LEARNING DYNAMICS OF CHAIN-OF-THOUGHT STATE TRACKING IN A SOLVABLE TRANSFORMER MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Chain-of-thought generation can turn a multi-step computation into a sequence of locally checkable state updates, but the training dynamics of such state tracking remain poorly understood. We study this question in a solvable setting: a simplified one-block transformer trained by supervised next-token prediction on state sequences generated by composing permutations. The architecture separates fixed-lag token retrieval from a specialized MLP logic module that applies the retrieved permutation to the current state. Using a statistical-physics mean-field description, we derive dynamics for three order parameters measuring correct-token retrieval in attention as well as on- and off-target logic alignment. These solutions quantitatively match simulations for the order parameters and, combined with a Gaussian approximation to the logit distribution, qualitatively predict the sharp transition in final rollout accuracy. The analysis reveals staged learning: the logic module first learns a mixed heuristic, after which attention locks onto the relevant tokens and efficient MLP alignment is enabled. The same mechanism emerges in standard one-block transformers and is missing when either subsystem is frozen at initialization. Together, these results provide a controlled mechanistic account of how attention-based retrieval and MLP-based logic co-develop during chain-of-thought state tracking.

1 INTRODUCTION

Transformers are highly successful across tasks such as translation, text classification, and question answering (Vaswani et al., 2017; Yang et al., 2019; Touvron et al., 2023; Le Scao et al., 2023; Liu et al., 2019). Chain-of-thought reasoning (Wei et al., 2022) has been a major recent advance: by inserting intermediate tokens that support step-wise answer production, large language models (LLMs) substantially improve on complex tasks (Wei et al., 2022). Alongside in-context learning (Brown et al., 2020; Von Oswald et al., 2023; Chan et al., 2022; Raventós et al., 2023), the sharp performance gain from chain-of-thought reasoning can be viewed as an emergent phenomenon – with respect to model scale in these studies and, as analyzed here, with respect to training time. This success has motivated extensive work in explainable AI (Elhage et al., 2021; Neo et al., 2024; Yun et al., 2020; Geva et al., 2021; Wang et al., 2022; Olsson et al., 2022; Bereska & Gavves, 2024; Sharkey et al., 2025; Chughtai et al., 2023) both through empirical analysis and analytically tractable, simplified settings. Prior work suggests that the attention block and multi-layer perceptron (MLP) inside a transformer serve two core functions: attention is mainly responsible for semantic and positional structure, while the MLP largely handles information storage and processing (Geva et al., 2021; Neo et al., 2024; Meng et al., 2022). Yet, the two mechanisms depend on each other, and their learning dynamics are coupled during training. This raises basic questions: how do attention and MLP influence one another during training, and when do they learn relative to each other?

In the present work, we disentangle these two core parts of a transformer. We let the attention mechanism handle positional information, deliberately without semantic complexity, and let the subsequent MLP act as an information storage and logic module. As a specific problem we employ a state-tracking task, in which the model sequentially composes permutations. This task has been used before to study length generalization of chain-of-thought computation in regular transformers (Huang et al., 2025). Due to its exact, discrete nature and its local state dependence on the prior

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

tokens, this task is well suited for detailed analysis. The sequential application of group elements can be solved by transformers (Marchetti et al., 2026; Chughtai et al., 2023; Huang et al., 2025), and in fact by a one-block transformer. In a student-teacher setup with an exact teacher, we fix selected parts of the architecture of a one-block transformer and obtain a solvable model that separates positional retrieval from logical composition learning. Our goal is not to model natural-language chain-of-thought reasoning in full-scale language models. We instead isolate the core of algorithmic chain-of-thought tasks: retrieving an operation from context and applying it to an evolving state. Fixed embeddings and values, together with the specialized MLP-style logic module, are deliberate simplifications that make the retrieval-logic coupling analytically tractable. The resulting model should therefore be read as a solvable mechanistic model of chain-of-thought state tracking, rather than as a complete model of standard transformer reasoning. To test the robustness of this picture, we relax these constraints individually and jointly in further experiments, finding the same qualitative ordering of learning stages.

Main results and contributions. We describe the learning process of a constrained one-block transformer for a permutation-composition task in a realizable teacher-student setup. We use three order parameters and derive differential equations for their dynamics. The student first learns a mixed heuristic in the MLP/logic block; afterwards, the attention head learns the relevant positional retrieval while the MLP logic promptly aligns with the teacher. The theory further predicts a steep increase in final rollout accuracy, qualitatively matching simulations. Experiments with architectural relaxations and with a standard one-block transformer support the mechanism of staged learning.

2 Related work

Explainable AI. Mechanistic interpretability aims to understand neural networks by reverse-engineering the algorithms they implement (Elhage et al., 2021; Neo et al., 2024; Yun et al., 2020; Geva et al., 2021; Wang et al., 2022; Olsson et al., 2022; Bereska & Gavves, 2024; Sharkey et al., 2025; Chughtai et al., 2023). Prior studies identified different roles for attention and the MLP, which perform contextual mappings and value mappings, respectively (Yun et al., 2020). Some works focus on attention heads (Elhage et al., 2021; Wang et al., 2022), others on factual knowledge stored in MLP layers (Meng et al., 2022), which act as key-value memories (Geva et al., 2021; Sukhbaatar et al., 2019). Neo et al. (2024) find attention heads that identify context cues relevant to next-token prediction and activate downstream neurons supporting that prediction.

Analytical models for transformers. Theoretical transformer models and limiting cases include a dynamical mean-field theory of large self-attention networks based on a mapping to asymmetric Hopfield networks (Poc-López & Aguilera, 2024; Treves & Amit, 1988), a solvable random-feature model of neural scaling laws (Maloney et al., 2022), and in-context learning in linear self-attention (Lu et al., 2025). Closer to our work, Yang et al. (2024) analyze gradient-flow training of a shallow transformer on word co-occurrence recognition and find two phases: a linear MLP first aligns with the target signals while softmax attention remains nearly static; later, both jointly increase the margin and drive the loss toward its minimum. Chen & Luo (2025) similarly argue for staged transformer optimization in a linearized setting, where attention parameters first condense toward task-relevant directions before key-query matrices participate more actively and normalized weights collapse toward low rank. In induction-head formation, training moves from a "lazy" local n-gram strategy to a richer in-context mechanism (Wang et al., 2025), again suggesting that distinct submodules enter the computation at different points in training. Our work is complementary: we study a chain-of-thought state-tracking task in which the model must maintain and update a latent state across steps, and we pair finite-width training runs with analytical predictions of the two-stage division of labor between attention-mediated retrieval and MLP-mediated logic update.

Attention and MLP dynamics. Tian et al. (2023) analyze attention during SGD training of a one-layer transformer, and Tian et al. (2024) report that MLP nonlinearity and residual structure shape how attention evolves in multilayer transformers, so that the two modules interact rather than learn independently. Empirically, feed-forward layers preferentially learn simple distributional associations such as bigrams, whereas attention layers are more closely tied to in-context reasoning (Chen et al., 2025), and over the course of pre-training, general-purpose components specialize, with FFNs

remaining comparatively stable and attention heads showing higher turnover (Hakimi et al., 2025). Both findings point to staged specialization rather than uniform co-development.

Reasoning theory and chain of thought. Transformers often solve complex tasks better when allowed to proceed step by step (Nye et al., 2021; Wei et al., 2022), and intermediate steps or explanations in context can further improve them (Zelikman et al., 2022; Lampinen et al., 2022). Prystawski et al. (2023) show that intermediate steps help when the training data consist of overlapping, strongly interacting local clusters of variables. In small transformers trained on iterative tasks, Cabannes et al. (2024) identify an "iteration head", an attention pattern that retrieves the current input token by its position so that the MLP can apply the state update, and study its emergence and transfer between tasks. Transformers can learn some reasoning tasks with far fewer layers than reasoning steps (Liu et al., 2022), but bounded-depth transformers cannot solve certain arithmetic tasks unless the model size grows super-polynomially with the input length (Feng et al., 2023); Takanami & Pehlevan (2026) address how to scale depth. Nadgir et al. (2026) model chain-of-thought reasoning as a tree process with a critical branching degree, below which reasoning hurts and above which an optimal tree depth minimizes the error. With chain of thought, constant-depth transformers can learn serial problems such as permutation composition (Li et al., 2024; Feng et al., 2023).

Permutation composition and state tracking. Many tasks require compositions of operations, e.g. mathematical or algorithmic. For sequential group composition, the required width and depth depend on the architecture (Marchetti et al., 2026). How small networks encode finite group operations is investigated by Chughtai et al. (2023) . Liu et al. (2022) show that composition tasks may be implemented through shortcuts, so that a network can solve the training task without learning the underlying rule; in our state-tracking task, we can directly follow which state the network tracks at every step, leaving little room for hidden shortcuts. Huang et al. (2025) prove that a one-layer transformer trained by gradient descent learns such state-tracking tasks with chain of thought and characterize length generalization through attention concentration. Prior work on these tasks thus addresses expressivity, learnability, length generalization, or reverse engineering of trained circuits, whereas we derive closed dynamics for macroscopic order parameters that describe how attention and the MLP-style logic module co-evolve, how the computation separates into retrieval and state update, and why the final rollout accuracy undergoes a sharp transition.

3 Problem and network setup

To isolate the model's ability to track latent states, we construct a synthetic environment that serves as a symbolic chain of thought and retains the core difficulty of serial reasoning: a sequence of intermediate steps must be carried out before the final answer is reached. A teacher with exact state tracking generates the data, and a student is trained to reproduce the resulting chain.

3.1 Permutation-chain task

Consider the first $N \geq 2$ positive integers $[N] := \{1, \dots, N\}$ and fix $d_g \leq N!$ permutations $\{\sigma_i\}_{i=1,\dots,d_g} \subseteq S_N$ acting on $[N]$. This subset forms the action vocabulary for the task; it need not be a group itself. We encode a sequence of permutations, called *actions*, and the resulting *states* as a chain of tokens that the network must reproduce. We draw $L \in \mathbb{N}$ actions (i.i.d.) from $\{\sigma_i\}_{i=1,\dots,d_g}$, represented by their indices $g_1, \dots, g_L \in \{1, \dots, d_g\}$, choose an initial state $s_0 \in [N]$, and recursively generate L states by applying the actions in chain form:

$$s_t = \sigma_{g_t}(s_{t-1}), \qquad t = 1, \dots, L. \tag{1}$$

This trajectory is concatenated into a single token sequence of total length $2L + 1$:

$$x_1 = g_1, \ \dots \ , \ x_L = g_L, \quad x_{L+1} = s_0, \quad x_{L+2} = s_1, \ \dots \ , \ x_{2L+1} = s_L. \tag{2}$$

Given the actions and the initial state, the student must generate the L subsequent states, ending with the final rollout token s_L: at inference, the prompt is the prefix $(g_1, \dots, g_L, s_0)$ and the output is the generated suffix $(s_1, \dots, s_L)$. By Cayley's theorem, any finite group admits a faithful representation by permutations, so this chain also covers the sequential application of group elements (Marchetti et al., 2026) and the state-tracking tasks with group actions studied by Huang et al. (2025).

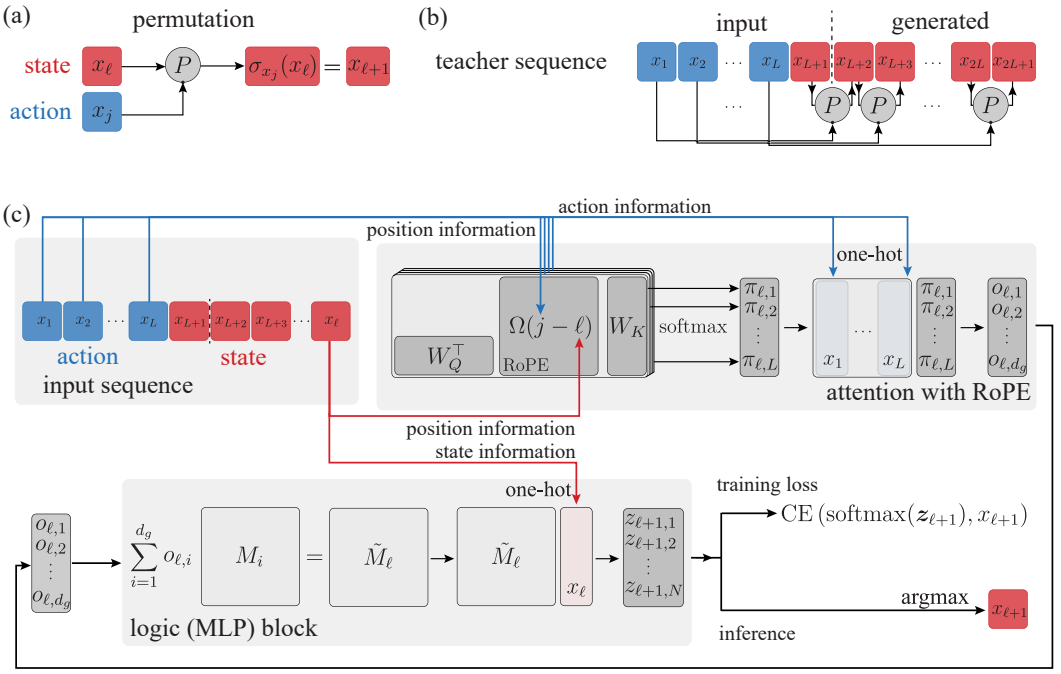

Figure 1: Task setup and specialized one-block transformer model. (a) Action token x_j selects permutation σ_{x_j} that maps the current state x_ℓ to $x_{\ell+1}$. (b) Teacher sequence from actions $x_1, \ldots, x_L$ and initial state x_{L+1}; for $i \in \{L+1, \ldots, 2L\}$, x_{i+1} is obtained by applying x_{i-L} to x_i. (c) Student prediction: RoPE attention retrieves the action at fixed lag $-L$, and the logic block applies the corresponding weighted sum of learnable matrices to the one-hot state, producing logits $z_{\ell+1}$.

As a toy example, let $N = 3$, $L = 3$, $s_0 = 3$, and $(g_1, g_2, g_3) = (2, 1, 2)$, with σ_1 switching $1 \leftrightarrow 2$ and σ_2 shifting $1 \to 2 \to 3 \to 1$. The generated states are $s_1 = \sigma_2(3) = 1$, $s_2 = \sigma_1(1) = 2$, $s_3 = \sigma_2(2) = 3$, and the full sequence is $(2, 1, 2 \mid 3, 1, 2, 3)$.

3.2 Network setup

We consider a minimal one-block transformer in which a modified attention head retrieves the relevant action at a fixed lag and passes it to an explicit logic module that performs the next-token prediction, thereby isolating retrieval from application. A single-layer transformer can encode the solution (Huang et al., 2025); we use this minimal depth to focus on the reasoning process through intermediate steps. Figure 1 summarizes the setup.

