Slower Generalization, Faster Memorization: A Sweet Spot in Algorithmic Learning

Shin So **Kyelim Lee** **Albert No***

Department of Artificial Intelligence, Yonsei University

Abstract

Once generalization becomes possible, does adding more data always make learning faster? In Needleman–Wunsch matrix generation, we find that small Transformers reach high exact-match validation accuracy in the fewest optimizer updates at an intermediate dataset size, while larger datasets converge more slowly. A random-suffix probe shows that, at larger dataset sizes, models fit the structured matrices before memorizing their random suffixes, suggesting that shared rule learning aids training-set fitting. To examine what makes rules easier to learn, we study elementary cellular automata (ECA), in which each binary cell updates from its own state and those of its two neighbors. These experiments identify lower-order predictive signals–output biases or predictive relationships involving only a few input bits–as a key factor in learnability. In Rule 90, finite-sample fluctuations create output biases and single-input dependencies absent from the uniform population. These signals weaken as the dataset grows. Biased training that introduces these signals accelerates generalization on an unchanged uniform validation distribution. These results support an explanation for faster convergence at intermediate dataset sizes: sampling fluctuations can facilitate rule acquisition, and larger datasets weaken this advantage.

1 Introduction

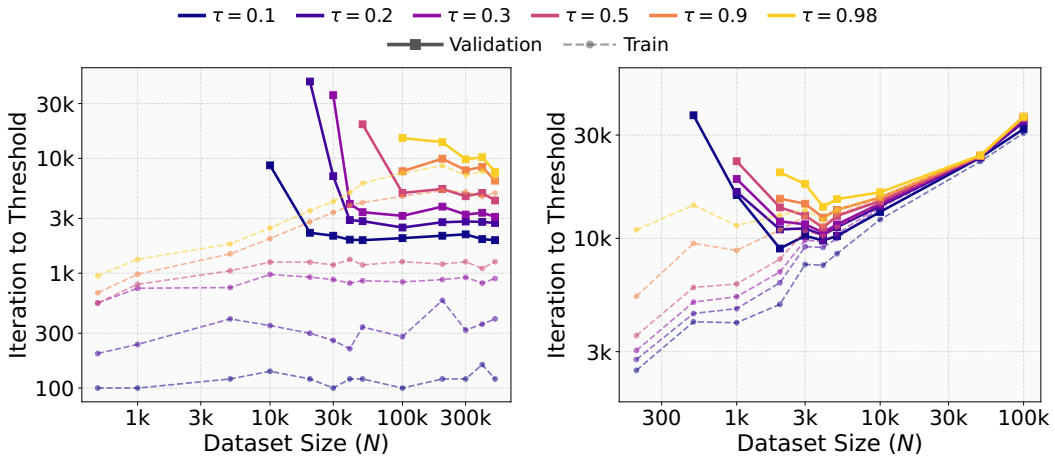
Once a model has enough examples to generalize, can more examples make learning slower? Empirical scaling laws establish the value of additional data for predictive performance when model size and compute are scaled appropriately [Hestness et al., 2017, Kaplan et al., 2020, Hoffmann et al., 2022]. In algorithmic learning, studies of grokking and critical dataset sizes further show that more data can enable generalization and shorten the delay before it emerges [Power et al., 2022, Liu et al., 2022, Zhu et al., 2024]. These findings make continued acceleration a natural expectation. We examine how far that expectation extends, distinguishing the amount of data that makes generalization possible from the amount that makes it fastest.

We study this question in Needleman–Wunsch (NW) matrix generation, a task derived from sequence alignment [Needleman and Wunsch, 1970]. Given two input strings, a small decoder-only Transformer learns to generate the score matrix defined by the NW dynamic program. A prediction counts as correct only when every matrix entry matches the target. We find that validation accuracy reaches a given threshold fastest at an intermediate dataset size (Figure 1). Beyond the optimal dataset size, larger datasets require more optimizer updates to reach the same threshold. We call this increase in convergence time among dataset sizes that already permit generalization the large-data slowdown.

The contrast with multiplication makes this finding particularly striking. Arithmetic tasks are widely used to study algorithmic learning and grokking [Power et al., 2022, Liu et al., 2022, Lee et al., 2023], making three-digit multiplication a natural comparison. In our baseline, which predicts the

*Correspondence to: albertno@yonsei.ac.kr

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Three-digit multiplication.

(b) NW matrix generation, $L = 5$, $D = 6$.

Figure 1: Dataset-size sweeps for multiplication and NW matrix generation. Each curve reports the mean recorded number of optimizer updates needed to reach exact-match threshold τ; lower values mean faster convergence. Colors denote thresholds τ, solid lines show validation crossings $I_V(\tau; N)$, and dashed lines show training crossings $I_T(\tau; N)$. Multiplication shows no pronounced post-threshold slowdown. NW reaches validation thresholds fastest at intermediate sizes and more slowly at the largest dataset size in the sweep.

product directly, validation convergence becomes faster and then roughly plateaus as the dataset grows, with no comparable slowdown over the tested range (Figure 1). Thus, additional examples of a deterministic rule can have qualitatively different effects on learning speed across tasks.

The slowdown extends the grokking picture of how training-set fitting and generalization unfold. Increasing dataset size can shorten the delay between fitting and generalization [Power et al., 2022, Liu et al., 2022]; NW exhibits a further regime in which this delay has narrowed, yet additional data slows convergence on both sets (Section 3). Near the onset of weak generalization, we also find that adding examples can accelerate training-set fitting, suggesting that partial rule learning already helps explain training predictions. To examine this interaction, we append a fixed, randomly assigned four-bit suffix to each NW target as a reference for example-specific memorization (Section 4). As generalization emerges, matrix training accuracy advances ahead of suffix accuracy, and their early gap approximately tracks validation accuracy. These observations support a contribution from partial rule learning to training-set fitting. Meanwhile, memorizing the suffix becomes slower as the dataset grows, revealing an increasing cost of fitting example-specific associations.

To investigate why the slowdown depends on the task, we study elementary cellular automata (ECA), in which each binary cell evolves according to a local rule based on its own state and those of its two neighbors (Section 5). We identify lower-order predictive signals—output biases or predictive relationships involving only a few input bits—as a key factor in learnability. For Rule 90, output biases and single-input predictive relationships are absent under the uniform population distribution but arise from finite-sample deviations in the training data. These signals weaken as the dataset grows. Biased training that introduces these signals accelerates generalization on an unchanged uniform validation distribution, while matched datasets balanced to eliminate first-transition single-input dependencies take longer to fit. These results support a mechanism by which sampling fluctuations can facilitate rule acquisition at intermediate dataset sizes. Together, the NW sweeps, random-suffix probe, and ECA experiments suggest complementary roles for the cost of example-specific fitting and the weakening of empirical predictive signals in the large-data slowdown.

2 Related Work

Grokking and train–validation timing. Grokking is delayed generalization, in which training accuracy becomes high long before validation accuracy rises on small algorithmic datasets [Power et al., 2022]. Subsequent work connects the delay to structured representations [Liu et al., 2022], circuit formation and efficiency [Nanda et al., 2023, Varma et al., 2023], implicit-bias transitions [Lyu

et al., 2023], and internal subnetworks [Minegishi et al., 2025]. Xu et al. [2026] derive quantitative bounds on grokking time in ridge regression as a function of training hyperparameters. Dataset-size sweeps in earlier work also show that more data can shorten the delay before generalization [Power et al., 2022, Liu et al., 2022]. Our work shares the focus on training–validation timing but asks whether faster generalization continues as the dataset grows.

Dataset size, scaling, and critical thresholds. Empirical scaling laws study how loss changes with data, parameters, and compute [Hestness et al., 2017, Kaplan et al., 2020, Hoffmann et al., 2022], while double descent shows that more data or training can sometimes hurt in certain effective-complexity regimes [Nakkiran et al., 2021]. Critical-data-size work asks when generalization first becomes possible and often finds that additional data beyond this onset accelerates validation convergence [Power et al., 2022, Liu et al., 2022, Zhu et al., 2024]. Our question concerns the fastest convergence among dataset sizes that already permit generalization. Larger datasets can also be less effective under a fixed training budget in other algorithmic settings. Charton and Kempe [2024] show benefits from repeating a smaller set of examples, Wen et al. [2025] report parity settings in which smaller datasets generalize while a larger dataset fails to learn within the budget, and Liu et al. [2026] connect sampling biases to relative layer growth and report parity speedups that persist after removing input biases and single-input label dependencies. Our NW fitting diagnostics and ECA signal interventions examine a complementary, task-dependent aspect of this phenomenon.

Memorization and rule-mediated fitting. Neural networks can memorize random labels [Zhang et al., 2017], yet often learn simple shared patterns before fitting noise [Arpit et al., 2017]. Memorization can also be useful for high-accuracy learning when simple shared patterns are insufficient [Feldman and Zhang, 2020]; in language models, memorization has been studied through unintended memorization, extraction, and capacity estimates [Carlini et al., 2019, 2021, Morris et al., 2025]. Mechanistic accounts of grokking suggest that memorizing and generalizing solutions can differ in efficiency [Nanda et al., 2023, Varma et al., 2023, Liu et al., 2023, Huang et al., 2024]. NW matrix generation lets us observe this interaction through dataset-size sweeps.

Algorithmic and structured-output tasks. Controlled algorithmic tasks are useful because exact generalization and memorization can be measured cleanly. Prior work has used modular arithmetic, parity, arithmetic, and synthetic symbolic tasks to study delayed generalization and hidden progress [Power et al., 2022, Barak et al., 2022, Lee et al., 2023, Charton, 2024]. NW matrix generation differs from scalar arithmetic tasks because a short input pair induces a full dynamic-programming table whose cells are mutually constrained. This places NW among structured dynamic-programming problems such as edit distance, local sequence alignment, Viterbi decoding, and context-free parsing [Wagner and Fischer, 1974, Smith and Waterman, 1981, Viterbi, 1967, Younger, 1967]. We use NW as the primary evidence task and multiplication as a contrast baseline. ECA provides a controlled comparison of task learnability at fixed output length and recurrence structure.

3 Dataset Size and the Speed of Algorithmic Learning

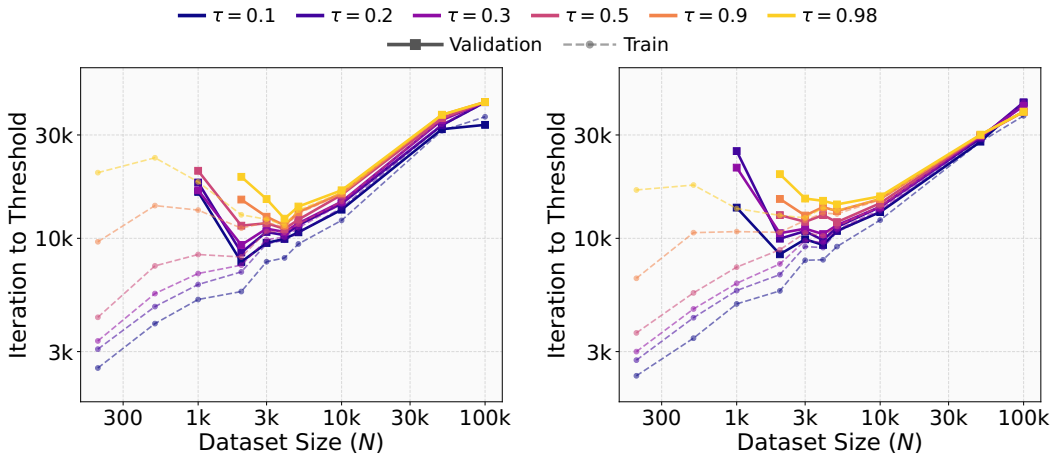
We examine how training dataset size affects the number of optimizer steps needed to reach a given validation accuracy. For each dataset size N, we record the first evaluated checkpoint at which validation exact-match accuracy reaches a specified threshold, keeping the model and training protocol fixed within each sweep. We also track the steps needed to reach training-accuracy thresholds to compare the timing of training-set fitting and generalization. Figure 1 summarizes these measurements for NW and multiplication, with lower values indicating faster convergence.

