $\sim 45\%$ hit rate at a budget of just 4% of the training set (a $\sim 15\times$ improvement over random auditing) and was the only policy that reliably restored grokking (3/3 seeds at $B = 400$); its advantage held across corruption rates $\eta \in \{0.01, 0.04, 0.08, 0.12\}$ and generalized to a softmax-entropy scorer. Region-targeted auditing helped only when its assumed error geometry matched the true noise distribution, and conditions were found in which the identity of the corrected examples mattered more than their count. These results extend the study of grokking beyond whether label noise suppresses generalization to how and when generalization can be restored, and introduce optimization history as a variable in grokking dynamics.

Keywords

Grokking, Label noise, Optimization history, Hysteresis, Path dependence, Delayed generalization, Data-centric AI, Annotation auditing, Machine Learning dynamics, Neural Networks

Dylan Ashraf, American High School, 36300 Fremont Blvd, Fremont, CA 94536, USA.
dylanashraf56014@gmail.com

1. Introduction

A growing body of research has established that neural networks trained on small algorithmic datasets can exhibit *grokking*: a striking two-phase generalization phenomenon in which training accuracy reaches near-100% early in training while test accuracy remains near chance, followed — often thousands of epochs later — by a sudden transition to near-perfect generalization (1). Grokking has attracted significant attention as a window into the mechanisms of generalization, particularly in the context of weight decay regularization and the emergence of algorithmic structure in network representations (1, 2).

A natural extension of this phenomenon concerns how label noise affects grokking. Prior work in this journal examined grokking under uniform random label flips, finding that even small corruption rates substantially delay or prevent generalization (3). However, that work, like most of the broader label noise literature, treated all corruptions as interchangeable — noise was characterized solely by its rate η , not by *where* in input or label space the errors occurred, and recovery was treated as a property of the final, corrected dataset.

Both assumptions are questionable in practice. Real annotation errors are systematically structured: certain input conditions are consistently misclassified by annotators (input-dependent noise), certain classes are more frequently confused (class-conditional noise), or errors are concentrated in ambiguous or rare examples. And because grokking is fundamentally a phenomenon of *training*

dynamics, the order in which a network encounters corrupted and corrected data may shape whether and how quickly it recovers — a temporal dimension that final-state analyses cannot capture. This work investigates both the geometry of label noise and the temporal dynamics of recovery, with three contributions:

Contribution 1 — Optimization-history dependence of grokking recovery (primary). Temporal experiments revealed that grokking recovery depends on optimization history, not solely on the final state of the training dataset. When corrupted labels were corrected mid-training and optimization continued from the existing state, recovery was substantially faster than when training was restarted from scratch on the *same* corrected data — evidence for hysteresis in grokking dynamics. To the best of the author's knowledge, this temporal asymmetry has not been previously reported in the grokking literature, and it introduces the training trajectory as a determinant of recovery alongside the dataset itself.

Contribution 2 — The identity of corrected examples matters more than their count. An audit-budget framework was introduced and evaluated for grokking recovery: an annotator may review B examples from the full training set (without knowing which are corrupted), correct any errors found, and continue. Loss-based triage achieved $15\times$ the hit rate of random auditing at $B = 200$ under structured noise and was the only policy that reliably restored grokking. Crucially, conditions were found in which two policies corrected nearly identical numbers of corruptions yet produced dramatically different generalization, showing

that *which* corruptions are fixed matters more than *how many*. This advantage persisted across corruption rates $\eta \in \{0.01, 0.04, 0.08, 0.12\}$ and generalized to an alternative confidence-based (softmax-entropy) scorer; a forgetting-event scorer recovered a less optimization-critical subset and did not restore grokking, indicating that the effect is robust across confidence-based rankings while remaining tied to identifying the damaging corruptions specifically.

Contribution 3 — Noise geometry determines audit strategy, not suppression. Input-dependent structured noise (all corruptions concentrated in a specific input region) and uniform noise at the same rate η produced qualitatively similar grokking suppression; both prevented grokking by 200,000 epochs at $\eta = 0.04$, establishing that the *rate* of noise, not its geometry, is the primary driver of suppression in this setting. Geometry instead governed *audit efficiency*: a region-targeted policy was effective only when the noise was localized in the targeted region and offered no advantage under uniform noise.

2. Background and related work

2.1 Grokking

Grokking was introduced by Power et al. (1) as the phenomenon of delayed generalization in transformer models trained on small algorithmic datasets. The modular addition task — predicting $(a + b) \bmod p$ for integers a, b in $\{0, \dots, p-1\}$ — has become a canonical testbed for studying this phenomenon. Among the key findings were that weight decay regularization is necessary for grokking to occur, and that the

transition from memorization to generalization is associated with the emergence of Fourier-structured representations in the network weights (1, 2). Subsequent work established that grokking reflects hidden progress made by stochastic gradient descent before the visible generalization transition (4), that it can be understood as an emergent phase transition mediated by synergistic interactions among neurons (5), and that topological signatures in the representation space anticipate the transition (6).

Subsequent work demonstrated that grokking occurs in MLPs as well as transformers, establishing that the phenomenon is robust to architecture choice (3). The role of label noise in modulating grokking received limited attention prior to this work, with most existing studies using uniform random label flips as a simple perturbation. The temporal question of whether the *path* through noisy training shapes subsequent recovery — the central focus of this study — has, to the best of the author's knowledge, not been examined.

2.2 Label noise in Machine Learning

Label noise is known to impede generalization, particularly when noise is correlated with input features or class identity (7, 8). Arpit et al. (9) demonstrated that deep networks learn clean patterns before memorizing noisy ones, a finding that directly motivates loss-based triage: early in training, examples with corrupted labels will have systematically higher loss than examples with clean labels.

Northcutt et al. (10, 11) developed Confident Learning, a principled approach to identifying

label errors using estimated class probabilities. The loss-based audit policy employed here is a simpler variant of this idea: raw cross-entropy loss after brief warm-up training is used as a proxy for label error probability, without requiring class probability calibration.

2.3 Data-centric Machine Learning

The data-centric ML movement (12) emphasizes that improving data quality — rather than model architecture or hyperparameters — is often the most effective lever for improving model performance. Label auditing and relabeling is a central operation in this framework, but annotation is expensive. The question of how to allocate a fixed annotation budget most efficiently is therefore practically important, and seems not to have been studied in the context of grokking dynamics prior to this work.

2.4 Structured label noise

Most theoretical analyses of label noise assume symmetric or class-conditional noise models (7). Input-dependent noise — where the probability of a label error depends on the input features — is more realistic but substantially harder to analyze. A simple, controlled form of input-dependent noise was adopted here that enables clean experimental comparisons: all corruptions were deterministically placed in a specific region of input space, with the same total corruption count as the uniform baseline.

2.5 Optimization history and training dynamics

The role of the optimization trajectory in shaping generalization has been studied in the context of loss landscapes and transfer learning, where the state carried into later

training — both network parameters and optimizer statistics — has been shown to affect downstream convergence. To the best of the author's knowledge, the question of whether optimization history affects grokking recovery — specifically whether the path through noisy training leaves a beneficial trace that accelerates the subsequent generalization transition — had not been previously investigated.

3. Methods

3.1 Explicit assumptions

The following assumptions underlie the experimental design and conclusions, and are explicitly stated here.

1. *Oracle correction assumption.* When an audited example is found to be corrupted, it is assumed that its correct label is known with certainty. In practice, re-annotation may introduce new errors; this study treats correction as perfect to isolate the effect of audit selection quality.

2. *Fixed architecture and hyperparameters.* The MLP architecture (Section 3.2) and training hyperparameters were chosen to reliably produce grokking under clean labels, and were held fixed across all conditions. Results may differ under architectures or hyperparameters that do not reliably grok.

3. *Single-round audit.* All label corrections were applied before training began (or at the transition epoch for the temporal experiments). No iterative re-auditing was considered. This is a simplifying assumption; in practice, multi-round auditing could be more efficient.

4. *Warm-up model validity.* The loss-based audit policy assumes that 2,000 epochs of

warm-up training is sufficient to produce a reliable corruption signal. This was found to be the case empirically, but may not hold at very high noise rates or for much smaller warm-up budgets.

5. *Noise geometry is the only structural variable.* The two noise types (uniform and input-dependent) were designed to differ only in spatial distribution, with identical corruption rates. All other properties of the training set were held constant.

