Exploring model depth and data complexity through the lens of cellular automata

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large language models excel at solving complex tasks, owing to their hierarchical architecture that enables the implementation of sophisticated algorithms through layered computations. In this work, we study the interplay between model depth and data complexity using elementary cellular automata (ECA) datasets. We demonstrate empirically that, given a fixed parameter count, deeper networks consistently outperform shallower variants. Our findings reveal that complex ECA rules require a deeper model to emulate. Finally, analysis of attention score patterns elucidates why shallower networks struggle to effectively emulate complex rules.

1 Introduction

Large Language Models (LLMs) are undergoing rapid scaling [18, 3, 14, 1], with significant increases in training FLOPs, dataset size, and parameter count. As traditional trial-and-error approaches become computationally intractable at these scales, researchers increasingly rely on scaling laws [11, 8, 2] to optimize model architectures and training regimes without exhaustive empirical validation. However, the emergence of unforeseen capabilities beyond certain scale thresholds [22, 15] introduces additional complexity to performance forecasting, rendering predictive models increasingly challenging and potentially unreliable. In particular, the benefit of depth in model scaling remains unclear [2, 7], although it has been shown to be helpful in many cases [19, 17, 24].

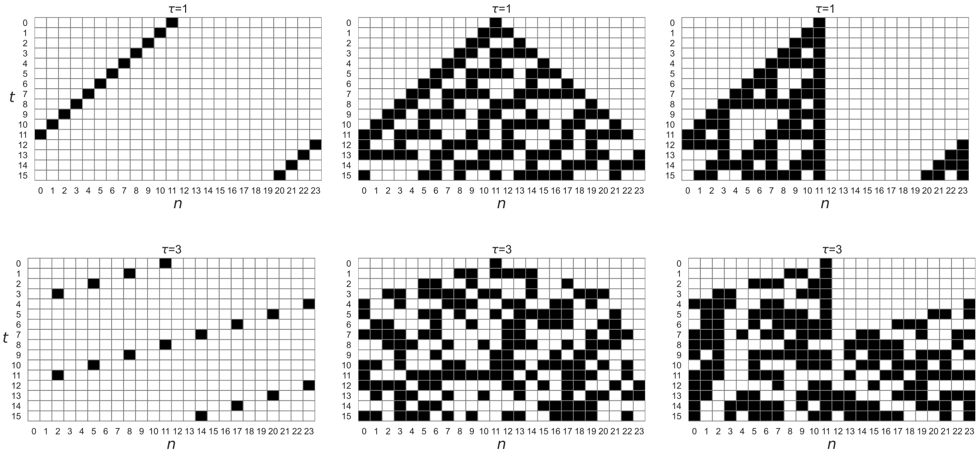
A significant factor contributing to the discrepancies observed in existing literature is the lack of systematic consideration of the interplay between data complexity and model architecture. To address this gap, our study investigates the impact of model depth concerning dataset complexity, utilizing Cellular Automata (CA) [20, 21, 6, 23] as controlled datasets. Specifically, we train GPT-like autoregressive models [4] of varying depths on Elementary Cellular Automata (ECA) [23] datasets, which offer the advantage of systematically controllable complexity [10, 23, 9] of data. This approach allows for a more rigorous examination of the relationship between model depth and data complexity, potentially reconciling inconsistencies in previous findings.

2 Elementary Cellular Automata

Elementary Cellular Automata (ECA) [23] are boolean-valued CA that live on a one-dimensional lattice. At each time step t, a given cell i in the lattice has a value $s_i(t) \in \{0, 1\}$. The state, which is the collection of all values in the lattice at time step t, is represented by a binary vector $\boldsymbol{s}(t) \in \{0, 1\}^n$, where n is the size of the lattice. The state at time step $t + 1$ is completely determined by the state at time t, following a 3-to-1 boolean-valued map:

$$s_i(t+1) = r_I(s_{i-1}(t), s_i(t), s_{i+1}(t)) \,, \tag{1}$$

Submitted to Workshop on Scientific Methods for Understanding Deep Learning, NeurIPS 2024.



(a) Rule 2 (b) Rule 30 (c) Rule 110

Figure 1: Visualization of ECA with periodic boundary conditions, where black cells represent value 1 and white cells represent value 0. First row: $\tau = 1$; Second row: $\tau = 3$. This comparison illustrates the diverse complexity patterns across different ECA classes and temporal scales.

where $I = \{0, 1, \cdots, 255\}$ for ECA[1].