The attention head. A single attention head retrieves the relevant action from the prefix. We use Rotary Positional Encoding (RoPE) (Su et al., 2024), which has been found to perform better than absolute encodings, e.g., for context-free grammars (Allen-Zhu & Li, 2025), and fix the values in order to focus on positional information. We define

$$\mathbf{q_i} := \mathrm{RoPE}_i(W_Q\,\mathbf{u_i}) \in \mathbb{R}^{d_h}\,, \quad \mathbf{k_j} := \mathrm{RoPE}_j(W_K\,\mathbf{u_j}) \in \mathbb{R}^{d_h}\,, \quad \mathbf{v_j} := \mathrm{Val}(x_j) \in \mathbb{R}^{d_g}\,, \tag{3}$$

where d_g is the size of the action vocabulary, $d_h \in 2\mathbb{N}$ the head dimension, and $\mathbf{u_i}$ the embedding of token x_i. RoPE at sequence position i is the linear map $\mathrm{RoPE}_i(\mathbf{W}) := \Omega(i)\,\mathbf{W}$, where $\mathbf{W}$ is a query or key vector and $\Omega(i)$ is a matrix of 2D-block rotations. The query-key dot products obey

$$\big(\mathrm{RoPE}_i(\mathbf{W_Q})\big)^\top \big(\mathrm{RoPE}_j(\mathbf{W_K})\big) = \mathbf{W_Q}^\top\,\Omega(j-i)\mathbf{W_K}\,. \tag{4}$$

Thus, they depend only on the relative offset $j-i$, which makes RoPE suitable for fixed-lag retrieval. The values are one-hot features for action tokens and zero for state tokens, i.e., $\mathrm{Val}(g_j) = \hat{\mathbf{e}}_{\mathbf{g_j}} \in \mathbb{R}^{d_g}$ for an action and $\mathrm{Val}(x_j) = \mathbf{0}$ otherwise. To isolate positional from semantic information, we fix the embedding vectors to $\mathbf{u_i} = E(x_i) = 1 \in \mathbb{R}^1$ for all $i = 1, \ldots, 2L+1$, which reduces the key

and query matrices to vectors; Appendix A.3 shows how this arises from bundled embeddings and matrix-shaped keys and queries. Attention is causal, and we extend the mask to the state columns, for which the value projections are zero. The attention entries and the head output are

$$\pi_{ij} = \mathrm{softmax}_{j\le \min\{i,L\}}\left(\frac{\mathbf{q_i}^\top \mathbf{k_j}}{\sqrt{d_h}}\right), \qquad \mathbf{o_i} := \sum_{j\le i} \pi_{ij}\, \mathbf{v_j}\ . \tag{5}$$

This d_g-dimensional output assigns a weight to each possible action. Perfect learning corresponds to a one-hot output identifying the correct action among all d_g possibilities.

The logic module. The attention output is supplied to a logic module, a constrained instantiation of a GLU-style MLP (Appendix A.3), to predict the next state. We represent permutations by matrices, $O(\sigma_a)\,\mathbf{e_j} = \mathbf{e}_{\sigma_\mathbf{a}(\mathbf{j})}$ for $j \in [N],\ a \in [d_g]$, where $\{\mathbf{e_j}\}_{j=1}^N$ is the standard basis of $\mathbb{R}^N$, and introduce a set $\{M_a\}_{a=1}^{d_g}$ of learnable matrices $M_a \in \mathbb{R}^{N\times N}$. With attention outputs $\mathbf{o_i} \in \mathbb{R}^{d_g}$, we define

$$\tilde{M}_i := \sum_{a=1}^{d_g} (o_i)_a\, M_a\ \in \mathbb{R}^{N\times N}\ . \tag{6}$$

After learning, this map approximates the correct permutation matrix. Before convergence, it is a weighted sum of the learnable matrices in which actions identified by the attention head as relevant receive larger weights than irrelevant actions. The current state $x_i = s_{i-L-1}$ enters as an N-dimensional one-hot vector $\mathbf{e}_{\mathbf{s_{i-L-1}}} \in \mathbb{R}^N$, and the logits are defined as $\mathbf{z_{i+1}} := \tilde{M}_i\, \mathbf{e}_{\mathbf{s_{i-L-1}}} \in \mathbb{R}^N$. The prediction $\hat{s}_{i-L}$ is the index of the maximal entry.

This setup preserves the core functionality of a transformer block with an attention head followed by an MLP. During training, the model sees every intermediate token. At inference, only the prefix $(g_1, \ldots, g_L, s_0)$ is given and the states $s_1, \ldots, s_L$ must be generated, the final answer being the last state, so the student must carry out a chain of thought. Under teacher forcing, the current state is locally available at the predictor token, so a single fixed-lag retrieval of the relevant action suffices. A more general model could include a second attention head copying the previous state token. Such supervision of every intermediate token is how autoregressive models are trained on chain-of-thought data before any reinforcement training, thus we regard teacher forcing as a realistic rather restricted method. At inference, the model must work with its own predicted states.

Teacher model. The teacher employs the same architectural template as the student. Its attention head attends from predictor positions $j + L$ to action positions j, i.e., to the relative offset $-L$. This means $\pi^\star_{j+L,j} = 1$ for $j = 1, \ldots, L$ and zero everywhere else, which can be realized arbitrarily well by an appropriate choice of keys and queries. The head output is $\mathbf{o_i^\star} = \mathbf{v_{i-L}} \equiv \hat{\mathbf{e}}_{\mathbf{g_{i-L}}}$. The student is not forced to align with this fixed-lag teacher, but we did not observe any other accumulation points in the training runs that are presented below. In the teacher logic module, the matrices are the exact permutation matrices, $M^\star_a := O(\sigma_a) \in \mathbb{R}^{N\times N}$ for $a = 1, \ldots, d_g$. Consequently, the per-step map becomes $\tilde{M}^\star_i = M^\star_{g_{i-L}} = O(\sigma_{g_{i-L}})$. The teacher analog of the student logit is the target vector $\mathbf{p^\star_{i+1}} = O(\sigma_{g_{i-L}})\, \mathbf{e}_{\mathbf{s_{i-L-1}}} = \mathbf{e}_{\mathbf{s_{i-L}}}$, whose maximal entry marks the correct next state s_{i-L}.

3.3 Training objective

We train the model autoregressively with teacher forcing on full sequences 2, but compute the loss only on the suffix state tokens $s_1 = x_{L+2}, \ldots, s_L = x_{2L+1}$, using the cross-entropy loss

$$\mathcal{L} = \left\langle \frac{1}{L} \sum_{i=L+1}^{2L} -\ln P(x_{i+1} \mid x_1, \ldots, x_i) \right\rangle_{x_1,\ldots,x_{2L+1}}\ . \tag{7}$$

The current state token $x_{j+L} = s_{j-1}$ is used to predict the next one, for which the model needs the action token $x_j = g_j$ inside the logic module. For each training sequence, it can be reliably found at the relative offset $j - (j + L) = -L$, a single fixed lag independent of the position, and the prediction is performed at all L state positions. As a baseline, we choose $N = d_g = 32$, $L = 10$, and 32,768 training sequences; all hyperparameters and details of the dataset generation are given in Appendix A.2. The model need not see all $d_g^L N = 32^{11} \approx 3.6 \times 10^{16}$ possible sequences to learn

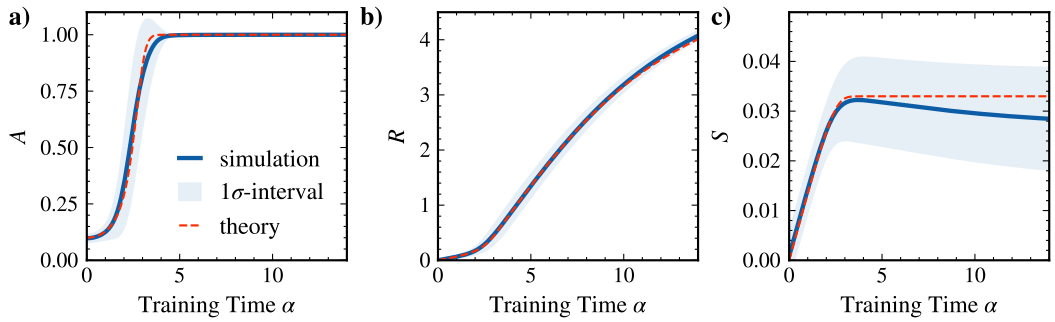

Figure 2: Dynamics of the order parameters A, R, and S in panels a), b), and c), respectively. Simulations are averaged over 100 seeds, and σ denotes the standard deviation. The theoretical predictions agree very well with the simulations (note different scale in panel c)).

the application of any action to any state at any position ($d_g NL = 10{,}240$ combinations). With 32,768 training sequences, almost certainly every action-state pair is observed during training, and generalization is tested on unseen sequences. Appendix C.1 reports sparse-coverage runs in which not all pairs are observed.

4 Learning Dynamics

4.1 Order Parameters and Mean-Field Loss

Order parameters. At prediction positions $i = L+1, \ldots, 2L$, the fixed-lag structure identifies the correct action position $i - L$. We therefore define

$$A := \frac{1}{L} \sum_{i=L+1}^{2L} \pi_{i,i-L} \,. \tag{8}$$

This attention order parameter describes the average attention mass on the correct action, as specified by the teacher, across all prediction steps in the chain. Since the attention entries are normalized, $A = 1$ corresponds to perfect retrieval at every step, $A = 1/L$ to random guessing if the entries are uniform, and $A \approx 0$ to actively avoiding the correct action. As logic order parameters, we define

$$R := \frac{1}{d_g} \sum_{a=1}^{d_g} \frac{1}{N} \operatorname{Tr}\Big[\big(M_a^\star\big)^\top M_a \Big] \,, \qquad S := \frac{1}{d_g(d_g-1)} \sum_{a=1}^{d_g} \sum_{\substack{b=1 \\ b \neq a}}^{d_g} \frac{1}{N} \operatorname{Tr}\Big[\big(M_a^\star\big)^\top M_b \Big] \,. \tag{9}$$

where $\{M_a\}_{a=1}^{d_g}$ and $\{M_a^\star\}_{a=1}^{d_g}$ are student and teacher matrices, respectively. R measures the alignment of the learned matrices with the corresponding teacher permutation matrices, and S the off-target overlap with the other teacher matrices. Both depend on the magnitude of the learnable matrices; Appendix C.1 shows that normalized overlaps give qualitatively the same results.

Mean-field loss and logits. With the next logits $\mathbf{z_{i+1}}$ at prediction position i, the cross-entropy loss is $\mathcal{L} = \big\langle \frac{1}{L} \sum_{i=L+1}^{2L} [-\mathbf{p^\star_{i+1}} \cdot \mathbf{z_{i+1}} + \ln(\sum_{r=1}^{N} \mathrm{e}^{z_{i+1,r}})] \big\rangle$, where the one-hot teacher vector $\mathbf{p^\star_{i+1}}$ singles out the correct logit $z_{i+1,k} = \mathbf{p^\star_{i+1}} \cdot \mathbf{z_{i+1}}$. To obtain an ensemble description, we make a mean-field ansatz and replace the correct and the other logits by their expectation values with respect to states and actions. By symmetry, the other logits behave identically on average, so that a single $z_{i+1,\overline{k}}$ describes any logit but the correct one, and the mean-field loss is

$$\ell_{\mathrm{CE}} := \frac{1}{L} \sum_{i=L+1}^{2L} \Big[-\langle z_{i+1,k} \rangle + \ln\Big(\mathrm{e}^{\langle z_{i+1,k} \rangle} + (N-1) \mathrm{e}^{\langle z_{i+1,\overline{k}} \rangle} \Big) \Big] \,. \tag{10}$$

In this way, the loss can be expressed through the order parameters and an initialization-dependent constant. By Jensen's inequality, the mean-field loss underestimates the full loss, with a leading

correction of second order in the logit fluctuations. Empirically, it describes both the single-seed and the seed-averaged loss very well (Appendix B.1). The expectation of the correct logit is taken over the local action and state. It reduces to an average over the independently and uniformly drawn actions and initial state, since permutation composition preserves the relevant statistics. In the ensemble, no step in the state sequence is preferred, so $\pi_{i,i-L} = A$ describes the relevant attention entries well, as also observed during training (Fig. 5 in the appendix). This yields the mean correct logit

$$\langle z_{i+1,k} \rangle = \left(A + \frac{1-A}{d_g} \right) R + \frac{d_g - 1}{d_g} (1-A) S \, . \tag{11}$$

The dependence on the sequence position i disappears after averaging. The mean other logits follow from the average total logit, independently of which state label is correct: the sum of the logits is conserved under cross-entropy training without weight decay, so that

$$\langle z_{i+1,\overline{k}} \rangle = -\frac{\langle z_{i+1,k} \rangle}{N-1} + \frac{1}{d_g N (N-1)} \sum_{a=1}^{d_g} \sum_{r,s=1}^{N} \left[M_a \right]_{rs} \, , \tag{12}$$

where the sum over the learnable matrices remains constant throughout training by the cross-entropy loss and is thus fixed by the initialization. As the mean correct logit grows during training, the remaining logits decrease, so the network progressively suppresses all incorrect classes.

4.2 Training dynamics

Training induces discrete updates of the microscopic network parameters along the negative gradient of the exact loss. Under the mean-field ansatz, a representative learnable matrix entry is updated as

$$[M_a]_{rs} \mapsto [M_a]_{rs} + \Delta [M_a]_{rs} \, , \qquad \Delta [M_a]_{rs} = -\eta \, \frac{\partial \ell_{\mathrm{CE}}}{\partial [M_a]_{rs}} \, , \tag{13}$$

with learning rate η and the mean-field loss ℓ_{CE} as a function of all network parameters. We define the *training time* $\alpha := \#\,\text{epochs}/(d_g N)$, so that one epoch corresponds to a small step $\Delta\alpha := 1/(d_g N)$. The mean-field loss allows us to translate these microscopic updates into dynamics for the macroscopic order parameters. To that end, we assume that the attention mass on the correct action is exchangeable across sequence positions ($\pi_{i,i-L} = A$), average arising teacher-teacher matrix overlaps over seeds, and use the empirical closure $d_g N [|\nabla_{\mathbf{W_Q}} A|^2 + |\nabla_{\mathbf{W_K}} A|^2] = L c_\omega A(1-A)$ for the attention-gradient factor. Here,

$$c_\omega := \frac{2 d_g N}{L(L-1)} \, \sigma^2 \cdot \frac{2}{d_h} \sum_{n=1}^{d_h/2} \left[1 - \frac{1}{L^2} \left(\frac{\sin(\omega_n L/2)}{\sin(\omega_n/2)} \right)^2 \right]^2 \tag{14}$$

is determined by the initialization. Details and reasons for the closure are given in Appendix B.3. Here, σ^2 is the variance of the key/query initialization (a default uniform initialization $\mathcal{U}(-d_{\mathrm{model}}^{-1/2}, +d_{\mathrm{model}}^{-1/2})$ gives $\sigma^2 = 1/3$ with $d_{\mathrm{model}} = 1$) and $\{\omega_n\}_{n=1}^{d_h/2}$ are the RoPE frequencies. Consequently, ΔA, ΔR, and ΔS become functions of the order parameters themselves and the network hyperparameters. We then consider the regime of the thermodynamic limit $d_g, N, L, d_h \to \infty$ while keeping d_g/N, N/L, and L/d_h finite. In this limit, $\Delta\alpha \to 0$, which allows us to pass from difference quotients to continuous derivatives, e.g., from $\Delta S/\Delta\alpha$ to $\mathrm{d}S/\mathrm{d}\alpha$, provided they converge uniformly on compact time intervals (Hairer et al., 1993). Here, this is consistent with the finite-ratio assumptions on the hyperparameters and with the initial conditions $A(0) = 1/L$ and $R(0) = S(0) = 0$. In the limiting regime and up to fluctuations, the macroscopic order parameters satisfy, under this mean-field closure,

$$\frac{\mathrm{d}A}{\mathrm{d}\alpha} = \eta \, \frac{1}{1 + \frac{1}{N-1} \, \mathrm{e}^{m(A,R,S)}} \, \frac{N}{N-1} \, \frac{d_g - 1}{d_g} \, c_\omega \cdot L A (1-A)(R-S) \, , \tag{15a}$$

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \eta \, \frac{1}{1 + \frac{1}{N-1} \, \mathrm{e}^{m(A,R,S)}} \cdot \left(A + \frac{1-A}{d_g} \right) , \tag{15b}$$

$$\frac{\mathrm{d}S}{\mathrm{d}\alpha} = \eta \, \frac{1}{1 + \frac{1}{N-1} \, \mathrm{e}^{m(A,R,S)}} \cdot \frac{1-A}{d_g} \, , \tag{15c}$$

where $m(A, R, S) = \frac{1}{L}\sum_{i=L+1}^{2L}(\langle z_{i+1,k}\rangle - \langle z_{i+1,\overline{k}}\rangle)$ is the logit margin, expressed through Eqs. 11 and 12. Figure 2 compares empirical results with numerical solutions of these equations. Attention shows a clear transition: for $\alpha \lesssim 1$, the parameter A remains near its initial value; later, it rises rapidly to saturation. In the same regimes, R first increases slowly and then quickly aligns with the teacher once the attention head has started learning, while the off-target overlap S increases relatively quickly at first, on a much smaller scale than R, and saturates once attention is nearly perfect. Since the prediction is the maximal argument of the logits, and not drawn by probability, perfect alignment is not needed to predict the next token correctly.