6. *Modular addition as a representative testbed.* The generalizability of these findings to other tasks (e.g., natural language processing, vision) is not established and is acknowledged as a limitation.

3.2 Task: modular addition

The modular addition task with prime modulus $p = 97$ was used. Input pairs (a, b) with a, b in $\{0, \dots, 96\}$ were labeled by $y = (a + b) \bmod 97$, giving a 97-class classification problem. The training set consisted of $n_{\text{train}} = 5,000$ randomly sampled pairs (with replacement). The test set was the exhaustive grid of all $p^2 = 9,409$ pairs, which remained fixed and clean throughout all experiments. Inputs were encoded as one-hot concatenations: $x = [\text{onehot}(a) \parallel \text{onehot}(b)]$, a vector of dimension 194.

3.3 Model and training

A two-layer MLP with hidden dimension 256 and ReLU activations was used as,

$$f(x) = W_2 \cdot \text{ReLU}(W_1 x + b_1) + b_2 \quad (\text{eq1}),$$
 where W_1 is 256×194 and W_2 is 97×256 . All models were trained with AdamW (13) with learning rate 1×10^{-3} and weight decay 2×10^{-2} for up to 200,000 epochs with batch size 512.

These hyperparameters were chosen to reliably produce grokking under clean labels, consistent with prior work. Models were evaluated every 2,000 epochs on both training and test sets. Early stopping was applied when test accuracy exceeded 0.95 for 5 consecutive evaluation points (the grokking criterion). All experiments were performed on Apple MPS (Metal Performance Shaders) using PyTorch float32. Three random seeds — 0, 1, and 2 — were used for all main experiments.

3.4 Noise models

The total corruption count was fixed at $k = \text{floor}(\eta \cdot n_{\text{train}}) = \text{floor}(0.04 \times 5000) = 200$ examples for main experiments. Corrupted labels were replaced by a uniformly random incorrect class (i.e., $y' \neq y$, chosen uniformly from the remaining $p - 1 = 96$ classes).

The two noise types differed only in *which* k examples were selected for corruption:

Uniform noise. The 200 corrupted examples were drawn uniformly at random from all 5,000 training examples. This is the standard noise model used in prior work and served as the control condition.

Input-dependent (structured) noise. All 200 corruptions were concentrated in the subset of training examples where $a < \text{floor}(p/4) = 24$. This region contained approximately $n_{\text{region}} \approx 1,237$ examples (roughly 25% of the training set at uniform draw probabilities). No examples outside this region were corrupted. This simulated a systematic annotation failure mode tied to a specific input condition.

A key property of this design was that the two noise types had identical corruption rates η but

radically different spatial distributions. Any difference in learning dynamics or audit efficiency between them was therefore attributable to noise geometry alone.

3.5 Temporal path-dependence experiments

To test whether grokking recovery depends on optimization history rather than solely on the final dataset state, four experimental conditions were evaluated in addition to a clean baseline. In all of them, the model was trained continuously; conditions differed only in when the labels were clean or noisy and whether training was restarted at the transition epoch $T = 50,000$.

Condition clean. Training was conducted on clean labels throughout. This established the baseline grokking timing.

Condition A. Training was conducted on noisy labels throughout with no correction. This established the baseline suppression effect.

Condition B (clean $\rightarrow$ noisy). The model was trained on clean labels for the first $T = 50,000$ epochs, after which label noise was injected and training continued from the existing state. This tested whether grokking, once achieved, could persist under subsequent label corruption.

Condition C (noisy $\rightarrow$ clean, continue). The model was trained on noisy labels for the first $T = 50,000$ epochs (the memorization regime), after which all corrupted labels were corrected. Training then continued from the existing optimization state — that is, retaining both the network weights and the optimizer statistics accumulated during the noisy phase. This tested whether grokking could recover without restarting optimization.

Condition D (noisy $\rightarrow$ clean, reinitialize). The model was trained on noisy labels for the first $T = 50,000$ epochs, after which all corrupted labels were corrected and the model was fully reinitialized (both network weights and optimizer statistics). Training then restarted from scratch on the corrected dataset. This established the counterfactual: what would happen with the same corrected data but no optimization history from the noisy phase?

The key comparison was Condition C versus Condition D. Both received an *identical* corrected dataset; they differed only in whether the accumulated optimization state from the noisy phase was carried forward (C) or discarded (D). Any difference in grokking outcome between C and D would therefore constitute evidence that optimization history — and not the corrected dataset alone — influences recovery. It is important to note that C and D differ in their entire accumulated state (weights, learned representations, and optimizer statistics together); the design establishes that optimization history matters but does not isolate which component is responsible (see Section 5.5).

All temporal experiments used $\eta = 0.04$ input-dependent and uniform noise, seeds 0–2 (five conditions $\times$ three seeds $\times$ two noise types = 30 runs total).

3.6 Audit budget policies

After noise was applied, an audit selected B training examples for human review, where B denotes the annotation budget (200, 400, or 800 in the experiments). Any examples whose labels were found to be incorrect were

corrected to their true labels before training began. Crucially, the audit policy did not have access to the ground-truth corruption mask — it selected examples using only the noisy training data and (for model-based audits) a briefly-trained model.

Five policies were evaluated in total:

Random audit. B examples were selected uniformly at random from all 5,000 training examples. This is the naive baseline. Under any noise distribution with corruption rate η , the expected hit rate (fraction of audited examples that are corrupted) is exactly $\eta = 4\%$.

Region audit. B examples were selected uniformly at random from the suspected high-error region $\{i : a_i < p/4\}$, filling any remaining budget from outside the region if B exceeded the region size. This policy encoded domain knowledge that the region $a < p/4$ is prone to labeling errors. Under input-dependent noise, where all 200 corruptions lie within this region of $\sim 1,237$ examples, the expected hit rate was $200/1237 \approx 16.2\%$ — approximately four times the random baseline. Under uniform noise, where corruptions were spread across all 5,000 examples, the expected hit rate was also $\approx \eta = 4\%$ — no improvement over random.

Loss-based audit. A fresh model was trained on the noisy training set for 2,000 warm-up epochs using the same architecture and hyperparameters as the main model. Per-example cross-entropy loss on the training set was then computed, and the top- B highest-loss examples were selected for review. This policy required no domain knowledge and was applicable to any noise structure.

Uncertainty audit. A warm-up model was trained as in the loss-based policy. The softmax

entropy of each example's predicted class distribution was computed, and the top- B highest-entropy examples were selected. Unlike raw loss, entropy is bounded and invariant to logit scale, representing a distinct inductive bias for identifying corrupted examples.

Forgetting-event audit. During warm-up training, the correctness of each example's prediction was tracked at regular evaluation intervals. The number of *forgetting events* — instances where an example flipped from correctly to incorrectly predicted across consecutive evaluations — was recorded per example. The top- B examples by forgetting count were selected for review, on the hypothesis that corrupted examples exhibit more volatile learning dynamics.

The hit rate (fraction of audited examples that are genuinely corrupted) and correction rate (fraction of all corruptions that are fixed) were recorded for each run as measures of audit efficiency.

3.7 Noise rate and scorer generalization

To assess whether the audit findings generalized beyond $\eta = 0.04$ and beyond the cross-entropy loss scorer, two additional sweeps were conducted:

Noise rate sweep. The main audit experiment was repeated at $\eta \in \{0.01, 0.08, 0.12\}$ with the same policies (none, random, region, loss-based) and representative budgets $B \in \{400, 800\}$, using seeds 0–2 and both noise types. The existing $\eta = 0.04$ results served as the midpoint.

Alternative scorer sweep. At $\eta = 0.04$, the uncertainty audit and forgetting-event audit

policies (Section 3.6) were evaluated alongside the loss-based audit, using budgets $B \in \{200, 400, 800\}$, seeds 0–2, and both noise types. This assessed whether the advantage of model-based triage over random auditing was specific to cross-entropy loss or generalizable across scoring functions.

4. Results

4.1 Baseline: clean labels and noisy no-audit

Under clean labels, the model reliably grokked at approximately 36,000–48,000 epochs (mean $\approx 40,700$), reaching final test accuracy ≥ 0.97 in all cases. This confirmed that the training pipeline was correctly configured for the grokking regime.

