What Happens During the Loss Plateau? Understanding Abrupt Learning in Transformers

Pulkit Gopalani [1] **Wei Hu** [1]

Abstract

Training Transformers on algorithmic tasks frequently demonstrates an intriguing *abrupt learning* phenomenon: an extended performance plateau followed by a sudden, sharp improvement. This work investigates the underlying mechanisms for such dynamics, primarily in shallow Transformers. We reveal that during the plateau, the model often develops an interpretable *partial solution* while simultaneously exhibiting a strong *repetition bias* in their outputs. This output degeneracy is accompanied by *internal representation collapse*, where hidden states across different tokens become nearly parallel. We further identify the slow learning of optimal attention maps as a key bottleneck. Hidden progress in attention configuration during the plateau precedes the eventual rapid convergence, and directly intervening on attention significantly alters plateau duration and the severity of repetition bias and representational collapse. We validate that these phenomena—repetition bias and representation collapse—are not artifacts of toy setups but also manifest in the early pre-training stage of LLMs like Pythia and OLMo.

1. Introduction

Training Transformers on mathematical or algorithmic tasks often exhibits "abrupt learning" in their training dynamics, where the model's performance plateaus at a suboptimal level for an extended period before suddenly and rapidly converging to the optimal solution (Nanda et al., 2023; Barak et al., 2022; Singh et al., 2024; Zhang et al., 2025; Wang et al., 2025) (Figures 1 and 2). This is often considered an example of the broader phenomenon of "emergence," where model capabilities appear to arise discontinuously with increasing amount of parameters, training data, or training steps (Wei et al., 2022).

The goal of this paper is to uncover universal characteristics and underlying mechanisms that define these training dynamics that are broadly applicable to a wide range of setups and tasks. We train small linear-Attention Transformers (1 or 2 layers) on a suite of simple algorithmic tasks such as moving-window-sum, permutation, and multi-digit addition, among others. These tasks have well-defined optimal solutions, allowing us to precisely measure the model's progress against a known ground truth. Furthermore, small models allow for tractable analysis and interpretation of internal model mechanisms.

We identify novel inductive biases that underlie the early plateau period of Transformer training: the model learns a partial solution while being biased toward degenerate patterns in its outputs and internal representations (see Figure 1 for an overview). We further study the pivotal role of attention map learning in driving these phenomena and overcoming the performance plateau. Finally, we demonstrate that key findings from these controlled small-scale studies extend to the pre-training dynamics of actual Large Language Models (LLMs).

2. Setup

We study the moving-window-sum (MWS) task, that involves computing the sliding-window sum (modulo p) of a length-n sequence over windows of size 2; i.e. $x_1, x_2, \ldots, x_n, \mathrm{SEP}, y_1, y_2, \ldots, y_n$,

$$y_i = \begin{cases} x_1 & i = 1 \\ (x_{i-1} + x_i) \bmod p & i \geq 2 \end{cases}$$

[1]University of Michigan, Ann Arbor, US. Correspondence to: Pulkit Gopalani <gopalani@umich.edu>.

Accepted at Methods and Opportunities at Small Scale (MOSS), ICML 2025, Vancouver, Canada.

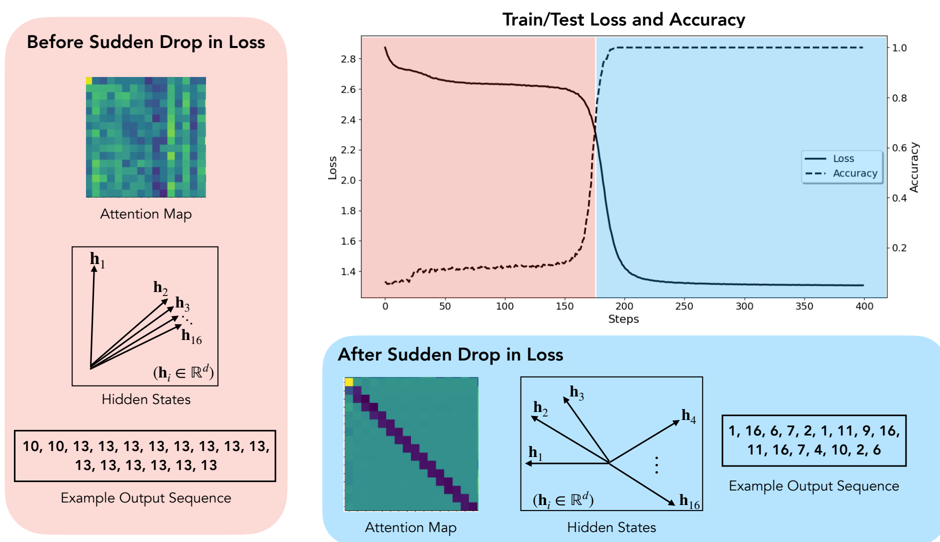


Figure 1. **Abrupt learning and related characteristics.** Training a shallow Transformer on algorithmic tasks like moving-window-sum exhibits *abrupt learning*: performance plateaus for an extended number of steps, before suddenly and sharply improving to optimum.

Here, $n = 16$, $x_i \sim \text{Unif}\{1, \ldots, 16\}, p = 17$, and $\text{SEP} = 17$ is a separator token. We train a 1-layer, 1-head Transformer with causal linear Attention on the above task in an online fashion (new batch of 256 samples from the distribution at every step), using Adam optimizer with a fixed stepsize 10^{-4} without weight decay. The training loss is the standard next-token-prediction cross-entropy loss, computed on the full sequence $(x_1, \ldots, x_n, \text{SEP}, y_1, \ldots, y_n)$. We measure accuracy over the output part $y_1, y_2, \ldots, y_n$. Please see Appendix A for implementation details, and Appendix B for details on measuring attention progress. We study other algorithmic tasks in Appendix E, and other model configurations / SGD in Appendix F.

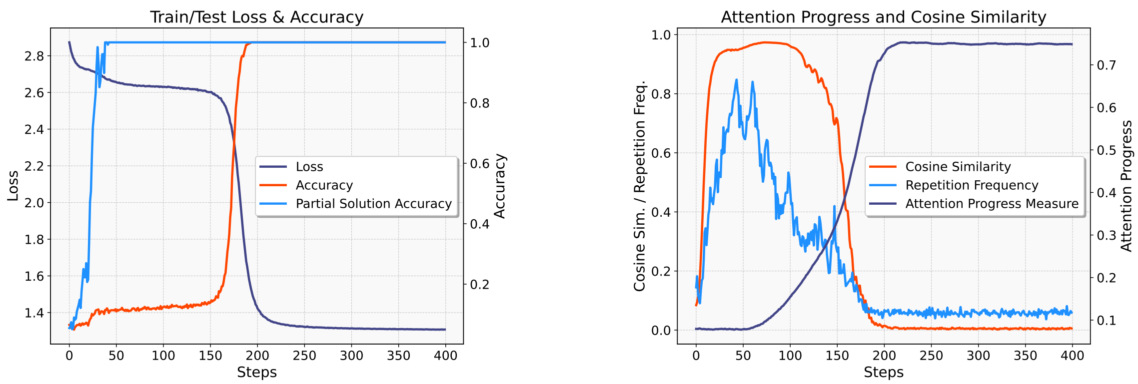
3. Inductive Biases in the Early Phase of Training

In this section, we characterize several key manifestations of inductive biases in the early phase of Transformer training.

Partial Solution. During the loss plateau, the model often has already learned to implement a *partial solution* to the task. This means it correctly predicts a subset of the output tokens, typically those corresponding to an intuitively simpler part of the problem, while failing on the more complex parts. For instance, in the MWS task, the model quickly learns to predict the first output token y_1 correctly, as it is simply x_1, while the overall loss remains high and accuracy on subsequent tokens is low (see Figure 2(a) for the first-token accuracy). Such partial solutions are observed for various algorithmic tasks (Table 1).

Repetition Bias. Concurrent with learning the partial solution, the model's outputs during the initial phase of training display a strong *repetition bias*, which refers to a tendency of the model to generate repetitive tokens of the form $x, x, x, \ldots$. To quantify such repetitions, we simply count the output tokens that equal the next one: for sequence $(y_1, y_2, \ldots, y_n)$, define its repetition frequency $\rho := \frac{1}{n-1} \sum_{i=1}^{n-1} \mathbf{1}[y_i = y_{i+1}]$. We observe that ρ increases rapidly during the early phase of training from a small initial value (see Figure 2(b)), while the optimal attention map has not been learned.

Representation Collapse. We further study the relation between the hidden representations at different output positions, and find a strong *representation collapse* phenomenon—these representations become nearly parallel in the early phase of training (except for the first output position which is correctly predicted in the partial solution). We measure the pairwise cosine similarity between hidden representations $\mathbf{h}_i, \mathbf{h}_j \in \mathbb{R}^d$ at positions i, j in the output sequence, $\text{COS}_{i,j} := \frac{\langle \mathbf{h}_i, \mathbf{h}_j \rangle}{\|\mathbf{h}_i\| \|\mathbf{h}_j\|}$ (this quantity is averaged over a random batch of sequences). We find that in the early phase of training, there is a rapid increase in $\text{COS}_{i,j}$—averaged over all output positions i, j except the first position, this quantity increases to ≈ 0.95 (Figure 2(b)). Similar to repetitions, representation collapse is not present at initialization and only appears after a few steps of training, in contrast to the *rank collapse* phenomenon for deep softmax-attention Transformers at initialization (Anagnostidis et al., 2022). While we measure the cosine similarity between hidden states just before the LM (classifier) head in the main paper, in Figure 29 we show that representation collapse happens in all intermediate layers.



(a) (b)

Figure 2. **Abrupt learning dynamics for the MWS task.** (a): Train/Test loss and Train/Test Accuracy (note that both train and test data metrics are near-identical in the online training setup, and thus we only report train metrics); (b): Attention Progress, Repetition Frequency, and Representation Cosine Similarity between hidden states.

4. The Role of Learning Attention

Observe that though the loss dynamics are abrupt, attention progress measure as well as repetitions and representation collapse are not (Figure 2(b)); that is, even when the loss is barely decreasing (between steps 50 and 150), attention progress measure notably increases, accompanied by a decrease in repetition frequency and representation collapse. We show in Appendix C that in the residual stream, representation collapse occurs after the attention layer during the early phase of training. Subsequently, via training-time interventions, we show that learning the attention map plays a crucial role in shaping the loss plateau as well as repetitions and representation collapse.