Interpretation. During early training, the attention output weights the actions in the sequence nearly uniformly, so the logic module cannot yet apply the correct action to the current state. Instead, a mixed heuristic is learned: the next state is predicted by applying a mixture of actions, the student matrices acquire relevant as well as irrelevant directions, and both the correct overlap R and the off-target overlap S increase. Once a large enough margin $R - S$ has been established, the attention head begins to identify the correct action token, since $\mathrm{d}A/\mathrm{d}\alpha \propto R - S$. This, in turn, reinforces the teacher-student alignment, because the relevant directions can now be learned effectively. Such an MLP-first regime is consistent with gradient-flow analyses of transformer training in which task-relevant MLP directions emerge before attention has appreciably specialized (Yang et al., 2024).

5 Final rollout accuracy

Ultimately, the model should predict the final token correctly, that is, carry out the chain of thought without errors along the way. Here we model the accuracy of the final token. At each sequence step, the model predicts either the correct next state or an incorrect one, so the probability of a correct prediction obeys the recursion

$$P(\hat{s}_{t+1} = s_{t+1}) = P(\hat{s}_t = s_t)\cdot P(\hat{s}_{t+1} = s_{t+1}|\hat{s}_t = s_t) + \big[1 - P(\hat{s}_t = s_t)\big]\cdot P(\hat{s}_{t+1} = s_{t+1}|\hat{s}_t \neq s_t)\,. \tag{16}$$

Since the initial token is given, we have $P(\hat{s}_0 = s_0) = 1$. In principle, the model can recover the correct token even from an incorrect current state, although this matters less as training progresses. Both conditional probabilities in Equation 16 depend on the logit distributions, which are empirically close to normal at initialization and in early training. Assuming that the correct logit follows the Gaussian distribution $\mathcal{N}(\mu_k, \sigma_k^2)$ and each incorrect logit $\mathcal{N}(\mu_{\overline{k}}, \sigma_{\overline{k}}^2)$ across the sequence, the next-token accuracy is $P(\hat{s}_{t+1} = s_{t+1}|\hat{s}_t = s_t) = \rho$ for any $t \in \{0, \ldots, L-1\}$, with

$$\rho = \int_{-\infty}^{+\infty} \mathrm{d}u\, \frac{\mathrm{e}^{-u^2/2}}{\sqrt{2\pi}}\, \Phi\!\left(\frac{\sigma_k}{\sigma_{\overline{k}}}\,u + \frac{\mu_k - \mu_{\overline{k}}}{\sigma_{\overline{k}}}\right)^{N-1}. \tag{17}$$

Furthermore, if the incorrect logits are symmetric in that they have no preference among themselves, the recovery term $P(\hat{s}_{t+1} = s_{t+1} \mid \hat{s}_t \neq s_t)$ can be written as $(1-\rho)/(N-1)$ and the final rollout accuracy becomes

$$P(\hat{s}_L = s_L) \approx \frac{N-1}{N}\left(\frac{N\rho - 1}{N-1}\right)^L + \frac{1}{N}\,. \tag{18}$$

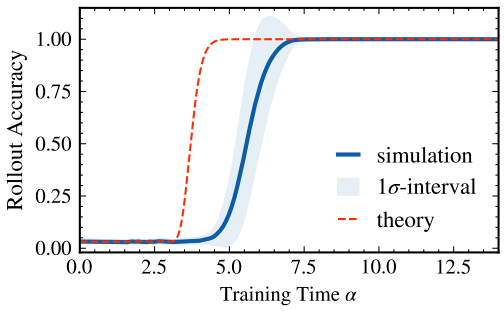

Figure 3: Final rollout accuracy, averaged over 100 model seeds; σ denotes the standard deviation. The theoretical curve uses μ_k and $\mu_{\overline{k}}$ from the solutions of Eqs. 15 and constant variances $\sigma_k^2 = \sigma_{\overline{k}}^2 \approx 0.044$ from initialization. The predicted rise occurs slightly too early because the empirical variances increase during training which introduces more uncertainty in the predictions and slows training; see Appendix B.6. Using the empirical variances, the predicted transition matches much better.

This expression is approximate, because training introduces correlations between the correct and incorrect logits, whereas the derivation assumes independence, and because the logit distributions

deviate from the Gaussian approximation as training proceeds. Even so, Equation 18 captures the qualitative form of the transition already for constant variances $\sigma_k^2, \sigma_{\bar{k}}^2$; see Figure 3. The rollout accuracy remains near its initial plateau of $1/N$, then rises sharply and saturates at unity. The reason is clear from Equation 18: the next-token accuracy enters to the L-th power, so even a gradual increase in one-step accuracy is converted into a sudden jump in the final rollout accuracy.

6 Further empirical tests of the staged-learning mechanism

The analytical predictions refer to the constrained model above. We relax the constraints one at a time and together, using a standard causal attention mask including the state columns, learnable embeddings, a learnable value matrix instead of one-hot values, and a regular MLP instead of the reduced logic module. The staged dynamics persist in all cases and in a size-comparable standard one-block transformer with residual connections and layer normalizations, trained with AdamW and a learning rate schedule (Appendix C.2). We measure logic alignment by probing the MLP with a perfectly retrieved action and the current state, yielding effective student matrices and analogs of R and S (Appendix C.2). While A remains on its initial plateau, the R and S probes already increase. The probes then rise or fall sharply as attention concentrates on the fixed-lag action position. The same ordering appears with two attention heads (A is the head average).

In further experiments, we freeze the attention matrix at its uniform initialization. The R and S probes still increase, but the final rollout accuracy remains near random-guessing level (Appendix C.3). With the MLP frozen at initialization instead, A rises modestly above its initial plateau, but the final rollout accuracy again remains near random-guessing level. The two subsystems thus depend on each other during learning: with attention frozen, the MLP lacks the retrieved permutation, while with the MLP frozen, attention does not reliably identify the relevant action.

7 Discussion

Summary. We studied a simplified transformer model for a compositional permutation task from a statistical-physics viewpoint, introducing macroscopic order parameters that describe the learning dynamics of the attention head and the logic module – a specialized MLP. The model first learns a heuristic, mixing relevant and irrelevant directions, before attention learns to retrieve the relevant action and the logic module aligns with the teacher. The order in which the subsystems learn is not known a priori from the construction. The two stages emerge from the coupled dynamics and are confirmed by training simulations. The dynamical equations provide a causal explanation for this sequence: the margin between on-target and off-target logic overlaps serves as a gating mechanism for efficient attention learning. We also derived a description of the final-token accuracy that qualitatively reproduces the observed sudden increase and traces it to the compositional structure of the task. The staged dynamics are also observed in models with relaxed constraints and in a standard one-block transformer. Taken together, these results clarify the roles of attention and the logic module in a controlled state-tracking problem.

Limitations. The architecture of the main model studied here is deliberately constrained and therefore does not represent a standard one-block transformer. Whether the same learning dynamics occur in full-scale transformers remains an open question. Our analytical results for the constrained model rely on a mean-field approximation to the loss and a Gaussian approximation to the logits used to predict final rollout accuracy. The accuracy of these approximations may differ in other models or on other tasks. With learnable embeddings and plain SGD, the model does not reliably predict the final rollout token within the training times studied, possibly because the learning rate schedule was tuned to fixed embeddings.

Outlook Future work should determine how far the present picture extends to transformer architectures at practical scale and depth, how robust the staged learning dynamics are, whether they extend to other tasks or natural language, and how reinforcement learning after pretraining affects them.

AI USE STATEMENT

In this work, we used generative AI tools for helping develop theoretical models or conceptual frameworks (establish a base model), providing critical ingredients for proving mathematical claims (outside the main results), and designing or providing feedback on research methodology or experiments (feedback on derivations and simulations). We have not used generative AI tools for generating synthetic data sets, formulating mathematical claims, assisting in the writing of proofs, assisting with translation, or interpreting results.

Proposing or refining hypotheses, implementing methods, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis are not applicable to this work.

Additionally, we used generative AI tools to suggest experimental parameters (model hyperparameters), create or edit software code (for simulations and plotting), draft parts of a research paper (suggesting prose), summarize or analyze existing literature, source/search for information, identify relevant literature, suggest a structure for a research paper, and propose a title or keywords for a research paper.

We have reviewed all AI-assisted work: we checked the LLM-generated research idea for potential plagiarism through a manual literature survey, verified and tested LLM-generated code for correctness, and checked any prose suggestions for appropriateness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We do not believe that ethical concerns are raised by this work.

REPRODUCIBILITY STATEMENT

The code for the simulations and plots is available in the supplementary material. Used hyperparameters are stated in Appendix A.1, descriptions of training and dataset generation are given in Appendix A.2. All analytical derivations can be found in Appendix B. Further experiments are described in Appendix C.

REFERENCES

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 1, learning hierarchical language structures. *arXiv preprint arXiv:2305.13673*, 2025.

Leonard Bereska and Efstratios Gavves. Mechanistic interpretability for ai safety–a review. *arXiv preprint arXiv:2404.14082*, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Vivien Cabannes, Charles Arnal, Wassim Bouaziz, Alice Yang, Francois Charton, and Julia Kempe. Iteration head: A mechanistic study of chain-of-thought. *arXiv preprint arXiv:2406.02128*, 2024. URL https://arxiv.org/abs/2406.02128.

Stephanie CY Chan, Ishita Dasgupta, Junkyung Kim, Dharshan Kumaran, Andrew K Lampinen, and Felix Hill. Transformers generalize differently from information stored in context vs in weights. *arXiv preprint arXiv:2210.05675*, 2022.

Lei Chen, Joan Bruna, and Alberto Bietti. Distributional associations vs in-context reasoning: A study of feed-forward and attention layers. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WCVMqRHWW5.

Zheng-An Chen and Tao Luo. From condensation to rank collapse: A two-stage analysis of transformer training dynamics. *arXiv preprint arXiv:2510.06954*, 2025.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. A toy model of universality: Reverse engineering how networks learn group operations. In *International Conference on Machine Learning*, pp. 6243–6267. PMLR, 2023.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36:70757–70798, 2023.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021.

Ernst Hairer, Syvert Paul Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin / New York, 2 edition, 1993.

Ahmad Dawar Hakimi, Ali Modarressi, Philipp Wicke, and Hinrich Schuetze. Time course MechInterp: Analyzing the evolution of components and knowledge in large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-acl.654/.

Yu Huang, Zixin Wen, Aarti Singh, Yuejie Chi, and Yuxin Chen. Transformers provably learn chain-of-thought reasoning with length generalization. *arXiv preprint arXiv:2511.07378*, 2025.

Andrew Lampinen, Ishita Dasgupta, Stephanie Chan, Kory Mathewson, Mh Tessler, Antonia Creswell, James McClelland, Jane Wang, and Felix Hill. Can language models learn from explanations in context? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 537–563, 2022.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*, 2023.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to solve inherently serial problems. In *International Conference on Learning Representations*, 2024.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. Technical report, 2019. URL https://arxiv.org/abs/1907.11692.

Yue M Lu, Mary Letey, Jacob A Zavatone-Veth, Anindita Maiti, and Cengiz Pehlevan. Asymptotic theory of in-context learning by linear attention. *Proceedings of the National Academy of Sciences*, 122(28):e2502599122, 2025.

Alexander Maloney, Daniel A Roberts, and James Sully. A solvable model of neural scaling laws. *arXiv preprint arXiv:2210.16859*, 2022.

Giovanni Luca Marchetti, Daniel Kunin, Adele Myers, Francisco Acosta, and Nina Miolane. Sequential group composition: A window into the mechanics of deep learning. *arXiv preprint arXiv:2602.03655*, 2026.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

Amrut Nadgir, Vijay Balasubramanian, and Pratik Chaudhari. How does chain of thought decompose complex tasks? *arXiv preprint arXiv:2604.08872*, 2026.

Clement Neo, Shay B Cohen, and Fazl Barez. Interpreting context look-ups in transformers: Investigating attention-mlp interactions. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16681–16697, 2024.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Michael D. Perlman. Jensen's inequality for a convex vector-valued function on an infinite-dimensional space. *Journal of Multivariate Analysis*, 4(1):52–65, 1974. ISSN 0047-259X. doi: https://doi.org/10.1016/0047-259X(74)90005-0.

Ángel Poc-López and Miguel Aguilera. Dynamical mean-field theory of self-attention neural networks. *arXiv preprint arXiv:2406.07247*, 2024.

Ben Prystawski, Michael Li, and Noah Goodman. Why think step by step? reasoning emerges from the locality of experience. *Advances in Neural Information Processing Systems*, 36:70926–70947, 2023.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. *Advances in neural information processing systems*, 36:14228–14246, 2023.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, et al. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*, 2025.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*, 2019.

Kaito Takanami and Cengiz Pehlevan. An asymptotic theory of chain-of-thought in in-context learning, 2026. URL https://arxiv.org/abs/2606.03217.

Yuandong Tian, Yiping Wang, Beidi Chen, and Simon Du. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. *arXiv preprint arXiv:2305.16380*, 2023. URL https://arxiv.org/abs/2305.16380.

Yuandong Tian, Yiping Wang, Zhenyu Zhang, Beidi Chen, and Simon Shaolei Du. JoMA: Demystifying multilayer transformers via joint dynamics of MLP and attention. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=LbJqRGNYCf.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Alessandro Treves and Daniel J Amit. Metastable states in asymmetrically diluted hopfield networks. *Journal of Physics A: Mathematical and General*, 21(14):3155–3169, 1988.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

Mingze Wang, Ruoxi Yu, Weinan E, and Lei Wu. How transformers get rich: Approximation and dynamics analysis. *arXiv preprint arXiv:2410.11474*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Hongru Yang, Bhavya Kailkhura, Zhangyang Wang, and Yingbin Liang. Training dynamics of transformers to recognize word co-occurrence via gradient flow analysis. *arXiv preprint arXiv:2410.09605*, 2024.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 15476–15488. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/639a9a172c044fbb64175b5fad42e9a5-Paper-Conference.pdf.

A Reproducibility and model details

A.1 Hyperparameters

Table 1: Hyperparameters for the training simulations.