The 256 Elementary Cellular Automata rules are categorized into four distinct classes based on their asymptotic behavior [23, 13]: (I) uniform, characterized by homogeneous final states; (II) periodic or stable, exhibiting regular patterns or fixed points; (III) chaotic, displaying aperiodic and seemingly random configurations; and (IV) complex, demonstrating localized structures reminiscent of Class II, but with intricate interactions.

Inspired by Israeli and Goldenfeld [10], we also consider ECA with τ-step evolution, with $\tau \geq 1$:

$$s_i((t+1)\tau) = r_I^\tau(s_{i-\tau}(t\cdot\tau), \cdots, s_i(t\cdot\tau), \cdots, s_{i+\tau}(t\cdot\tau)), \tag{2}$$

where the τ-step evolution can be equivalently viewed as a 1-step evolution with a $(2\tau+1)$-to-1 map described by a new rule r_I^τ. Note that this does not mean a rule with a larger τ is harder than the one with a smaller τ. As the nature of r_I^τ heavily depends on r_I itself and also the τ choice [10].

In this work, we focus on Rule 2, 30, and 110, which are representative of Classes II, III, and IV, respectively. Rule 30 is the hardest to predict out of those three rules for any τ, as it is known to exhibit chaotic behavior, and no simplification has been found for $\tau > 1$. Rule 2 is the simplest since it always converges to some stable or periodic patterns. Rule 110 has an intermediate hardness [23, 10]. Figure 1 illustrates the spatiotemporal patterns generated by these rules with varying τ. We also show the rule icon which fully characterize the rules in Appendix B.

3 Transformers Trained on ECA

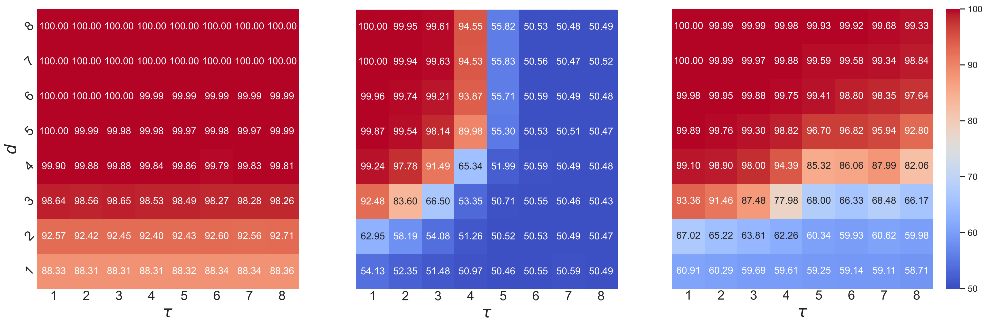
Training: We focus on ECA on a lattice size of $n = 24$ with periodic boundary conditions. We follow the standard train-test split where different initial conditions were used to generate those states. All training states are flattened into a sequence with the form $\mathrm{vec}(\boldsymbol{s}(0), \boldsymbol{s}(\tau), \cdots, \boldsymbol{s}(t_{\mathrm{train}}\tau))$, with $t_{\mathrm{train}} = 7$, so the training context length is 192 for all τ values.

Evaluation: We measure test performance in two different settings:

(`Eval 1`) Next-token-prediction accuracy: Given a test sequence up to t time steps ($24 \cdot t$ tokens), we measure the next token prediction accuracy, averaged over all tokens.

(`Eval 2`) Sequence-matching accuracy with length generalization: We give the sequences at times $[t, t+1, ...t+6]$ and ask the model to predict for time step $t+7$. Repeating this for $t = 1$ to $t = 8$, the prediction is marked correct *only* if the model predicts *all* cells within the time step correctly. Note that in this case we test up to twice of the evolving steps for training, to check length generalization.

[1] There are $2^3 = 8$ possible patterns for a given triplet. A rule needs to decide, for each pattern, whether the cell will be a 1 or a 0 in the next time step. So in total $2^8 = 256$ possible rules.