Under $\eta = 0.04$ noise with no audit, neither noise type produced grokking within 200,000 epochs (0/3 seeds for both). Final test accuracy plateaued at 0.419 ± 0.015 for input-dependent noise and 0.430 ± 0.009 for uniform noise — both near the memorization plateau and far from generalization. The similarity of these values confirmed that noise rate, not geometry, primarily determined the degree of grokking suppression.

4.2 Grokking recovery is optimization-history dependent

The central result of this study is that grokking recovery depends on the optimization trajectory, not solely on the final corrected dataset.

Table 1. Temporal experiment results across all five conditions, seeds 0–2, and both noise types (30 runs; 6 runs per condition). "Grokked" = test accuracy ≥ 0.95 for 5 consecutive evaluation points within 200k epochs.

Condition	Description	Grokked	Mean t_{grok}
Clean	Baseline clean	6/6	$\sim 40,700$
A	Noisy throughout	0/6	—
B	Clean $\rightarrow$ noisy at 50k	4/6	$\sim 37,000$ (grokked seeds only)
C	Noisy $\rightarrow$ clean at 50k, continue	6/6	$\sim 78,700$
D	Noisy $\rightarrow$ clean at 50k, reinitialize	6/6	$\sim 91,300$

Condition C vs. Condition D — the hysteresis finding. Both conditions received identical corrected datasets (all 200 corruptions restored to true labels). Despite this, Condition C (continue training from the existing state) grokked consistently 10–16k epochs faster than Condition D (reinitialize and retrain) across all six seed–noise-type combinations.

Table 1A. Grokking comparison between conditions C and D

Seed	Noise	C t_grok	D t_grok	C advantage
0	input_dep	80,000	96,000	16,000
0	uniform	82,000	96,000	14,000
1	input_dep	80,000	90,000	10,000
1	uniform	80,000	90,000	10,000
2	input_dep	74,000	88,000	14,000
2	uniform	76,000	88,000	12,000

This result was consistent across all seeds and both noise types, establishing that *grokking recovery is path-dependent*: the optimization history accumulated during noisy training contributed positively to subsequent generalization once labels were corrected. Discarding that history (Condition D) was uniformly slower, even though the corrected dataset was identical. This constitutes evidence for hysteresis in grokking dynamics — the final dataset state alone is insufficient to predict recovery speed. Figure 1 shows representative learning curves for all five conditions; Figure 2 isolates the per-seed C-versus-D effect; and Figure 3 summarizes mean grokking times across conditions.

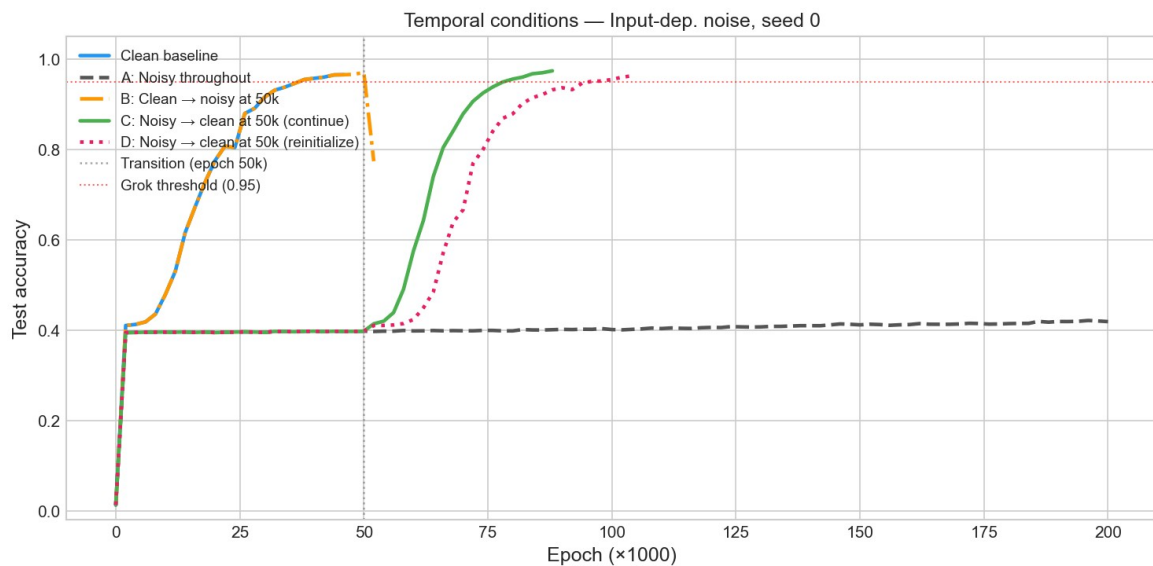


Figure 1. Test accuracy over training for all five temporal conditions under input-dependent noise (seed 0). The vertical dotted line marks the transition epoch (50,000). Clean baseline and Condition B (clean → noisy) grok before the transition. Condition A (noisy throughout) never groks. Conditions C and D both receive identical corrected datasets at epoch 50,000 but diverge thereafter: C (continue, green) crosses the grokking threshold ~16,000 epochs before D (reinitialize, pink dotted), demonstrating that optimization history accelerates recovery.

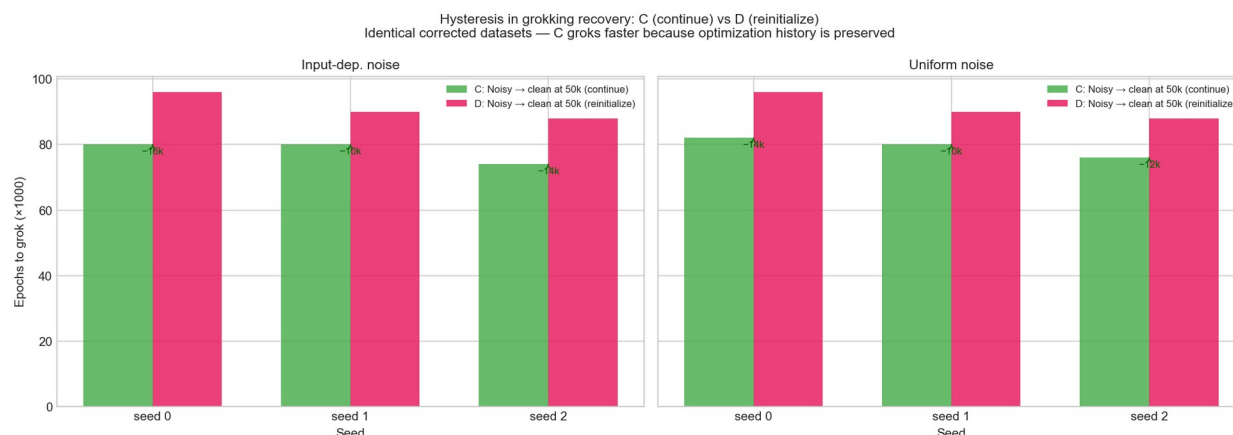


Figure 2. Epochs to grokking for Conditions C (continue) and D (reinitialize) across all three seeds and both noise types. Both conditions received identical corrected datasets at epoch 50,000. Condition C grokked 10–16k epochs faster than Condition D in every seed–noise combination, providing consistent evidence for hysteresis in grokking dynamics. The C advantage (annotated in each bar pair) reflects the contribution of the optimization history accumulated during noisy training.

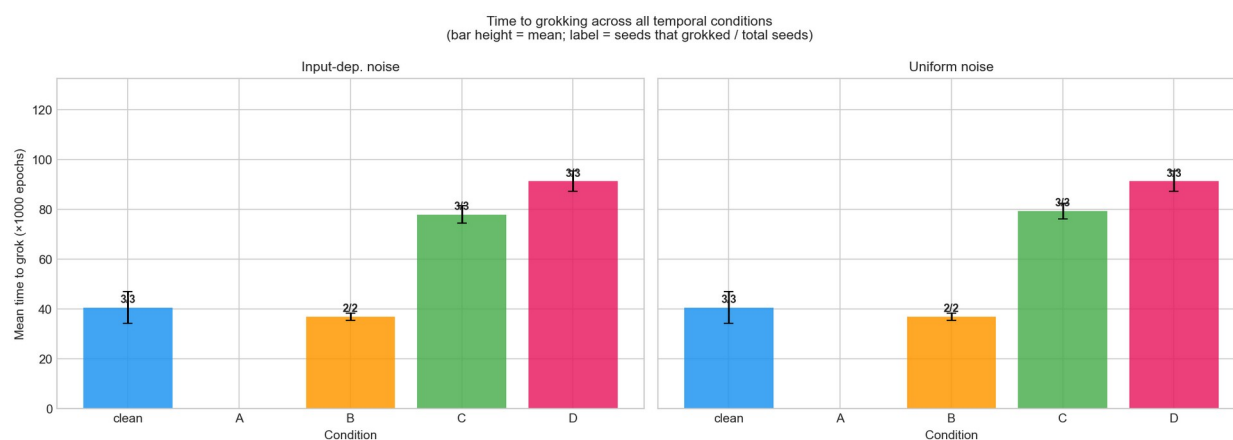


Figure 3. Mean time to grokking across all five temporal conditions for input-dependent (left) and uniform (right) noise. Bar labels show the number of seeds that grokked out of total seeds run. Condition A never groks (bar absent). Condition B groks in 2/3 seeds at timing similar to the clean baseline. Conditions C and D both grok in 3/3 seeds but take roughly twice as long as the clean baseline, with C consistently faster than D.