Variable	Value	Description
N	32	Magnitude of permutation set $\{1, \ldots, N\}$
L	10	Sequence steps (actions $g_1, \ldots, g_L$; predicted states $s_1, \ldots, s_L$)
d_g	32	Vocabulary size of permutations
d_{model}	1	Embedding dimension
d_h	128	RoPE encoding dimension (key & query dimension)
$n_{\text{train_seq}}$	32,768	Number of training examples
$n_{\text{test_seq}}$	256	Number of test examples
batch_size	32,768	Batch size for gradient descent (full-batch if equal to $n_{\text{train_seq}}$)
lr	5×10^{-1}	Learning rate
weight_decay	0	Weight decay (Adam optimizer)
rope_theta	10,000	RoPE frequency base (power-law distributed)
train_epochs	20,000	Number of training epochs
eval_every	50	Evaluation frequency (in epochs)
seeds	100	Number of seeds (one model per seed)
optimizer	`sgd`	Optimizer type (`sgd` or `adamw`)
fix_embedding	`True`	Restrict embedding vectors to unity

Table 1 lists all hyperparameters used for the chief simulations. The RoPE frequencies are power-law distributed and set as $\omega_n := 2\pi\,\theta_{\text{RoPE}}^{-2(n-1)/d_h}$ with θ_{RoPE} (`rope_theta`) as the base and $n = 1, \ldots, d_h/2$.

A.2 ARCHITECTURE SETUP

For the model training, we use `PyTorch`. For the sake of reproducibility, we sketch the numeric setup as follows:

- Set seed with `numpy`.
- Set RoPE frequencies.
- Draw permutation vocabulary (d_g) using `rng.permutation(N)` with the default `numpy` random number generator of the seed, then create the permutation matrices.
- Generate dataset (sequences):
  - Draw L actions from the permutations and one initial index in $[N]$.
  - Build the teacher token sequence by sequential application of the permutations on the current state.
- Define class with `pytorch` using `nn.Module`.
- Initialize two linear layers (keys and queries) and a parameter tensor for the learnable matrices uniformly.
- Forward function:
  - Embed the token, compute key and query, apply RoPE.
  - Compute attention matrix (causal and restricted to actions).
  - Compute attention output with Values.
  - Compute logic module output with learnable matrices and current token.
- Train the model:
  - Use train and test data loader for the dataset.
  - Set optimizer (here `SGD` with batch size equal to train set size)
  - For each epoch, predict the tokens, compute the loss, apply the optimizer, backwards propagation, and optimizer step.
  - Evaluate every few epochs the train/test accuracies, the order parameters, and other computable quantities.

The code is available in the supplementary material. To train the model, execute the Python script with command arguments `--seed <int> --dir "str"` for one seed, the output is saved as a numpy file in the given directory. The hyperparameters are set within the script's code. To inspect the data, compute the theoretical predictions, and generate the figures displayed in this work, use the Python Notebook.

For training, several GPUs and a cluster were used. A T4 GPU takes 30 minutes to train one model (one seed), an NVIDIA Geforce RTX 4060 around 20 minutes. We used seeds 0 to 99.

Many hyperparameter combinations were tested, hence more computational time was occupied than strictly necessary for the results shown here. The model learns robustly for changes in the hyperparameters N, d_g, L, d_h as well as the learning rate, RoPE frequency base and number of training examples.

Variations in d_g, N (between 8 and 32) or L (between 5 and 20) or d_h (between 8 and 128) also produce fully learned models. Fewer frequencies (d_h) limit the learnability, as do small embedding dimensions ($d_{\text{model}} \lesssim 128$) if learnable embeddings are turned on. Linearly distributed frequencies yield a similar learning behavior as the power-law distributed ones used here. Appendix C.5 shows figures for another set of hyperparameters than used in the main text.

A.3 REALIZING THE LOGIC MODULE AS A CONSTRAINED MLP

The logic module used in the main text differs from the feed-forward block of a standard transformer because the module output is written directly as a weighted sum of learnable matrices. Nevertheless, this computation can be implemented by a GLU-style MLP once the MLP input is arranged so that the first d_g coordinates contain the action information $\mathbf{o} \in \mathbb{R}^{d_g}$ coming from attention and the last

N coordinates contain the current state $\mathbf{s} \in \mathbb{R}^N$. The state vector is one-hot, whereas $\mathbf{o}$ may contain soft attention outputs.

Ignoring sequence indices, the desired logic computation is

$$\mathbf{z} := \sum_{a=1}^{d_g} o_a M_a \mathbf{s} = \begin{pmatrix} M_1, \dots, M_{d_g} \end{pmatrix} \begin{pmatrix} o_1 \mathbf{s} \\ \vdots \\ o_{d_g} \mathbf{s} \end{pmatrix} , \tag{19}$$

where $M_a \in \mathbb{R}^{N \times N}$, $a = 1, \dots, d_g$, are the learnable action matrices. We now show that the vector in the second factor can be produced by fixed up- and gate-projections.

Consider a GLU-style MLP without biases,

$$\mathbf{z}_{\text{MLP}} := W_d \left[W_u \begin{pmatrix} \mathbf{o} \\ \mathbf{s} \end{pmatrix} \odot \phi\left(W_g \begin{pmatrix} \mathbf{o} \\ \mathbf{s} \end{pmatrix} \right) \right] . \tag{20}$$

We take the activation to be the identity, $\phi(x) = x$. The up-projection is fixed to copy the state vector into d_g blocks,

$$W_u := \begin{pmatrix} 0_{N \times d_g} & I_N \\ \vdots & \vdots \\ 0_{N \times d_g} & I_N \end{pmatrix} \in \mathbb{R}^{d_g N \times (d_g + N)} , \qquad W_u \begin{pmatrix} \mathbf{o} \\ \mathbf{s} \end{pmatrix} = \begin{pmatrix} \mathbf{s} \\ \vdots \\ \mathbf{s} \end{pmatrix} . \tag{21}$$

The gate projection is fixed to copy each action coordinate over one state-sized block. Let $T_a \in \mathbb{R}^{N \times d_g}$ denote the matrix whose a-th column is all ones and whose other entries are zero. Let $\mathbf{1_N} \in \mathbb{R}^N$ denote the all-ones vector. Set

$$W_g := \begin{pmatrix} T_1 & 0_{N \times N} \\ \vdots & \vdots \\ T_{d_g} & 0_{N \times N} \end{pmatrix} \in \mathbb{R}^{d_g N \times (d_g + N)} , \qquad W_g \begin{pmatrix} \mathbf{o} \\ \mathbf{s} \end{pmatrix} = \begin{pmatrix} o_1 \mathbf{1}_N \\ \vdots \\ o_{d_g} \mathbf{1}_N \end{pmatrix} . \tag{22}$$

Therefore, multiplying the two projections of Eqs. 21 and 22 element wise gives the rightmost vector of Equation 19. Finally, choose the down-projection as $W_d := \begin{pmatrix} M_1, \dots, M_{d_g} \end{pmatrix} \in \mathbb{R}^{N \times d_g N}$. With this choice, Eqs. 19–22 imply $\mathbf{z}_{\text{MLP}} = \mathbf{z}$ for the linear activation $\phi(x) = x$. Thus, the specialized logic module is representable as a GLU-style MLP with fixed up-projection and gate-projection weights and learnable down-projection blocks.

Embeddings. Similarly to the logic module, the attention head can be implemented from a default attention architecture. With the bundled embedding structure introduced above, i.e., a $(d_g + N)$-dimensional embedding with one-hot vectors for the actions and states, we define key and query matrices $W_K, W_Q \in \mathbb{R}^{d_h \times (d_g + N)}$ as

$$W_K = \begin{pmatrix} (W_K)_1 & \cdots & (W_K)_1 \\ \vdots & & \vdots \\ (W_K)_{d_h} & \cdots & (W_K)_{d_h} \end{pmatrix} , \quad W_Q = \begin{pmatrix} (W_Q)_1 & \cdots & (W_Q)_1 \\ \vdots & & \vdots \\ (W_Q)_{d_h} & \cdots & (W_Q)_{d_h} \end{pmatrix} . \tag{23}$$

The first column is repeated $d_g + N$ times. In that regard, the weight matrix reduces to one learnable vector. Then, any embedded vector $\mathbf{u_i} \in \mathbb{R}^{d_g + N}$ is one-hot, because it is either an action or a state. This yields

$$W_K \begin{pmatrix} \mathbf{g} \\ \mathbf{0}_N \end{pmatrix} = \begin{pmatrix} (W_Q)_1 \\ \vdots \\ (W_Q)_{d_h} \end{pmatrix} , \qquad W_K \begin{pmatrix} \mathbf{0}_{d_g} \\ \mathbf{s} \end{pmatrix} = \begin{pmatrix} (W_K)_1 \\ \vdots \\ (W_K)_{d_h} \end{pmatrix} , \tag{24}$$

and equally for W_Q, for any one-hot encoded action $\mathbf{g}$ or state $\mathbf{s}$. This is completely equivalent to choosing a unit embedding and key/query vectors such that $\mathbf{W_K u_i} = \mathbf{W_K}$ with $\mathbf{u_i} = 1 \in \mathbb{R}^1$. Consequently, we may write the keys and queries as vectors and omit the embedding in the attention head.

B ANALYTICAL DERIVATIONS

B.1 MEAN-FIELD LOSS CORRECTIONS

The true loss contains a function of the form

$$f : \mathbb{R}^N \longrightarrow \mathbb{R}, \qquad f(z) = \ln\left(\sum_{r=1}^{N} e^{z_r}\right) \tag{25}$$

which can be shown to be convex, i.e., for every $x, y \in \mathbb{R}^N$ and every $t \in [0, 1]$,

$$f\big((1-t)x + ty\big) \leq (1-t)f(x) + tf(y)\,. \tag{26}$$

The decisive reason is that the Hessian of f is positive semi-definite. Because f is convex, Jensen's inequality for a vector-valued function (Perlman, 1974) can be employed,

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]\,, \tag{27}$$

where both the expectations of the random variable X and of $f(X)$ are finite in magnitude. In our case,

$$\log\left(\sum_{r} e^{\mathbb{E}[Z_r]}\right) \leq \mathbb{E}\left[\log\left(\sum_{r} e^{Z_r}\right)\right] \tag{28}$$

such that $\ell_{\mathrm{CE}} \leq \mathcal{L}$. We abbreviate $\mu = \mathbb{E}[Z]$, $Z = \mu + \delta$, then write

$$\Delta = \mathbb{E}[f(\mu + \delta)] - f(\mu) \tag{29}$$

and find

$$\Delta = \mathbb{E}\left[\frac{1}{2}\sum_{r=1}^{N} p_r h_r^2\right] + \mathcal{O}(h_r^3) =: \Delta_2 + \mathcal{O}(h_r^3) \tag{30}$$

where

$$p_r = \frac{e^{\mu_r}}{\sum_{s=1}^{N} e^{\mu_s}}\,, \qquad h_r = \delta_r - \sum_{r=1}^{N} p_r \delta_r\,. \tag{31}$$

Using the covariances defined by

$$\Sigma_{rs} := \mathbb{E}[(Z_r - \mu_r)(Z_s - \mu_s)] = \mathbb{E}[\delta_r \delta_s]\,, \tag{32}$$

the leading correction term, in fact a second-order correction, to the mean-field loss is

$$\Delta_2 = \frac{1}{2}\left[\sum_{r} p_r \Sigma_{rr} - p^\top \Sigma p\right] = \frac{1}{2}\,\mathrm{Tr}\big[(\mathrm{diag}(p) - pp^\top)\Sigma\big]\,. \tag{33}$$

Then

$$\mathcal{L} - \ell_{\mathrm{CE}} = \frac{1}{L}\sum_{i=L+1}^{2L} \Delta_2^{(i)} \tag{34}$$

where i indexes each sequence step. The evolution of full loss and mean-field loss are shown in Figure 4.

B.2 DERIVATION OF THE MEAN CORRECT/OTHER LOGITS

Mean correct logit. The next correct logit $z_{i+1,k}$ following position i in the L-sequence can be computed directly. Predictions are made only for the states following positions $i = L+1, \ldots, 2L$; and for these prediction positions, we have

$$z_{i+1,k} = \mathbf{p}^{\star}_{\mathbf{i+1}} \cdot \mathbf{z_{i+1}} \tag{35}$$

where $\mathbf{p}^{\star}_{\mathbf{i+1}}$ is the teacher "logit" vector, a one-hot vector with unity at entry k. This gives

$$z_{i+1,k} = \sum_{a,b=1}^{d_g} \mathbf{e}_{\mathbf{x_i}}^{\top} (M_a^{\star})^{\top} M_b \mathbf{e}_{\mathbf{x_i}} \left(\sum_{j \le i} \pi_{ij} (v_j)_b \right) (v_{i-L})_a \,. \tag{36}$$

Here, $\mathbf{e}_{\mathbf{x_i}} \in \mathbb{R}^N$ is a state token written as a one-hot encoded (OHE) vector. Similarly, the value vector $\mathbf{v}_j$ from the transformer is an OHE vector for the actions, but zero for the states:

$$\mathbf{v}_j := \begin{cases} \hat{\mathbf{e}}_{\mathbf{g_j}} \,, & j = 1, \ldots, L \\ \mathbf{0} \,, & j \ge L+1 \end{cases} \tag{37}$$

with unit vectors in $\mathbb{R}^{d_g}$. Each g_j, $j = 1, \ldots, L$, specifies the action among the d_g permutations in the vocabulary.

We combine the facts that $i \ge L+1$ here and that $\mathbf{v}_j$ is identically zero for $j \ge L+1$ to rewrite the inner sum:

$$z_{i+1,k} = \sum_{a,b=1}^{d_g} \Big[(M_a^{\star})^{\top} M_b \Big]_{x_i x_i} \left(\sum_{j=1}^{L} \pi_{ij} \delta_{g_j,b} \delta_{g_{i-L},a} \right) \tag{38}$$

$$= \sum_{a,b=1}^{d_g} \Big[(M_a^{\star})^{\top} M_b \Big]_{x_i x_i} \left(\pi_{i,i-L} \delta_{g_{i-L},b} \delta_{g_{i-L},a} + \sum_{\substack{j=1 \\ j \ne i-L}}^{L} \pi_{ij} \delta_{g_j,b} \delta_{g_{i-L},a} \right) . \tag{39}$$

One model trains on a large number of training examples. To account for the average behavior, we have to take an expectation value w.r.t. the current state x_i and all the actions g_j, $j \in \{1, \ldots, L\}$. Here, all actions are needed, because the attention output depends on all those actions via the Values. Because the current state can be obtained by the chain of permutations, the average w.r.t. $x_i = s_{i-L-1}$ is equivalent to an average over the initial state s_0, when simultaneously averaging over the actions. We assume a uniform probability distribution for all quantities. These two averages are then statistically independent.