Biasing the Attention Map. To study the role of attention map, we slightly modify the training process starting at different time points in training, biasing it towards (or away from) the optimal attention map to check if repetitions, representation collapse, and loss plateau are reduced (resp. amplified). We do the following: at training time, starting at step t_0, we multiply the attention map values for output tokens except the first position at Ω (i.e. optimal attention map positions) by a constant $c > 0$; for $c > 1$, this implies biasing the model towards the final (optimal) attention map, whereas for $0 < c < 1$, this implies biasing the model away from the optimal attention map. We find that, for $c > 1$ and various values of t_0, such a scaling leads to lower average cosine similarity between hidden states, lower frequency of repetitions, and faster convergence (Figures 3 and 6). Whereas, for $0 < c < 1$, we find the opposite: the model is in representation collapse state for a longer time and converges later compared to the non-scaled ($c = 1$) case, while the repetition frequency remains large throughout the plateau (Figure 7, Appendix C). Moreover, we also show that fixing attention map to optimal value at the start of training leads to essentially no loss plateau or repetitions / representation collapse, while the same does not occur if we fix MLP / embeddings only to their optimal value (Figure 8, Appendix C). *Hence, learning the optimal attention map has a direct effect on shaping the loss dynamics as well as repetitions and representation collapse.*

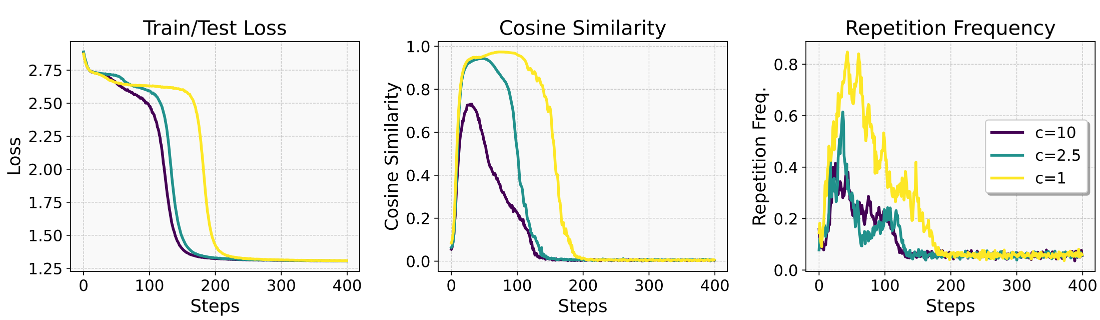


Figure 3. **Biasing attention map by** $c > 1$. We find that multiplicative biasing the attention map towards more weight to optimal positions leads to faster convergence, accompanied by less repetitions and average cosine similarity.

5. A Further Look at Repetition Bias

Having observed that Transformer models exhibit a strong repetition bias in the early phase of training, which co-occurs with the loss plateau, we now take a further look at this repetition bias and study how it might be affected by the amount of repetitions in the training sequences.

Beyond Repetitions in Consecutive Tokens. One hypothesis for the reason behind repetition bias is that the training data may consist of some repetitions, and the model may pick up these patterns and amplify them in the early phase of training. We show in Appendix D.1 that on the prefix sum task with low repetitions in the training data, there are still repetitions in output, albeit of a different nature than the contiguous repetitions seen for MWS.

Repetitive Sequences Are Easier to Learn. Motivated by evidence for repetition bias, we show in Appendix D.2 that training Transformers on such repetitive sequences leads to no loss plateau, and that the model can learn (inaccurate) repetitions quite early (after ≈ 10 training steps), verifying the early-phase repetition bias.

6. Repetition Bias and Representation Collapse in LLMs

Having shown that degenerate patterns of repetition bias and representation collapse are prevalent in small Transformers trained on algorithmic tasks, we check whether such phenomena occur during the early pre-training phase of LLMs as well. We show that this is indeed the case, using checkpoints of open-source LLMs Pythia (Biderman et al., 2023) and OLMo-2 (OLMo et al., 2024).

For Pythia models with 14M, 1B, 1.4B, and 2.8B parameters, we find strong representation collapse in the early training steps in their last layers (Figure 4). Specifically, we use 100 questions from the test split of the AI2 ARC-Easy dataset (Clark et al., 2018). For each question, we generate 8 tokens and compute the pairwise cosine similarity of the hidden states (see Appendix A for details). Figure 4 shows that at initialization, the average cosine similarity is relatively low (0.4-0.65), but within a few steps of training for all models, it sharply increases to > 0.9. These results remain similar if we use random sampling instead of greedy decoding (Figure 30). Further, the outputs for many prompts in the greedy decoding case are trivial repetitions of the same token, e.g., newline '\n', a clear manifestation of repetition bias.

For OLMo-2, at its earliest available training checkpoint (step 150, `OLMo-2-1124-7B`), the average representation cosine similarity in the above setup is ≈ 0.93; for the next checkpoint at step 600, this value has already decreased to ≈ 0.43 (similar for both greedy decoding and random sampling strategies). *Hence, repetition bias and representation collapse occur in the early pre-training phase of LLMs, validating our findings beyond toy settings.*

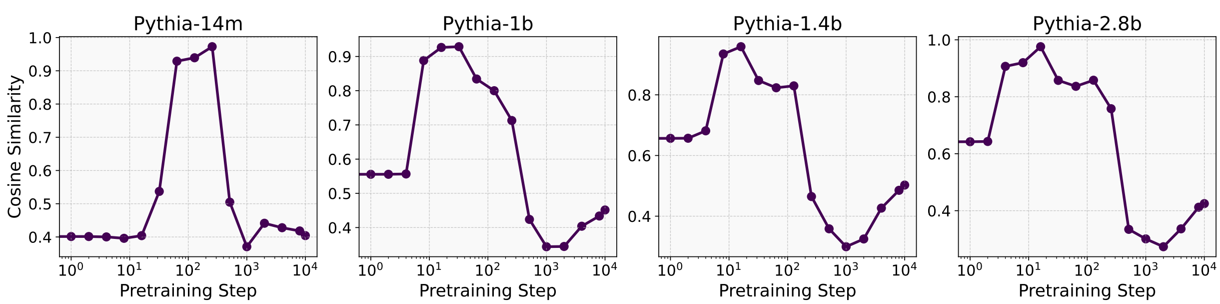


Figure 4. **Representation Collapse during Pythia Pretraining.** Representation collapse at different Pythia pretraining checkpoints; tokens generated via greedy decoding. We find that representation collapse occurs in the early phase of pre-training for these models as well, where average pairwise cosine-similarity for output tokens approaches ≈ 1.0 and reduces as training progresses.

7. Discussion

We identified repetition bias and representation collapse as key characteristics of the early-phase inductive biases of Transformer training, which are closely connected to the commonly observed loss plateau. The questions of *why* such representation collapse / repetition bias exists during early time training, *why* the initial rate of learning attention map is slow for algorithmic tasks, and how it connects to the intuitive "complexity" of the task are interesting questions for future research. We discuss related work on abrupt learning, grokking, repetitions etc. in Appendix H.

References

Anagnostidis, S., Biggio, L., Noci, L., Orvieto, A., Singh, S. P., and Lucchi, A. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=FxVH7iToXS.

Barak, B., Edelman, B. L., Goel, S., Kakade, S. M., eran malach, and Zhang, C. Hidden progress in deep learning: SGD learns parities near the computational limit. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=8XWP2ewX-im.

Barbero, F., Banino, A., Kapturowski, S., Kumaran, D., Araújo, J. G. M., Vitvitskyi, A., Pascanu, R., and Veličković, P. Transformers need glasses! information over-squashing in language tasks, 2024. URL https://arxiv.org/abs/2406.04267.

Belrose, N., Pope, Q., Quirke, L., Mallen, A., and Fern, X. Neural networks learn statistics of increasing complexity, 2024. URL https://arxiv.org/abs/2402.04362.

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Boix-Adsera, E., Littwin, E., Abbe, E., Bengio, S., and Susskind, J. Transformers learn through gradual rank increase, 2023. URL https://arxiv.org/abs/2306.07042.

Chen, A., Shwartz-Ziv, R., Cho, K., Leavitt, M. L., and Saphra, N. Sudden drops in the loss: Syntax acquisition, phase transitions, and simplicity bias in MLMs. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=MO5PiKHELW.

Chen, S., Sheen, H., Wang, T., and Yang, Z. Training dynamics of multi-head softmax attention for in-context learning: Emergence, convergence, and optimality, 2024b. URL https://arxiv.org/abs/2402.19442.

Choshen, L., Hacohen, G., Weinshall, D., and Abend, O. The grammar-learning trajectories of neural language models, 2022. URL https://arxiv.org/abs/2109.06096.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Cui, H., Behrens, F., Krzakala, F., and Zdeborová, L. A phase transition between positional and semantic learning in a solvable model of dot-product attention, 2024. URL https://arxiv.org/abs/2402.03902.

Edelman, E., Tsilivis, N., Edelman, B. L., eran malach, and Goel, S. The evolution of statistical induction heads: In-context learning markov chains. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=qaRT6QTIqJ.

Fu, Z., Lam, W., So, A. M.-C., and Shi, B. A theoretical analysis of the repetition problem in text generation, 2021. URL https://arxiv.org/abs/2012.14660.

Garg, S., Tsipras, D., Liang, P., and Valiant, G. What can transformers learn in-context? a case study of simple function classes, 2023. URL https://arxiv.org/abs/2208.01066.

Gopalani, P., Lubana, E. S., and Hu, W. Abrupt learning in transformers: A case study on matrix completion. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=O9RZAEp34l.

Hiraoka, T. and Inui, K. Repetition neurons: How do language models produce repetitions?, 2025. URL https://arxiv.org/abs/2410.13497.

Hoffmann, D. T., Schrodi, S., Bratulić, J., Behrmann, N., Fischer, V., and Brox, T. Eureka-moments in transformers: Multi-step tasks reveal softmax induced optimization problems, 2024. URL https://arxiv.org/abs/2310.12956.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration, 2020. URL https://arxiv.org/abs/1904.09751.

Hoogland, J., Wang, G., Farrugia-Roberts, M., Carroll, L., Wei, S., and Murfet, D. Loss landscape degeneracy drives stagewise development in transformers, 2025. URL https://arxiv.org/abs/2402.02364.

Inoue, M., Park, H., and Okada, M. On-line learning theory of soft committee machines with correlated hidden units –steepest gradient descent and natural gradient descent–. *Journal of the Physical Society of Japan*, 72(4):805–810, April 2003. ISSN 1347-4073. doi: 10.1143/jpsj.72.805. URL http://dx.doi.org/10.1143/JPSJ.72.805.

Jacot, A., Ged, F., Şimşek, B., Hongler, C., and Gabriel, F. Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity, 2022. URL https://arxiv.org/abs/2106.15933.

Karpathy, A. Karpathy/mingpt: A minimal pytorch re-implementation of the openai gpt (generative pretrained transformer) training, 2022. URL https://github.com/karpathy/minGPT.