Condition B (clean → noisy). The temporal experiments also probed the reverse direction. Grokking emerged during the clean first phase in the seed 0 and seed 2 runs (four runs across both noise types; phase-1 test accuracy ≈ 0.97 at epoch 50,000). After noise was injected at epoch 50,000, test accuracy in these four runs declined to 0.76–0.80 by the end of training — a substantial degradation from the grokked state, but well above the ≈ 0.40 chance level of the no-audit baseline, indicating that a consolidated generalizing solution was partially

robust to subsequent corruption. The seed 1 runs (both noise types) did not achieve sustained grokking during the clean phase: phase-1 test accuracy reached 0.968 but did not satisfy the strict criterion of ≥ 0.95 for five consecutive evaluations before the transition.

Following noise injection, these two runs collapsed to chance (≈ 0.43). The contrast suggests that the robustness of grokking to later corruption depends on how fully the generalizing solution had consolidated before noise was introduced (Figure 4).

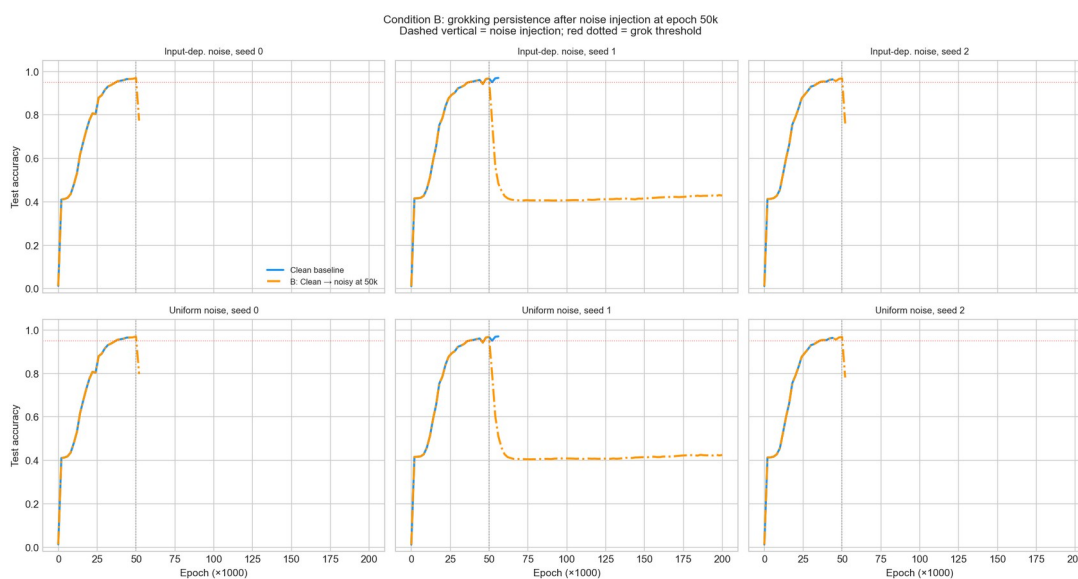


Figure 4. Grokking persistence after noise injection at epoch 50,000 (Condition B), shown for all three seeds and both noise types. Each panel overlays the clean baseline (blue) and Condition B (orange). In seeds 0 and 2 — where the network had fully grokked during the clean phase (test accuracy ≈ 0.97) — test accuracy degraded after noise injection but remained well above chance (≈ 0.76 – 0.80), indicating partial persistence of the generalizing solution. In seed 1 (both noise types), the network had not satisfied the strict grokking criterion before the transition (test accuracy reached 0.968 but not for five consecutive evaluations), and test accuracy collapsed to chance (≈ 0.43) after noise injection. The contrast suggests that the degree of generalization consolidation at the time of noise injection determines whether the solution persists.

Having established that *when* correction occurs shapes recovery, the remaining results turn to the complementary practical question: given a fixed annotation budget, *which* examples should be corrected.

4.3 Audit hit rates: geometry determines efficiency

Table 2 summarizes mean hit rates across seeds for each policy–noise combination, averaged

across budgets. Random auditing achieved approximately $\eta = 4\%$ hit rate in both conditions, exactly as predicted. Region auditing achieved $\approx 16.5\%$ under input-dependent noise — a $4.1\times$ improvement — but was statistically indistinguishable from random auditing under uniform noise (4.1% vs. 4.2%). This directly demonstrated that region audit's advantage is specific to noise geometry: it was

effective only when noise was localized in the targeted region.

Loss-based auditing substantially outperformed both other policies under input-dependent noise. At $B = 200$, it achieved a 45.3% hit rate — correcting ~ 91 of the 200 corrupted examples with a budget of only $200/5000 = 4\%$

of the training set. This represented a $15\times$ improvement over random auditing. Loss-based auditing also achieved high hit rates under uniform noise (41.7% at $B = 200$), where region auditing provided no advantage, demonstrating that the loss signal captured label corruption regardless of its spatial distribution.

Table 2. Mean hit rate (corrupted examples found / budget), averaged across $B \in \{200, 400, 800\}$ and seeds 0–2.

Policy	Input-dep. noise	Uniform noise
Random audit	4.0%	4.2%
Region audit	16.5%	4.1%
Loss-based audit	30.8%	25.9%

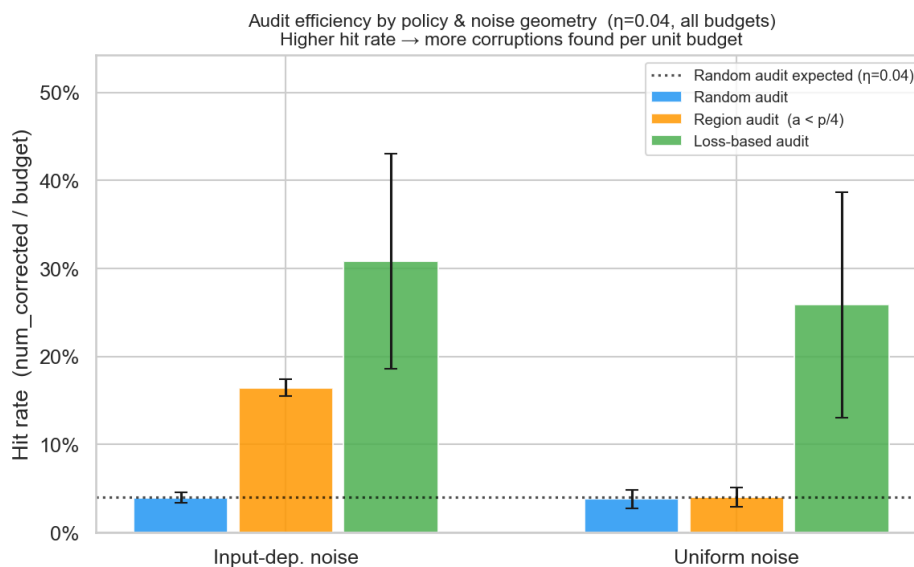


Figure 5. Audit efficiency by policy and noise geometry ($\eta = 0.04$). Bar height shows mean hit rate (fraction of audited examples that were genuinely corrupted) averaged across budgets $B \in \{200, 400, 800\}$ and seeds 0–2. Error bars show standard deviation. The dotted line marks the theoretical expected hit rate for random selection ($\eta = 4\%$). Region auditing achieves a $4\times$ improvement over random under input-dependent noise but provides no advantage under uniform noise. Loss-based auditing substantially outperforms both other policies under either noise type.