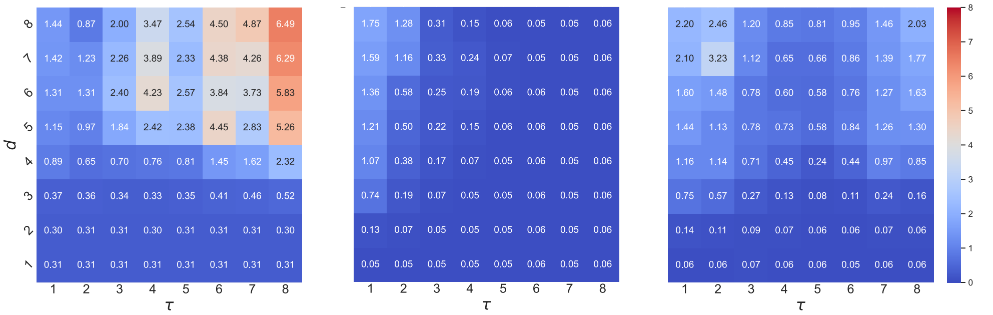


(a) Rule 2 (b) Rule 30 (c) Rule 110

Figure 2: Depth vs τ phase diagrams for selected ECA rules: Test accuracy (`Eval 1`) on states generated from unseen initial conditions. (a) Rule 2 exhibits consistent performance across all τ due to its convergence to stable/periodic states, enabling arbitrary length generation. (b) The chaotic nature of Rule 30 results in much worse accuracy for large τ.

Crucial role of model depth: In Figure 2, we demonstrate the correlation between model performance and the complexity of the three aforementioned ECA rules. It illustrates the performance of models across varying depths d and minimal evolving steps τ for each rule. To isolate the effect of depth, we maintain a constant number of non-embedding parameters $N = 2^{22} (\approx 4.2\text{M})$ across all models by adjusting the width accordingly. Additional experiments with $N = 2^{23} (\approx 8.4\text{M})$ and $N = 2^{24} (\approx 16.8\text{M})$, achieved by increasing model width, yielded qualitatively similar results, see Appendix C. Other details of these experiments, including training setups and hyperparameters, see in Appendix A.

We measure the next token prediction accuracy (`Eval 1`) for the test set. For Rule 2, the model achieves perfect performance across all τ when the network depth $d \geq 3$, demonstrating the simplicity of the rule and the capacity of the model to capture its dynamics fully. In contrast, for the more complex Rule 110, we observe a gradual degradation in model performance as τ increases, indicating the increased computational complexity and the necessity for deeper models to capture its behavior accurately. In contrast, for the chaotic Rule 30, the model can not do better than random guessing for $\tau \geq 6$, regardless of the depth.



(a) Rule 2 (b) Rule 30 (c) Rule 110

Figure 3: Depth-τ phase diagrams for selected ECA rules with length generalization (`Eval 2`). (c) Rule 110 demonstrates non-monotonic performance with respect to τ, reflecting the existence of simplified descriptions for r_{110}^{τ} at certain larger τ values.

Sequence length generalization: In Figure 3, we evaluate the length generalization ability beyond t_{train}. For Rule 2, we observe robust length generalization for large τ values, attributable to its convergence to stable or periodic states, as we have demonstraed in panel (a) of Figure 1. This allows the model to achieve generalization by iteratively applying a few simple learned compositional rules. For small τ, the model does not perform as well in the length generalization. This is due to finite $t_{\text{train}} = 7$ we used, where the dynamics have not converged to a stable and periodic pattern.

In contrast, the chaotic nature of Rule 30 and the lack of simplified representations for larger τ results in rapid error propagation, making length generalization challenging for $\tau \geq 3$. This difficulty stems from the inability of the model to formulate a concise implementation of r_{30}^{τ}. The highly chaotic