Kumar, T., Bordelon, B., Gershman, S. J., and Pehlevan, C. Grokking as the transition from lazy to rich training dynamics, 2024. URL https://arxiv.org/abs/2310.06110.

Lee, N., Sreenivasan, K., Lee, J. D., Lee, K., and Papailiopoulos, D. Teaching arithmetic to small transformers, 2023. URL https://arxiv.org/abs/2307.03381.

Li, H., Lan, T., Fu, Z., Cai, D., Liu, L., Collier, N., Watanabe, T., and Su, Y. Repetition in repetition out: Towards understanding neural text degeneration from the data perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=WjgCRrOgip.

Liu, Z., Kitouni, O., Nolte, N., Michaud, E. J., Tegmark, M., and Williams, M. Towards understanding grokking: An effective theory of representation learning, 2022. URL https://arxiv.org/abs/2205.10343.

Lubana, E. S., Kawaguchi, K., Dick, R. P., and Tanaka, H. A percolation model of emergence: Analyzing transformers trained on a formal language, 2024. URL https://arxiv.org/abs/2408.12578.

Lyu, K., Jin, J., Li, Z., Du, S. S., Lee, J. D., and Hu, W. Dichotomy of early and late phase implicit biases can provably induce grokking, 2024. URL https://arxiv.org/abs/2311.18817.

Merrill, W., Tsilivis, N., and Shukla, A. A tale of two circuits: Grokking as competition of sparse and dense subnetworks, 2023. URL https://arxiv.org/abs/2303.11873.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

OLMo, T., Walsh, P., Soldaini, L., Groeneveld, D., Lo, K., Arora, S., Bhagia, A., Gu, Y., Huang, S., Jordan, M., Lambert, N., Schwenk, D., Tafjord, O., Anderson, T., Atkinson, D., Brahman, F., Clark, C., Dasigi, P., Dziri, N., Guerquin, M., Ivison, H., Koh, P. W., Liu, J., Malik, S., Merrill, W., Miranda, L. J. V., Morrison, J., Murray, T., Nam, C., Pyatkin, V., Rangapur, A., Schmitz, M., Skjonsberg, S., Wadden, D., Wilhelm, C., Wilson, M., Zettlemoyer, L., Farhadi, A., Smith, N. A., and Hajishirzi, H. 2 olmo 2 furious, 2024. URL https://arxiv.org/abs/2501.00656.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Prieto, L., Barsbey, M., Mediano, P. A. M., and Birdal, T. Grokking at the edge of numerical stability, 2025. URL https://arxiv.org/abs/2501.04697.

Reddy, G. The mechanistic basis of data dependence and abrupt learning in an in-context classification task, 2023. URL https://arxiv.org/abs/2312.03002.

Rende, R., Gerace, F., Laio, A., and Goldt, S. A distributional simplicity bias in the learning dynamics of transformers, 2025. URL https://arxiv.org/abs/2410.19637.

Saad, D. and Solla, S. A. On-line learning in soft committee machines. *Phys. Rev. E*, 52:4225–4243, Oct 1995. doi: 10.1103/PhysRevE.52.4225. URL https://link.aps.org/doi/10.1103/PhysRevE.52.4225.

Singh, A. K., Moskovitz, T., Hill, F., Chan, S. C. Y., and Saxe, A. M. What needs to go right for an induction head? a mechanistic study of in-context learning circuits and their formation, 2024. URL https://arxiv.org/abs/2404.07129.

Varma, V., Shah, R., Kenton, Z., Kramár, J., and Kumar, R. Explaining grokking through circuit efficiency, 2023. URL https://arxiv.org/abs/2309.02390.

Wang, M., Yu, R., E, W., and Wu, L. How transformers get rich: Approximation and dynamics analysis, 2025. URL https://arxiv.org/abs/2410.11474.

Wang, W., Li, Z., Lian, D., Ma, C., Song, L., and Wei, Y. Mitigating the language mismatch and repetition issues in llm-based machine translation via model editing, 2024. URL https://arxiv.org/abs/2410.07054.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=yzkSU5zdwD. Survey Certification.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Huggingface's transformers: State-of-the-art natural language processing, 2020. URL https://arxiv.org/abs/1910.03771.

Xu, J., Liu, X., Yan, J., Cai, D., Li, H., and Li, J. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation, 2022. URL https://arxiv.org/abs/2206.02369.

Xu, Z., Wang, Y., Frei, S., Vardi, G., and Hu, W. Benign overfitting and grokking in relu networks for xor cluster data, 2023. URL https://arxiv.org/abs/2310.02541.

Yao, J., Yang, S., Xu, J., Hu, L., Li, M., and Wang, D. Understanding the repeat curse in large language models from a feature perspective, 2025. URL https://arxiv.org/abs/2504.14218.

Zhang, Y., Singh, A. K., Latham, P. E., and Saxe, A. Training dynamics of in-context learning in linear attention, 2025. URL https://arxiv.org/abs/2501.16265.

A. Detailed Setup and Experimental Details

Model Architecture. We use a 1-layer, 1-head Transformer with causal masking and linear attention. This simple architecture can already solve the MWS task to perfect accuracy. Formally, for a sequence of tokens $(s_1, \ldots, s_L)$, the Transformer output is,

$$\mathrm{TF}_\theta(s_1, s_2, \ldots, s_L) = \mathrm{LM} \circ (\mathrm{Id} + \mathrm{MLP}) \circ (\mathrm{Id} + \mathrm{Attn}) \circ \mathrm{Embed}(s_1, s_2, \ldots, s_L)$$

where Embed outputs sum of token and absolute positional embeddings $h_i \in \mathbb{R}^d$, and Attn denotes the causal-linear-Attention operation that combines tokens such that output at i^{th} position is,

$$[\mathrm{Attn}(h_1, h_2, \ldots, h_L)]_i = W_O \left(\sum_{j=1}^{i} (h_j^\top W_K^\top W_Q h_i) W_V h_j \right); \quad W_O, W_K, W_Q, W_V \in \mathbb{R}^{d \times d}.$$

MLP denotes the 2-layer neural net $h_i \mapsto W_2(\sigma(W_1 h_i))$ for $W_2 \in \mathbb{R}^{d \times 4d}, W_1 \in \mathbb{R}^{4d \times d}$, and σ the GELU activation. LM is a linear layer that maps the hidden state $h_i \in \mathbb{R}^d$ to logits $v_i \in \mathbb{R}^{|V|}$ (V denotes the vocabulary for a task; for instance, for MWS task, $V = \{0, 1, \ldots, 17\}$). Note that all linear maps above implicitly include a bias term, and we use pre-LayerNorm so that before Attn, MLP, and LM, a LayerNorm operation is applied to the hidden states h_i. For generating sequences, we use greedy decoding i.e. output token is determined by the maximum logit over the vocabulary. We use linear attention to avoid vanishing gradient issues from softmax attention being a contributing factor toward abrupt learning, which was argued in (Hoffmann et al., 2024). We also show similar results on softmax attention, multi-layer / multi-head models, and models with varying d in Appendix F.

Training. The model is trained to minimize the standard next-token-prediction cross-entropy loss over the full sequence i.e. $(x_1, \ldots, x_n, \mathrm{SEP}, y_1, \ldots y_n)$ for the MWS task. We evaluate accuracy over the output portion of the sequence, i.e., $y_1, \ldots, y_n$, averaged over these n positions. We use the Adam optimizer with a constant learning rate 10^{-4} and no weight decay. The training is conducted in an online / single-epoch fashion, where a new batch of 256 training samples is drawn from the data distribution at each training step. Note that in this setup, the training and test losses essentially coincide. For completeness, we also show similar results on the SGD optimizer in Appendix F.

Causal Linear Attention. Linear attention transformer is obtained simply by removing the softmax activation function when computing the attention map, and setting the causal mask to 0 instead of $-\infty$. We use the existing minGPT implementation (Karpathy, 2022) (MIT licence) for our experiments, modifying the code as above and wherever required.

LLM Experiments. We use Pythia (Biderman et al., 2023) / OLMo-2 (OLMo et al., 2024) pretrained models (Apache 2.0 Licence) hosted on Huggingface Transformers (Wolf et al., 2020) and run them on the ARC-Easy dataset (Clark et al., 2018) (CC-BY-SA 4.0 Licence). We set the `use_cache=False` in the `generate` function, and use the hidden state used for predicting each of the 8 output tokens. For random sampling, we use `do_sample=True` (using default temperature value), using `do_sample=False` for our greedy decoding results.

B. Abrupt Learning and Attention Map

Abrupt Learning. Following the training procedure described above will result in a characteristic *abrupt learning* curve, where the training/test loss is stuck at some sub-optimal value for a significant number of steps, before suddenly and rapidly decreasing to its optimal value (Figure 2(a)). This drop in loss is accompanied by a similarly rapid increase in accuracy, indicating that the optimal solution is learned abruptly.

Attention Map. We analyze the attention map at different points during training. We find that the attention map shows a sparse, interpretable pattern after the sudden loss drop, while no such pattern is shown before the sudden drop (Figure 1). For the MWS task, this optimal attention pattern corresponds to each output token y_i attending only to the input tokens relevant to its computation, i.e., attending to x_1 for y_1, and to x_i, x_{i-1} for $y_i, i \geq 2$. We further use an *Attention Progress Measure (APM)* to record the progress of the attention map toward its optimal pattern during training, defined as

$$\mathrm{APM} := \frac{\sum_{(i,j)\in\Omega} |A_{ij}|}{\sum_{(i,j)} |A_{ij}|},$$

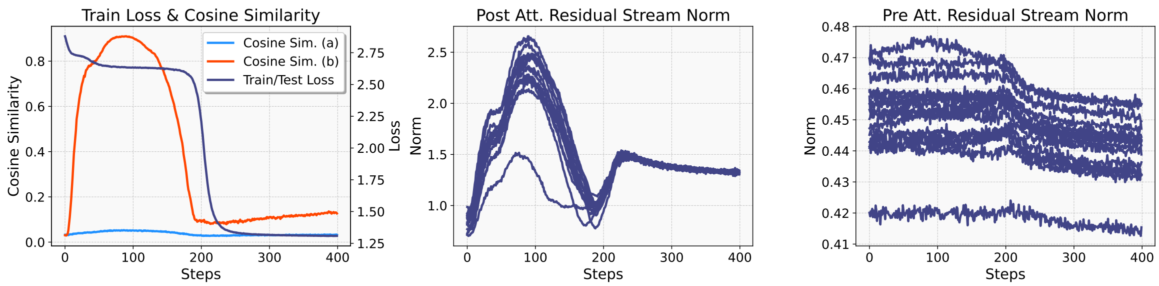


Figure 5. Norm and representation collapse dynamics for (a) pre- and (b) post-attention residual streams for all positions i, j except the first position.

where A_{ij} denotes the attention score allocated to the j^{th} token when computing output at the i^{th} position in the sequence, and Ω is the set of position pairs in the optimal attention map. This measure is defined with absolute values due to our choice of linear attention so that A_{ij} could be positive or negative. In experiments, we calculate APM averaged over a random batch of sequences. Figure 2(b) shows that the APM monotonically increases from near 0 to near 0.8 during training, and its increase is more gradual than the loss/accuracy dynamics. In particular, APM already increases to a nontrivial value during the loss plateau and before the sudden loss drop.