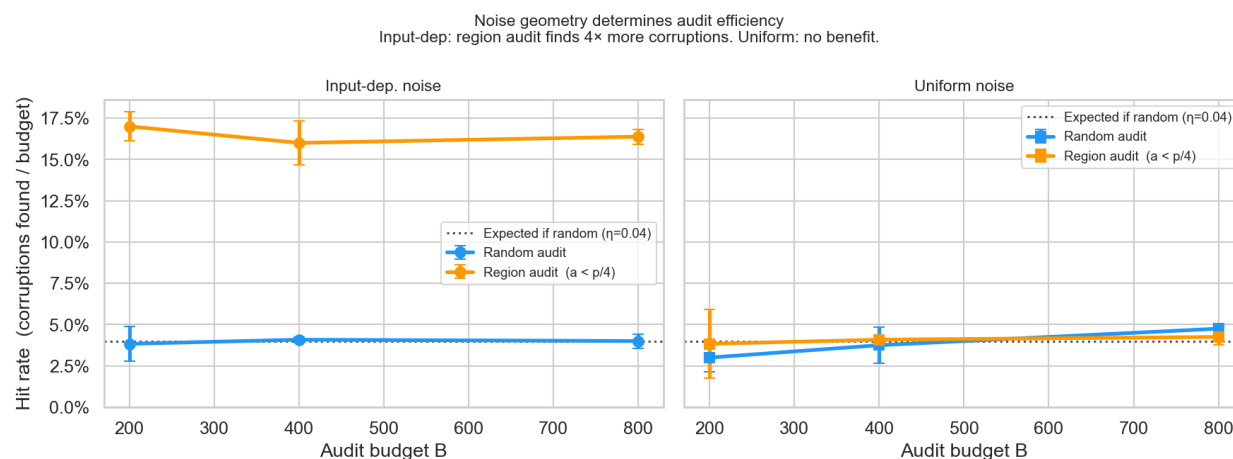


Figure 6. Hit rate of random and region auditing as a function of audit budget B , shown separately for input-dependent (left) and uniform (right) noise. Region auditing consistently achieves ~16–17% hit rate under input-dependent noise regardless of budget, a 4× improvement over the random baseline (dotted line). Under uniform noise, region auditing is indistinguishable from random auditing at all budgets, directly demonstrating that its advantage is entirely attributable to the match between audit strategy and noise geometry.

Hit rates for loss-based auditing decreased with lower corruption density. The overall pattern of budget (from ~45% at $B = 200$ to ~18% at $B = 800$) because the total number of corruptions audit efficiency across policies and noise geometries is shown in Figure 5, and the was fixed at 200: once the highest-loss geometry-specificity of region auditing is corrupted examples had been selected, isolated in Figure 6. additional budget drew from a larger pool with

4.4 Grokking recovery under audit policies

Table 3. Grokking outcomes (test accuracy ≥ 0.95 for 5 consecutive evaluations), 200k epoch budget.

Policy	Budget	Input-dep. grokked	Uniform grokked
None	—	0/3	0/3
Random	200–800	0/3	0/3
Region	200–800	0/3	0/3
Loss	200	1/3	0/3
Loss	400	3/3	0/3
Loss	800	3/3	1/3

Random auditing and region auditing never auditing correcting up to 65% of total restored grokking under either noise type corruptions at $B = 800$ under input-dependent across any budget tested. Despite region noise (vs. only 16% for random), the remaining

uncorrected examples were sufficient to prevent grokking.

Loss-based auditing restored grokking reliably at $B = 400$ under input-dependent noise (3/3 seeds, mean $t_{\text{grok}} \approx 120,000$ epochs) and at $B = 800$ under input-dependent noise (3/3 seeds, mean $t_{\text{grok}} \approx 79,000$ epochs, approximately 34% faster than at $B = 400$). Under uniform

noise, loss-based auditing substantially accelerated learning but did not consistently achieve the strict grokking criterion; all three seeds reached final test accuracy ≥ 0.90 at $B = 400$ and $B = 800$. Representative learning curves for each policy are shown in Figure 7, and final test accuracy as a function of budget is summarized in Figure 8.

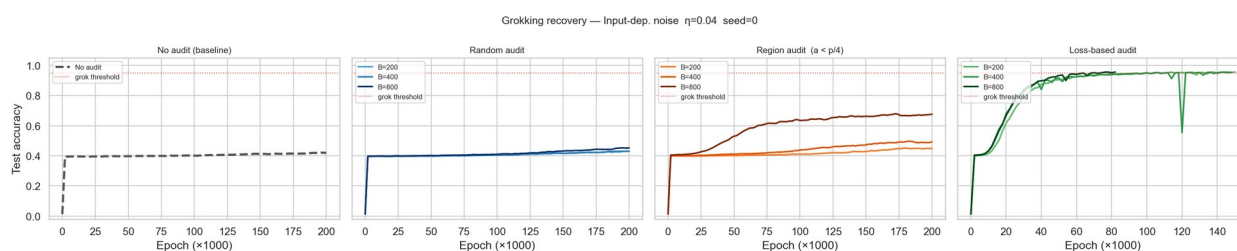


Figure 7. Test accuracy learning curves for each audit policy at all budgets (input-dependent noise, seed 0, $\eta = 0.04$). No-audit and random audit baselines plateau near 40% test accuracy and never grok. Region auditing improves final accuracy modestly at $B = 800$ but does not restore grokking. Only loss-based auditing produces the characteristic sharp grokking transition, crossing the 0.95 threshold (red dotted line) at all three budget levels.

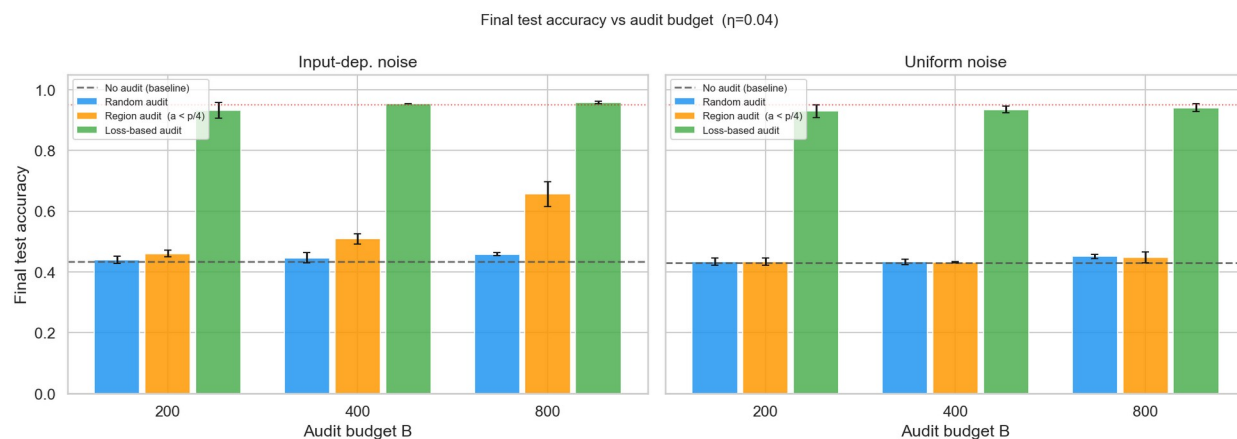


Figure 8. Final test accuracy (at epoch 200,000) as a function of audit budget B , for each policy and both noise types ($\eta = 0.04$, mean $\pm$ SD across seeds 0–2). Dashed horizontal line shows the no-audit baseline ($\sim 42\%$). Under input-dependent noise, loss-based auditing reaches near-perfect accuracy at $B = 400$ and $B = 800$. Under uniform noise, loss-based auditing achieves $\sim 93\%$ final accuracy across all budgets. Random and region auditing remain at or near the no-audit baseline under both noise types.