We apply the expectation value and rewrite $\delta_{g_{i-L},b} \delta_{g_{i-L},a} = \delta_{ab} \delta_{g_{i-L},a}$:

$$\langle z_{i+1,k} \rangle = \frac{1}{N} \sum_{x_i=1}^{N} \frac{1}{d_g^L} \sum_{g_1=1}^{d_g} \sum_{g_2=1}^{d_g} \cdots \sum_{g_L=1}^{d_g} \tag{40}$$

$$\times \left(\sum_{a,b=1}^{d_g} \Big[(M_a^{\star})^{\top} M_b \Big]_{x_i x_i} \left(\pi_{i,i-L} \delta_{ab} \delta_{g_{i-L},a} + \sum_{\substack{j=1 \\ j \ne i-L}}^{L} \pi_{ij} \delta_{g_j,b} \delta_{g_{i-L},a} \right) \right) . \tag{41}$$

Then, we get

$$\langle z_{i+1,k} \rangle = \sum_{a,b=1}^{d_g} \frac{1}{N} \operatorname{Tr} \Big[(M_a^{\star})^{\top} M_b \Big] \left(\pi_{i,i-L} \frac{\delta_{ab}}{d_g} + \sum_{\substack{j=1 \\ j \ne i-L}}^{L} \pi_{ij} \frac{1}{d_g^2} \right) \tag{42}$$

$$= \sum_{a,b=1}^{d_g} \frac{1}{N} \operatorname{Tr} \Big[(M_a^{\star})^{\top} M_b \Big] \left[\pi_{i,i-L} \left(\frac{\delta_{ab}}{d_g} - \frac{1}{d_g^2} \right) + \frac{1}{d_g^2} \right] \tag{43}$$

$$= \pi_{i,i-L} R + (1 - \pi_{i,i-L}) \frac{1}{d_g^2} \sum_{a,b=1}^{d_g} \frac{1}{N} \operatorname{Tr} \Big[(M_a^{\star})^{\top} M_b \Big] \tag{44}$$

$$= \left(\pi_{i,i-L} + \frac{1 - \pi_{i,i-L}}{d_g} \right) R + \frac{d_g - 1}{d_g} (1 - \pi_{i,i-L}) S \,. \tag{45}$$

Let us assume that there is no preferred step in the sequence; then we can use the ansatz $\pi_{i+L,i} = A$. This is empirically confirmed by the attention entries's evolution during training shown in Fig. 5. Consequently,

$$\langle z_{i+1,k} \rangle = \left(A + \frac{1 - A}{d_g} \right) R + \frac{d_g - 1}{d_g} (1 - A) S \,. \tag{46}$$

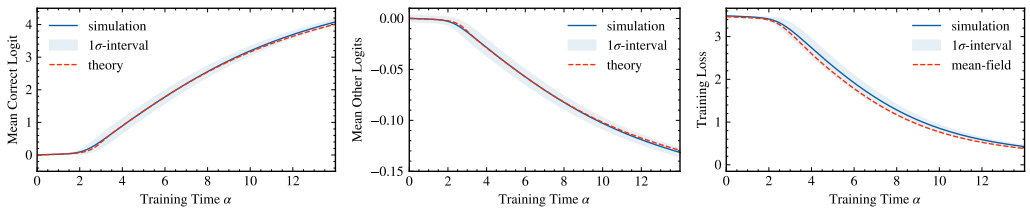

Figure 4: (Left and Middle) Mean correct and other logits. (Right) Mean-field loss ($\ell_{\mathrm{CE}}(A, R, S)$) compared to the actual loss

Mean other logits. Starting with the expression for the logit vector, we calculate the average logit first. For any $r = 1, \ldots, N$, we have

$$z_{i+1,r} = \mathbf{z}_{i+1} \cdot \mathbf{e}_r = \sum_{a=1}^{d_g} \mathbf{e}_r^\top M_a \mathbf{e}_{x_i} \sum_{j=1}^{L} \pi_{ij} (v_j)_a = \sum_{a=1}^{d_g} \big[M_a\big]_{rx_i} \sum_{j=1}^{L} \pi_{ij} \delta_{g_j a} \tag{47}$$

and in the expectation value,

$$\langle z_{i+1,r} \rangle = \frac{1}{N} \sum_{x_i=1}^{N} \frac{1}{d_g^L} \sum_{g_1=1}^{d_g} \sum_{g_2=1}^{d_g} \cdots \sum_{g_L=1}^{d_g} \sum_{a=1}^{d_g} \big[M_a\big]_{rx_i} \sum_{j=1}^{L} \pi_{ij} \delta_{g_j a} \tag{48}$$

$$= \frac{1}{d_g N} \sum_{s=1}^{N} \sum_{a=1}^{d_g} \big[M_a\big]_{rs} \sum_{j=1}^{L} \pi_{ij} \, . \tag{49}$$

The sum over the attention rows are exactly 1 for all $i \geq L + 1$ with our construction. The sum of all logits remains constant throughout the training due to the combination of cross-entropy loss and gradient flow. Hence, we define the constant parameter ζ_{init} as the sum of the logits:

$$\zeta_{\mathrm{init}} := \frac{1}{d_g N} \sum_{a=1}^{d_g} \sum_{r,s=1}^{N} \big[M_a\big]_{rs} \, . \tag{50}$$

Now, we obtained the average expression for *any* logit. That is, for some initializations, r is the correct logit's index; for many others, r is the index for some other logit. As we are interested in the definite, non-correct logit here and already have an identity for the mean correct logit, we define

$$\langle z_{i+1,\bar{k}} \rangle := \frac{1}{N-1} \Big(\zeta_{\mathrm{init}} - \langle z_{i+1,k} \rangle \Big) \, . \tag{51}$$

As ζ_{init} is fundamentally a quantity depending on the model initialization, different seeds will yield different dynamics of the mean other logits and thus of the order parameters.

B.3 Differential equations for the order parameters

The micro-parameter change, obtained from the discrete learning update step, is described by

$$\Delta [M_a]_{rs}(\alpha) := [M_a]_{rs}(\alpha + \Delta\alpha) - [M_a]_{rs}(\alpha) = -\eta \, \frac{\partial \ell_{\mathrm{CE}}}{\partial [M_a]_{rs}} \, . \tag{52}$$

We write the derivative w.r.t. the micro-parameters via the mean correct/other logits with the chain rule. For them, we find the relation

$$\frac{\partial \ell_{\mathrm{CE}}}{\partial \langle z_{i+1,k} \rangle} = - \frac{\partial \ell_{\mathrm{CE}}}{\partial \langle z_{i+1,\bar{k}} \rangle} \quad \left(= - \frac{1}{1 + \frac{1}{N-1} \, \mathrm{e}^{m(A,R,S)}} \right) \tag{53}$$

where $m(A, R, S) = \frac{1}{L} \sum_{i=L+1}^{2L} \langle z_{i+1,k} \rangle - \langle z_{i+1,\bar{k}} \rangle$.

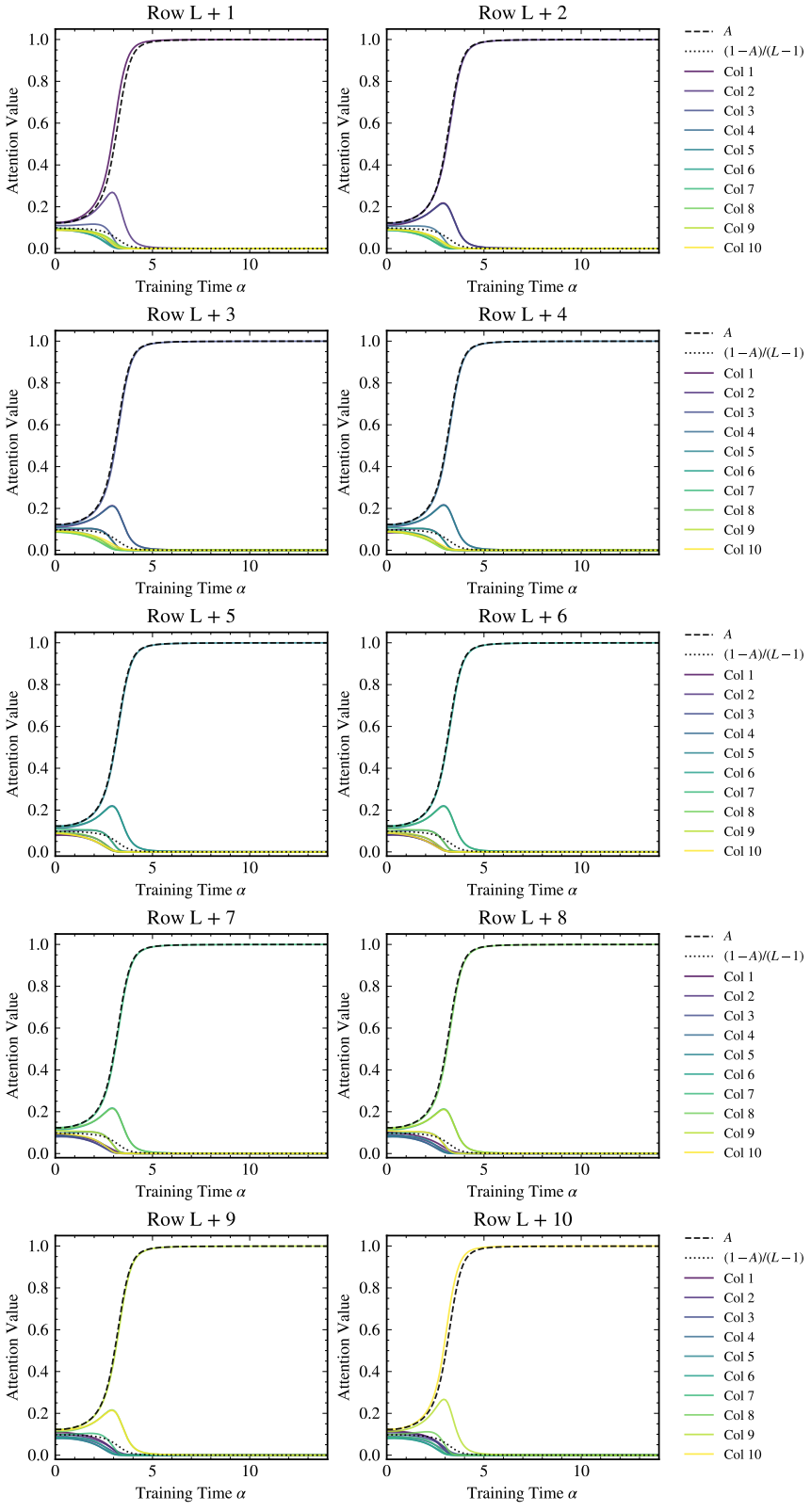

Figure 5: Entries of the attention block as a function of the learning time.

Mean correct/other logit derivatives. Using the expression for the mean correct logit from Equation 11, it is

$$\frac{\mathrm{d}\langle z_{i+1,k}\rangle}{\mathrm{d}[M_b]_{rs}} = \frac{\partial\langle z_{i+1,k}\rangle}{\partial R}\frac{\mathrm{d}R}{\mathrm{d}[M_b]_{rs}} + \frac{\partial\langle z_{i+1,k}\rangle}{\partial S}\frac{\mathrm{d}S}{\mathrm{d}[M_b]_{rs}} \tag{54}$$

$$= c_R\,\frac{\mathrm{d}R}{\mathrm{d}[M_b]_{rs}} + c_S\,\frac{\mathrm{d}S}{\mathrm{d}[M_b]_{rs}}\,. \tag{55}$$

There is no derivative of A here, because A does not depend on the learned permutation matrices at all. The parameters c_R, c_S are given by

$$c_R := A + \frac{1-A}{d_g}\,, \quad c_S = \frac{d_g-1}{d_g}(1-A) = 1 - c_R\,. \tag{56}$$

One can show that

$$\frac{\mathrm{d}R}{\mathrm{d}[M_b]_{rs}} = \frac{1}{d_g N}\,[M_b^\star]_{rs}\,, \qquad \frac{\mathrm{d}S}{\mathrm{d}[M_b]_{rs}} = \frac{1}{d_g(d_g-1)N}\sum_{\substack{c=1\\c\neq b}}^{d_g}[M_c^\star]_{rs}\,. \tag{57}$$

Thus,

$$\frac{\mathrm{d}\langle z_{i+1,k}\rangle}{\mathrm{d}[M_b]_{rs}} = \frac{c_R}{d_g N}\,[M_b^\star]_{rs} + \frac{c_S}{d_g(d_g-1)N}\sum_{\substack{c=1\\c\neq b}}^{d_g}[M_c^\star]_{rs}\,. \tag{58}$$

Take the expression for the mean other logit from Equation 51, then

$$\frac{\mathrm{d}\langle z_{i+1,\bar{k}}\rangle}{\mathrm{d}[M_b]_{rs}} = -\frac{1}{N-1}\frac{\mathrm{d}\langle z_{i+1,k}\rangle}{\mathrm{d}[M_b]_{rs}} + \frac{1}{N-1}\frac{1}{d_g N}\,. \tag{59}$$

Before we resolve this expression, it is best to advance to the update for $[M_b]_{rs}$.

The micro-parameter update. Due to the opposing minus signs in the mean correct/other logits in the loss expression, and by Equation 53, we get a simplification in

$$\Delta[M_b]_{rs} = -\eta\left(\frac{\partial\ell_{\mathrm{CE}}}{\partial\langle z_{i+1,k}\rangle}\frac{\mathrm{d}\langle z_{i+1,k}\rangle}{\mathrm{d}[M_b]_{rs}} + \frac{\partial\ell_{\mathrm{CE}}}{\partial\langle z_{i+1,\bar{k}}\rangle}\frac{\mathrm{d}\langle z_{i+1,\bar{k}}\rangle}{\mathrm{d}[M_b]_{rs}}\right) \tag{60}$$

$$= -\eta\,\frac{\partial\ell_{\mathrm{CE}}}{\partial\langle z_{i+1,k}\rangle}\frac{N}{N-1}\left(\frac{\mathrm{d}\langle z_{i+1,k}\rangle}{\mathrm{d}[M_b]_{rs}} - \frac{1}{d_g N^2}\right) \tag{61}$$

and we can now plug in the mean correct logit derivative:

$$\Delta[M_b]_{rs} = -\frac{\eta}{d_g N}\frac{\partial\ell_{\mathrm{CE}}}{\partial\langle z_{i+1,k}\rangle}\frac{N}{N-1}\left[c_R\,[M_b^\star]_{rs} + \frac{c_S}{d_g-1}\sum_{\substack{c=1\\c\neq b}}^{d_g}[M_c^\star]_{rs} - \frac{1}{N}\right]. \tag{62}$$

Order parameter updates. With the order parameter definitions of R and S, we find

$$\Delta R = \frac{1}{d_g}\sum_{a=1}^{d_g}\frac{1}{N}\,\mathrm{Tr}\Big[(M_a^\star)^\top \Delta M_a\Big] \tag{63}$$

$$= -\frac{\eta}{d_g N}\frac{\partial\ell_{\mathrm{CE}}}{\partial\langle z_{i+1,k}\rangle}\frac{N}{N-1}\left(c_R + c_S\tau - \frac{1}{N}\right) \tag{64}$$

because the teacher matrices are fixed for one seed, and

$$\Delta S = \frac{1}{d_g(d_g-1)}\sum_{a=1}^{d_g}\sum_{\substack{b=1\\b\neq a}}^{d_g}\frac{1}{N}\,\mathrm{Tr}\Big[(M_a^\star)^\top \Delta M_b\Big] \tag{65}$$

$$= -\frac{\eta}{d_g N}\frac{\partial\ell_{\mathrm{CE}}}{\partial\langle z_{i+1,k}\rangle}\frac{N}{N-1}\left(c_R\tau + c_S\,\frac{d_g-2}{d_g-1}\,\sigma + \frac{c_S}{d_g-1} - \frac{1}{N}\right). \tag{66}$$

Two constants appear, τ and σ. They are defined as

$$\tau := \frac{1}{d_g(d_g-1)N} \sum_{a=1}^{d_g} \sum_{\substack{c=1 \\ c\neq a}}^{d_g} \mathrm{Tr}\Big[\big(M_a^\star\big)^\top M_c^\star\Big] \tag{67}$$

and

$$\sigma := \frac{1}{d_g(d_g-1)(d_g-2)N} \sum_{a=1}^{d_g} \sum_{\substack{b=1 \\ b\neq a}}^{d_g} \sum_{\substack{c=1 \\ c\neq a,b}}^{d_g} \mathrm{Tr}\Big[\big(M_a^\star\big)^\top M_c^\star\Big] \tag{68}$$

and depend on the seed. We readily find $\sigma = \tau$ and are left with one such permutation matrix trace. As permutation matrices are not orthogonal, there is some overlap.