3.1 Task Setup: NW Matrix Generation and Multiplication

In Needleman–Wunsch (NW) matrix generation, a model predicts a sequence-alignment score matrix from two input strings [Needleman and Wunsch, 1970]. Let $\Sigma = \{A, C, G, T\}$ and let $x, y \in \Sigma^L$. Each input pair has a target matrix $M(x, y) \in \mathbb{Z}^{(L+1)\times(L+1)}$. Boundary entries use a fixed gap penalty, and interior entries obey

$$M_{i,j} = \max\{M_{i-1,j-1} + s(x_i, y_j),\ M_{i-1,j} + g,\ M_{i,j-1} + g\},$$

where s is the match/mismatch score and g is the gap penalty. Unless otherwise stated, we use match score 5, mismatch score -4, and gap penalty -5. The main experiments use $L = 5$, so each input has



(a) Depth 4.

(b) Depth 5.

Figure 2: NW multi-threshold sweeps across depths $D \in \{4, 5\}$ at $L = 5$. Colors denote exact-match thresholds, solid lines show validation crossings, and dashed lines show training crossings. Across depths, validation crossing time is non-monotonic in dataset size, giving an intermediate-data sweet spot. In the same dataset-size band where weak validation thresholds first become reachable, training crossings can also become faster as N increases, supporting the partial-rule interpretation.

ten symbols and the target is a 6×6 matrix. The model predicts the serialized matrix autoregressively, and exact-match accuracy counts an example as correct only if every matrix entry is correct. We use three-digit multiplication as a comparison task. Each input is a pair of three-digit integers and the target is the exact decimal product.

Both tasks use decoder-only Transformers with D layers, D attention heads, embedding width $64D$, and feed-forward width $256D$. Figure 1 uses $D = 6$, and Figure 2 checks depths four and five. AdamW and the training schedule are fixed within each sweep. Additional details are given in Appendix A, with model and training settings in Appendix B.

3.2 Crossing Times, Critical Size, and Optimal Size

Let $\mathcal{D}_N$ denote a training set of N examples, and let $A_T(t; N)$ and $A_V(t; N)$ denote the measured training and validation exact-match accuracies after t optimizer updates. Each run has a budget of $T_{\text{budget}} = 50{,}000$ updates. For an individual run, let $\mathcal{T}_{\text{eval}}$ denote the evaluated update counts within this budget. For threshold $\tau \in (0, 1)$, the crossing times are the first evaluated checkpoints at which accuracy reaches at least τ:

$$I_T(\tau; N) = \min\{t \in \mathcal{T}_{\text{eval}} : A_T(t; N) \geq \tau\},$$
$$I_V(\tau; N) = \min\{t \in \mathcal{T}_{\text{eval}} : A_V(t; N) \geq \tau\}.$$

A run is censored when the threshold is not reached within the budget, and no finite crossing time is assigned. A crossing records first attainment, without requiring the accuracy to remain above the threshold at subsequent checkpoints. We use $\tau = 0.98$ as the strict threshold and lower thresholds to detect partial competence. Appendix C describes the treatment of non-crossing runs.

For the NW dataset-size summaries, let $\overline{I}_V(\tau; N)$ denote the mean validation crossing time over runs that reach the threshold within the budget, and let $\mathcal{N}_\tau$ contain the tested sizes with at least one recorded validation crossing. The *critical dataset size* is the smallest such tested size,

$$N_c(\tau) = \min \mathcal{N}_\tau.$$

An *optimal dataset size* minimizes the mean recorded validation crossing time,

$$N^\star(\tau) \in \arg\min_{N \in \mathcal{N}_\tau} \overline{I}_V(\tau; N).$$

Non-crossing runs are excluded from the mean; a dataset size is excluded only if it has no recorded crossing. These empirical summaries are conditional on threshold attainment and depend on the threshold, training budget, and tested sizes. A *dataset-size sweet spot* exists when an optimal size lies strictly between the smallest and largest tested sizes.

3.3 Multiplication versus NW

Figure 1 compares training and validation crossing times across dataset sizes and accuracy thresholds. In multiplication, validation crossings generally become earlier as more examples are added, then remain stable across larger datasets. Additional data therefore initially accelerates generalization, after which the gains largely saturate. At lower accuracy thresholds, training crossings remain substantially earlier than the corresponding validation crossings. This separation narrows at higher thresholds, but the validation curves show no pronounced upward turn at the large-data end of the sweep.

NW exhibits a different response. As dataset size increases from the smaller-data range, training and validation crossing times move closer together. At larger sizes, the two families of curves become closely aligned and both turn upward: reaching the same accuracy on either split requires more optimizer updates. Thus, training and validation thresholds can be reached close together in time while both take longer to attain as the dataset grows. Unlike the approximate plateau observed in multiplication, NW validation crossing times attain their minima at intermediate dataset sizes.

3.4 Two Findings: Faster Fitting and Slower Generalization

The NW curves motivate two complementary questions about the effects of additional data. Near the onset of weak generalization, can adding examples make the training set faster to fit? Among larger datasets that already permit generalization, does more data continue to reduce validation crossing time? These questions concern different dataset-size ranges. We examine the strict training threshold alongside lower validation thresholds to distinguish nearly complete training-set fitting from emerging validation competence.

Finding 1: more data can accelerate training-set fitting. In the dataset-size range where lower validation thresholds first become reachable, adding examples can reduce $I_T(0.98; N)$, allowing the model to reach nearly perfect training accuracy in fewer updates despite the larger training set. Figure 2 shows this behavior across model depths. Here, faster memorization refers to earlier completion of training-set fitting, not necessarily faster memorization of individual examples. The coincidence with emerging validation competence suggests that partial rule learning helps fit the training set. Even an incomplete rule can explain predictions across multiple training examples, reducing what remains to be memorized individually.

Finding 2: generalization is fastest at an intermediate dataset size. The NW validation curves exhibit a dataset-size sweet spot: the optimal dataset size $N^\star(\tau)$ lies strictly between the smallest and largest tested sizes. Beyond $N^\star(\tau)$, larger datasets with recorded validation crossings require more optimizer updates on average to reach the same threshold, i.e., more data slows generalization.

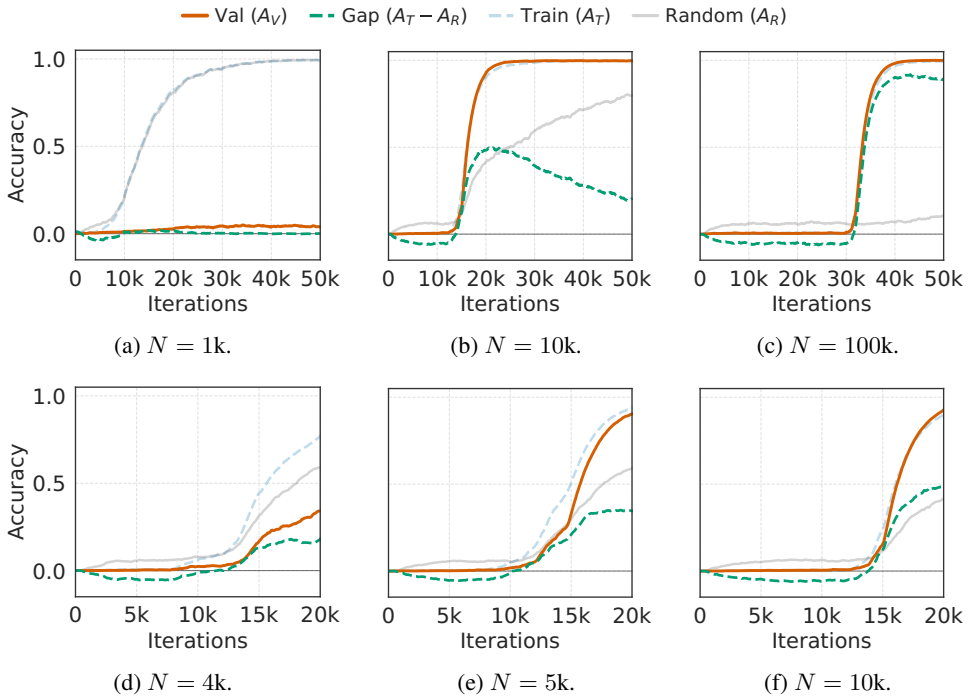
To better understand these findings, Section 4 uses a random-suffix probe to examine how shared rule learning and example-specific fitting contribute to training progress. Section 5 then investigates the large-data slowdown through the lens of task learnability.

4 Rule Learning and the Cost of Example-Specific Fitting

Partial rule learning can accelerate fitting, yet larger NW datasets still generalize more slowly beyond the optimal dataset size. If a shared rule makes training examples easier to fit, why does learning from more examples of the same rule require so many additional updates? The benefit of rule learning may be offset by a growing burden of example-specific fitting. We examine that burden with a random-suffix probe, attaching a fixed, randomly assigned suffix to each NW target.

4.1 Shared Rules Make Training Examples Easier to Fit

By *residual exact fitting*, we mean fitting training predictions that the model's currently learned rule does not yet explain. Those predictions may be resolved by learning more of the shared rule or through example-specific fitting, meaning memorization of individual input–target associations. A random suffix provides a reference for the latter because its assignments cannot be predicted from the NW rule, rather than a calibrated measure of the residual fitting cost of the original NW matrix.



(a) $N = 1$k. (b) $N = 10$k. (c) $N = 100$k.

(d) $N = 4$k. (e) $N = 5$k. (f) $N = 10$k.

Figure 3: Random-suffix probe and train–random gap. Each NW target includes a four-bit random suffix. Faint curves show training accuracy on the NW matrix A_T and random suffix A_R. Dark curves show NW validation accuracy A_V and $G = A_T - A_R$. **Top (a–c)**: full trajectories. As dataset size grows, the structured matrix is fit before the shorter random suffix. **Bottom (d–f)**: enlarged early-training views. Validation initially rises alongside the gap, suggesting that rule-mediated progress contributes to both training and validation accuracy. The approximate agreement is local to acquisition onset, not the entire trajectory.

We append a four-bit random binary suffix to each NW target. Each suffix is sampled once and held fixed throughout training. The matrix follows the shared NW rule, whereas suffix assignments are independent across examples and must be memorized for the training inputs. The NW rule cannot predict a new example's suffix. The short suffix provides an unstructured comparison with the longer rule-generated matrix. Let $A_V(t)$ be validation accuracy on the NW matrix, $A_T(t)$ training accuracy on the NW matrix, and $A_R(t)$ training accuracy on the random suffix.

Figure 3(a–c) compares these component accuracies. At small N, matrix accuracy and suffix accuracy rise on similar timescales while validation remains low. At larger N, the model fits the rule-generated matrix before the much shorter random suffix. This pattern is consistent with shared rule learning helping fit the matrix, while A_R tracks progress on the separately assigned random associations.

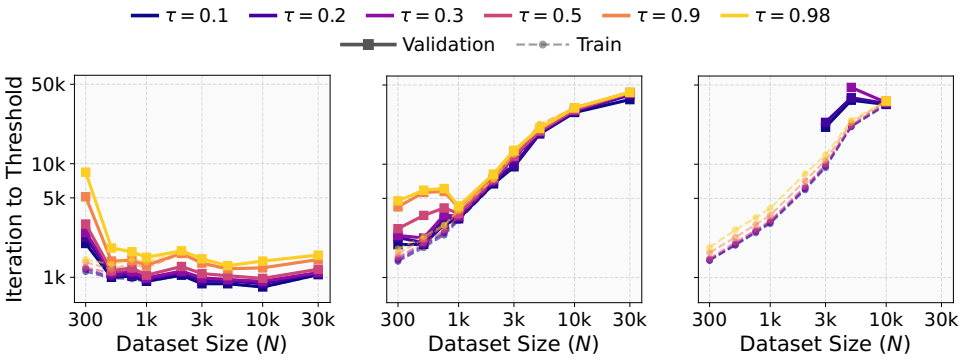
4.2 Early Fitting Combines Rule Learning and Memorization

The train–random gap measures NW training accuracy beyond the random-suffix fitting reference,

$$G(t) = A_T(t) - A_R(t).$$

The *early transition* is the initial rise of NW validation accuracy from zero. Figure 3(d–f) shows the gap becoming positive alongside validation, as matrix accuracy exceeds suffix accuracy.