4.5 The identity of corrected examples matters more than the number corrected

A striking result emerged from comparing region audit at $B = 800$ with loss-based audit at $B = 200$ under input-dependent noise:

Region audit, $B = 800$: corrected 131/200 corruptions (65.5%), did **not** grok, final test ≈ 0.656

Loss audit, $B = 200$: corrected 91/200 corruptions (45.5%), **did** grok (1/3 seeds), $t_{0.80} \approx 33,000$ epochs

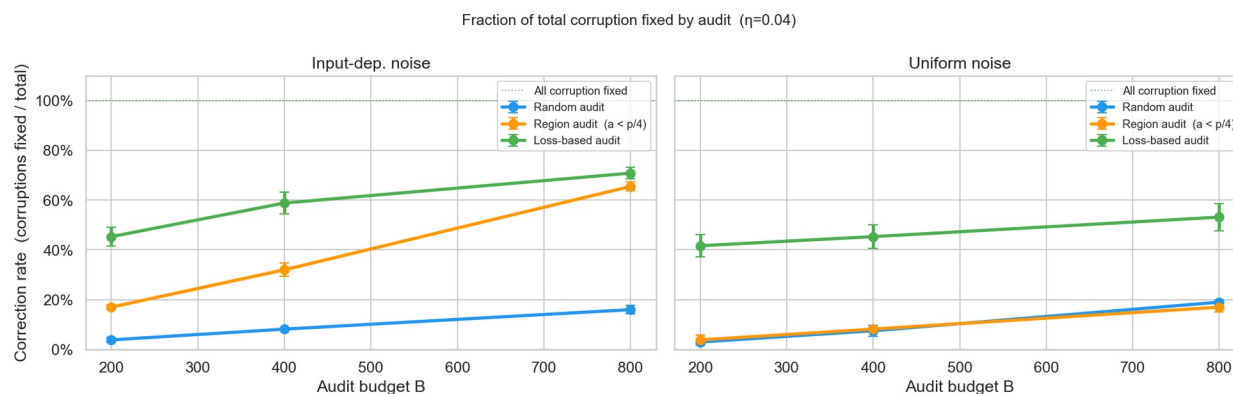


Figure 9. Fraction of total corruptions corrected as a function of audit budget B , for each policy and both noise types ($\eta = 0.04$, mean $\pm$ SD across seeds 0–2). Under input-dependent noise, loss-based auditing corrects $\sim 46\%$ of all 200 corruptions at $B = 200$ and $\sim 71\%$ at $B = 800$. Region auditing reaches $\sim 66\%$ at $B = 800$ but fails to restore grokking despite fixing more corruptions in absolute terms than loss auditing at $B = 200$ — illustrating that correction rate alone does not predict grokking recovery.

Despite fixing *fewer* corruptions in absolute terms, loss-based auditing at $B = 200$ produced dramatically better generalization. This suggested that the loss signal identifies not just corrupted examples, but the *most damaging* corrupted examples — those that most directly interfered with the model's ability to learn the underlying algorithmic structure. Figure 9 contrasts the fraction of total corruptions corrected by each policy, showing that correction count alone does not predict grokking recovery.

4.6 Generalization across noise rates

The noise rate sweep at $\eta \in \{0.01, 0.04, 0.08, 0.12\}$ revealed a recoverability threshold between $\eta = 0.04$ and $\eta = 0.08$, and produced

the most direct evidence for the "identity matters" claim in the entire study.

$\eta = 0.01$ (50 corruptions out of 5,000). At the lowest tested noise rate, loss-based auditing was the only policy that restored grokking across both noise types. Loss audit at $B = 400$ grokked at a mean epoch of 50,700 (input-dependent) and 57,300 (uniform); at $B = 800$, at 49,300 and 52,700, respectively (means over three seeds). Random auditing and region auditing failed to grok under all conditions. The critical comparison was region audit versus loss audit at $B = 800$ under input-dependent noise: the two policies corrected almost identical fractions of the 50 corruptions (correction rate 0.733 for region vs. 0.727 for

loss — region actually corrected marginally *more*), yet only loss audit restored grokking (final test acc = 0.819 for region vs. 0.963 for loss). Because the correction *count* was effectively identical — indeed slightly favoring region — the difference in grokking outcome could only be attributed to the *identity* of the corrected examples. This provides direct experimental evidence that which corruptions are fixed determines grokking recovery, not merely how many.

$\eta = 0.04$ (200 corruptions). Results are as reported in Sections 4.3–4.5. Loss audit at $B \geq 400$ restored grokking under input-dependent noise (3/3 seeds); no other policy restored grokking at any budget.

$\eta = 0.08$ (400 corruptions). No policy met the strict grokking criterion within 200,000 epochs at either noise type. However, loss audit at $B = 800$ achieved a final test accuracy of 0.931 (input-dependent) and 0.895 (uniform) — substantially higher than region audit (0.511 / 0.397) and random audit (0.390 / 0.398) at the same budget. At $B = 800$, loss audit corrected ~74% of corruptions under input-dependent noise (correction rate 0.741), leaving ~100 corruptions uncorrected; this residual corruption was sufficient to keep the model just below the strict ≥ 0.95 criterion despite the high final accuracy. The advantage of loss-based triage over all other policies was thus maintained beyond the recoverability threshold.

$\eta = 0.12$ (600 corruptions). No policy restored grokking. Loss audit at $B = 800$ again achieved the highest final accuracy (0.918 input-dependent; 0.800 uniform), while all other policies remained at or below 0.44 under input-dependent noise (region 0.436, random 0.373). The qualitative ordering of policies — loss $>$ region $\geq$ random — was preserved at every noise rate, with region auditing exceeding random only under input-dependent noise (consistent with the geometry-specific advantage of region auditing established in Section 4.3).

In summary, the qualitative advantage of loss-based auditing over random and region auditing was confirmed at every η tested. The absolute ability to restore grokking was limited to $\eta \leq 0.04$, where the number of corruptions remaining after a targeted audit fell below the suppression threshold; at higher noise rates loss audit still dominated on final test accuracy without meeting the strict grokking criterion.

4.7 Alternative scoring functions

The alternative scorer sweep evaluated uncertainty audit (softmax entropy) and forgetting-event audit alongside the loss-based audit at $\eta = 0.04$, budgets $B \in \{200, 400, 800\}$, seeds 0–2, and both noise types (an additional 72 runs, including random-audit references, beyond the main experiment).

Table 4. Alternative-scorer results under input-dependent noise ($\eta = 0.04$), means over three seeds. "Hit" is the fraction of audited examples that were genuinely corrupted; "Acc" is final test accuracy; "Grok" is the number of seeds (of three) meeting the strict criterion (test accuracy ≥ 0.95 for five consecutive evaluations).

Scorer	Hit (B=200)	Acc (B=200)	Grok (B=200)	Acc (B=400)	Grok (B=400)	Acc (B=800)	Grok (B=800)
Random	0.038	0.439	0/3	0.446	0/3	0.459	0/3
Loss	0.453	0.932	1/3	0.954	3/3	0.958	3/3
Uncertainty	0.452	0.930	1/3	0.947	2/3	0.958	3/3
Forgetting	0.297	0.607	0/3	0.620	0/3	0.639	0/3

The results dissociate the scorers into two groups. Loss-based and uncertainty-based auditing performed almost identically at every budget and under both noise types: their hit rates differed by less than 0.002 (e.g., 0.453 vs. 0.452 at $B = 200$), their correction rates were equal to three decimal places, and their final test accuracies differed by at most ~ 0.01 (with the single exception of uniform noise at $B = 200$, where uncertainty lagged loss, 0.843 vs. 0.929). Both restored grokking at the same budgets — by $B = 800$ each grokked in 3/3 seeds under input-dependent noise and 1/3 under uniform noise. Because softmax entropy is bounded and invariant to logit scale, this near-equivalence demonstrates that the recovery effect is not an artifact of the raw cross-entropy scale but holds for a qualitatively distinct, confidence-based ranking of examples.

The forgetting-event audit, by contrast, was consistently weaker. It identified fewer corrupted examples (hit rate 0.297 vs. ~ 0.45 at $B = 200$) and did not restore grokking at any tested budget under either noise type (maximum final test accuracy 0.639). It nonetheless outperformed random auditing (which reached only ~ 0.46), confirming that

prediction-volatility carries *some* corruption signal — just a markedly less precise one than direct prediction confidence.

This dissociation directly addresses whether different ranking metrics recover similar or distinct subsets of corruptions with different effects on recovery. Loss and entropy — both direct measures of the warm-up model's per-example confidence — recover nearly the same subset of corruptions and produce the same recovery, whereas forgetting-event count, an indirect measure of learning instability, recovers a smaller and less optimization-critical subset and fails to restore grokking. Rather than showing that "any scorer works," the results show that the effect is robust across confidence-based scorers while remaining genuinely tied to identifying the optimization-critical corruptions — strengthening the central claim that the *identity* of the corrected examples is what matters.