C. The Role of Learning Attention

Representation Collapse Occurs After the Attention Layer. We verify whether the attention layer is responsible for representation collapse during the early phase of training. To this end, we plot the cosine similarity of the residual stream for output tokens just before and after the attention layer. Formally, let the residual stream before attention layer (i.e., token + positional embeddings) be $\mathbf{h}_i \in \mathbb{R}^d$, and the residual stream after attention layer be $\mathbf{h}'_i \in \mathbb{R}^d$, we measure the norm and pairwise cosine similarity for $\mathbf{h}_i$ and $\mathbf{h}'_i$ in Figure 5.

We find that in the early phase of training, the cosine similarity between different positions in the post-attention residual stream representations approaches 1.0 rapidly, which is not the case for pre-attention. Furthermore, the norm of $\mathbf{h}'_i$ grows rapidly in this phase, while the norm of $\mathbf{h}_i$ remains near-constant. *Hence, in the residual stream, representation collapse occurs after the attention layer during the early phase of training.*

Biasing the Attention Map. We do the following: at training time, starting at step t_0, we multiply the attention map values for output tokens except the first position at Ω (i.e. optimal attention map positions) by a constant $c > 0$; for $c > 1$, this implies biasing the model towards the final (optimal) attention map, whereas for $0 < c < 1$, this implies biasing the model away from the optimal attention map.

We find that, for $c > 1$ and various values of t_0, such a scaling leads to lower average cosine similarity between hidden states, lower frequency of repetitions, and faster convergence (Figures 3 and 6). Whereas, for $0 < c < 1$, we find the opposite: the model is in representation collapse state for a longer time and converges later compared to the non-scaled

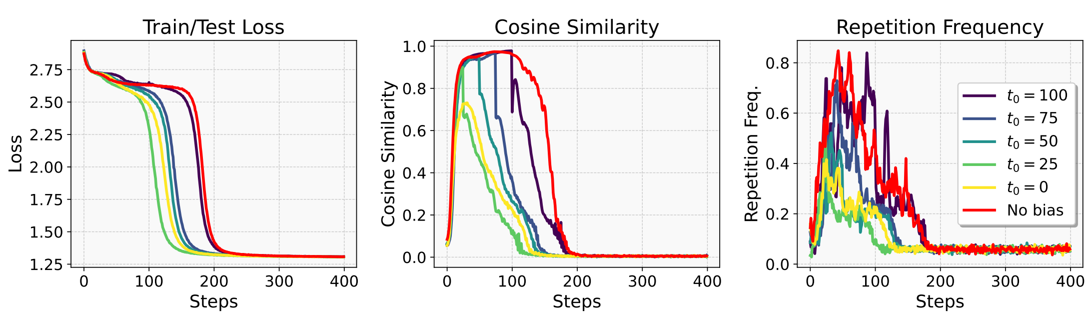


Figure 6. Biasing attention map by $c = 10$ at different t_0 during training.

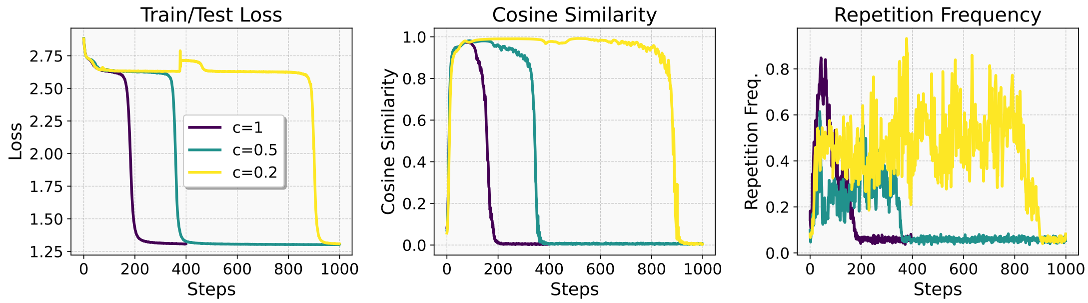


Figure 7. **Biasing attention map by** $c < 1$. We find that biasing the attention map to have lesser weight at optimal positions leads to slower convergence, and more representation collapse and repetitions.

$(c = 1)$ case, while the repetition frequency remains large throughout the plateau (Figure 7).

For example, for $t_0 = 0, c = 10$, i.e. scaling $10\times$ from the start of training, we find that the peak cosine similarity attained during training is ≈ 0.6, much smaller than the ≈ 0.95 attained for $c = 1$, and further the peak for $c = 10$ is for negligible duration compared to that for $c = 1$. Later values of $t_0 = 25, 50, 75$ show similar results wherein the cosine similarity drops immediately on the above biasing operation, followed by lower repetition frequency and convergence to optimal solution (Figure 6).

On the other hand, for $t_0 = 0, c = 0.2, 0.5$, the model takes much longer to converge and is in representation collapse / large repetition frequency state for much longer. This is in line with our expectation that lower attention map values for the optimal positions lead to slower learning and prolonged representation collapse. *Hence, learning the optimal attention map has a direct effect on shaping the loss dynamics as well as repetitions and representation collapse.*

Training with Optimal Attention. In this test, we initialize with the optimal attention map by fixing embeddings, LayerNorm for attention layer and attention layer weights to their final values at the end of a normal training run, so that at initialization, the correct attention map is already available to subsequent layers. We re-train the subsequent non-fixed layers starting from random initialization.

For the attention layer, we choose the set of parameters to initalize in 2 ways: (a) only Key, Query (W_K, W_Q) weights, and (b) All of Key, Query, Value, Output (W_K, W_Q, W_V, W_O) weights. We find that in both of these cases, learning only the subsequent layers (i.e. MLP, LM Head) take significantly shorter time than training the full model, without any significant representation collapse, repetitions or plateau in loss (Figure 8). Further, between (a) and (b), we find that additionally having W_O, W_V layers initialized to optimal values slightly speeds up learning, and average cosine similarity goes up to approx 0.15 instead of ≈ 0.45 when only initializing W_K, W_Q weights. This indicates that W_O, W_V layers also play a non-trivial role in causing representation collapse. *This result confirms that attention map is a major bottleneck that leads to early representation collapse and loss plateau.*

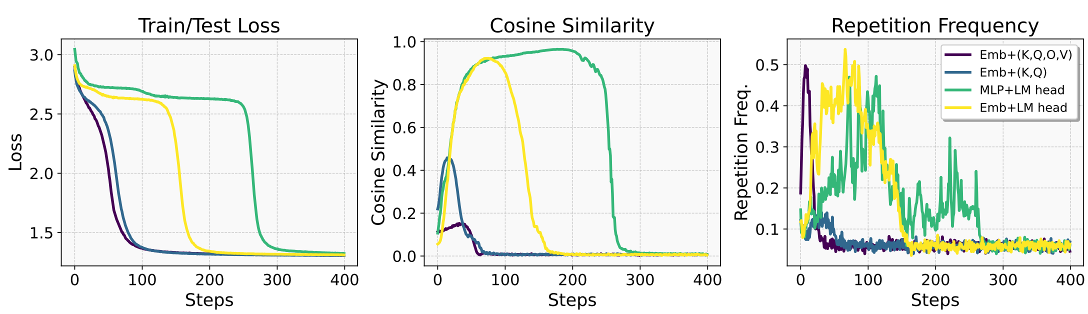


Figure 8. **Different optimal initializations and effect on training.** We find that fixing attention and embedding weights (i.e. attention map) to optimal value, and training other components leads to faster convergence and lesser representation collapse / repetitions. Similar effect does not hold for fixing optimal MLP or Embeddings. $(K, Q, O, V$ respectively denote the parameters W_K, W_Q, W_O, W_V.)

Optimal MLP or Embeddings Do Not Help. On the other hand, fixing MLP or embeddings (together with LM head) to their final optimal values and re-training the other components does not qualitatively change the training dynamics from the full training case, i.e., a significant loss plateau, repetition bias, and representation collapse still occur (Figure 8). *This indicates that there is little benefit from having the optimal MLP or embeddings at initialization compared to attention map.*

D. A Further Look at Repetition Bias

D.1. Beyond Repetitions in Consecutive Tokens

One hypothesis for the reason behind repetition bias is that the training data may consist of some repetitions, and the model may pick up these patterns and amplify them in the early phase of training. To investigate this, we consider a task with low repetitions in the training data. In particular, we consider the *prefix sum* task, where the outputs $y_1, \dots, y_n$ are defined as $y_i = (\sum_{j=1}^{i} x_j) \bmod p$. Our choice of the input distribution ensures that there is no repetition in consecutive output positions (i.e., $y_i \neq y_{i+1}$ for all i). Indeed, training a Transformer on the prefix sum task does not result in a significant increase in the repetition frequency at any point in training, unlike the MWS task. Nevertheless, in the early training phase, we still observe that only a few tokens appear repeatedly in the model output though not contiguously as in the MWS task. Therefore, we consider an alternative measure of repetitions based on entropy: for an output sequence $y_1, y_2, \dots, y_n$, we define

$$\mathrm{SeqEnt}(y_1, \dots, y_n) := \sum_{i=1}^{|V|} p_i \log(1/p_i); \quad p_i = \frac{|\{y_j = v_i, j \in [n]\}|}{n}$$

i.e. simply the entropy of the empirical distribution of tokens in the sequence. Intuitively, the entropy is lower if most probability mass is concentrated at a few tokens, and larger if the tokens are more uniformly distributed. We find that the model output entropy quickly goes to quite low values early in training compared to the entropy of ground-truth data (Figure 9), indicating that the model still has a form of repetition bias. Further, representation collapse still happens in the early phase, with the average cosine similarity going to 0.8 during the plateau.

Hence, we find that repetition bias might take different forms depending on the task, but still robustly occurs in the early phase of training.