During the early transition, the empirical relation $A_T(t) \approx A_V(t) + A_R(t)$ is consistent with contributions from both transferable rule learning and example-specific fitting. The agreement between $A_T(t) - A_R(t)$ and $A_V(t)$ suggests that the rule improving validation predictions also helps fit the training set. Generalization therefore emerges while example-specific fitting is still incomplete, rather than beginning only after memorization is finished. Because A_R measures a separate target, the relation provides a diagnostic of the two contributions rather than an exact partition of training examples. Once validation accuracy is high, A_T saturates while A_R continues to rise (Figure 3(b–c)), so the additive account describes the early transition, not the entire learning trajectory.



(a) Rule 110. (b) Rule 90. (c) Rule 150.

Figure 4: Dataset-size regimes in ECA. Rules 110, 90, and 150 use the same input and output format and the same training settings. Colors denote exact-trajectory thresholds. Solid-square curves show validation crossings, and dashed-circle curves show training-dataset diagnostic crossings. Rule 110 quickly reaches a near-flat update requirement, Rule 90 exhibits a large-data slowdown after the training–validation delay narrows, and Rule 150 remains difficult to generalize within the budget.

These observations suggest that acquiring shared predictions and fitting example-specific associations can contribute simultaneously to training progress. The increasing cost of suffix memorization motivates investigating whether analogous residual fitting costs contribute to the original NW slowdown. The probe alone does not determine why the response differs between NW and multiplication. Section 5 therefore examines *task learnability*, asking what makes a rule easier to acquire and how larger datasets can weaken that advantage.

5 Why More Data Slows Learning: Task Learnability

We investigate the large-data slowdown through task learnability. NW and multiplication differ in both their rules and output formats. Elementary cellular automata (ECA) repeatedly update each binary cell using its own state and those of its two neighbors. ECA let us compare rules at fixed output length, recurrence structure, and training settings. We test whether lower-order signals make a rule easier to learn and whether larger datasets weaken those signals.

5.1 Learning Regimes in a Controlled ECA Setting

In an elementary cellular automaton (ECA), each cell updates its binary state based only on its own current state and those of the two adjacent cells. Let $x_i^t \in \{0,1\}$ denote the state of cell i at time t, and let $x^t = (x_1^t, \ldots, x_L^t)$ denote the full row of states. Starting from an initial row x^0, the same rule $f_r : \{0,1\}^3 \to \{0,1\}$ [Wolfram, 1983] updates all cells simultaneously,

$$x_i^t = f_r(x_{i-1}^{t-1}, x_i^{t-1}, x_{i+1}^{t-1}).$$

We call the three preceding-row states $(x_{i-1}^{t-1}, x_i^{t-1}, x_{i+1}^{t-1})$ the *parents* of x_i^t. Cyclic boundaries make the first and last cells adjacent. The model receives an initial row x^0 of $L = 20$ cells and generates the next $T = 8$ rows $x^1, \ldots, x^8$, concatenated with row separators. Under the uniform population distribution, the initial states x_i^0 are independent Bernoulli$(1/2)$ variables. We use the same initial rows for all three rules. Appendix H describes the model, optimizer, dataset, and evaluation.

Rule 110 is asymmetric and nonlinear, whereas Rules 150 and 90 compute parity over three and two parents, respectively. Rule 90 uses the left and right parents. We describe their observed learning regimes as easy, hard, and intermediate, respectively. These rules exhibit distinct learning regimes despite identical output lengths, recurrence structures, and training settings.

Easy: initial speedup, then a plateau. Rule 110 generalizes in fewer updates as the dataset first grows (Figure 4(a)). At larger dataset sizes, the required updates remain roughly constant, and training and validation improve on similar timescales. The pattern is consistent with grokking studies in which additional data shortens the delay between fitting and generalization [Power et al., 2022, Liu et al., 2022, Zhu et al., 2024].

Hard: little generalization within the budget. Rule 150 (Figure 4(c)) fits the training diagnostic at smaller dataset sizes but rarely generalizes. With more data, it takes longer to reach the training thresholds and eventually fails to reach them within the budget. Thus, a short and exactly specified parity rule can still be hard for the models to learn.

Intermediate: narrowing train–validation delay, then slowdown. Rule 90 (Figure 4(b)) exhibits two contrasting responses within the same learnable task. At smaller dataset sizes, training accuracy rises well before validation, exhibiting a delay characteristic of grokking [Power et al., 2022, Liu et al., 2022]. At larger dataset sizes, the mean recorded training and validation crossing times nearly coincide and increase together with N. Unlike Rule 110, Rule 90 shows no plateau in acquisition time as N grows. It also generalizes across a wider range of dataset sizes than Rule 150. At fixed output length and recurrence structure, we next examine which rule properties make learning easier.

5.2 Lower-Order Signals Make Rules Easier to Learn

Useful prediction can begin before the model learns the full rule. The model can first capture patterns that require little input information, such as an imbalance in target frequencies or a predictive relationship involving only a few inputs. Such patterns need not determine the output to improve accuracy and reduce loss relative to uniform guessing. A *signal* is the predictive structure that makes such improvement possible. We quantify the *k-order signal*, $G_k(P)$, as the largest reduction in target uncertainty an ideal predictor can obtain under P using at most k parent bits.

At a fixed position i and transition t, let $Y = x_i^t$ denote the target and $\mathcal{Y}$ its set of possible labels. Let P be the distribution of the preceding row. For $U \subseteq \{i-1, i, i+1\}$, write $X_U = (x_j^{t-1})_{j \in U}$ for the observed parents, and let $\mathcal{U}_k$ contain all such subsets with at most k elements, including the empty set. The k-order signal is

$$G_k(P) = \log_2 |\mathcal{Y}| - \min_{U \in \mathcal{U}_k} H_P(Y \mid X_U) = G_0(P) + \max_{U \in \mathcal{U}_k} I_P(Y; X_U).$$

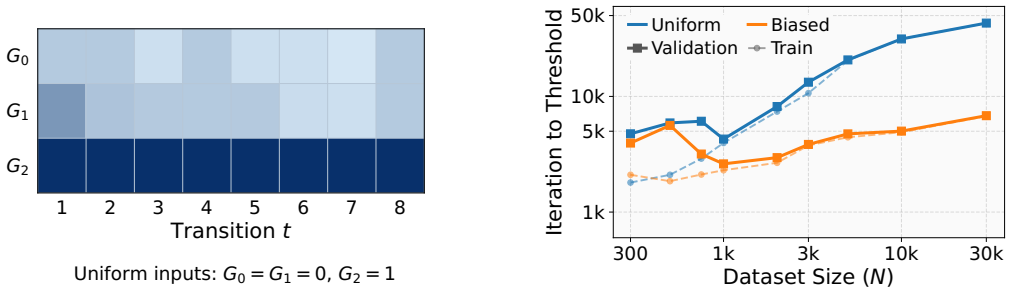
The reference predictor assigns equal probability to every possible label, giving an uncertainty of $\log_2 |\mathcal{Y}|$ bits. In our ECA task, $\mathcal{Y} = \{0, 1\}$, so this reference uncertainty is $\log_2 2 = 1$ bit. The conditional entropy $H_P(Y \mid X_U)$ measures the uncertainty remaining after observing the parents in U, and the minimum selects the most informative allowed subset. Thus, $G_k(P)$ measures how much target uncertainty can be reduced using at most k parents. Equivalently, it combines the output-bias signal $G_0(P) = \log_2 |\mathcal{Y}| - H_P(Y)$, arising from unequal target frequencies, with the additional signal from the observed parents, measured by mutual information $I_P(Y; X_U)$.

The order k bounds the number of observed parents, rather than isolating a pure k-way interaction. The quantities describe available signals, not signals necessarily learned by the model. Every ECA rule has $G_3(P) = 1$ because its three parents determine the output exactly. The full three-parent signal therefore does not distinguish the rules. Signals available from fewer or no parent bits are *lower-order signals*, which can support useful prediction before the full rule is known. Comparing G_0, G_1, and G_2 reveals whether the rules offer similarly easy ways to begin learning.

Rule 110 offers useful predictions from fewer parents than Rules 90 and 150 (Figure 5). Under uniform one-step inputs, Rule 110 has positive G_0 and G_1, although no pair fully determines the output ($G_2 < 1$). Rule 90 has the larger second-order signal, $G_2 = 1$, but $G_0 = G_1 = 0$, and Rule 150 requires all three parents for useful prediction. Perfect prediction from two parents does not make Rule 90 easier than Rule 110. Lower-order signals can matter more for acquisition than a stronger signal that requires combining parents. For staircase targets, gradient-based learning can acquire higher-order features by building on lower-order features learned earlier [Abbe et al., 2021, 2023]. The staircase perspective suggests that Rule 110 is easier because useful lower-order predictions can aid learning before relationships involving more parents are acquired. Uniform Rules 90 and 150 require multiple parents to improve on chance.

	G_0	G_1	G_2	G_3
Rule 110	0.046	0.094	0.750	1
Rule 90	0	0	1	1
Rule 150	0	0	0	1

Figure 5: Uniform one-step k-order signals. Darker blue indicates larger values, with zero shown in white.



(a) Inputs with 13–15 ones.

(b) Rule 90, $\tau = 0.98$.

Figure 6: Biased training reduces Rule 90's slowdown. **(a)**: k-order signals across eight transitions for inputs with 13–15 ones. Biased inputs introduce lower-order signals absent under uniform inputs ($G_0 = G_1 = 0$). White denotes zero and darker blue denotes larger G_k. **(b)**: mean updates to reach 0.98 exact eight-row accuracy. Validation remains unchanged and uniform. Solid squares and dashed circles mark validation and training-dataset diagnostic crossings, respectively. Within each condition, both means use runs that reach validation accuracy 0.98.

5.3 Adding Lower-Order Signals Accelerates Rule 90 Acquisition

We test whether adding lower-order signals makes Rule 90 faster to learn. We hold the XOR rule, model, and optimization settings fixed and manipulate the distribution of initial rows. Uniformly sampled rows serve as the baseline, while biased sampling restricts the 20-bit initial rows to states containing 13–15 ones. The biased distribution introduces positive G_0 and G_1 without changing the rule to be learned (Figure 6(a)). Both training conditions use the same uniform validation set, so the comparison tests whether providing lower-order signals during training accelerates acquisition on the original task. Appendix H details the construction and signal calculation.

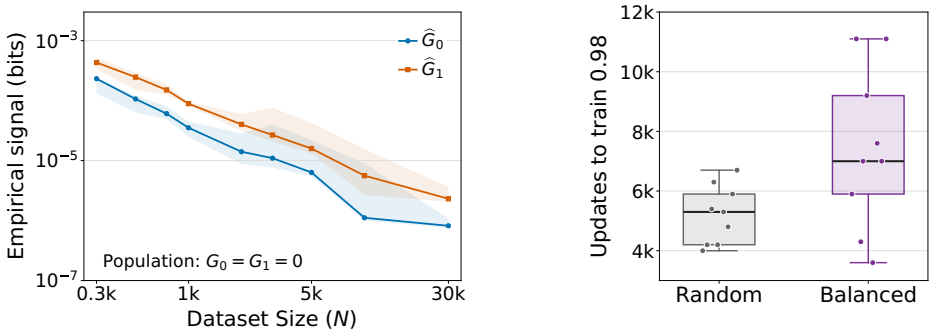
Biased training reaches the validation threshold in substantially fewer updates at larger dataset sizes (Figure 6(b)). Since the biased distribution introduces lower-order signals, this result supports a role for such signals in rule acquisition on uniform validation inputs. The bias here was deliberate. We next examine signals introduced by random sampling, which can change with dataset size.

5.4 More Data, Weaker Signals, Slower Learning

Smaller datasets can supply lower-order signals absent from the population. Under uniform initial cells, Rule 90 has $G_0(P) = G_1(P) = 0$ for the transitions studied. A sampled dataset's empirical distribution $\widehat{P}_N$ need not match the population exactly. Chance deviations create output imbalance and single-parent dependence, producing positive $\widehat{G}_k = G_k(\widehat{P}_N)$ for $k = 0, 1$ [Paninski, 2003]. Such deviations are typically larger in smaller samples. As N grows, the empirical distribution approaches the population and the signals decrease toward zero (Figure 7(a)). The accidental patterns can help early training predictions even though they need not generalize to the population.

We test whether the empirical signals affect learning by comparing *Random* and *Balanced* datasets at $N = 2{,}000$. The target trajectories and their frequencies are identical, and only the initial rows differ. Both datasets have first-transition $G_0 = 0$, but Random retains $G_1 > 0$ while Balanced has $G_1 = 0$. Balanced takes more updates, on average, to reach training accuracy 0.98 (Figure 7(b)). The comparison supports the learning benefit of empirical lower-order signals, although balancing also changes other input statistics. Appendix H gives the construction.