4.8 Time to reach 0.80 test accuracy

For conditions that successfully restored learning (loss audit under both noise types), the time to reach test accuracy ≥ 0.80 served as a continuous measure of recovery speed. Under

input-dependent noise, loss audit reduced $t_{0.80}$ from "never reached" (no-audit baseline) to $33,300 \pm 2,100$ epochs at $B = 200$, $29,300 \pm 1,200$ epochs at $B = 400$, and $27,300 \pm 1,100$ epochs at $B = 800$ (Figure 10). The diminishing returns with budget were consistent with the declining marginal value of additional corrections as the most damaging corruptions had already been fixed.

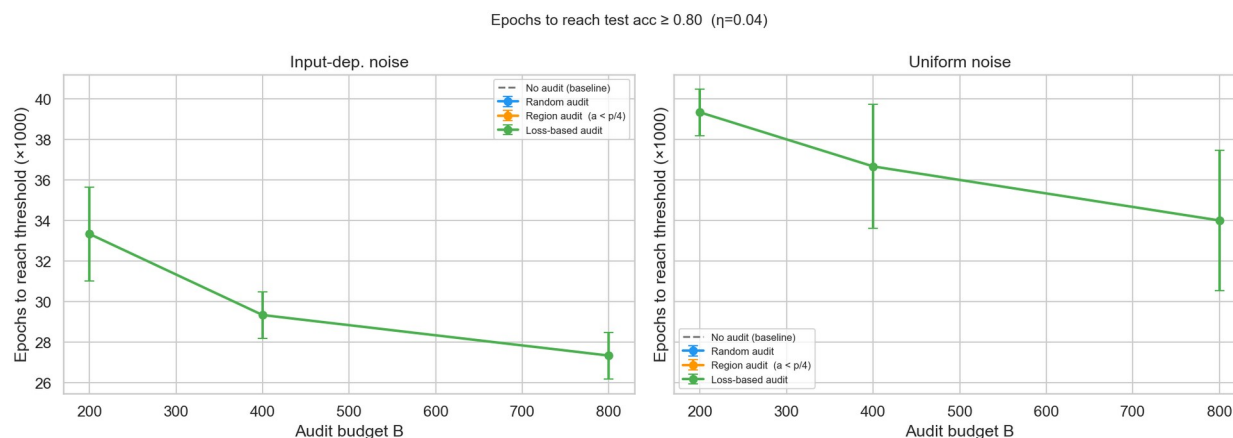


Figure 10. Epochs required to reach test accuracy ≥ 0.80 under loss-based auditing, as a function of budget B , for input-dependent (left) and uniform (right) noise (mean $\pm$ SD across seeds). Only loss-based auditing ever reached this threshold; random and region auditing are omitted because they never did. Time to reach 0.80 decreases with budget, from $\sim 33,000$ epochs at $B = 200$ to $\sim 27,000$ at $B = 800$ under input-dependent noise, reflecting diminishing marginal returns as the most damaging corruptions are fixed first.

5. Discussion

5.1 Optimization history as a resource for grokking recovery

The temporal experiments revealed an unexpected phenomenon: the state accumulated during training on noisy data was not merely a source of interference to be discarded upon label correction. Rather, it functioned as a resource that accelerated subsequent grokking. Condition C, which retained this optimization history, consistently grokked 10–16k epochs faster than Condition D, which discarded it — even though both received the identical corrected dataset. Because the data were the same, recovery speed cannot be explained by

the corrected dataset alone; the trajectory by which the model arrived at that dataset mattered.

One interpretation is that noisy training partially explored the weight space in directions that were subsequently useful for generalization: even though the noisy labels prevented full grokking, the network accumulated parameters, partial representations, and adaptive optimizer statistics that were favorable for the eventual generalization transition. This is consistent with the view of grokking as a structured phase transition in which the path through weight space matters, not just the endpoint. An

alternative interpretation is that the noisy phase served as implicit regularization — training in the presence of noise encouraged the model toward flatter, more generalizable regions of the loss landscape that were then exploited once clean labels were available. Distinguishing between these interpretations, and decomposing the contribution of retained weights versus optimizer statistics (Section 5.5), is a direction for future mechanistic work.

5.2 The geometry of noise determines audit strategy, not audit outcome

The control condition — region auditing under uniform noise — was a key result for interpreting the audit experiments. Region auditing under uniform noise achieved the same 4% hit rate as random auditing and never restored grokking regardless of budget. This established that region auditing's advantage under input-dependent noise was entirely attributable to the match between audit strategy and noise geometry — not to any inherent property of region-based inspection.

This finding has a practical implication: domain-knowledge-based audit strategies are only valuable when the domain knowledge correctly identifies where errors are concentrated. In the absence of reliable prior knowledge about noise geometry, loss-based triage is the safer choice.

5.3 Loss triage as a geometry-agnostic recovery method

Loss-based auditing achieved high hit rates under both noise types (45% for input-dependent, 42% for uniform at $B = 200$). This was consistent with the finding of Arpit et al.

(9) that neural networks learn clean patterns before memorizing noisy ones: early in training, corrupted labels produced higher loss because they were inconsistent with the patterns the network was learning. The warm-up model used for loss triage captured this signal regardless of where in input space the corruptions occurred.

The fact that loss triage worked nearly as well under uniform noise as under structured noise was an important clarification of the paper's claims. The geometry of noise determined the *relative* advantage of region auditing vs. random auditing. Loss-based triage was powerful under any noise structure, because it exploited the loss signal rather than spatial location.

5.4 Implications for data-centric Machine Learning

These findings provided concrete, quantitative guidance for practitioners with limited annotation budgets:

1. Random auditing is almost never worth it at small budgets. Fixing 3–4% of corrupted examples had no measurable effect on grokking dynamics.
2. Loss-based triage is the default recommendation when no reliable domain knowledge about noise geometry is available. At a budget of just 4% of the training set ($B = 200$ out of 5,000), it fixed 45% of all corruptions and restored substantial generalization.
3. Region auditing is valuable only when noise geometry is known and confirmed. It offered a 4× improvement in hit rate over random under

the matching noise type, but zero improvement under the non-matching type.

4. The identity of fixed examples matters more than their count. This suggested that hit rate (not correction rate) should be the primary metric for evaluating audit policies.

5. Mid-training label correction is preferable to retraining from scratch when grokking recovery is the goal. The optimization history accumulated during noisy training was a resource, not a liability.

5.5 Limitations

Several limitations bound the generalizability of these findings. First, modular addition is a highly structured algorithmic task; the grokking phenomenon and the behavior of structured noise may differ substantially in natural language or vision domains. Second, the input-dependent noise model used here is a maximally structured extreme case (all corruptions in one region); real-world structured noise is likely more diffuse. Third, all temporal experiments used a fixed transition epoch of $T = 50,000$; the hysteresis effect may vary in magnitude depending on when correction occurs relative to the grokking transition. Fourth, the oracle correction assumption (Section 3.1) may not hold in practice, where re-annotation introduces new errors. Fifth, and most important for the interpretation of the central finding, the temporal design establishes that optimization history matters but does not isolate the responsible component. Conditions C and D differ not only in retained optimizer statistics but also in network weights, learned representations, and feature geometry at the time of correction. The experiments therefore

support the conclusion that optimization history influences recovery, but they do not distinguish whether the observed hysteresis arises from retained optimizer statistics, retained learned representations, or both. The interpretation throughout this manuscript is accordingly framed in terms of optimization-history dependence rather than optimizer-state dependence. A rigorous decomposition would require additional controls — for example, retaining the network weights while resetting only the optimizer statistics, versus retaining both weights and optimizer statistics, versus fully reinitializing both — which is left to future work. Sixth, the scorer comparison (Section 4.7) evaluated two alternatives to cross-entropy loss — softmax entropy and forgetting-event count — but did not test the gradient-norm, influence-estimate (14), or representation-space-outlier scorers; these are more computationally expensive per example and are left to future work, although the close agreement between the loss and entropy scorers suggests the effect is governed by per-example prediction confidence rather than any single metric.

5.6 Perspectives

Natural extensions of this work include: [1] decomposing and characterizing the mechanistic basis of the optimization-history advantage in Condition C using the weight-versus-optimizer controls described above, weight trajectory analysis, and representation geometry; [2] studying how noise geometry interacts with the *type* of corruption (e.g., systematic nearest-neighbor flips vs. random); [3] applying these audit policies and temporal conditions to transformer models on more

complex algorithmic tasks; and [4] combining loss triage with active learning to iteratively improve audit efficiency across multiple rounds.