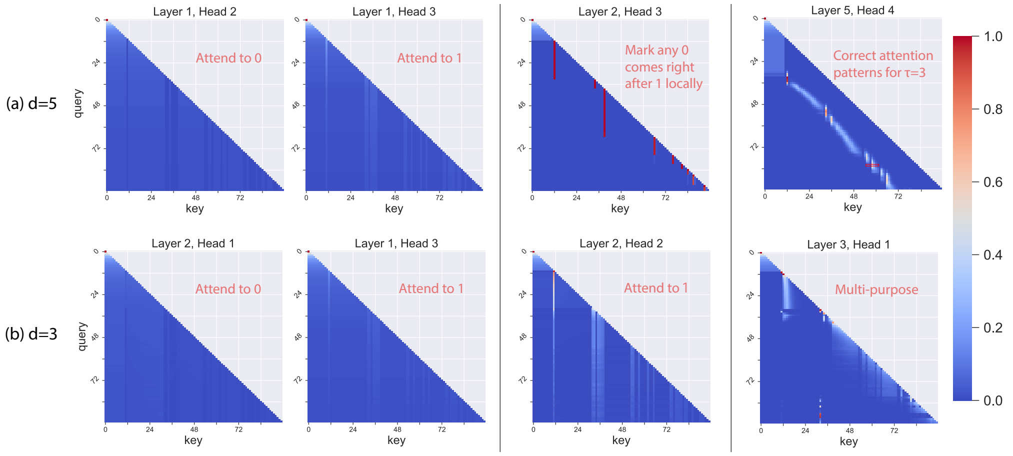


Figure 4: Attention scores of $d = 3$ and $d = 5$ models trained on Rule 30 with $\tau = 3$. We feed the sequence shown in Figure 1(b) $\tau = 3$ panel to both models, which is not in the training set.

nature of Rule 30 and the irreducibility of r_{30}^{τ} for $\tau > 1$ [10] present two potential strategies for the model: (i) direct implementation of the complex rule for $\tau > 1$, or (ii) composition of multiple smaller τ steps. The former approach necessitates a substantially larger training steps t_{train}, while the latter relies more heavily on increased model depth. We believe in both cases, the model needs an extensive dataset to perform for large τ due to the difficulty of the task. Nevertheless, deeper architectures consistently outperform shallower ones for a given τ, highlighting the importance of model depth for implementing complex algorithms.

Interestingly, Rule 110 exhibits non-monotonic length generalization performance with respect to τ for fixed network depths. This phenomenon likely arises from the existence of simplified representations of r_{110}^{τ} for certain larger τ values, as suggested by Israeli and Goldenfeld [10]. Such simplified representations may facilitate more effective learning and generalization for those τ values.

4 Interpretability

In this section, by visualizing the attention score, we give some explanation on the necessity of a certain depth. We focus on models with depths $d = 3$ and $d = 5$, trained on Rule 30 with a minimal evolving step $\tau = 3$. To make a reasonable prediction for this chaotic rule, a model must infer the underlying rule and apply it to a 7-cell collection $\{s_{i-3}(t), \cdots, s_{i+3}(t)\}$ to predict $s_i((t+1)\tau)$.

In Figure 4, we selected heads from both models that might contribute non-trivially to the algorithm. We find in both cases, models use heads from lower layers to identify the information of 0 and 1. For $d = 5$ model, the layer 2 head correctly identifies the underlying data-generating process. This is done by only looking at 0s that come right after 1s within one iteration ahead. After intermediate layers process the information, the head from the last layer builds a correct attention pattern that mainly localizes around the 7-cell collection from the previous iteration. In contrast, the $d = 3$ model does not have enough space to process the information, resulting in a multi-purpose last layer head that fails to perform well. For unlisted heads, see Appendix D for more details.

5 Discussion

In this paper, we studied the importance of model depth for handling chaotic ECA rules. However, as we mentioned in the paper, there is more than one possibility for implementing a given ECA rule. We believe it is important to understand in more detail under what conditions a model would implement the same algorithm differently.

Another interesting question would be how to extend this dataset to build connections with real-world settings. Here we list two possibilities while leaving the detailed research for the future: (i) mix different rules with varying complexity to emulate real-world settings; (ii) emulate real-language with Rule 110, which is Turing complete [5]. Then systematically study the translated "language".

References

[1] Google gemini-1.5 team, 2024. URL https://arxiv.org/abs/2403.05530.

[2] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws, 2024. URL https://arxiv.org/abs/2404.05405.

[3] Anthropic. Claude 3.5, 2024. URL https://www.anthropic.com/news/claude-3-5-sonnet.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

[5] Matthew Cook. Universality in elementary cellular automata. Department of Computation and Neural Systems, Caltech, 2004. Available at: https://www.complex-systems.com/pdf/15-1-1.pdf.

[6] Martin Gardner. Mathematical games. *Scientific American*, 223(4):120–123, 1970. ISSN 00368733, 19467087. URL http://www.jstor.org/stable/24927642.

[7] Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. The unreasonable ineffectiveness of the deeper layers, 2024. URL https://arxiv.org/abs/2403.17887.

[8] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030. Curran Associates, Inc., 2022.