More data can therefore describe the population better while weakening empirical signals that make learning easier. In Rule 90, weakening lower-order signals helps explain why larger datasets take longer to learn. The NW suffix probe provides complementary evidence that fitting arbitrary example-specific associations becomes more costly as the dataset grows (Section 4). Together, these experiments suggest that weakening empirical signals and increasing fitting costs may contribute to the observed slowdowns.



(a) Empirical lower-order signals.

(b) Matched $N = 2$k datasets.

Figure 7: Empirical signals and training-set fitting in Rule 90. **(a)**: empirical G_0 and G_1 approach their population value of zero as dataset size grows. Lines and bands show the median and range over three datasets after averaging over the eight transitions. **(b)**: number of updates to reach 0.98 training accuracy at $N = 2{,}000$. Targets and first-transition $G_0 = 0$ are matched, but $G_1 > 0$ for Random and $G_1 = 0$ for Balanced. Balanced fits more slowly on average. Boxes show medians and interquartile ranges over nine runs. Dots show individual runs.

6 Discussion and Limitations

What the evidence supports. The large-data slowdown is not limited to full NW tables or a single training configuration. Depth, batch-size, weight-decay, and cell-accuracy checks support the slowdown across multiple settings (Appendices D and J). Edit-distance tables and partial NW outputs retain a sweet spot, while scratchpad multiplication also slows at larger dataset sizes (Appendix I). In ECA, rules with identical output length and recurrence structure exhibit different learning regimes. Output format alone therefore cannot explain the contrast. Lower-order signals help explain differences in task learnability and why larger Rule 90 datasets can learn more slowly.

When can smaller datasets help? Our results suggest that the benefits of additional examples can be offset by changes in the predictive signals available during training and by the cost of fitting example-specific associations. In Rule 90, sampling fluctuations supply lower-order signals that weaken as the dataset grows. The NW suffix probe shows that memorizing example-specific associations takes longer with more examples. The account connects to the benefits of repeated examples reported by Charton and Kempe [2024] and the sampling-bias explanation of Liu et al. [2026]. Useful biases need not arise only by chance. Deliberately biased Rule 90 training accelerates learning on an unchanged uniform validation distribution (Section 5.3). The result motivates testing signal-rich synthetic datasets and curricula that gradually broaden the training distribution in code and reasoning tasks. Dataset size and composition should therefore be treated as training choices, rather than assuming that more distinct examples always make learning faster.

Limitations and open questions. We do not yet predict which task properties place a problem in the intermediate regime or determine its optimal dataset size. The lower-order-signal account is tested in Rule 90, but comparable signals have not been measured directly in NW. Defining those signals and identifying how rule learning and example-specific fitting interact inside the model would test the proposed explanation more directly. Experiments with larger models and variable-length inputs are also needed to establish how broadly the findings apply.

7 Conclusion

NW reaches high validation accuracy fastest at an intermediate dataset size, even though adding examples can initially accelerate training-set fitting. Controlled ECA experiments connect the differing data-size responses to task learnability. Biased Rule 90 training that introduces lower-order signals accelerates acquisition on uniform validation inputs, while empirical signals from random sampling weaken with dataset size. Together, the Rule 90 interventions and the NW random-suffix diagnostic suggest complementary pressures that may contribute to the large-data slowdown. More examples can help reveal a rule without reducing the number of updates needed to learn it.

Acknowledgments and Disclosure of Funding

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) in 2026 (Grant No. RS-2026-25612350, Project Title: "Development of an End-to-End Design and Validation Framework for Mathematically Principled Next-Generation AI Architectures and Learning Algorithms"), the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. RS-2024-00410005), and the Pioneer Research Center Program through the National Research Foundation of Korea funded by the Ministry of Science, ICT & Future Planning (RS-2022-NR067569).

Competing interests. The authors declare that, during the 36 months preceding submission, they had no financial relationships outside the submitted work that could reasonably be perceived as influencing this research.

References

Emmanuel Abbe, Enric Boix-Adsera, Matthew S. Brennan, Guy Bresler, and Dheeraj Nagaraj. The staircase property: How hierarchical structure can guide deep learning. In *NeurIPS*, 2021.

Emmanuel Abbe, Enric Boix Adserà, and Theodor Misiakiewicz. SGD learning on neural networks: Leap complexity and saddle-to-saddle dynamics. In *COLT*, 2023.

Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *ICML*, 2017.

Boaz Barak, Benjamin Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. Hidden progress in deep learning: Sgd learns parities near the computational limit. In *NeurIPS*, 2022.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *USENIX Security*, 2019.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *USENIX Security*, 2021.

Francois Charton. Learning the greatest common divisor: explaining transformer predictions. In *ICLR*, 2024.

François Charton and Julia Kempe. Emergent properties with repeated examples. *arXiv preprint arXiv:2410.07041*, 2024.

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In *NeurIPS*, 2020.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Yufei Huang, Shengding Hu, Xu Han, Zhiyuan Liu, and Maosong Sun. Unified view of grokking, double descent and emergent abilities: A perspective from circuits competition. *arXiv preprint arXiv:2402.15175*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Nayoung Lee, Kartik Sreenivasan, Jason D Lee, Kangwook Lee, and Dimitris Papailiopoulos. Teaching arithmetic to small transformers. *arXiv preprint arXiv:2307.03381*, 2023.

Jingwen Liu, Ezra Edelman, Surbhi Goel, and Bingbin Liu. Less data, faster training: Repeating smaller datasets speeds up learning via sampling biases. In *ICML*, 2026.

Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In *NeurIPS*, 2022.

Ziming Liu, Eric J Michaud, and Max Tegmark. Omnigrok: Grokking beyond algorithmic data. In *ICLR*, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Kaifeng Lyu, Jikai Jin, Zhiyuan Li, Simon S Du, Jason D Lee, and Wei Hu. Dichotomy of early and late phase implicit biases can provably induce grokking. *arXiv preprint arXiv:2311.18817*, 2023.

Gouki Minegishi, Yusuke Iwasawa, and Yutaka Matsuo. Bridging lottery ticket and grokking: Understanding grokking from inner structure of networks. *TMLR*, 2025.

John X Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G Edward Suh, Alexander M Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize? *arXiv preprint arXiv:2505.24832*, 2025.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *ICLR*, 2023.

Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 1970.

Liam Paninski. Estimation of entropy and mutual information. *Neural Computation*, 2003.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 1981.

Vikrant Varma, Rohin Shah, Zachary Kenton, János Kramár, and Ramana Kumar. Explaining grokking through circuit efficiency. *arXiv preprint arXiv:2309.02390*, 2023.

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

Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 1967.

Robert A. Wagner and Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 1974.

Kaiyue Wen, Huaqing Zhang, Hongzhou Lin, and Jingzhao Zhang. From sparse dependence to sparse attention: Unveiling how chain-of-thought enhances transformer sample efficiency. In *ICLR*, 2025.

Stephen Wolfram. Statistical mechanics of cellular automata. *Reviews of Modern Physics*, 1983.

Mingyue Xu, Gal Vardi, and Itay Safran. To grok grokking: Provable grokking in ridge regression. In *ICML*, 2026.

Daniel H. Younger. Recognition and parsing of context-free languages in time n^3. *Information and Control*, 1967.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.

Xuekai Zhu, Yao Fu, Bowen Zhou, and Zhouhan Lin. Critical data size of language models from a grokking perspective. *arXiv preprint arXiv:2401.10463*, 2024.

A Task Details

A.1 Needleman–Wunsch Task Details

We use fixed-length Needleman–Wunsch matrix prediction as a controlled structured-output task. Each input consists of two sequences $x = (x_1, \ldots, x_m)$ and $y = (y_1, \ldots, y_m)$ over a finite alphabet $\mathcal{A}$. The target is the full dynamic-programming score matrix $M \in \mathbb{Z}^{(m+1)\times(m+1)}$ computed by the Needleman–Wunsch recurrence. The main experiments use $m = 5$ and $|\mathcal{A}| = 4$.

The score matrix is initialized by

$$M_{0,0} = 0, \qquad M_{i,0} = i \cdot g, \qquad M_{0,j} = j \cdot g,$$

where g is the gap penalty. For $i, j \geq 1$, the recurrence is

$$M_{i,j} = \max\left\{M_{i-1,j-1} + s(x_i, y_j), M_{i-1,j} + g, M_{i,j-1} + g\right\},$$

where $s(x_i, y_j)$ is the match/mismatch score. We use $s(a, b) = 5$ (match), -4 (mismatch) and gap penalty $g = -5$ throughout the main experiments.

The model predicts the score matrix as a sequence of discrete output tokens. Unless otherwise stated, we evaluate exact full-matrix accuracy: an example is counted as correct only when all predicted matrix entries match the Needleman–Wunsch target matrix. Cell-level accuracy is used only for diagnostic analyses and is not the main convergence metric.

For each dataset size N, we sample N training examples from the finite task universe and evaluate on a fixed held-out validation set. Unless otherwise stated, the validation set is held fixed across runs for the same task configuration. Dataset sizes are swept over $N \in \{200, 500, 1k, 2k, 3k, 4k, 5k, 10k, 50k, 100k\}$.

A.2 Addition and Multiplication Baselines

We use decimal addition and multiplication as short-output algorithmic baselines. These tasks are included to test whether exact-match thresholding alone produces an intermediate-data sweet spot. The main claims of the paper concern Needleman–Wunsch matrix generation; the arithmetic tasks serve as contrast controls.

For d-digit addition, each input is a pair of nonnegative decimal integers (a, b), where $a, b \in \{0, \ldots, 10^d - 1\}$, and the target is the exact decimal representation of $a + b$. For d-digit multiplication, the input is again a pair (a, b), with $a, b \in \{0, \ldots, 10^d - 1\}$, and the target is the exact decimal representation of $a \cdot b$. The operand lengths are $d = 3$ for multiplication (Figure 1a) and $d = 10$ for the addition task (Figure 15).

Inputs are blank-padded to length d. Targets are first written in canonical decimal form without leading zeros and then blank-padded to the maximum target length for batching and autoregressive prediction. Exact-match accuracy is computed on the full padded target sequence.

For arithmetic baselines, training examples are sampled from the finite input universe and validation examples are held out from the same task family. The multiplication sweep uses

$$N \in \{500, 1\text{k}, 5\text{k}, 10\text{k}, 20\text{k}, 30\text{k}, 40\text{k}, 50\text{k}, 100\text{k}, 200\text{k}, 300\text{k}, 400\text{k}, 500\text{k}\}.$$

The addition sweep uses

$$N \in \{500, 600, 700, 800, 900, 1\text{k}, 5\text{k}, 10\text{k}, 20\text{k}, 30\text{k}, 40\text{k}, 50\text{k}, 100\text{k}, 200\text{k}, 300\text{k}, 400\text{k}, 500\text{k}\}.$$

Optimizer, model-size, threshold-crossing, and censoring conventions follow NW unless noted.

B Full Model and Training Details

B.1 Dataset sweep and model

For NW at length L, the finite task universe has size $|\mathcal{D}_{\text{full}}| = |\Sigma|^{2L} = 4^{2L}$. At $L = 5$, this gives $4^{10} = 1{,}048{,}576$ possible input pairs. For each dataset size N, we sample a training set $\mathcal{D}_N \subset \mathcal{D}_{\text{full}}$ and evaluate on held-out examples from the same universe. The NW sweep uses

$$N \in \{200, 500, 1\text{k}, 2\text{k}, 3\text{k}, 4\text{k}, 5\text{k}, 10\text{k}, 50\text{k}, 100\text{k}\}.$$

We train decoder-only Transformers [Vaswani et al., 2017]. A depth-D model has D layers, D attention heads, embedding dimension $64D$, and feed-forward width $256D$. The main depth sweep uses $D \in \{3, 4, 5, 6\}$. Models are trained with AdamW [Loshchilov and Hutter, 2019] and a cosine learning-rate schedule from 10^{-3} to 10^{-4}, with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and weight decay 0.1. The effective batch size is 160, obtained by gradient accumulation, and each run is trained for at most $50{,}000$ optimizer updates with evaluation every 100 updates. Batch size, optimizer settings, tokenization, and evaluation frequency are held fixed within each task sweep. We treat all non-crossing runs as censored rather than assigning them finite crossing times. Each task and dataset-size configuration is run with five random seeds unless otherwise stated.