Many seeds give rise to many choices for the set of permutations. We can average over the seeds to get $\overline{\tau} = 1/N + \varepsilon$ with a correction $\varepsilon = -(1-1/N)/(N!-1)$, which is negligible for non-small N. Details on this average are given in Appendix B.4. The final difference quotients using our mean-field cross-entropy then read

$$\Delta R = \frac{\eta}{d_g N} \frac{1}{1+\frac{1}{N-1}\,\mathrm{e}^{m(A,R,S)}} \left(A + \frac{1-A}{d_g}\right) \tag{69a}$$

$$\Delta S = \frac{\eta}{d_g N} \frac{1}{1+\frac{1}{N-1}\,\mathrm{e}^{m(A,R,S)}} \frac{1-A}{d_g} \tag{69b}$$

with $m(A,R,S) = \frac{1}{L}\sum_{i=L+1}^{2L} \langle z_{i+1,k}\rangle - \langle z_{i+1,\overline{k}}\rangle$.

The attention order parameter. To see the dynamics of A, we write, for the moment, $\mathbf{W} = (\mathbf{W}_\mathbf{K}^\top, \mathbf{W}_\mathbf{Q}^\top)^\top$ as a vector of all key and query weights. We calculate

$$d_g N(\nabla_\mathbf{W} A)\cdot(-\eta)(\nabla_\mathbf{W}\ell_{\mathrm{CE}}) = \nabla_\mathbf{W} A \cdot \frac{\Delta \mathbf{W}}{\Delta\alpha} \to \nabla_\mathbf{W} A \cdot \frac{\mathrm{d}\mathbf{W}}{\mathrm{d}\alpha} = \frac{\mathrm{d}A}{\mathrm{d}\alpha}\,. \tag{70}$$

Applying the chain rule once more for $\nabla_\mathbf{W}\ell_{\mathrm{CE}} = (\nabla_\mathbf{W} A)\frac{\mathrm{d}\ell_{\mathrm{CE}}}{\mathrm{d}A}$ (only A depends on the weights) gives

$$\frac{\mathrm{d}A}{\mathrm{d}\alpha} = -\eta d_g N (\nabla_\mathbf{W} A)^2 \frac{\mathrm{d}\ell_{\mathrm{CE}}}{\mathrm{d}A} \tag{71}$$

when assuming the right-hand side is finite in the thermodynamic limit. It will be seen that this is indeed the case.

Define attention logits $l_{ij} = \mathbf{q_i}^\top \mathbf{k_j}/\sqrt{d_h} = \mathbf{W}_\mathbf{Q}^\top \Omega(j-i)\mathbf{W}_\mathbf{K}/\sqrt{d_h}$. Then the gradient $\nabla_\mathbf{W} A$ becomes

$$\nabla_\mathbf{W} A = \sum_{i=L+1}^{2L} \frac{\partial A}{\partial \pi_{i,i-L}} \nabla_\mathbf{W} \pi_{i,i-L} = \sum_{i=L+1}^{2L} \frac{\partial A}{\partial \pi_{i,i-L}} \sum_{k=1}^{L} \frac{\partial \pi_{i,i-L}}{\partial l_{ik}} \nabla_\mathbf{W} l_{ik} \tag{72}$$

Let us define the function

$$c(A) := d_g N(\nabla_\mathbf{W} A)^2 = d_g N \left| \sum_{i,k=1}^{L} \frac{\partial A}{\partial \pi_{i+L,i}} \frac{\partial \pi_{i+L,i}}{\partial l_{i+L,k}} \Big(\nabla_\mathbf{W} l_{i+L,k}\Big) \right|^2 \tag{73}$$

then the differential equation reads

$$\frac{\mathrm{d}A}{\mathrm{d}\alpha} = \eta\, c(A) \frac{1}{1+\frac{1}{N-1}\,\mathrm{e}^{m(A,R,S)}} \frac{N}{N-1} \frac{d_g-1}{d_g} (R-S) \tag{74}$$

so that merely $c(A)$ needs to be yet determined. Given the derivatives

$$\frac{\partial A}{\partial \pi_{i+L,i}} = \frac{1}{L} \tag{75}$$

$$\frac{\partial \pi_{i+L,i}}{\partial l_{i+L,k}} = \pi_{i+L,i}(\delta_{ik} - \pi_{i+L,k}) \tag{76}$$

$$\nabla_{\mathbf{W_Q}} l_{i+L,k} = \frac{\Omega(k-i-L)\mathbf{W_K}}{\sqrt{d_h}} \tag{77}$$

$$\nabla_{\mathbf{W_K}} l_{i+L,k} = \frac{\Omega(i+L-k)\mathbf{W_Q}}{\sqrt{d_h}} \,, \tag{78}$$

some linear algebra yields, for the approximation $\pi_{i+L,j} = A\delta_{ij} + \frac{1-A}{L-1}(1-\delta_{ij})$ of an expected seed-averaged behavior,

$$c(A) = d_g N A^2 (1-A)^2 \frac{\mathbf{W_K}^\top (\mathbb{I}-\Lambda)^2 \mathbf{W_K} + \mathbf{W_Q}^\top (\mathbb{I}-\Lambda)^2 \mathbf{W_Q}}{d_h} \tag{79}$$

where $\Lambda = \mathrm{diag}(\lambda_1, \lambda_1, \lambda_2, \lambda_2, \ldots, \lambda_{d_h/2}, \lambda_{d_h/2})$ with $\lambda_n = 1$ for $\omega_n \in 2\pi\mathbb{Z}$ or otherwise

$$\lambda_n = \frac{1}{L-1}\left[\frac{1}{L}\left(\frac{\sin(\omega_n L/2)}{\sin(\omega_n/2)}\right)^2 - 1\right]. \tag{80}$$

While the diagonal part in $\pi_{i+L,j} = A\delta_{ij} + \frac{1-A}{L-1}(1-\delta_{ij})$ approximates the attention entries well, the off-diagonal entries behave differently during training. This elongates the smooth increase of A in the learning regime. Without the uneven behavior, the slope of A would grow too steeply, because $A^2(1-A)^2$ acts as a catalyst in the differential equation for A. At the end of training, just as at the very beginning, this approximation describes the system well, however. Hence, the above expression for $c(A)$ may be used to determine the value at initialization, where the ansatz is still correct. We find that the empirical trajectory is approximated by $c(A) = Lc_\omega \cdot A(1-A)$ with

$$c_\omega := \frac{d_g N}{L}\frac{L-1}{L^2}\left\langle \frac{\mathbf{W_K}^\top (\mathbb{I}-\Lambda)^2 \mathbf{W_K} + \mathbf{W_Q}^\top (\mathbb{I}-\Lambda)^2 \mathbf{W_Q}}{d_h}\bigg|_{\alpha=0} \right\rangle_{\mathbf{W_Q},\mathbf{W_K}}. \tag{81}$$

At initialization, $\mathbf{W_Q}$ and $\mathbf{W_K}$ are random vectors, such that, for some random $\mathbf{X} \in \mathbb{R}^{d_h}$,

$$\langle \mathbf{X}^\top (\mathbb{I}-\Lambda)^2 \mathbf{X}\rangle_{\mathbf{X}} = \mathrm{Tr}\langle \mathbf{X}^\top (\mathbb{I}-\Lambda)^2 \mathbf{X}\rangle_{\mathbf{X}} = \mathrm{Tr}\Big[(\mathbb{I}-\Lambda)^2 \langle \mathbf{X}\mathbf{X}^\top\rangle_{\mathbf{X}}\Big] \tag{82}$$

exploiting the linearity of the trace and expectation value. $\langle \mathbf{X}\mathbf{X}^\top\rangle_{\mathbf{X}}$ is the covariance matrix of $\mathbf{X}$. An i.i.d. initialization yields a scaled unit matrix with, say, σ^2 as the variance. Then $\langle \mathbf{X}^\top(\mathbb{I}-\Lambda)^2\mathbf{X}\rangle_{\mathbf{X}} = \sigma^2 \,\mathrm{Tr}\big[(\mathbb{I}-\Lambda)^2\big]$ and

$$c_\omega = \frac{d_g N}{L}\frac{L-1}{L^2}\,\sigma^2 \frac{2}{d_h}\,\mathrm{Tr}\big[(\mathbb{I}-\Lambda)^2\big]\,. \tag{83}$$

Plugging in the definition of Λ followed by some algebraic manipulations leads to the final expression in Equation 14.

In principle, we can determine a discrete update step ΔA via a Taylor expansion in the Δl_{ij} of the softmax activation. These, in turn, are expressible via $\Delta\mathbf{W_Q}, \Delta\mathbf{W_K}$ resulting in a series in orders of the learning rate. Small learning rates allow us to neglect them, yielding a first-order expression equivalent to the above derivative. Let us note that an alternative definition of the training time incorporating the learning rate would allow us to take the gradient-flow limit $\eta \to 0$ to neglect the higher-order terms in the expansion and make the differential equations exact for finite hyperparameters without considering the regime of the thermodynamic limit.

B.4 Seed-averaged permutation matrix overlap

Our goal is to find the quantity $\mathrm{Tr}\big[(M_a^\star)^\top M_b^\star\big]$ for some $a \neq b$, $a, b \in \{1, \ldots, d_g\}$ on average. Because a product of two permutation matrices is again a permutation matrix, we reduce the problem

of finding the overlap of two random matrices by computing the average trace of a permutation. Thus, w.l.o.g., we can assume one of them to be the unit matrix. Since the trace is a linear operator, we can compute the average permutation matrix itself. As each entry in the first row has a probability of $1/N$ to have a non-zero entry when constructing such a matrix, and because each row must behave alike due to symmetry, the average permutation matrix is filled with the entries $1/N$.

Now, we only want to consider overlaps of different permutation matrices, thus we have to consider only non-trivial matrices when assuming the other one is the identity. To account for that, we write

$$\langle M^\star \rangle_{M^\star \in S^N \setminus \{\mathbb{I}_N\}} = \frac{1}{N!-1}\left(\sum_{i=1}^{N!} M_i^\star - \mathbb{I}_N\right) = \frac{1}{N!-1}\left(\frac{N!}{N} J_N - \mathbb{I}_N\right) \tag{84}$$

with $(J_N)_{rs} = 1 \; \forall r, s = 1, \ldots, N$ and $N \geq 2$. Hence,

$$\langle \operatorname{Tr} M^\star \rangle_{M^\star \in S^N \setminus \{\mathbb{I}_N\}} = \frac{N!}{N!-1} - \frac{N}{N!-1} = \frac{1 - \frac{1}{(N-1)!}}{1 - \frac{1}{N!}} \; . \tag{85}$$

With this intermediate finding, we may write

$$\left\langle \operatorname{Tr}\left[(M_a^\star)^\top M_b^\star \right] \right\rangle_{M_a^\star \neq M_b^\star \in S^N} = \frac{1 - \frac{1}{(N-1)!}}{1 - \frac{1}{N!}} \tag{86}$$

when $a \neq b$. We immediately realize that this constant approaches 1 for $N \to +\infty$. Therefore,

$$\overline{\tau} := \langle \tau \rangle_{M_a^\star \neq M_b^\star \in S^N} = \frac{1}{N} \frac{1 - \frac{1}{(N-1)!}}{1 - \frac{1}{N!}} = \frac{1}{N} - \frac{1}{N} \frac{N-1}{N!-1} \; . \tag{87}$$

B.5 Early training regime and attention learning

At the start of training, A remains on a plateau of $A = A_0$, which is $1/L$ on average (every attention entry is equal with random key/query initializations). A Taylor expansion of the loss up to linear order describes the system well during the early training process. The differential equations are solvable exactly in this regime:

$$A(\alpha) = A_0 \; , \qquad R(\alpha) = \eta \, \frac{N-1}{N} \left(A_0 + \frac{1 - A_0}{d_g} \right) \cdot \alpha \; , \qquad S(\alpha) = \eta \, \frac{N-1}{N} \frac{1 - A_0}{d_g} \cdot \alpha \; . \tag{88}$$

The matrix overlaps R and S increase linearly before higher-order terms in the loss become relevant and the attention mechanism starts to learn. We observe that the correct overlap increases faster than the overlap with wrong permutations.

In the main text, we observed initially a flat attention order parameter A, followed by a rapid increase at the start of the second learning phase (see left panel of Fig. 2). Using this separation of learning into the two phases, we can analytically obtain an approximate solution of the differential equations for the order parameters that describe the onset of attention learning. Combining Eqs. 15b and 15c, in the linearized regime of the first phase of training (Appendix B.5) where $A(0) = 1/L$, we find

$$\frac{\mathrm{d}(R - S)}{\mathrm{d}\alpha} \approx \eta \, \frac{N-1}{N} \frac{1}{L} \; . \tag{89}$$

In the limit of large N, the solution is thus given by $R - S \approx \eta\alpha/L$. Using this in the differential equation for the attention order parameter, Equation 15a, yields, to leading order in $1/L$,

$$\frac{\mathrm{d}A}{\mathrm{d}\alpha} \approx \eta^2 c_\omega \alpha A \tag{90}$$

$$\Rightarrow A(\alpha) \approx \frac{1}{L} \exp\left[\frac{c_\omega}{2} \eta^2 \alpha^2 \right] \; . \tag{91}$$

This approximation is valid until the onset of the second learning phase. The timescale of the increase is set by $\sqrt{2/(c_\omega \eta^2)} = O(1)$.

B.6 FINAL ROLLOUT ACCURACY MODELING

To model the final rollout accuracy, we start with the recursion relation

$$P(\hat{s}_{t+1} = s_{t+1}) = P(\hat{s}_t = s_t) \cdot P(\hat{s}_{t+1} = s_{t+1} | \hat{s}_t = s_t) + \big[1 - P(\hat{s}_t = s_t)\big] \cdot P(\hat{s}_{t+1} = s_{t+1} | \hat{s}_t \neq s_t) \tag{92}$$

with $P(\hat{s}_0 = s_0) = 1$. We shorten the notation:

$$\varrho_t := P(\hat{s}_t = s_t) \qquad \mathcal{A}_t := P(\hat{s}_{t+1} = s_{t+1} | \hat{s}_t = s_t) \qquad \mathcal{B}_t := P(\hat{s}_{t+1} = s_{t+1} | \hat{s}_t \neq s_t)\,. \tag{93}$$

Then

$$\varrho_{t+1} = \varrho_t(\mathcal{A}_t - \mathcal{B}_t) + \mathcal{B}_t \tag{94}$$

with $\varrho_0 = 1$ for the initial token, and we seek ϱ_L from this recursion relation. It holds

$$\varrho_L = \mathcal{A}_0 \prod_{t=1}^{L-1} (\mathcal{A}_t - \mathcal{B}_t) + \sum_{t=1}^{L-1} \left[\mathcal{B}_t \prod_{i=t+1}^{L-1} (\mathcal{A}_i - \mathcal{B}_i) \right]. \tag{95}$$

The next state is chosen by the maximal argument of the logit vector. That is, $\mathcal{A}_t$ is the probability of choosing the correct logit (when operating on s_t). From the definition of the logit, we see that the logit is the s_t-th column of our action-conditioned map, which does not depend on the last state token s_{t-1}. If we assume that the other logits are equally likely on average, and that the prediction of state i behaves the same as that of state j, then we may write

$$\mathcal{B}_t \approx \frac{1 - \mathcal{A}_t}{N - 1}\,. \tag{96}$$

Next, we need to find an expression for the probability of choosing the correct logit (given the current state is correct). For this, we model the correct logit as a Gaussian variable. A numerical check quickly supports this assumption: the correct logit is for different training examples and for each sequence step almost exactly distributed normally.