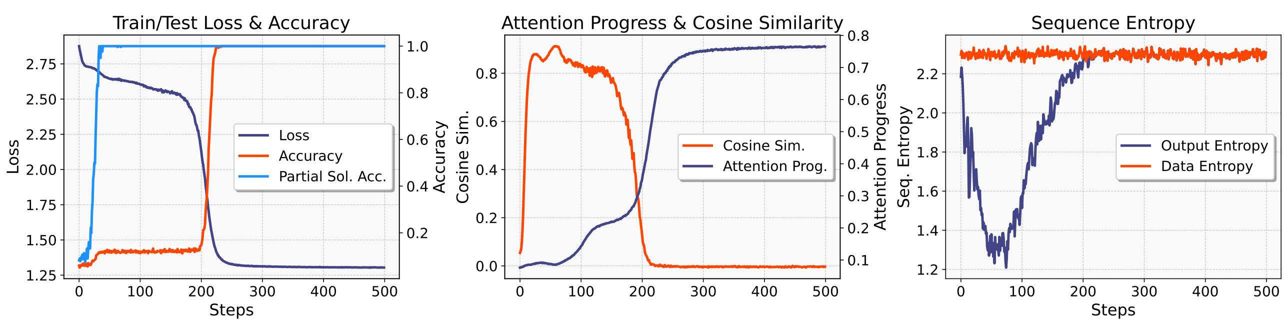


Figure 9. **Prefix sum task training dynamics.** While the usual contiguous repetitions do not occur for this task, an alternate form of repetition occurs in terms of having only a few distinct tokens in the output sequence. 'Sequence entropy' quantifies this repetition by measuring the entropy of the empirical distribution of tokens in a sequence, and averaging this entropy over a batch of sequences.

D.2. Repetitive Sequences Are Easier to Learn

On the other hand, we study what happens when the ground-truth data have a lot of repetitions. We consider a simple task REPEAT_1 of the form $x_1, x_2, \dots, x_n, \mathrm{SEP}, y_1, y_2, \dots, y_n$, where $y_i = x_1 \; \forall i$. Unlike other tasks, the loss curve for REPEAT_1 does not have any noticeable plateau, though the accuracy still shows a small plateau period (Figure 10). This observation indicates that such repetitive sequences are easier from an optimization perspective and hence likely "preferred" during the early stage of training. In fact, just one gradient step is sufficient to bring the average representation cosine similarity to ≈ 0.5.

To further understand the early training phase model output, we define another metric α_1 that measures to what extent the

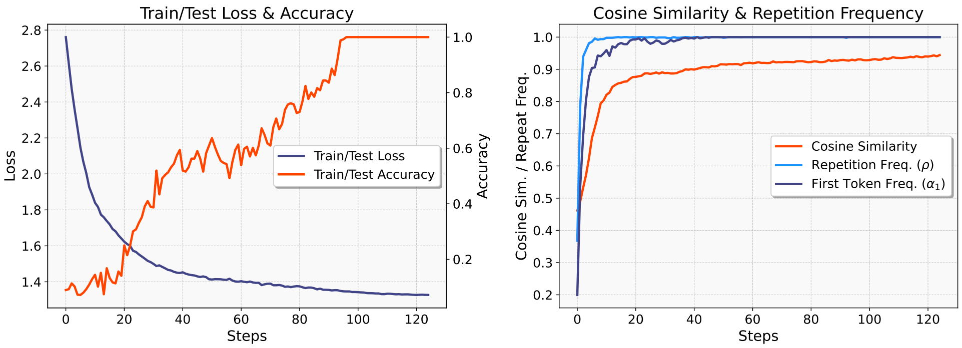


Figure 10. **REPEAT$_1$ training dynamics.** We find that there is no observable plateau in loss on training a 1-layer transformer on the REPEAT$_1$ task. Moreover, metrics like ρ (repetition frequency) and α_1 increase to near-perfect values rapidly in the early phase of training, indicating a model bias towards learning repetitive sequences.

model simply outputs the same token for all output positions: $\alpha_1 = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[y_i = y_1]$. Note that this is distinct from accuracy, in that the model might output the wrong y_1, however repeats y_1 at $y_2, \ldots, y_n$. We find that α_1 rapidly increases to near perfect values (> 0.9) in the early phase of training, showing that the model tends to repeat the first token identically at most positions, even though the output token itself might be incorrect. *Hence, repetitive sequences appear to be inherently easier for the Transformer to learn, and this is likely the reason for repetition bias in the early phase of training.*

We define 2 variants of the above task, REPEAT$_2$ and REPEAT$_4$ denoting the number of distinct repetitive blocks in the sequences for those tasks. The training dynamics for these tasks in Figures 11(a) and 12(a) show that similar to REPEAT$_1$, the training loss does not exhibit any plateau. Moreover, the repetition frequency ρ and metric α_1 increase rapidly to ≈ 1.0 early on in training.

REPEAT$_2$ This task is defined as,

$$y_i = \begin{cases} x_1 & 1 \leq i \leq 8 \\ (x_1 + 1) \bmod 17 & 9 \leq i \leq 16 \end{cases}$$

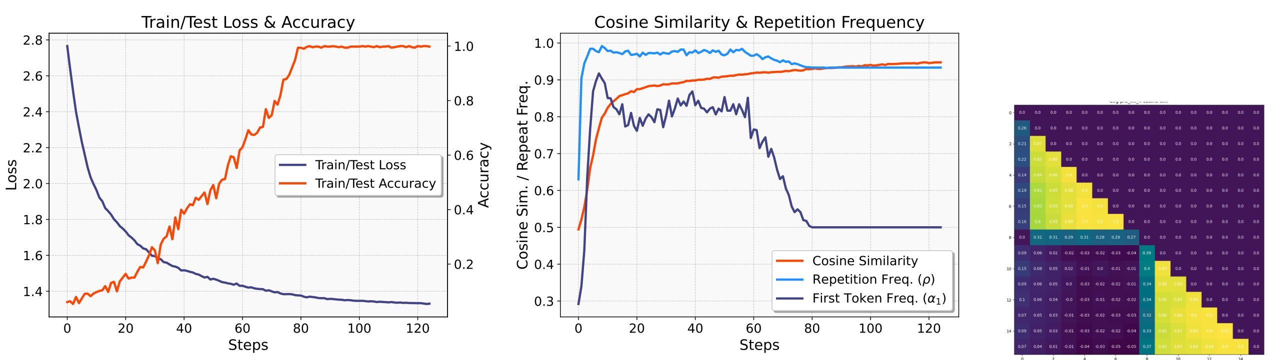


Figure 11. **REPEAT$_2$ training dynamics.** Note that there is no plateau in loss, similar to the REPEAT$_1$ task. Further, the pairwise cosine similarity for hidden states take a specific form indicating the blocks of repeated tokens in the output (marked yellow), over which we compute the average cosine similarity reported in (a).

REPEAT$_4$ This task is defined as

$$y_i = \begin{cases} x_1 & 1 \leq i \leq 4 \\ (x_1 + 1) \bmod p & 5 \leq i \leq 8 \\ (x_1 + 2) \bmod p & 9 \leq i \leq 12 \\ (x_1 + 3) \bmod p & 13 \leq i \leq 16 \end{cases}$$

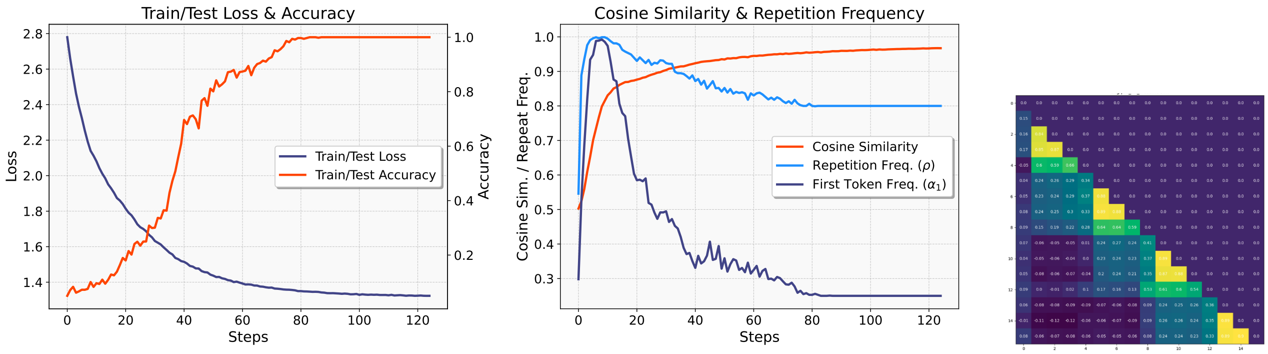


Figure 12. **REPEAT_4 training dynamics.** Similar to REPEAT_2, there is no plateau in loss. The pairwise cosine similarity for hidden states takes a form (marked yellow) indicating the blocks of repeated tokens in the output, over which we compute the average cosine similarity reported in (a).

E. Results for Other Algorithmic Tasks

This section presents results on a suite of algorithmic tasks, verifying the generality of our identified phenomena.

Table 1. Algorithmic tasks that show abrupt learning and partial solution during plateau

Task	**Description**	**Partial Solution**
Moving Window Sum (MWS)	Sum over moving window of 2 elements, copy 1st element	First input element
Prefix Sum (PRE)	Compute prefix sum of a given n–length sequence	First input element
Permutation (PER)	Permute an $n-$length sequence by given permutation	Incorrect permutation of input sequence
Multi-Digit Addition (ADD)	Add atmost–n–digit numbers	First digit (0 or 1) i.e. total carry-over from n digits
Histogram (HIST)	Compute counts of each element in n–length sequence	$\approx 100\%$ Repetitive sequences
Reverse (REV)	Reverse n–length input sequence	Repetitive sequences[1]
Copy (COPY)	Copy n–length input sequence	Repetitive sequences[1]

([1]*The loss plateau is very brief, hence a partial solution like other cases is not applicable.*)

E.1. Multi-Digit Addition

This task involves adding 2 atmost 4–digit numbers; if the numbers are represented as $a = \overline{a_1 a_2 a_3 a_4},\ b = \overline{b_1 b_2 b_3 b_4}$ and their sum $a + b = c = \overline{c_0 c_1 c_2 c_3 c_n}$ then the training sequences for ADD are of the form

$$a_1, a_2, a_3, a_4, +, b_1, b_2, b_3, b_4, =, c_4, c_3, c_2, c_1, c_0$$

Note that the output sequence is reversed, following the observations from (Lee et al., 2023). We find similar abrupt learning characteristics (Figure 13), partial solution in this case being c_0 i.e. total carry-over from 4 single digit add operations. An interpretable attention map learnt for the output sequence is shown in Figure 14.