The baseline effective batch of 160 combines eight sequences per microbatch with 20 gradient-accumulation steps. Models are trained from random initialization. Shaded regions report one standard deviation across runs with a recorded crossing at the plotted threshold.

If a run does not reach the threshold within $50{,}000$ updates, we treat it as censored and do not assign it a finite crossing time. Figure 1 uses the representative $D = 6$ setting, Figure 2 shows $D = 4, 5$, and Appendix D provides the full depth sweep.

Compute resources. The original experiments were run on NVIDIA RTX 3090 GPUs with 24 GB memory. A single NW run at L=5 and depth D=6 takes approximately 10 GPU-hours for 50,000 optimizer updates. The original experimental suite used approximately 2,600 GPU-hours, including the main dataset-size sweep, depth sweep, sequence-length robustness experiments, and random-suffix probes. Preliminary or failed runs used approximately 1,077 additional GPU-hours.

C Robustness and Measurement Scope

The robustness evidence includes the original depth and sequence-length sweeps, seed variability, epoch normalization, and full random-suffix trajectories (Appendices D–F). Further checks vary effective batch size, weight decay, and model depth, and compare cell accuracy with exact match (Appendix J). Additional algorithmic tasks are reported in Appendix I.

Depth robustness. Figure 2 repeats the NW sweep across depths. The qualitative pattern is stable: validation crossing time is non-monotonic in N, and train crossing time can decrease near the onset of weak validation competence. The optimal dataset size shifts with depth, but the two findings do not rely on a single representative model.

Threshold robustness. The use of multiple thresholds is essential. A single strict threshold would show only the late stage of learning. The lower thresholds reveal that validation competence emerges gradually and can precede strict train convergence. These crossings are consistent with transferable structure being learned while training-set fitting remains incomplete. The strict threshold $\tau = 0.98$ is used for the headline sweet spot because it measures high exact-match accuracy; weaker thresholds diagnose the transition into that regime.

For the NW and arithmetic plots with recorded cutoffs spaced by $1/80$, the displayed crossing is selected at the smallest recorded cutoff at least as high as τ. Curves labeled $\tau = 0.98$ therefore use the recorded 0.9875 crossing. These cutoffs are equivalent for exact-match evaluation on 80 examples, but need not be equivalent at other evaluation sizes or for cell accuracy.

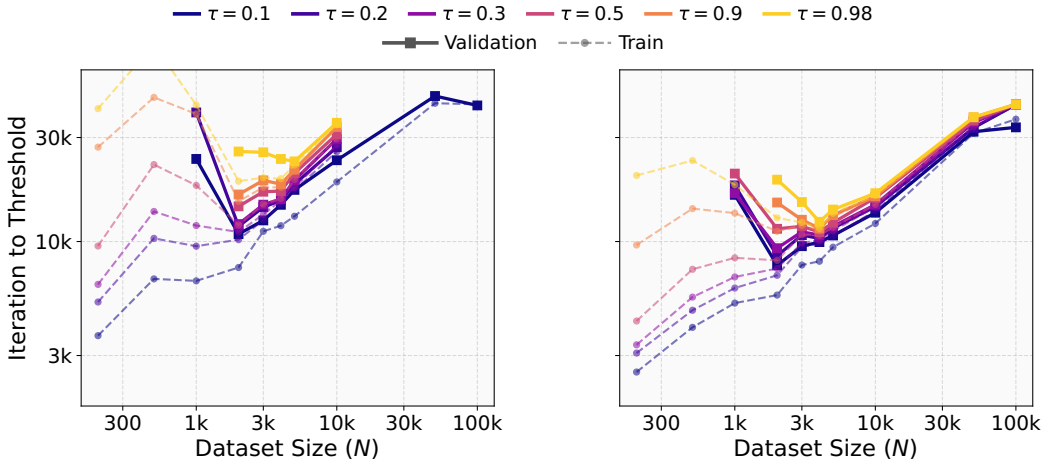
NW crossing times are the first evaluated checkpoints that meet the threshold, with evaluation every 100 optimizer updates. A recorded crossing does not require accuracy to remain above the threshold at later checkpoints.

Update-based scope. The slowdown we report is specifically update-based. One optimizer update sees a fixed batch size, not a fixed fraction of the dataset. As N grows, the same number of updates covers fewer epochs. Thus, our result should not be read as saying that larger datasets are intrinsically worse, or that they reduce asymptotic accuracy. It says that, for strict exact-match NW generation under the training protocol studied here, the fastest path in gradient-update count occurs at an intermediate dataset size. Epoch-normalized curves answer a different question: how many passes over the data are needed, rather than how many parameter updates are needed.

Censoring and interpretation. Censoring is applied to individual runs: a run that does not reach a threshold within the budget contributes no finite crossing time. For the NW and arithmetic sweeps, training and validation means at each threshold use the runs with recorded crossings on the corresponding split. A dataset size remains eligible if at least one run reaches the threshold. Thus, the reported means condition on within-budget threshold attainment and do not include the probability of attainment. The critical size $N_c(\tau)$ is the smallest tested size with a recorded validation crossing, and the optimal size $N^\star(\tau)$ minimizes the mean among sizes with such crossings. The observed sweet spot compares intermediate and larger datasets with recorded crossings; its interpretation is conditional on the contributing runs.

D Robustness Across Depth

We test whether Findings 1 and 2 are specific to the representative $D = 6$, $L = 5$ setting or whether they reproduce across model depth. Figure 8 reports the multi-threshold trajectories of Figure 1b for $D \in \{3, 4, 5, 6\}$ at $L = 5$. The qualitative pattern is preserved across depths: validation crossing time is non-monotonic in N (Finding 2), and the strict training crossing $I_T(0.98; N)$ can decrease near the onset of weak validation competence (Finding 1). The optimal dataset size varies with model depth.



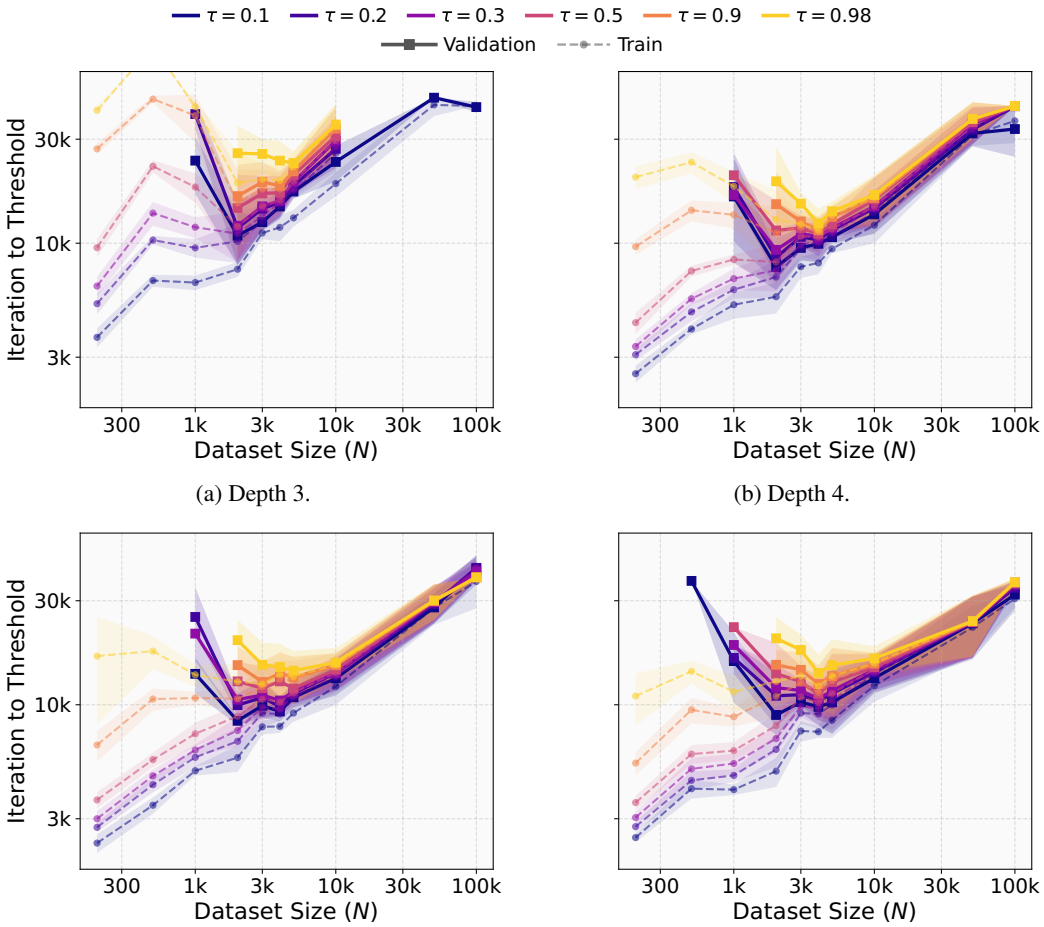
(a) Depth 3.

(b) Depth 4.

(c) Depth 5.

(d) Depth 6.

Figure 8: NW multi-threshold sweeps across depths $D \in \{3, 4, 5, 6\}$ at $L = 5$. Colors denote exact-match thresholds, solid lines show validation crossings, and dashed lines show training crossings. Across depths, validation crossing time is non-monotonic in dataset size, giving an intermediate-data sweet spot. In the same dataset-size band where weak validation thresholds first become reachable, training crossings can also become faster as N increases, supporting the partial-rule interpretation.



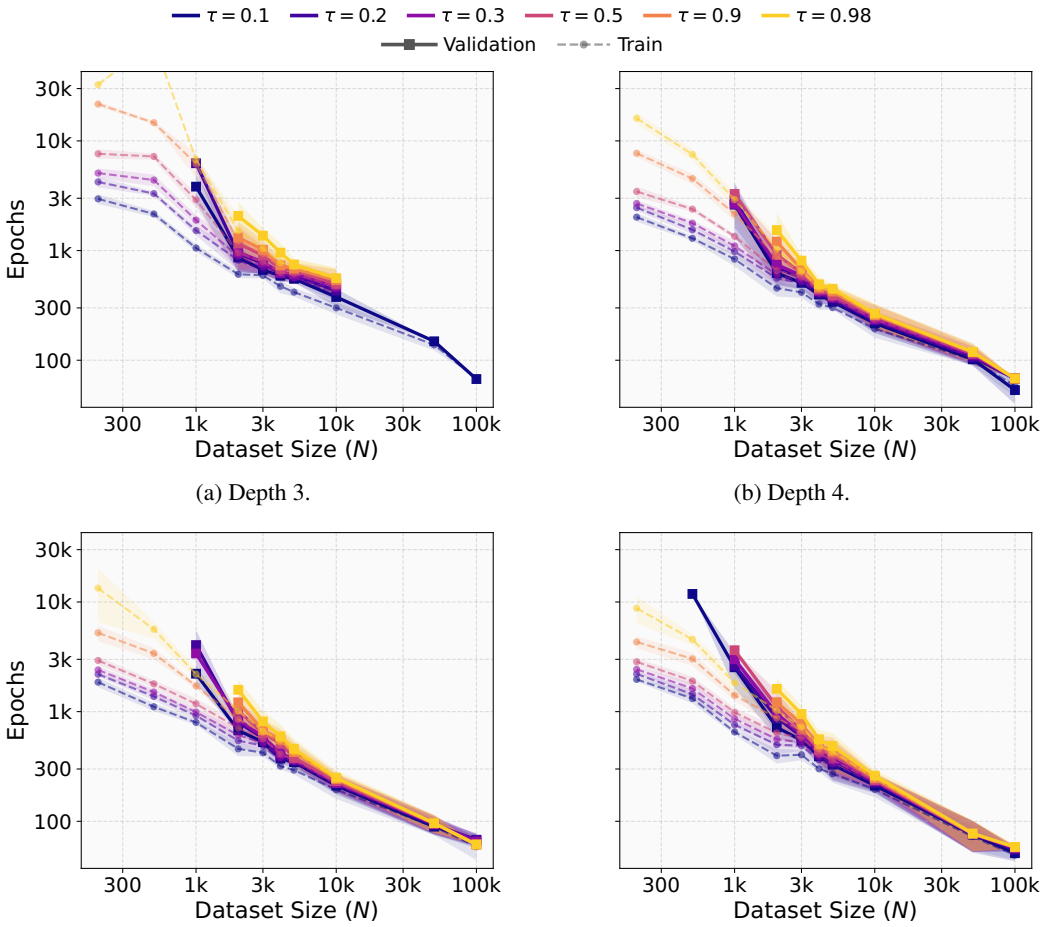
(a) Depth 3.