Viewing every sequence-step as the same, we can replace $\mathcal{A}_t$ by $\overline{\mathcal{A}} := \mathbb{E}_{t \in [L]}[\mathcal{A}_t]$ such that

$$\varrho_L \approx \frac{N-1}{N} \left(\frac{N\overline{\mathcal{A}} - 1}{N - 1} \right)^L + \frac{1}{N} \tag{97}$$

after some rearrangements and exploitation of a geometric sum with $\sum_{n=0}^{L-2} r^n = (1 - r^{L-1})/(1-r)$ and $r = \frac{N\overline{\mathcal{A}}-1}{N-1}$. Here, the singularity is hit when $\overline{\mathcal{A}} = 1$, which cannot occur as the student is never able to achieve perfect predictions.

Immediately, we realize that $\varrho_L \to 1$ when $\overline{\mathcal{A}} \to 1$. Random guessing ($\overline{\mathcal{A}} = 1/N$) would yield $\varrho_L = 1/N$ – a random choice. Yet ϱ_L grows much slower: each token in the sequence has to be predicted correctly in a row.

The probability of the correct choice given the correct current logit can be expressed as

$$\overline{\mathcal{A}} = P(z_k > \max_{f \neq k} z_f). \tag{98}$$

This is equivalent to determining the probability of every other logit being at most x (a certain correct logit z_k) and then integrating over x including the Gaussian PDF of $x = z_k$:

$$\overline{\mathcal{A}} = \int_{-\infty}^{+\infty} \mathrm{d}x \, \frac{\mathrm{e}^{-(x-\mu_k)^2/(2\sigma_k^2)}}{\sqrt{2\pi}\sigma_k} \, \Phi\!\left(\frac{x - \mu_{\overline{k}}}{\sigma_{\overline{k}}} \right)^{N-1} \tag{99}$$

where Φ is the CDF of the Gaussian PDF:

$$\Phi(u) := \int_{-\infty}^{u} \mathrm{d}y \, \frac{\mathrm{e}^{-y^2/2}}{\sqrt{2\pi}}\,. \tag{100}$$

Upon substituting $u = (x - \mu_k)/\sigma_k$:

$$\overline{\mathcal{A}} = \int_{-\infty}^{+\infty} \mathrm{d}u \, \frac{\mathrm{e}^{-u^2/2}}{\sqrt{2\pi}} \, \Phi\!\left(\frac{\sigma_k}{\sigma_{\overline{k}}} \, u + \frac{\mu_k - \mu_{\overline{k}}}{\sigma_{\overline{k}}} \right)^{N-1}. \tag{101}$$

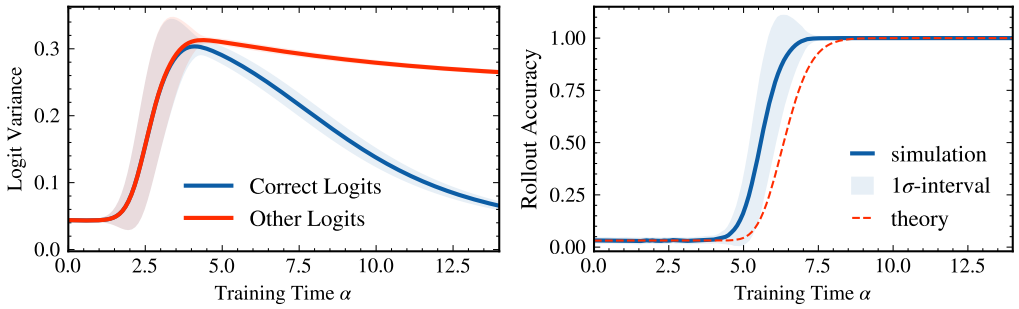

Figure 6: (a) Variances for the mean correct and other logits during training. (b) Simulated rollout accuracy for the empirical logit means and variances, where the empirical and theoretically predicted logit means had a near-perfect agreement.

C Supplementary Experiments

C.1 Normalized alignment metrics and sparse coverage

Normalized alignment. With the normalized versions of the logic order parameters utilizing the Frobenius norm $\|\cdot\|_F$,

$$R := \frac{1}{d_g}\sum_{a=1}^{d_g}\frac{\mathrm{Tr}\left[(M_a^\star)^\top M_a\right]}{\|M_a^\star\|_F\|M_a\|_F}\,, \qquad S := \frac{1}{d_g(d_g-1)}\sum_{a=1}^{d_g}\sum_{\substack{b=1\\ b\neq a}}^{d_g}\frac{\mathrm{Tr}\left[(M_a^\star)^\top M_b\right]}{\|M_a^\star\|_F\|M_b\|_F}\,, \tag{102}$$

we can determine if the norm growth of R and S is a byproduct of mere shifts in the entries of all student matrices or if alignment is indeed taking place. The teacher matrices have constant norm $\|M_a^\star\|_F = \sqrt{N}$. Figure 7 shows A and the normalized order parameters R, S for the baseline training set of 32,768 sequences, together with a sweep over smaller training sets ranging from 50 to 1,000. For the baseline, we the normalized on-target overlap increases to about 0.8 and slowly begins to saturate near the end of the depicted training runs, but still rising toward maximal value 1. The off-target overlap shows a similar curve compared to the non-normalized version displayed in Figure 2: it decreases to zero after an initial rise. The growth of the raw R in Figure 2 thereby reflects genuine directional alignment together with norm growth. The rollout accuracy, see Figure 8, saturates much earlier: a correct prediction requires only the maximal argument of each column of the learned composed permutation $\tilde{M}_i$. Almost all columns of the student matrices attain their maximum at the teacher entry long before the cosine similarity approaches unity.

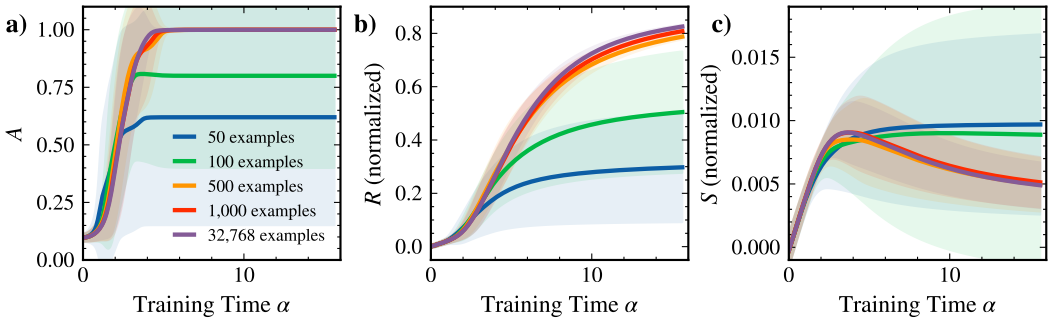

Figure 7: Attention order parameter A and normalized logic overlaps R, S for training sets of 50 to 32,768 sequences (10 seeds each). Shaded bands denote one standard deviation).

Sparse coverage. We perform training runs with smaller training set size, such that nor every action-state pair is observed during training. The realized pair coverage, measured per run and averaged over some seeds, ranges from about 39% for 50 sequences to 100% for 1,000 sequences.

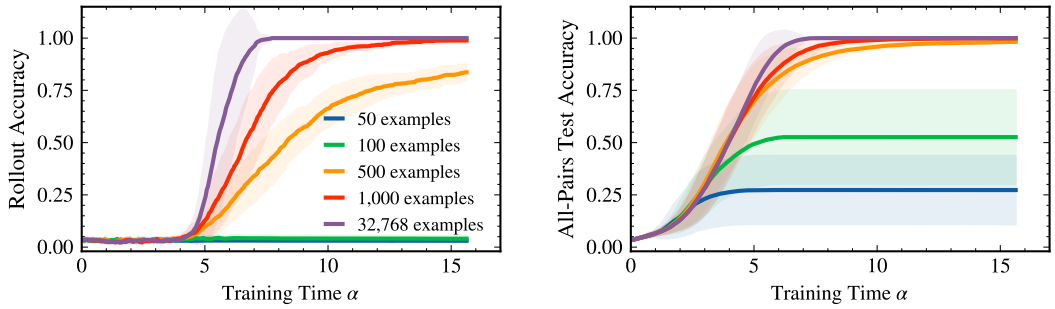

Figure 8: Final rollout accuracy for training sets of 50 to 32,768 sequences for an unseen test dataset and the corresponding step accuracy evaluated over all action-state pairs (10 seeds each).

The staged ordering is unchanged: R and S grow first, and A shows its steep increase at a similar training time across training-set sizes. However, A and the normalized R saturate at lower values. When evaluating the step accuracy under perfect attention retrieval over all $d_g N$ action-state pairs, we observe a range from 0.27 ± 0.16 to 1.00 ± 0.00 from 50 to 1,000 training examples. An unobserved entry of a teacher matrix is in general not identifiable from the observed ones, so unseen pairs cannot be inferred. The generalization tested throughout this work is the composition of observed pairs along unseen sequences.

C.2 Architectural relaxations and standard-block dynamics

We conduct several ablation studies and simulations of a standard one-block transformer to test whether the staged-dynamics picture extends to less constrained architectures. Each constraint of Section 3 is relaxed separately, and all four jointly: the mask is extended to a full causal mask (state columns are no longer neglected), the token embeddings are learned ($d_{\text{model}} = 128$, random initialization), the value matrix is learned instead of being one-hot, and the set of matrices is replaced by a standard MLP block. In addition, we train a fully standard one-block transformer with learned embeddings ($d_{\text{model}} = 128$), learned query / key / value / head output projections (Q/K/V/O), a GeGLU MLP, multiple residual connections and layer normalizations, as well as a variant of it with two attention heads. These are trained with the AdamW optimizer (0.01 weight decay, $\beta_1 = 0.9$, $\beta_2 = 0.95$), gradient clipping at value 1, and a learning rate schedule consisting of a warmup phase (500 steps with initial $\eta = 0.001$) and subsequent decline. Once the set of student matrices is replaced, R and S are no longer defined, and we construct a probe to measure an analog of each as follows. The logic module gets the inputs $\mathbf{o}$ from the attention head and the current state $\mathbf{e_s}$, then computes

$$\mathbf{z} = f(\mathbf{o}, \mathbf{e_s}) = \sum_{a=1}^{d_g} o_a M_a \mathbf{e_s} = \sum_{a=1}^{d_g} o_a (\mathbf{M_a})_{:,\mathbf{s}} \,, \tag{103}$$

where $(:, s)$ denotes the s-th column. Given a one-hot attention output $\mathbf{o} = \hat{\mathbf{e}}_\mathbf{b}$, the module's output will be $\mathbf{z} = (\mathbf{M_b})_{:,\mathbf{s}}$. We can thus reconstruct the set of matrices from the logic module, and we translate this to the regular MLP. With its output function f_{MLP}, we set $(\mathbf{F}_\mathbf{b}^{\text{probe}})_{:,\mathbf{s}} := \langle f_{\text{MLP}}(\mathbf{v_b}, \mathbf{e_s}) \rangle_{\text{context}}$ when providing a perfect attention output ($\mathbf{v_b}$ instead of $\hat{\mathbf{e}}_\mathbf{b}$ when value vectors and not one-hot vectors represent the actions). As embeddings make the predictions context-dependent, we average over 32 random contexts per action. Then, we identify $(\mathbf{M_b})_{:,\mathbf{s}} \leftrightarrow (\mathbf{F}_\mathbf{b}^{\text{probe}})_{:,\mathbf{s}}$ and replace R and S by R_{probe} and S_{probe} using the identified probe values.

Figures 9-13 show the order parameters, or their probes, for the single and the joint relaxations, and Figure 15 those of the standard one-block transformer with one and two heads. In every case, the logic overlaps grow from the start of training while A remains on its initial plateau, and A rises only afterwards. With the full causal mask, the plateau lies below $1/L$, because the initial attention mass is spread over the state positions as well. Table 2 compactly lists the ablations and documents the final rollout accuracies. The runs with learned embeddings, alone and combined with the other relaxations, show the staged dynamics but reach rollout accuracies of 0.24 ± 0.04 and 0.53 ± 0.14

after two to three times the reference training duration and are still improving. The fully standard block with the same learned embeddings reaches an accuracy of 1.0, so we attribute the remaining gap to the plain SGD schedule with a learning rate tuned to the frozen-embedding scale, which yields small gradients through the embeddings, rather than to the mechanism.

Table 2: Summary of the supplementary experiments: architectural relaxations, standard one-block transformers, freezing interventions, and additional testbeds. Rollout accuracies are means and standard deviations over 10 seeds at the end of training.

Experiment	**Staging preserved?**	**Final rollout accuracy**
Learned token embeddings	yes	Rollout: 0.24 ± 0.04 after $\alpha = 100$, still increasing
Learned value matrix	yes	1.0 ± 0.0
Full causal mask	yes	1.0 ± 0.0
Standard MLP replacing matrix bank	yes (with probes)	1.0 ± 0.0
Four relaxations jointly	yes	0.53 ± 0.14 after $\alpha = 100$, still increasing
Fully standard one-block transformer	yes (with probes, $A \approx 0.7$ after hitting 0.9 maximum)	1.0 ± 0.0
Two-head standard block	yes (with probes, A is mean of both heads)	1.0 ± 0.0
Attention frozen uniformly	no ($A = 1/L$, logic proxies approach different plateaus)	0.03 ± 0.01 after $\alpha = 100$
MLP frozen at initialization	no ($R_{\text{probe}} = S_{\text{probe}} \approx 0$, $A \approx 0.26$	0.03 ± 0.01 after $\alpha = 100$
Normalized matrix alignment	yes	1.0 ± 0.0
Sparse-data sweep	yes (39% to 100% pair coverage, 0.27 ± 0.16 to 1.00 ± 0.00 all-pair perfect-attention step accuracy)	0.03 ± 0.00 to 1.00 ± 0.00
Cyclic vocabulary C_{32}	yes	1.0 ± 0.0

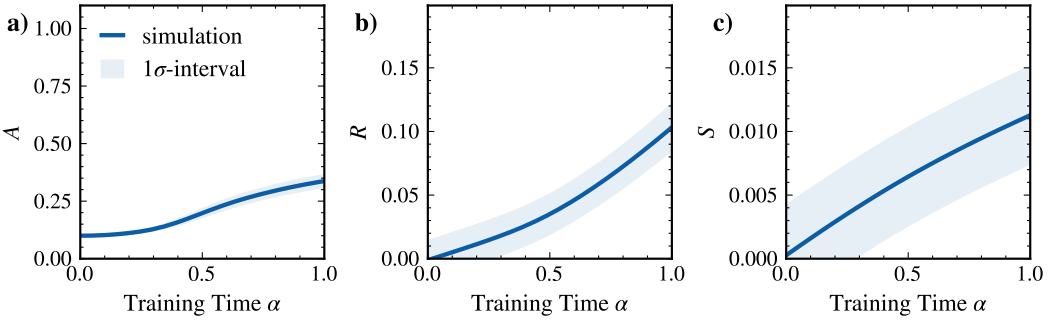

Figure 9: Constraint relaxation: learnable token embeddings ($d_{\text{model}} = 128$) instead of fixed $u = 1$ embeddings, all other constraints as in the main text. Mean over seeds; shaded bands denote one standard deviation.