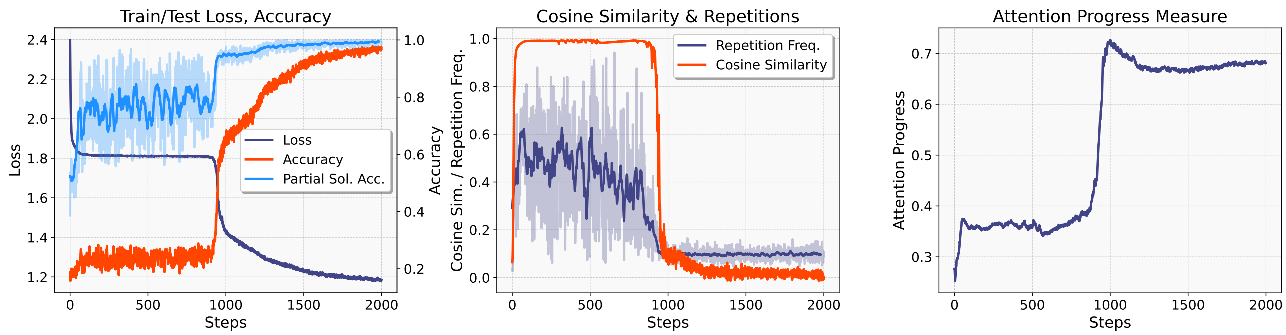


Figure 13. Training dynamics for Add task. (left) Train/Test Loss, Accuracy and Partial solution progress (c_0 accuracy); (middle) Repetition frequency and representation collapse; (right) Attention progress measure.

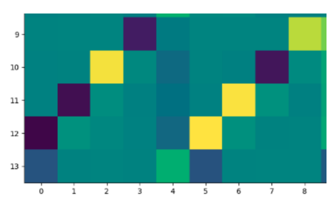


Figure 14. Attention map for add task, note that the model attends to the relevant digits in the input numbers, and to somewhat lesser extent to the preceding digits as well (highlighted positions show entries with larger magnitude).

E.2. Prefix sum

This task involves computing the cumulative (prefix) sum of an $n-$length sequence of integers, so that the training sequences in PRE are of the form $(n = 16, \mathrm{SEP} = 17)$,

$$x_1, x_2, \ldots, x_n, \mathrm{SEP}, y_1, y_2, \ldots, y_n$$

$$y_i = \left(\sum_{j=1}^{i} x_j\right) \bmod 17 \quad \forall i \in [n]$$

Training dynamics for this task are shown in Fig. 15 which show similar abrupt learning behavior as MWS and partial solution learning for y_1. The interpretable attention map learnt for this task is shown in Figure 16.

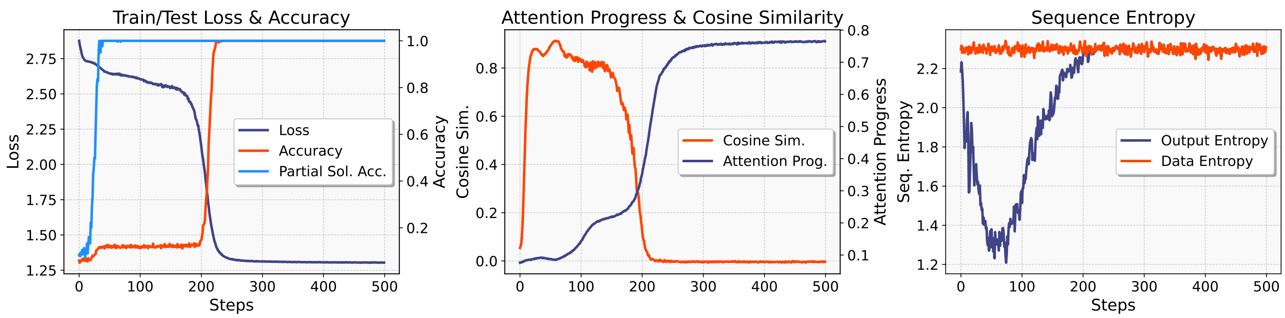


Figure 15. Training dynamics for Prefix sum task. (left) Train/Test Loss, Accuracy and Partial solution progress (y_1 accuracy); (middle) Attention progress and representation collapse; (right) SeqEnt for data and model output sequences.

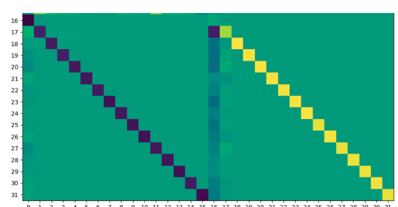


Figure 16. Attention map for Prefix sum task, that uses the relevant token in the input, as well as the previous token in the output to track prefix sum (highlighted positions show entries with larger magnitude).

E.3. Permutation

This task involves training a 2-layer, 1-head Transformer on permuting a length$-n$ sequence using the permutation π, which is generated at random and is distinct for each training sequence. Formally, for a sequence of positive integers $(x_1, \ldots, x_n)$ and a permutation $(\pi_1, \ldots, \pi_n)$ over $[n]$, training sequences for PER, $k = 0, 1, 2, \ldots$ are given by

$$x_1, \ldots, x_n, \text{SEP}, \pi_1, \ldots, \pi_n, \text{SEP}, x_{\pi_1}, \ldots, x_{\pi_n}$$

where $x_i \sim \text{Unif}\{17, 18, \ldots, 32\}, n = 16, \text{SEP} = 0$. The partial solution in this case is the output sequence being an permutation of the input sequence $x_1, \ldots, x_n$ i.e., it learns to copy the tokens correctly, but in wrong order (Figure 17).

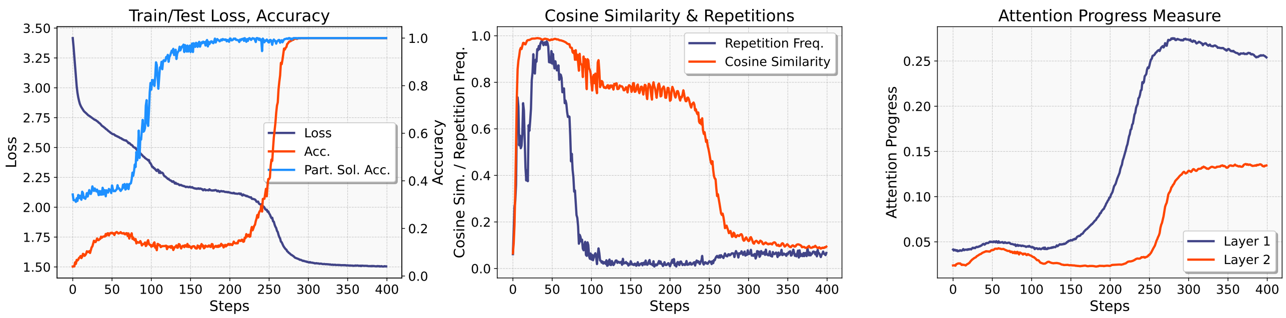


Figure 17. Training dynamics for Permutation task. (left) Train/Test Loss, Accuracy and Partial solution progress; (middle) Repetition frequency and representation collapse; (right) Attention progress measure. Note that the repetition frequency decreases by step 100, which is followed by the partial solution.

(a) Layer 1 (b) Layer 2

Figure 18. Attention maps for the 2 layer Transformer used for Permutation task; highlighted positions show entries with larger magnitude. (a) Attention map in Layer 1 where rows are attention weights over the input part $x_1, x_2, \ldots, x_n$ of the sequence. The highlighted positions are attending to $\pi_1, \pi_2, \ldots, \pi_n = 5, 15, 4, 14, 3, 13, 6, 10, 11, 9, 16, 1, 12, 8, 2, 7.$ for index $i \in [n]$.; (b) Attention map in Layer 2; the rows (output part of the sequence) are attention scores over the part of sequence to which Layer 1 attention map copies the correctly permuted tokens. This implies that this attention map simply copies the correct token from the residual stream after Layer 1.

E.4. Histogram

This task (Cui et al., 2024) involves computing the counts of elements in the input sequence, and training sequences are of the form

$$x_1, x_2, \dots, x_n, \mathrm{SEP}, y_1, y_2, \dots, y_n$$

$$y_i = \sum_{j=1}^{n} \mathbf{1}[x_j = x_i]$$

where $x_i \sim \mathrm{Unif}\{1, 2, \dots, 12\}, n = 16, \mathrm{SEP} = 0$. We train a 2-layer, 1-head transformer for this task, with gradient clipping (1.0) to avoid loss spikes (Figure 19). We note that the repetition bias in this case is quite strong which leads to $\approx 100\%$ repetitions in the early phase of training, and which we characterize as partial solution for this task. Further we only consider the attention map from layer 1 Figure 20 since this is the most consistent and clearly interpretable across runs, and indicates an identity-map-like function.

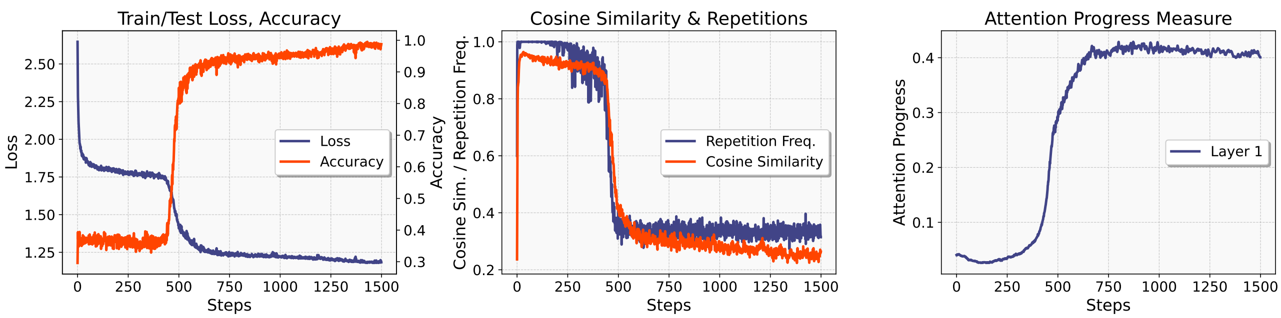


Figure 19. Training dynamics for Histogram task. (left) Train/Test Loss and Accuracy; (middle) Repetition frequency and representation collapse; (right) Attention progress measure. We only measure attention progress for the 1st layer, since that is the one that consistently and clearly shows an interpretable pattern (Figure 20).