6. Conclusion

This study investigated how structured label noise affects grokking dynamics and how limited annotation resources can be allocated to restore generalization. Three principal findings emerged.

First, noise geometry strongly influenced audit efficiency but had comparatively little effect on grokking suppression itself. At matched corruption rates, both uniform and input-dependent noise prevented grokking, indicating that corruption rate was the dominant determinant of suppression in this modular-addition setting. In contrast, audit performance depended strongly on the spatial distribution of errors. Region-targeted auditing provided a substantial advantage only when corruption was concentrated within the targeted region, demonstrating that audit strategies must be aligned with the underlying structure of annotation errors.

Second, the identity of corrected examples was found to be more important than the total number corrected. Loss-based auditing consistently outperformed random and region-based auditing across all noise rates tested and was the only policy that reliably restored grokking at moderate corruption levels. Importantly, conditions were observed in which two audit policies corrected nearly identical numbers of corrupted labels yet produced dramatically different generalization

outcomes. These results indicate that certain corrupted examples exert disproportionate influence on the emergence of algorithmic generalization, and that successful recovery depends on identifying these optimization-critical errors rather than maximizing correction count alone.

Third, and most significantly, the results demonstrate that grokking recovery is path-dependent. Temporal experiments showed that models trained initially on corrupted data and later corrected recovered substantially faster when training continued from the existing optimization state than when training was restarted from scratch on the identical corrected dataset. Across all seeds and both noise types, retaining optimization history reduced the time to grokking by approximately 10,000–16,000 epochs. Because the final corrected dataset was identical in both cases, recovery speed could not be explained solely by the data available after correction. Instead, the results indicate that information accumulated during earlier stages of optimization continues to influence subsequent generalization dynamics. This hysteresis effect suggests that the trajectory by which a model arrives at a dataset may be as important as the dataset itself.

Taken together, these findings extend the study of grokking beyond the question of whether noisy labels suppress generalization. The results show that recovery depends not only on the amount and structure of label noise, but also on which errors are corrected and when correction occurs during training. From a practical perspective, targeted auditing can dramatically improve the efficiency of label

correction efforts, while mid-training history accelerates recovery may provide new insight into how neural networks transition from memorization to generalization and may ultimately contribute to a more complete theory of grokking.

correction may be preferable to discarding accumulated optimization progress and retraining from scratch. More broadly, the observed hysteresis effect introduces optimization history as a potentially important variable in grokking dynamics and suggests a new direction for future mechanistic investigations of delayed generalization.

Future work should determine whether these phenomena extend beyond modular arithmetic to larger neural architectures and real-world datasets, examine additional forms of structured noise, and investigate the mechanistic origins of the observed path dependence. Understanding why optimization

This research was conducted independently, without external funding or advising. All experiments were conducted on a personal MacBook Pro M4 Max using PyTorch with Apple MPS acceleration. Portions of this work, including experimental code development and manuscript preparation, were assisted by Claude (Anthropic), an AI language model. All experimental designs, interpretations, and conclusions are the author's own.

Acknowledgments

7. References

1. Power, A., Burda, Y., Edwards, H., Babuschkin, I., & Misra, V. (2022). Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv*. <https://doi.org/10.48550/arXiv.2201.02177>
2. Nanda, N., Chan, L., Lieberum, T., Smith, J., & Steinhardt, J. (2023). Progress measures for grokking via mechanistic interpretability. *International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.2301.05217>
3. Singavarapu, H. (2025). Investigating the impact of low to moderate level label noise on neural network grokking dynamics. *Journal of High School Science*, 9(4), 346–372. <https://doi.org/10.64336/001c.154269>
4. Barak, B., Edelman, B., Goel, S., Kakade, S., Malach, E., & Zhang, C. (2022). Hidden progress in deep learning: SGD learns parities near the computational limit. *Advances in Neural Information Processing Systems (NeurIPS)*, 35. <https://doi.org/10.48550/arXiv.2207.08799>
5. Clauw, K., Stramaglia, S., & Marinazzo, D. (2024). Information-theoretic progress measures reveal grokking is an emergent phase transition. *ICML 2024 Workshop on Mechanistic Interpretability*. <https://doi.org/10.48550/arXiv.2408.08944>

6. Tang, Y., Wang, Q., García-Redondo, I., & Monod, A. (2026). Topological signatures of grokking. *arXiv*. <https://doi.org/10.48550/arXiv.2605.06352>
7. Natarajan, N., Dhillon, I. S., Ravikumar, P. K., & Tewari, A. (2013). Learning with noisy labels. *Advances in Neural Information Processing Systems (NeurIPS)*, 26, 1196–1204. <https://proceedings.neurips.cc/paper/2013/hash/3871bd64012152bfb53fdf04b401193f-Abstract.html>
8. Frénay, B., & Verleysen, M. (2014). Classification in the presence of label noise: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5), 845–869. <https://doi.org/10.1109/TNNLS.2013.2292894>
9. Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., et al. (2017). A closer look at memorization in deep networks. *International Conference on Machine Learning (ICML)*. <https://doi.org/10.48550/arXiv.1706.05394>
10. Northcutt, C. G., Athalye, A., & Mueller, J. (2021). Pervasive label errors in test sets destabilize machine learning benchmarks. *NeurIPS 2021 Track on Datasets and Benchmarks*. <https://doi.org/10.48550/arXiv.2103.14749>
11. Northcutt, C. G., Jiang, L., & Chuang, I. L. (2021). Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70, 1373–1411. <https://doi.org/10.1613/jair.1.12125>
12. Zha, D., Bhat, Z. P., Lai, K. H., Yang, F., Jiang, Z., Zhong, S., et al. (2023). Data-centric artificial intelligence: A survey. *arXiv*. <https://doi.org/10.48550/arXiv.2303.10158>
13. Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. *International Conference on Learning Representations (ICLR)*. <https://doi.org/10.48550/arXiv.1711.05101>
14. Koh, P. W., & Liang, P. (2017). Understanding black-box predictions via influence functions. *International Conference on Machine Learning (ICML)*. <https://doi.org/10.48550/arXiv.1703.04730>

Appendix A: Hyperparameter Summary

Parameter	Value
Modulus p	97
Training set size	5,000
Test set size	9,409 (exhaustive)
Input dimension	194 (one-hot)
Hidden dimension	256
Optimizer	AdamW
Learning rate	1×10^{-3}
Weight decay	2×10^{-2}
Batch size	512
Max epochs	200,000
Eval frequency	2,000 epochs
Grok threshold	0.95 (5 consecutive evals)
Noise rate η	0.04 (main); 0.01, 0.08, 0.12 (sweep)
Corrupted examples	200 ($\eta = 0.04$)
Loss warmup epochs	2,000
Random seeds	0, 1, 2
Temporal transition epoch	50,000

Appendix B: Reproducibility

All code and experimental results are available at

<https://github.com/gnarlylace51842/noise-geometry-determines-audit-efficiency>

Main experiments can be reproduced with:

Temporal path-dependence experiments

```
python3 run_temporal_experiment.py \
  --conditions clean A B C D \
  --transition-epoch 50000 --eta 0.04 \
  --seeds 0 1 2 --noise-types input_dep uniform
```

Main audit experiment ($\eta = 0.04$)

```
python3 run_audit_experiment.py \
  --seeds 0 1 2 --etas 0.04 \
  --noise-types input_dep uniform \
  --policies none random_audit region_audit loss_audit \
  --budgets 0 200 400 800
```

Noise rate sweep

```
python3 run_audit_experiment.py \
```

```
--seeds 0 1 2 --etas 0.01 0.08 0.12 \  
--noise-types input_dep uniform \  
--policies none random_audit region_audit loss_audit \  
--budgets 0 400 800
```

Alternative scorers

```
python3 run_audit_experiment.py \  
--seeds 0 1 2 --etas 0.04 \  
--noise-types input_dep uniform \  
--policies none random_audit loss_audit uncertainty_audit forgetting_audit \  
--budgets 0 200 400 800
```

All random seeds were fixed via `torch.manual_seed`, `numpy.random.seed`, and `random.seed` at the start of each run. Checkpointing enables exact resumption of interrupted runs.