(b) Depth 4.

(c) Depth 5.

(d) Depth 6.

Figure 9: Seed variability for the $L = 5$ NW multi-threshold sweeps. Curves show the same depth sweep as Figure 8, and shaded regions denote one standard deviation across runs with recorded crossings. Although exact crossing times vary, the qualitative pattern remains stable: validation crossings are non-monotonic in N, and training crossings improve near the onset of weak validation competence.



(a) Depth 3.

(b) Depth 4.

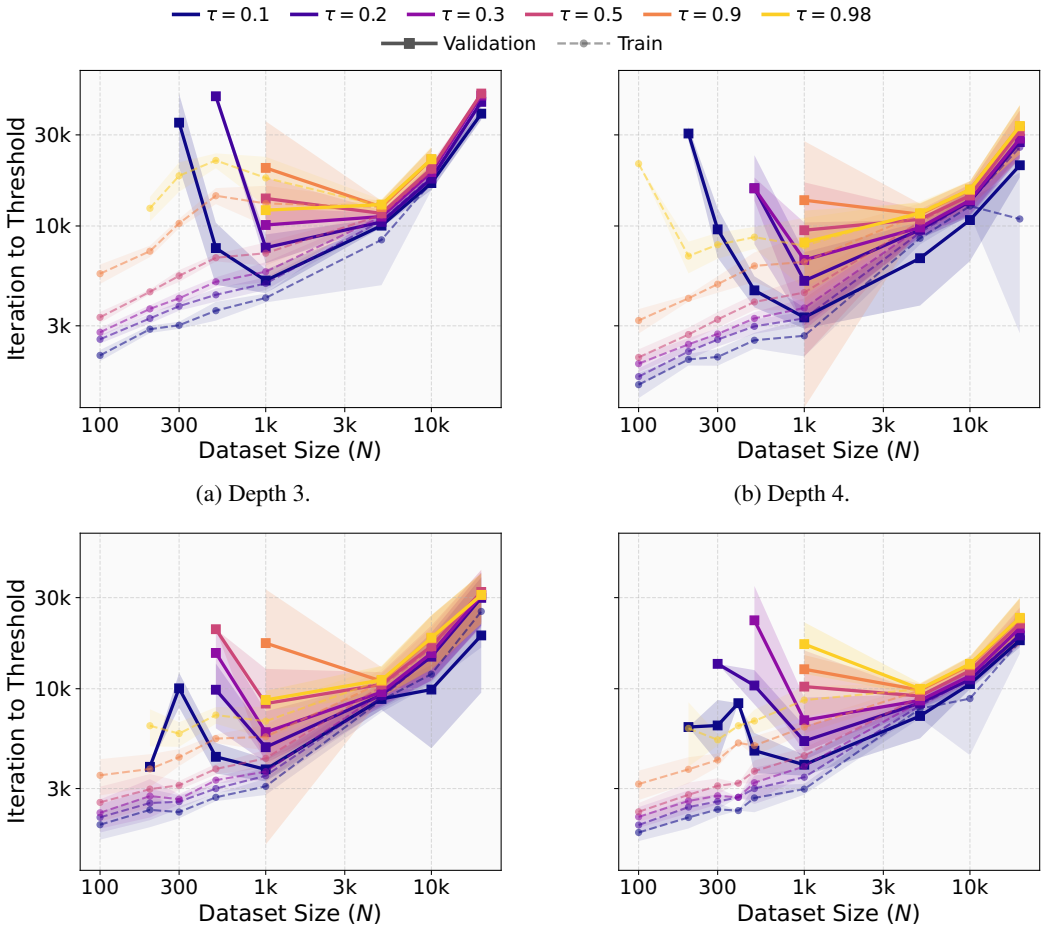
(c) Depth 5.

(d) Depth 6.

Figure 10: Epoch-normalized convergence for the $L = 5$ NW depth sweep. The same threshold crossings are measured in epochs rather than optimizer updates. Epoch normalization emphasizes data exposure and recovers the more conventional view in which larger datasets require fewer passes over the data. The main-text sweet spot therefore concerns updates, not passes over the data.

E Robustness Across Sequence Length

To test whether the timing pattern is specific to the $L = 5$ Needleman–Wunsch task, we repeat the same analysis at sequence length $L = 4$. The full task universe is smaller, but the qualitative structure is similar: validation convergence is non-monotonic in N, and the strict training crossing $I_T(0.98; N)$ can decrease near the onset of weak validation competence. Figures 11 and 12 provide seed-variance and epoch-normalized views of the same experiments.



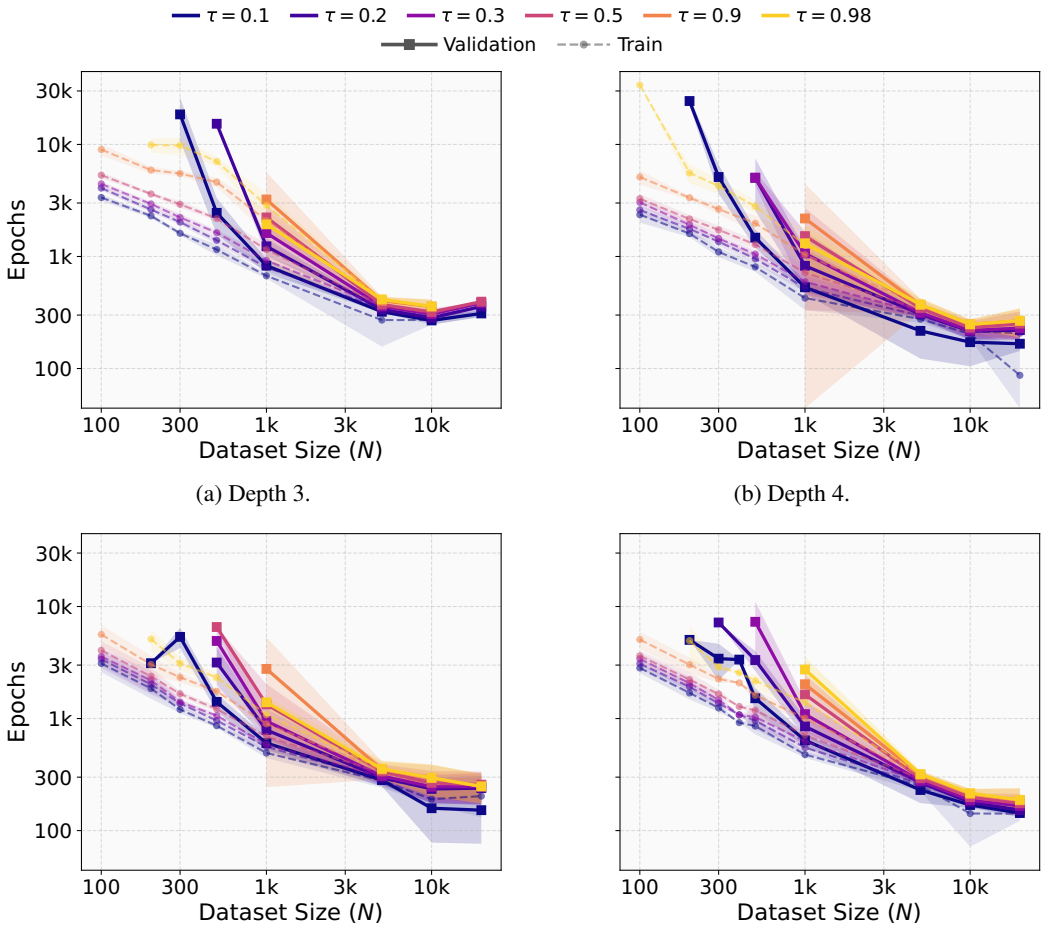
(a) Depth 3.

(b) Depth 4.

(c) Depth 5.

(d) Depth 6.

Figure 11: Seed variability for the $L = 4$ NW sweeps. Shaded regions denote one standard deviation across runs with recorded crossings. Exact threshold-crossing times vary, but the intermediate-data regime again shows overlap between weak validation competence and faster training-threshold crossings.



(a) Depth 3.

(b) Depth 4.

(c) Depth 5.

(d) Depth 6.

Figure 12: Epoch-normalized convergence for the $L = 4$ NW task. We report epochs to threshold for the same $L = 4$ configurations as Figure 11. As in the $L = 5$ experiments, epoch normalization emphasizes data exposure rather than parameter-update efficiency and recovers a conventional monotone-data view. This reinforces that the sweet spot studied in the main text is update-based.

F Random-Suffix Experiments

We extend the main-text random-suffix views to the full dataset-size sweep. Figures 13 and 14 show component-wise trajectories and the train–random gap, respectively. The suffix provides a separate reference for fitting randomized associations; its accuracy is not a calibrated estimate of residual fitting in the NW component. The correspondence between $A_T - A_R$ and A_V is interpreted locally near acquisition onset.

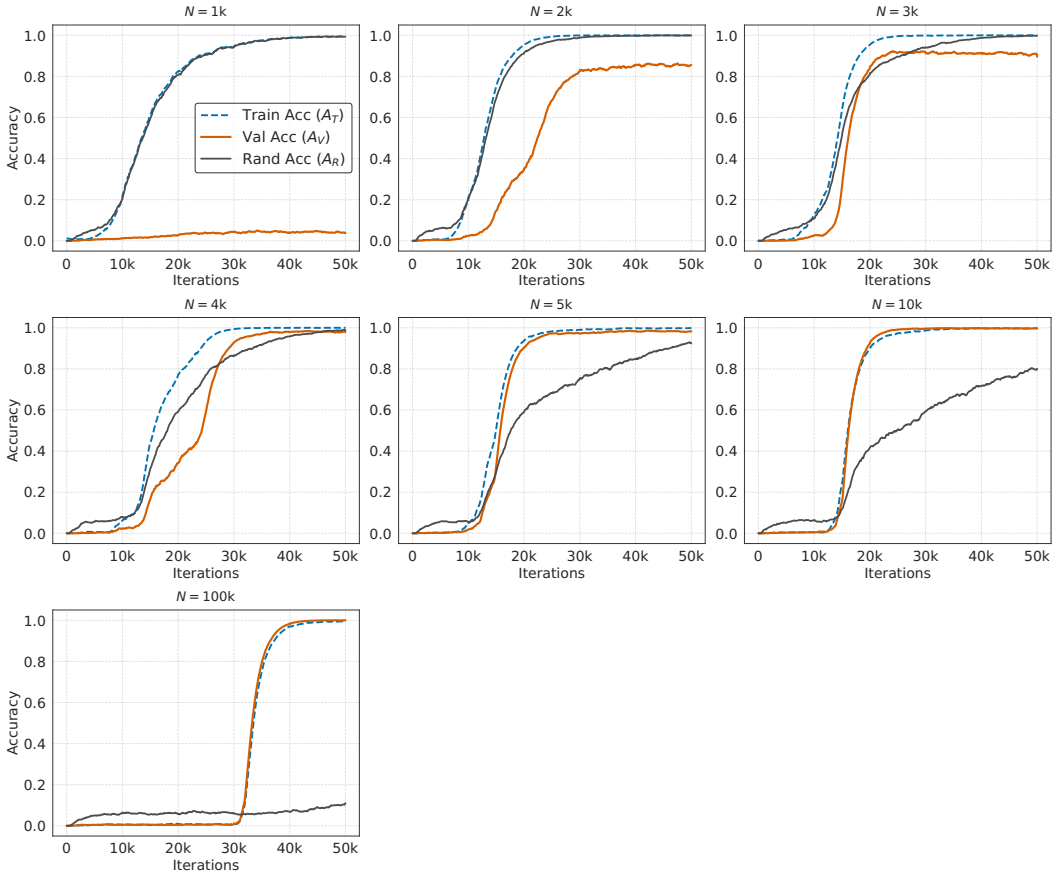


Figure 13: Full random-suffix component trajectories. Each NW target is augmented with a four-bit random suffix. We track training accuracy on the structured NW matrix A_T, validation accuracy on the NW matrix A_V, and training accuracy on the random suffix A_R. At small N, A_T and A_R rise together while A_V remains near zero. At larger N, the structured NW component reaches high training accuracy before the much shorter random suffix, matching Figure 3.