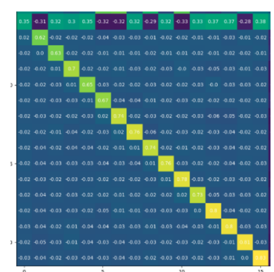


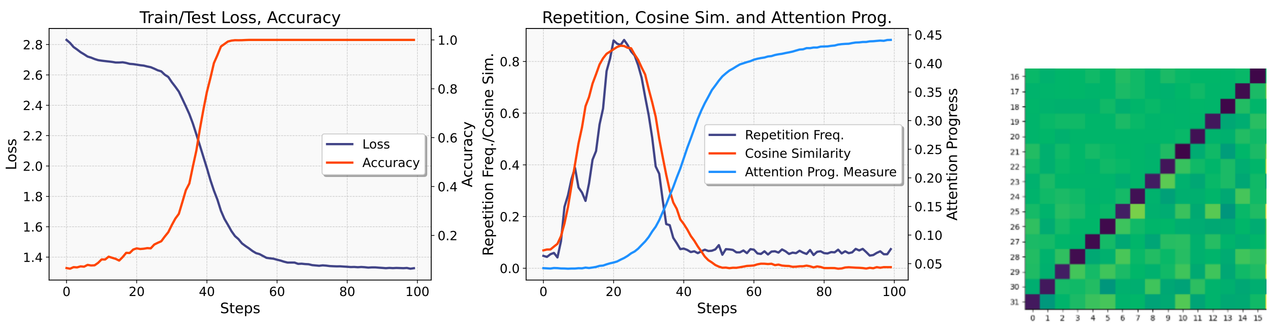
Figure 20. Attention map in layer 1 for histogram task, where rows for the latter half of the sequence compute attention weights over the input tokens x_i, similar to an identity map (highlighted positions show entries with larger magnitude).

E.5. Reverse

This is the task of reversing the input sequence, so that the training sequences for reverse task REV are given as,

$$x_1, x_2, \ldots, x_n, \mathrm{SEP}, x_n, x_{n-1}, \ldots, x_1$$

for $x_i \sim \mathrm{Unif}\{1, 2, \ldots, 16\}, n = 16, \mathrm{SEP} = 0$. The training dynamics are shown in Figure 21(a) and the interpretable attention map is shown in Figure 21(b).



(a)

Figure 21. **Training dynamics for Reverse task.** We see Abrupt Learning, Representation Collapse and Repetitions, though to a lesser extent than MWS task. Note that the plateau is much shorter compared to MWS, possibly explained by the fact that reversing a sequence is 'easier' than computing the moving window sum.

E.6. Copy

This is the trivial task of copying the input sequence as is, so that the training sequences for copy task COPY are given as,

$$x_1, x_2, \ldots, x_n, \mathrm{SEP}, x_1, x_2, \ldots, x_n$$

for $x_i \sim \mathrm{Unif}\{1, 2, \ldots, 16\}, n = 16, \mathrm{SEP} = 0$. The training dynamics are shown in Figure 22(a) and the interpretable attention map is shown in Figure 22(b).

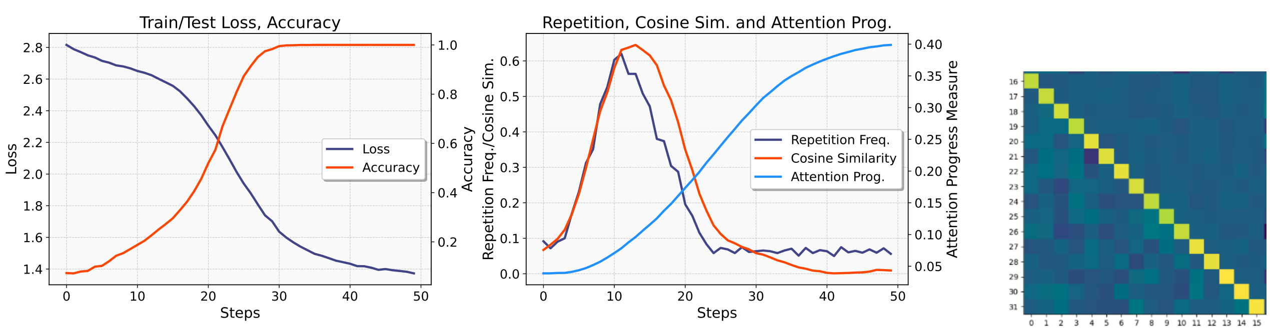


Figure 22. **Training dynamics for Copy task.** Similar to reverse task, we observe Abrupt Learning, Representation Collapse and Repetitions for Copy task, but this time to an even lesser extent than reverse task itself.

F. Varying Configurations

We demonstrate below that abrupt learning, representation collapse and repetition bias occur across model hyperparameter variations for the MWS task (Figures 23 to 27). We also show that training using SGD instead of Adam also demonstrates similar abrupt learning characteristics (Figure 28).

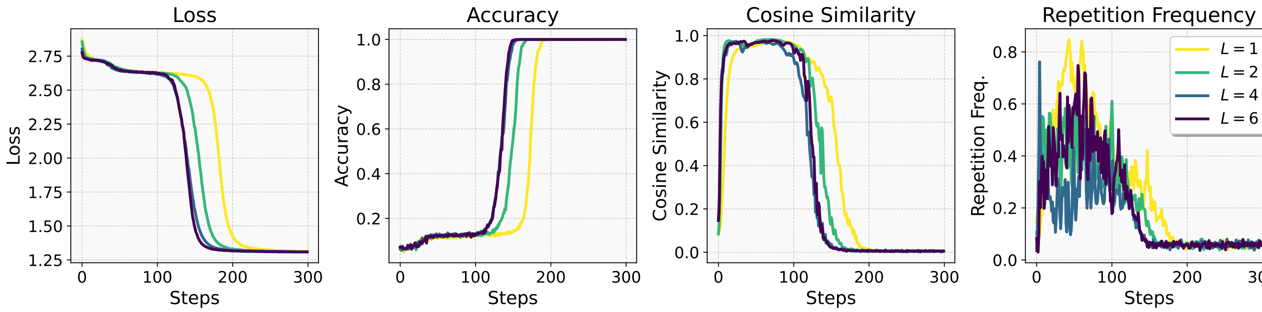


Figure 23. **Number of Layers** (L). We show that abrupt learning, representation collapse in the last layer and repetitions occur for multi (2, 4, 6)–layer models as well.

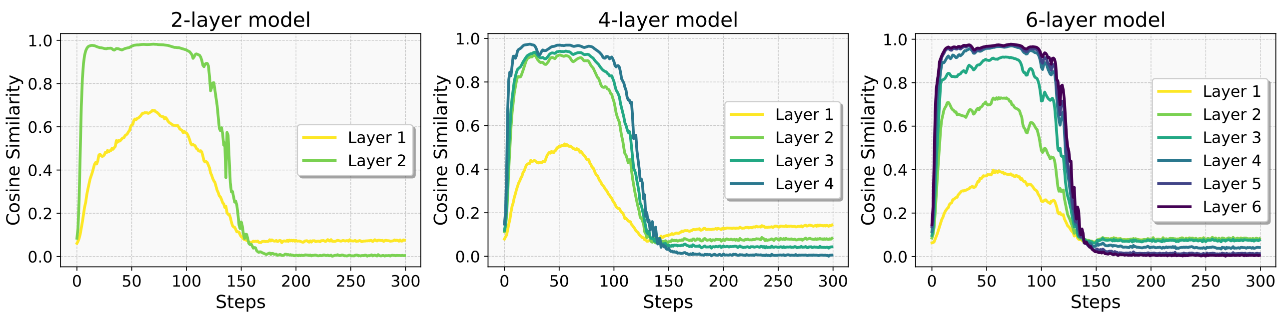


Figure 24. **Extent of Representation collapse at various intermediate layers.** Cosine similarity values showing the extent of representation collapse after each intermediate layer in multi-layer models. Note that the representation collapse is not so severe in the early layers of multi-layer models, but the cosine similarity becomes close to 1.0 as we progress to the final layer.

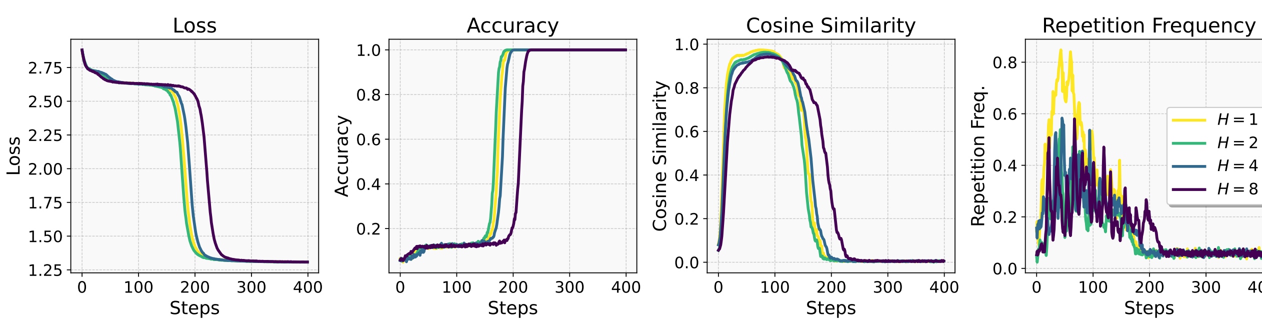


Figure 25. **Number of attention heads** (H)**.** We show that abrupt learning with representation collapse and repetition bias occurs in 1-layer multi-attention head models as well.

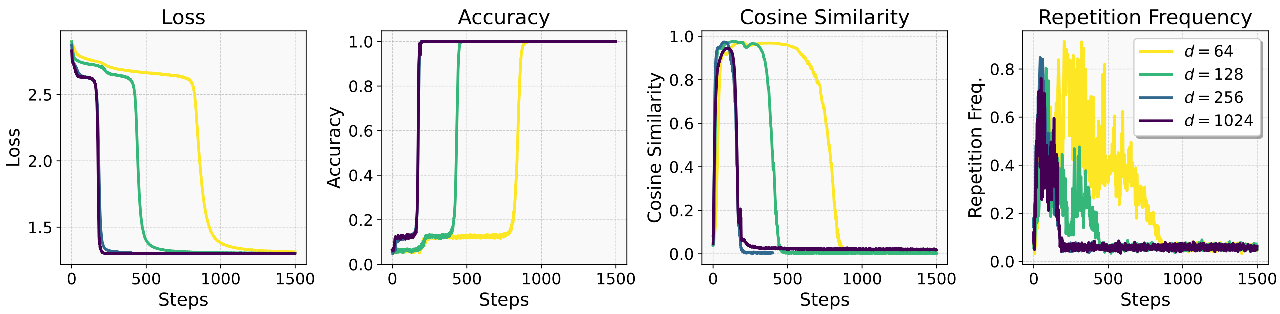


Figure 26. **Embedding dimension** (d)**.** We show that abrupt learning with representation collapse and repetition bias occurs in 1-layer 1-head models with different embedding dimension. Note that the convergence is delayed for models with smaller values of $d = 64, 128$.