C.3 Attention and MLP freezing interventions

To test the gating mechanism of Equation 15a in the standard one-block transformer, we freeze one subsystem at initialization and train the other (Figure 15). With attention frozen at its uniform initialization, the probes R_{probe} and S_{probe} initially follow the trajectories of the unfrozen transformer block – the mixed heuristic of the first phase – but diverge toward different plateaus once the full block starts to learn attention. The final rollout accuracy remains near zero even after about 100

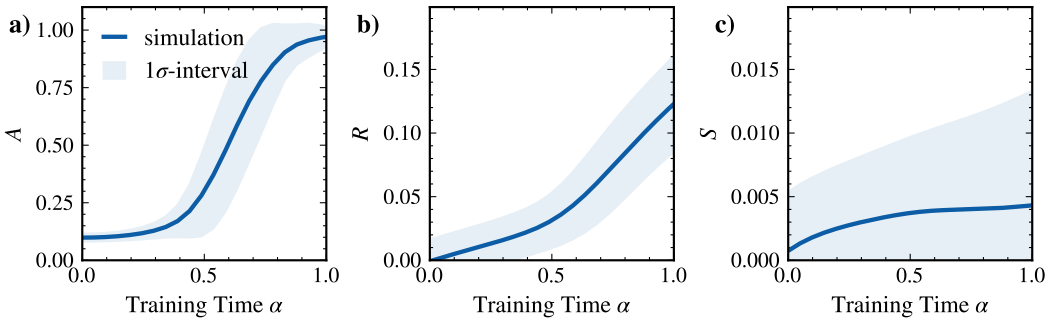

Figure 10: Constraint relaxation: learnable value matrix instead of fixed one-hot values, all other constraints as in the main text (without embeddings, these matrix entries are not the typical value projections). Mean over seeds; shaded bands denote one standard deviation.

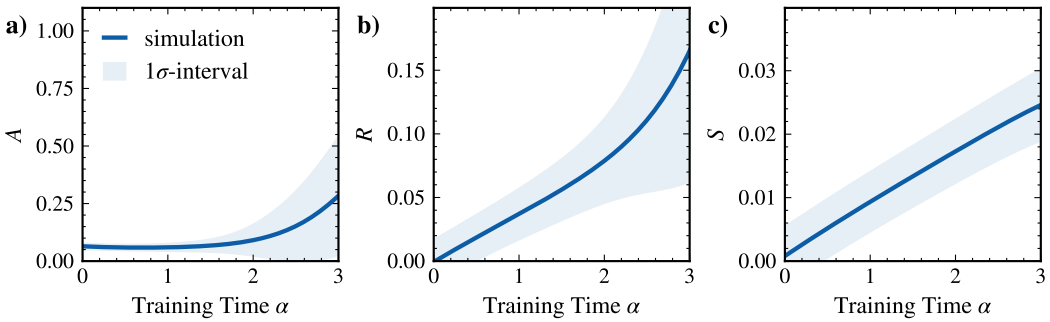

Figure 11: Constraint relaxation: full causal attention mask including the state columns, all other constraints as in the main text. Mean over seeds; shaded bands denote one standard deviation.

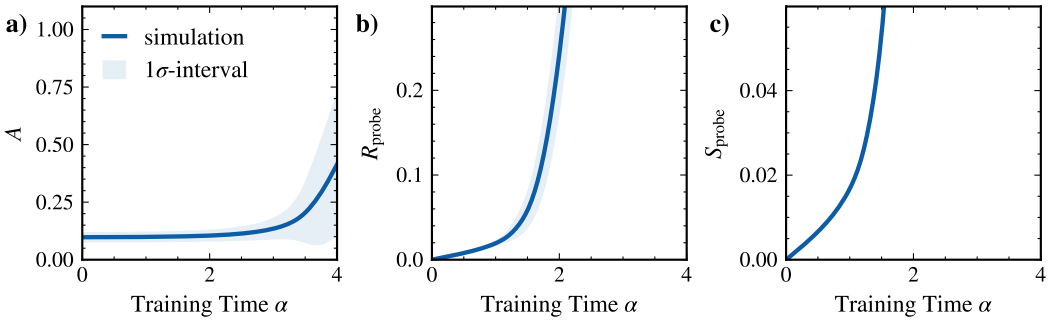

Figure 12: Constraint relaxation: MLP logic instead of the logic module, with a GeGLU activation function with width $4N$, all other constraints as in the main text. Mean over seeds; shaded bands denote one standard deviation.

times the training duration that the full block needs to reach perfect accuracy. With the MLP frozen at initialization, no discrimination margin forms: A rises more slowly and saturates far below the value reached by the full block (about 0.26 versus about 0.7 after extended training), and the rollout accuracy again stays at random-guessing level. Each mechanism therefore behaves differently when the other is held fixed, in line with the coupling in Equations 15: the logic module supplies the gradient signal that attention needs, and attention supplies the retrieved action that the logic module needs. Table 2 includes short summaries of these runs as well.

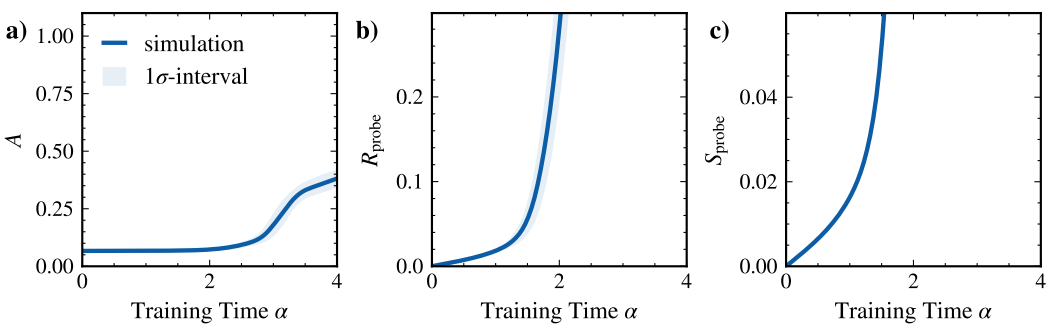

Figure 13: Constraint relaxation: all four relaxations jointly, with a GeGLU MLP in place of the logic module; R_{probe} and S_{probe} are the probe analogs of R and S.

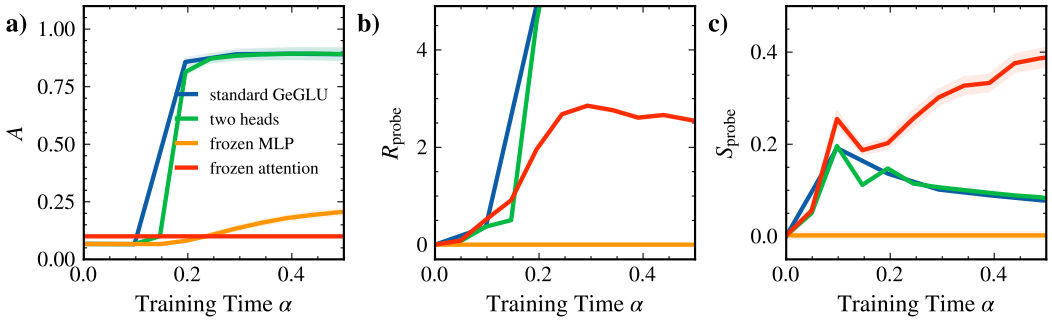

Figure 14: Standard one-block transformer with a GeGLU MLP (one and two attention heads) and freezing interventions in the one-head block. For the latter, either the MLP is frozen at initialization or the attention is held at uniform from initialization onward. For these runs, the training time α is defined with the number of optimizer steps in place of the number of epochs. Shaded bands denote one standard deviation over seeds.

C.4 Other testbeds

Cyclic group. We apply the baseline model to a subset of permutations comprising all representations of the cyclic group C_{32}. The size of the action vocabulary stays at $d_g = 32$ and thereby does not change, the same is true for the number of states N. The dynamical equations use teacher-teacher overlap statistics for arbitrary permutation matrices (Appendix B.3). These must be adapted if the action vocabulary changes. In this case, we must recompute them for the chosen subgroup of permutations, which is quite straightforward as cyclic shifts exhibit vanishing matrix overlap. Recall from the derivation of the differential equations the difference quotients

$$\frac{\Delta R}{\Delta \alpha} = \eta \, \frac{1}{1 + \frac{1}{N-1} \mathrm{e}^{m(A,R,S)}} \, \frac{N}{N-1} \left(c_R + c_S \tau - \frac{1}{N} \right) , \tag{104a}$$

$$\frac{\Delta S}{\Delta \alpha} = \eta \, \frac{1}{1 + \frac{1}{N-1} \mathrm{e}^{m(A,R,S)}} \, \frac{N}{N-1} \left(c_R \tau + c_S \, \frac{d_g - 2}{d_g - 1} \, \tau + \frac{c_S}{d_g - 1} - \frac{1}{N} \right) , \tag{104b}$$

which hold for any action vocabulary and where τ is the teacher-teacher matrix overlap defined by Equation 67. With $\tau = 0$ in the case of cyclic matrices, and let us additionally set $d_g = N$ as in the simulations, the subsequent ODEs are

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \eta \, \frac{1}{1 + \frac{1}{N-1} \mathrm{e}^{m(A,R,S)}} \cdot A \, , \tag{105a}$$

$$\frac{\mathrm{d}S}{\mathrm{d}\alpha} = -\eta \, \frac{1}{1 + \frac{1}{N-1} \mathrm{e}^{m(A,R,S)}} \cdot \frac{A}{N-1} \, . \tag{105b}$$

While R increases, S becomes negative.

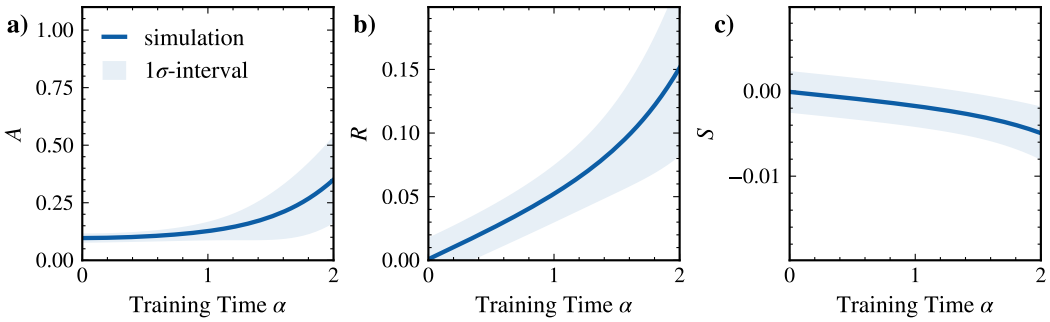

Figure 15: Baseline model with C_{32} as the action vocabulary. Shown are the means over 10 seeds. Shaded bands denote one standard deviation over seeds.

C.5 Robustness concerning hyperparameters

We perform simulation runs for the baseline model but with a different set of hyperparameters. We now set $d_g = N = 24$ and $d_h = 8$ covering some of the most important parameters for the task, the rest are as in Table 1. The RoPE frequencies are again set as $\omega_n := 2\pi\,\theta_{\mathrm{RoPE}}^{-2(n-1)/d_h}$ with θ_{RoPE} (`rope_theta`) as the base and $n = 1, \ldots, d_h/2$.

The results are shown in Figures 16, 17, 18, and 19.

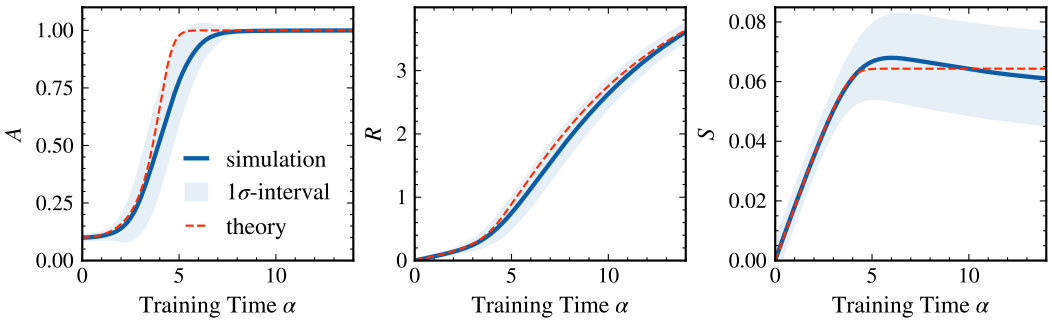

Figure 16: Order parameters from theory and simulations.

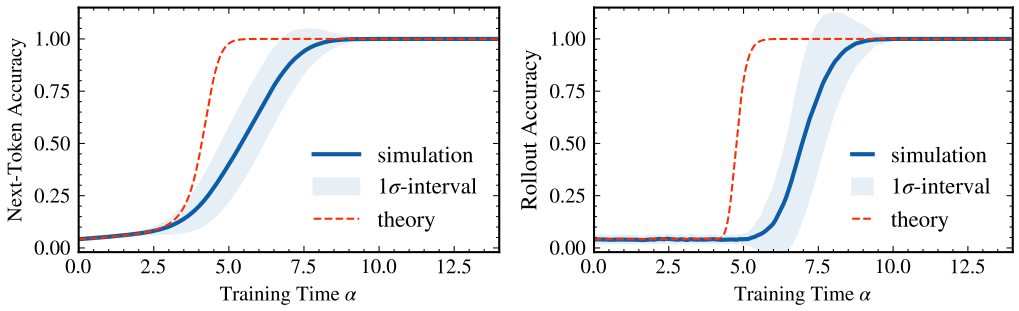

Figure 17: Next-token accuracy and final rollout accuracy for logit variances held fixed at their initial values of $\sigma_k^2 = 0.0466$, $\sigma_{\bar{k}}^2 = 0.0467$.

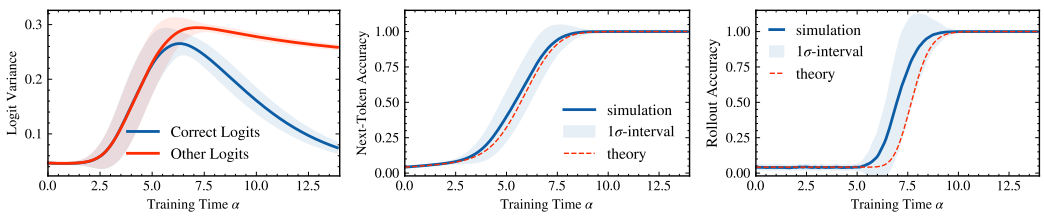

Figure 18: Next-token accuracy and final rollout accuracy with the empirical logit variances.

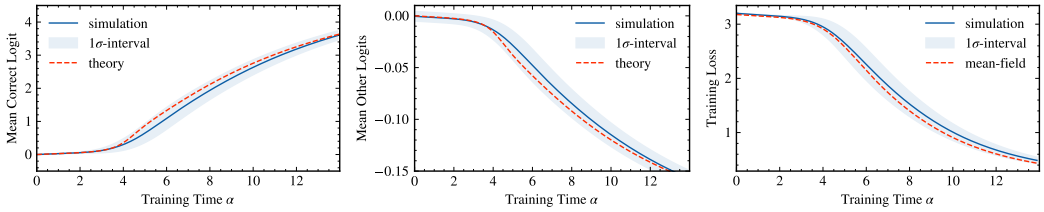

Figure 19: Mean correct/other logits and loss function.