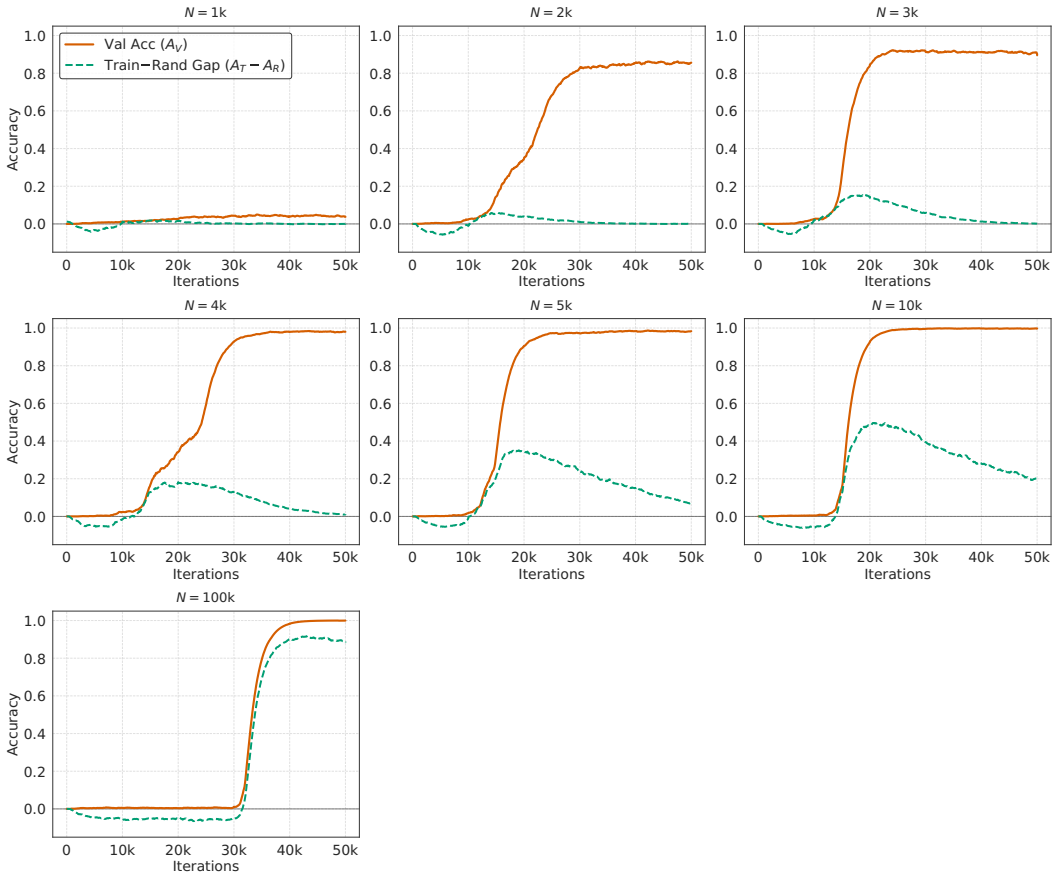


Figure 14: Full dataset-size sweep for the train–random gap. For each N, we compare NW validation accuracy A_V with $G = A_T - A_R$, where A_T is training accuracy on the NW matrix and A_R is training accuracy on the random suffix. The representative cases in Figure 3 reflect the full sweep. At small N, the gap remains small and validation remains low. At larger N, both rise together.

G Task-Scope Controls

Figure 15 compares three tasks under exact-match evaluation. Addition has similar training and validation crossing times. Multiplication improves as data grows over much of the tested range, whereas NW has an intermediate-data sweet spot. The task contrasts alone do not isolate the effects of output length or computation. The ECA comparison in Figure 4 holds output length and recurrence structure fixed and still produces different learning regimes.

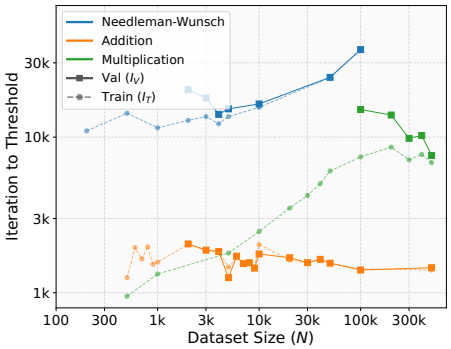


Figure 15: Task-scope controls at $\tau = 0.98$. Within these sweeps, a sweet spot is observed for NW but not for the direct arithmetic baselines.

The additional edit-distance and scratchpad-multiplication results extend the slowdown beyond NW, and partial NW shows that predicting the full matrix is not required (Appendix I). The random-suffix probe further shows that a short unstructured target can take longer to fit than a larger rule-generated target. Learning time therefore depends on the predictive signals and example-specific associations available to the model, rather than on output length alone. Rule learning and example-specific fitting can both contribute to accurate prediction, consistent with work on information not captured by shared patterns [Feldman and Zhang, 2020].

H ECA Experimental and Measurement Details

The ECA experiments use rows of length $L = 20$ and predict $T = 8$ updates from the initial row. The decoder-only Transformer has six layers, six attention heads, and hidden width 384. AdamW uses weight decay 0.1, with a cosine learning-rate schedule from 10^{-3} to 10^{-4}. The effective batch is 320 complete sequences, distinct from the NW batch size. Within the ECA comparison, all rules use the same initial-row identities, nested dataset sizes, and training settings. Each dataset size crosses three dataset draws with three model initializations.

Exact-trajectory accuracy requires all eight output rows and formatting tokens to be correct in free generation. Validation uses a fixed set separate from training. The plotted training accuracy is measured on a fixed training-dataset diagnostic subset, not the entire training dataset.

For Figure 4, we identify the highest displayed threshold reached at each rule, size, and split. The same runs reaching that threshold contribute to the crossing-time means at every lower displayed threshold. Training and validation are aggregated separately. Consequently, lower-threshold means condition on reaching the higher threshold, and the contributing cohorts can differ across dataset sizes and splits. Unreached thresholds are censored, rather than assigned a finite crossing time. For Figure 6, within each training condition, runs reaching validation accuracy 0.98 contribute to both training and validation means. This pairs the two splits within a condition. These cohort rules differ from the threshold-specific aggregation used for NW.

The population signals in Figure 5 are computed exactly from the eight equally likely three-parent configurations at the first transition. The signal profiles in Figure 6 instead enumerate the initial-row distribution restricted to 13–15 ones and calculate the resulting parent–target distribution separately at each transition. Biased sampling changes the training distribution, while validation remains uniform. The intervention preserves the rule and output format but changes multiple properties of the input distribution, so it does not isolate a single G_k.

Empirical signals. For Figure 7(a), we compute $\widehat{G}_k = G_k(\widehat{P}_N)$ from the observed parent–target frequencies in each sampled training dataset. The empirical signals are averaged over the eight transitions within each dataset. Lines and bands then show the median and range over three dataset draws. The plotted values describe prediction under the empirical distribution, whose accidental dependencies need not persist under the uniform population distribution.

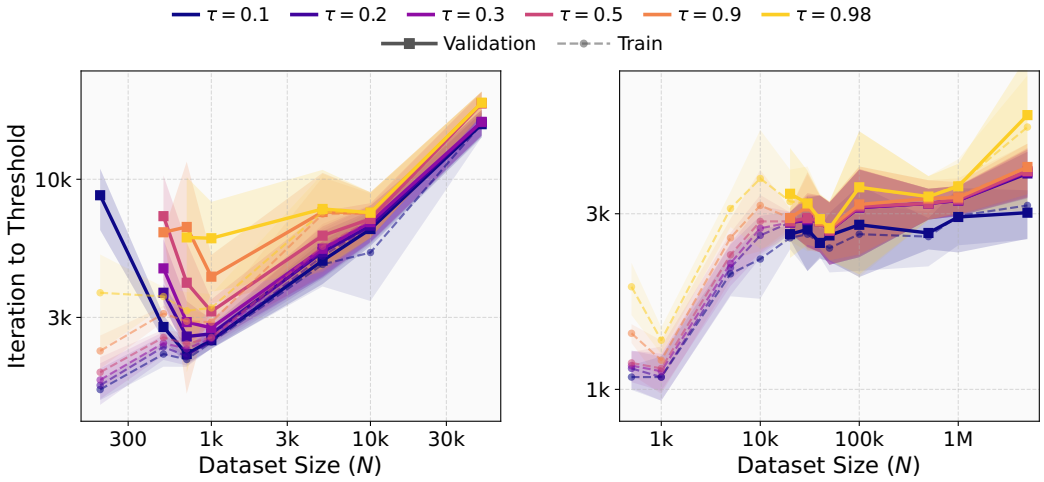
Matched Random and Balanced datasets. For length-20 cyclic rows, Rule 90 has a four-element kernel, so four distinct initial rows produce the same next row. For each sampled first target row y, we also use its bitwise complement $\bar{y}$. Each block contains two inputs mapping to y and two mapping to $\bar{y}$, so each target trajectory appears twice in both conditions. Balanced selects each input together with its bitwise complement. At every position in the first transition, the construction makes each single-parent–target joint distribution uniform, giving $G_0 = G_1 = 0$. Random instead selects two distinct preimages uniformly for each target, preserving every target trajectory and its multiplicity without enforcing zero single-parent dependence. All initial states are unique. The model and training settings are the same in both conditions.

Both conditions match output frequencies, all subsequent target rows, and trajectory multiplicities. Balanced also changes input marginals and the Hamming distances between inputs sharing a target. The intervention therefore tests the relevance of empirical lower-order signals without isolating G_1 from all other input properties. Figure 7(b) includes all nine training-threshold crossings in each condition. Balancing applies only at the first transition.

I Additional Algorithmic Tasks and Partial Outputs

We extend the dataset-size sweep to edit-distance table generation and scratchpad multiplication. Edit distance maps two length-four strings over a four-symbol alphabet to the complete Levenshtein cost table, using unit insertion, deletion, and substitution costs [Wagner and Fischer, 1974]. Scratchpad multiplication takes two four-digit integers and generates four place-value partial products followed by their final sum. Edit distance exhibits an intermediate-data validation optimum, and scratchpad multiplication requires more updates at larger dataset sizes (Figure 16).

Both task sweeps use five runs per dataset size. For each split and threshold, means and standard deviations use only runs with a recorded crossing, so these summaries condition on attainment within the budget. The highest displayed threshold, 0.98, uses the recorded 0.9875 crossing. The cutoffs agree for exact match on 80 evaluation examples but need not agree for cell accuracy. Scratchpad validation at $N = 10$k is omitted from Figure 16; the corresponding measurements are retained in the archived data.



(a) Edit-distance table generation.

(b) Scratchpad multiplication.

Figure 16: Slowdown beyond NW. **(a)**: edit-distance table generation exhibits an intermediate-data validation optimum. **(b)**: scratchpad multiplication predicts partial products and their sum, and validation convergence slows at larger dataset sizes. Colors denote exact-output thresholds. Solid squares show validation and dashed circles show training crossings. Lines and shading show means and one standard deviation across recorded crossings. Scratchpad validation at $N = 10$k is omitted.

The partial-NW experiment predicts nine cells of the length-five NW matrix, comprising its first interior row and the remaining cells of its first interior column. Each dataset size has one recorded run. Figure 17 retains an intermediate-data validation optimum, showing that the observed slowdown can also occur without generating the full matrix. The result complements the ECA comparison, which controls output length and recurrence structure across rules.