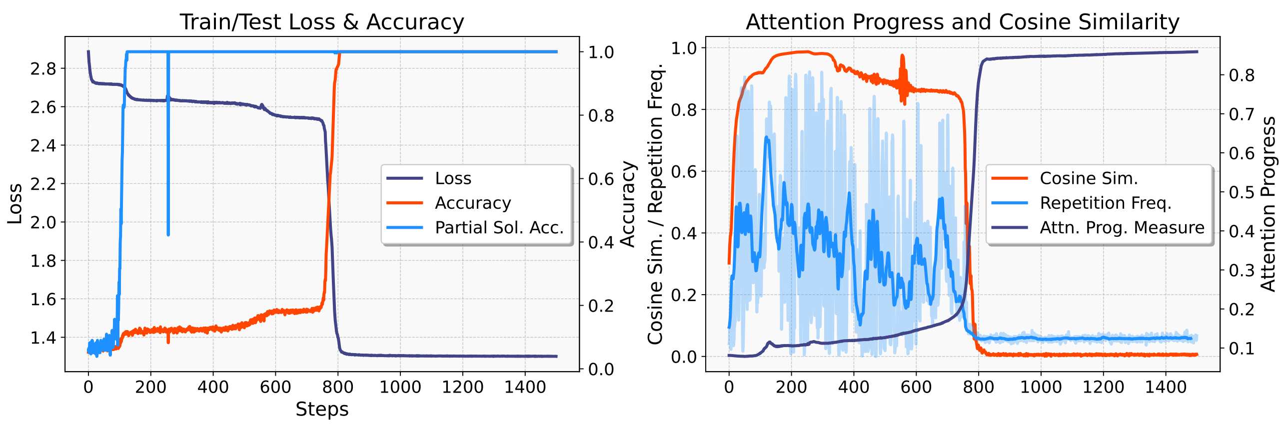


Figure 27. **Softmax Attention**. For completeness we show that repetition bias and early-phase representation collapse are not limited to linear transformers but are observed in softmax attention transformers as well. Note that the loss plateau is longer than that for linear attention.

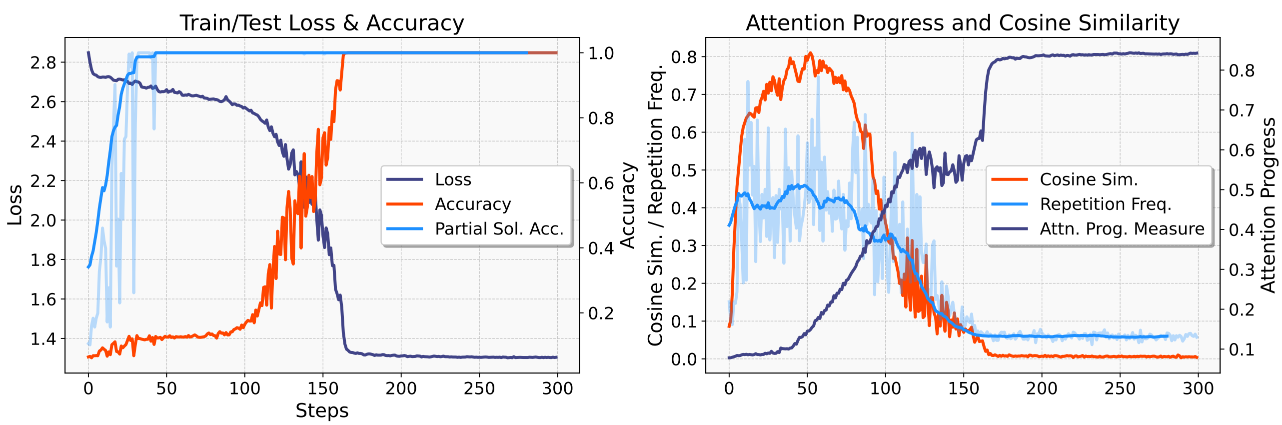


Figure 28. **SGD instead of Adam for loss optimization**. We show that abrupt learning is not limited to Adam optimizer, and occurs with SGD $(\eta = 0.1)$ as well. We chose this value of η since smaller values typically lead to much longer periods of little decrease in loss, without increase in accuracy.

G. Additional Figures

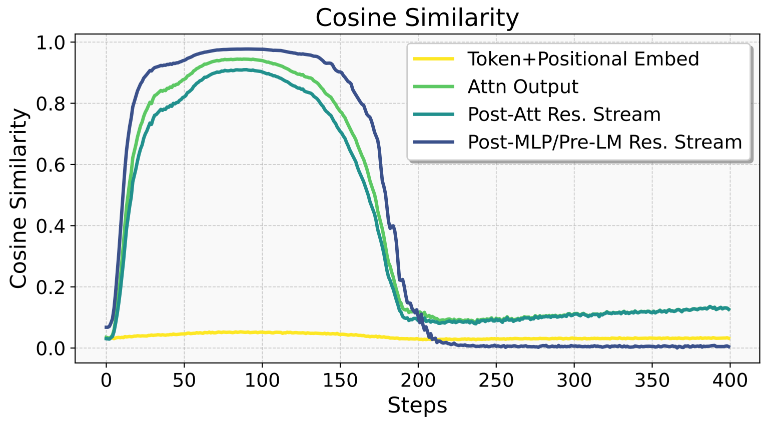


Figure 29. Cosine similarity at various points in residual stream for 1-layer, 1-head Transformer trained on MWS task.

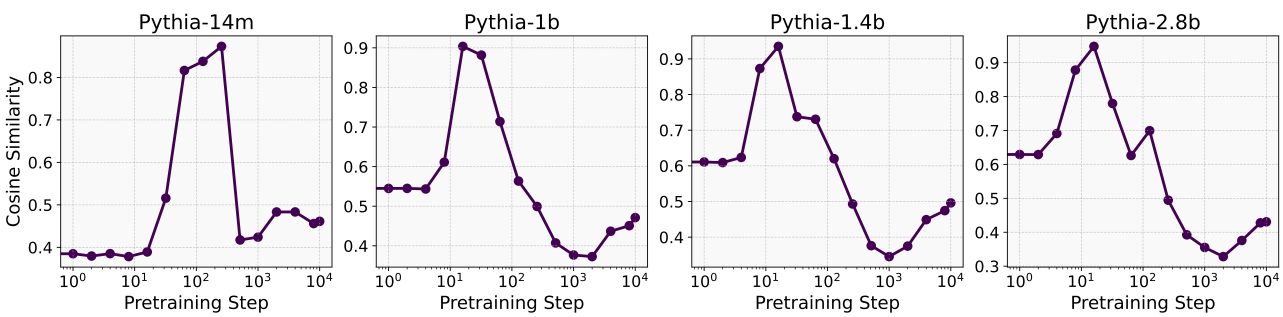


Figure 30. Representation collapse at different Pythia pretraining checkpoints; inference with random sampling.

H. Related Work

Abrupt learning has been studied in multiple settings; (Barak et al., 2022) studied it for parity tasks for multiple neural net architectures, while (Edelman et al., 2024) studied abrupt learning for a Markov chain task with Transformers. (Gopalani et al., 2024) showed that training a BERT model on matrix completion leads to abrupt learning, while (Lubana et al., 2024) connected abrupt learning for a grammar data setup to graph percolation. For abrupt learning in in-context learning (Garg et al., 2023), there has been a line of recent works (Zhang et al., 2025; Singh et al., 2024; Wang et al., 2025; Chen et al., 2024b; Reddy, 2023) that proposed various theoretical and empirical explanations. We aim to understand a unifying reason behind such observations, in an algorithmic setup for multiple tasks with multi-token output sequences, without restrictive assumptions on our model or training setup.

Repetition in language models is a well studied problem (Holtzman et al., 2020; Hiraoka & Inui, 2025; Li et al., 2023; Fu et al., 2021; Yao et al., 2025; Wang et al., 2024; Xu et al., 2022). However these works focus not on the early phase of training, but on how repetition may arise in pretrained models, and how to mitigate such phenomena. (Choshen et al., 2022) remarked that in the early phase of training language models, the output might contain some word repetitions, but understanding this occurrence is not the main focus of their work. Rank collapse is a related phenomenon for deep softmax transformers at initialization that might hinder training (Anagnostidis et al., 2022); however, our representation collapse phenomenon is different in that (i) we use shallow (1 or 2 layers) Transformers instead of deep ones; (ii) we use linear attention instead of softmax; (iii) our observed representation collapse occurs only *after* a few steps of training, not at initialization.

A line of recent work focused on understanding a related phenomenon of grokking (Power et al., 2022), which is abrupt generalization after an extended phase of memorization of training data by the model. Multiple works have studied grokking from the perspective of circuits (Nanda et al., 2023; Varma et al., 2023; Merrill et al., 2023), representation learning (Liu et al., 2022), delay in feature learning (Kumar et al., 2024; Lyu et al., 2024; Xu et al., 2023), and learning syntactic structures for linguistic data (Chen et al., 2024a). We focus on abrupt learning in the online training regime where there isn't a fixed training set, which is a different phenomenon from grokking. Grokking has also been attributed to the softmax activation in attention (Hoffmann et al., 2024; Prieto et al., 2025), which does not apply to our linear-attention setup.

Saddle-to-saddle dynamics (Boix-Adsera et al., 2023; Jacot et al., 2022) have also been used to explain plateau and sudden drop in loss during training. However, these results require very small scale of initialization ($\to 0$) for their results to hold, which does not hold in our setups.

The interplay of simplicity bias and Transformer learning dynamics has been studied recently in (Rende et al., 2025; Belrose et al., 2024). In (Belrose et al., 2024) authors show that neural nets learn lower-order moments of data earlier in training, and that the embedding statistics of Transformer models and token n-gram frequencies are related, explaining a specific distributional simplicity bias during training. (Rende et al., 2025) show that Transformer-based models progressively learn higher-order ('many-body') interactions between tokens in the sequence.

Ideas from statistical physics have also been used towards understanding initial loss plateaus in neural net training (Saad & Solla, 1995; Inoue et al., 2003); they work in a 2-layer teacher-student neural net setup, where the second layer is fixed during training, and use order parameters to study training. They show that there is a permutation symmetry in the weight vectors of the first layer during the early plateau stage, and exiting this symmetry state is what leads to drop in loss.

There has also been recent work towards understanding training dynamics of Transformers using techniques from singular learning theory (Hoogland et al., 2025). The core idea in this approach is to estimate the Local Learning Coefficient (LLC) during training, and use this quantity to explain degeneracy in the loss landscape, and consequently the stage-wise training dynamics of Transformers.

Note that we study representation collapse in the early phase of training, which is distinct from the notion of representation collapse in (Barbero et al., 2024); they show that for 2 sequences $(v_1, v_2, \ldots, v_n)$ and $(v_1, v_2, \ldots, v_n, v_n)$, as n grows large, the pretrained model's hidden state representation for the last token becomes identical for both sequences (Theorem 4.2, (Barbero et al., 2024)).