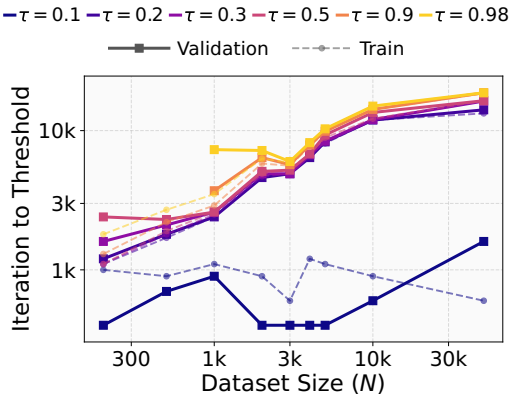
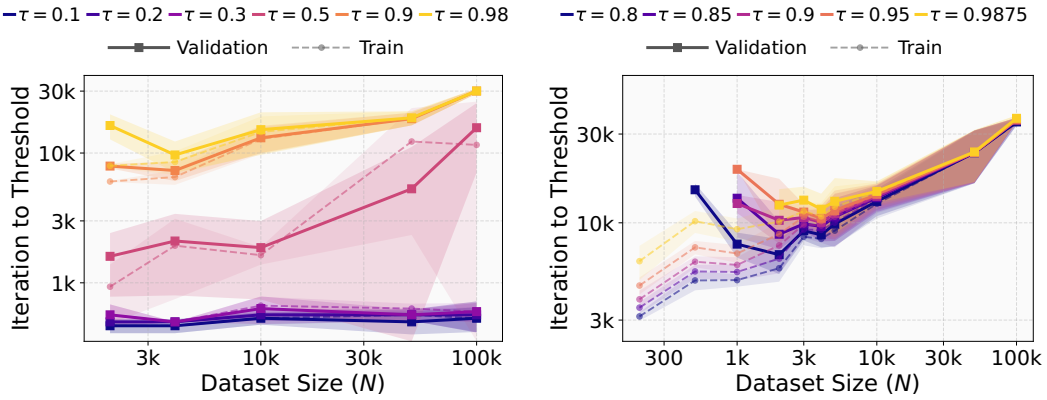


Figure 17: Partial NW predicts only nine cells from the first interior row and column of the NW matrix. Validation crossing time still has an intermediate-data optimum. Colors denote exact-output thresholds, with solid-square validation and dashed-circle training curves. Each curve shows the recorded crossing time for an individual run at each dataset size.

J Additional NW Robustness Results

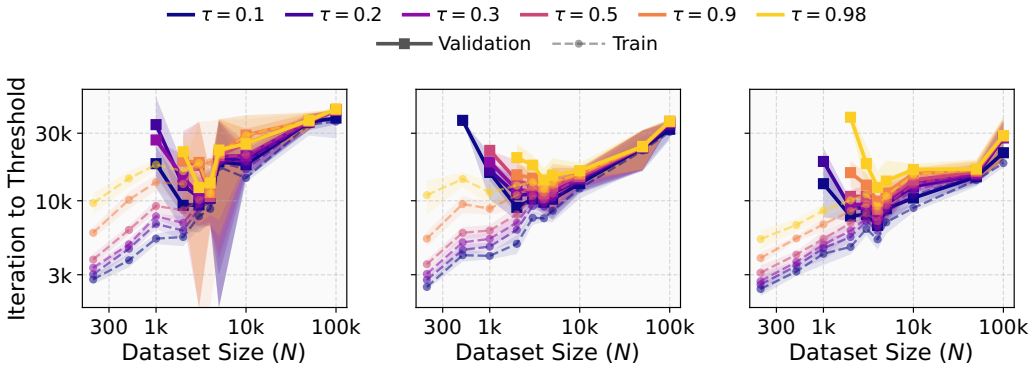
We extend the NW checks to an eight-layer model, cell accuracy, effective batch sizes 80 and 320, and weight decay 0, 0.01, and 0.03. The reference uses six layers, effective batch 160, and weight decay 0.1. Depth changes follow the model scaling in Appendix B, including the corresponding attention-head count and hidden width. The remaining comparisons change the indicated setting while keeping the reference architecture and other training settings fixed.



(a) Eight-layer model, exact matrix accuracy.

(b) Six-layer model, average cell accuracy.

Figure 18: NW capacity and measurement checks. An intermediate-data validation optimum remains visible with eight layers and when accuracy is averaged over matrix cells. Each panel has its own threshold legend because cell accuracy uses higher cutoffs. Solid squares show validation and dashed circles show training crossings, with shading showing one standard deviation.



(a) Effective batch 80.

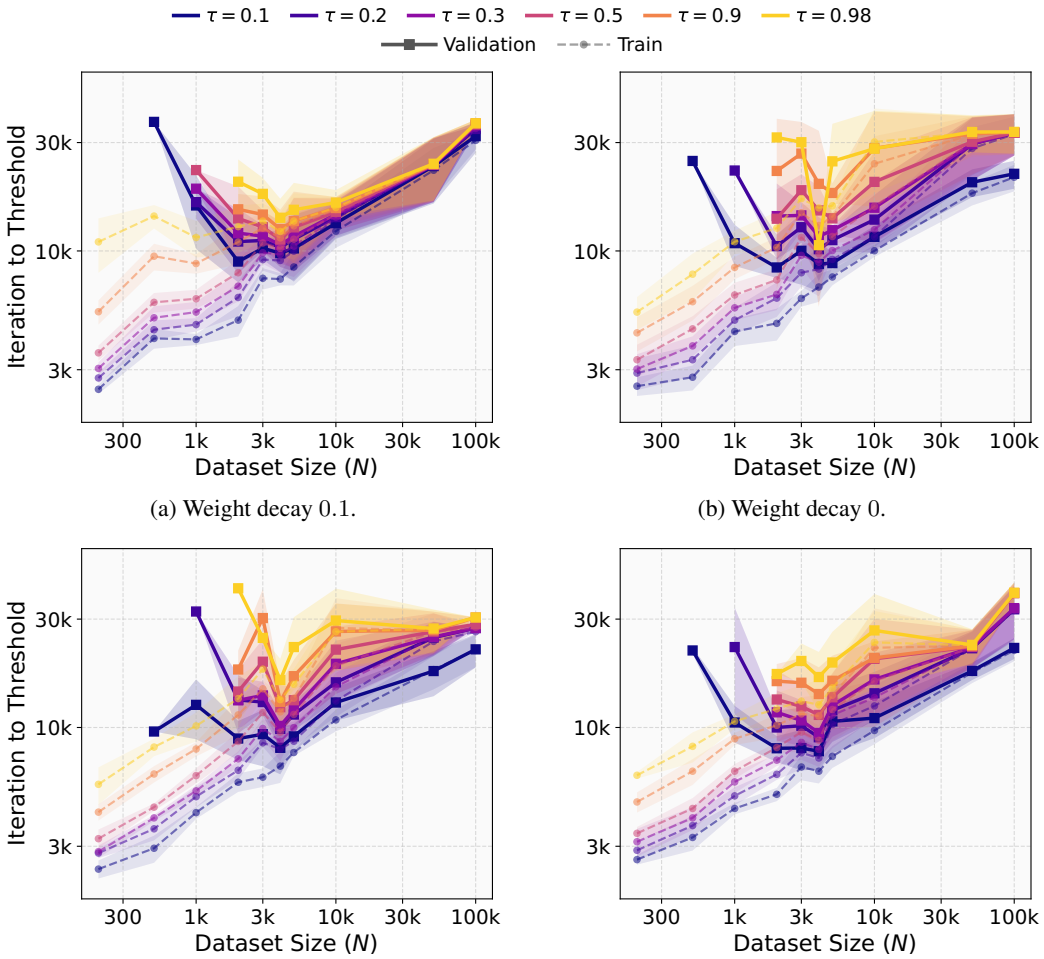
(b) Effective batch 160.

(c) Effective batch 320.

Figure 19: NW effective-batch comparison. Panels vary the number of examples per optimizer update across 80, 160, and 320, including gradient accumulation. The six-layer architecture and weight decay 0.1 are fixed. The intermediate-data optimum shifts, and validation still takes more updates at larger dataset sizes. Colors denote exact-matrix thresholds. Solid-square validation and dashed-circle training curves show means, with one-standard-deviation shading.

The eight-layer sweep uses three runs per dataset size. The cell-accuracy and optimization sweeps use five, while the reference has four recorded runs at $N = 10$k and five elsewhere. At each threshold, training and validation means are computed separately over runs with a recorded crossing. Shading shows one standard deviation of those crossing times. The number of contributing runs can vary across sizes, splits, and thresholds.

Across the batch-size and positive-weight-decay comparisons, validation curves rise again at larger dataset sizes, although the fastest size shifts. With zero weight decay, high-threshold crossings are recorded for fewer runs, especially at smaller sizes. The curves support slowdown under several optimization settings, with stronger evidence where the threshold is attained more consistently.



(a) Weight decay 0.1.

(b) Weight decay 0.

(c) Weight decay 0.01.

(d) Weight decay 0.03.

Figure 20: NW weight-decay comparison. The six-layer model and effective batch 160 are fixed, and panels vary weight decay across 0.1, 0, 0.01, and 0.03. Large-data delay appears in several settings, although the fastest size and threshold attainment change. Colors denote exact-matrix thresholds. Solid-square validation and dashed-circle training curves show means and one-standard-deviation shading across recorded crossings. High-threshold means with zero weight decay use fewer runs.

K Broader impact

This work is foundational and studies synthetic algorithmic datasets rather than deployed systems. Its main potential benefit is a clearer understanding of memorization–generalization dynamics under controlled conditions. We do not release pretrained models, scraped datasets, personal data, or user-facing systems. We therefore do not anticipate direct societal harms, although the general study of memorization may inform future work on privacy, robustness, and data efficiency.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the paper's main empirical claims and scope: NW matrix generation reaches high validation exact-match accuracy fastest at an intermediate dataset size, the result is update-based under a fixed optimization protocol, and multiplication is used as a contrast baseline. The paper also explicitly limits the claim to update-based convergence rather than asymptotic accuracy or epoch-normalized sample efficiency.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a Discussion and Limitations section. It states that the result should not be interpreted as a universal property of algorithmic tasks or as evidence that more data is generally harmful, and it discusses limits arising from the controlled NW setting, exact-match evaluation, small decoder-only Transformers, fixed-length inputs, and the lack of circuit-level identification.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper is empirical and does not present formal theorems or proof-based theoretical results. The equations are used to define empirical threshold-crossing times, and dataset-size summaries for organizing the observed dynamics rather than to state proved theoretical claims.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper specifies the synthetic NW task, scoring rule, finite task universe, dataset-size sweep, held-out validation protocol, model family, optimizer, learning-rate schedule, batch size, training budget, evaluation frequency, exact-match metric, threshold-crossing definitions, and censoring rule. Additional task and training details are provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The experiments use synthetically generated datasets, and the supplemental material includes anonymized code for data generation, model training, threshold-crossing evaluation, and plotting. The documentation describes the required environment and commands for reproducing the main figures.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 3 and the appendix specify the NW and multiplication tasks, train/validation construction, dataset-size sweep, Transformer architecture, optimizer, learning-rate schedule, weight decay, effective batch size, training budget, evaluation cadence, exact-match metric, and threshold/censoring definitions.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The appendix reports seed variability for the main multi-threshold timing experiments. Shaded regions are explicitly defined as one standard deviation across seeds, and the L=4 robustness appendix also reports seed variability and notes that exact threshold-crossing times vary across random seeds while the qualitative pattern remains visible.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the GPU type and memory, approximate runtime per run, and estimated total compute for the main dataset-size sweeps, depth sweeps, sequence-length robustness runs, and random-suffix probes.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

Justification: The research uses synthetic algorithmic datasets and small Transformer models, does not involve human subjects, personal data, private data, scraped datasets, or deployed systems, and is intended for scientific analysis of learning dynamics. We have reviewed the NeurIPS Code of Ethics and are not aware of any violations.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The paper is foundational research on synthetic algorithmic tasks and has no direct deployment pathway. Its positive impact is to clarify memorization–generalization dynamics in controlled settings. Potential negative impacts are limited because the paper does not release pretrained models, scraped datasets, private data, or user-facing systems.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk models, pretrained language models, image generators, scraped datasets, private datasets, or systems intended for deployment. The experiments use synthetic algorithmic data and small models.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the original works for the Transformer architecture and AdamW optimizer. The experiments use synthetically generated data rather than external datasets. The supplemental material documents the software libraries and any reused code assets together with their licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The supplemental material includes anonymized code for generating the synthetic NW and arithmetic datasets, training models, computing threshold-crossing times, and reproducing the main plots. The documentation describes the intended use, dependencies, limitations, and license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, human-subject experiments, participant instructions, or compensation.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects, crowdsourcing, personal data, or participant-facing studies, so IRB or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as an important, original, or non-standard component of the research methodology. Any use of language models for writing, editing, or formatting does not affect the scientific method, experiments, or claims.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.