[9] Lyman P. Hurd. Formal language characterizations of cellular automaton limit sets. *Complex Systems*, 1: 69–80, 1987.

[10] Navot Israeli and Nigel Goldenfeld. Coarse-graining of cellular automata, emergence, and the predictability of complex systems. *Phys. Rev. E*, 73:026203, Feb 2006. doi: 10.1103/PhysRevE.73.026203. URL https://link.aps.org/doi/10.1103/PhysRevE.73.026203.

[11] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

[13] Genaro J. Martinez. A note on elementary cellular automata classification, 2013. URL https://arxiv.org/abs/1306.5577.

[14] OpenAI. Openai o1, 2024. URL https://openai.com/o1/.

[15] Rylan Schaeffer, Brando Miranda, and Sanmi Koyejo. Are emergent abilities of large language models a mirage? In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 55565–55581. Curran Associates, Inc., 2023.

[16] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023. URL https://arxiv.org/abs/2104.09864.

[17] Google Gemma-2 Team. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

[18] Meta Llama-3 Team. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

[19] Matus Telgarsky. benefits of depth in neural networks. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 1517–1539, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR. URL https://proceedings.mlr.press/v49/telgarsky16.html.

[20] J. von Neumann. The general and logical theory of automata. In A. H. Taub, editor, *Collected Works*, volume 5, page 288. Pergamon Press, 1963.

[21] J. von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, 1966.

[22] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL `https://openreview.net/forum?id=yzkSU5zdwD`. Survey Certification.

[23] Stephen Wolfram. Statistical mechanics of cellular automata. *Rev. Mod. Phys.*, 55:601–644, Jul 1983. doi: 10.1103/RevModPhys.55.601. URL `https://link.aps.org/doi/10.1103/RevModPhys.55.601`.

[24] Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of language models: Part 2.1, grade-school math and the hidden reasoning process, 2024. URL `https://arxiv.org/abs/2407.20311`.

A Training Details

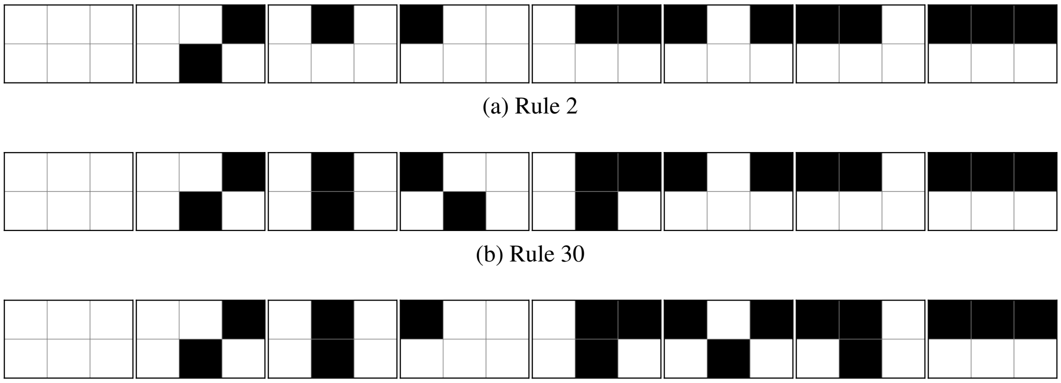
Model All of our models have the same architecture design and initialization as GPT-2, where the only difference is we use rotary positional embedding [16].

Optimization We train all of our models in Figures 2 and 3 using AdamW optimizer [12] with $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 10^{-8}$ and batch size $B = 256$ for 50k steps. For each model we select learning rate η from $\{0.00003, 0.0001, 0.0003\}$ and weight decay λ from $\{0.1, 1.0, 2.0\}$. We use a linear warmup and cosine decay scheduler, where the learning rate is linearly increased for the first 5k steps from 0.01η, then cosine decayed to 0.1η at the end of training.

Dataset We use a training set with $N_{\text{train}} = 2^{17}$ initial conditions, where the test set is the same size. The possible states generated, including initial conditions in the training (test) set, is $2^{17} * 16 = 2^{21}$, where the total possible states for $n = 24$ lattice are 2^{24}. So, training and test sets each have 12.5% number of states out of the total possibilities, which leads to an ignorable overlap in training and test sets.

B Rule Icons

Rule icons demonstrate all eight fundamental 3-to-1 maps for $\tau = 1$ ECA, see Figure 5.



(a) Rule 2

(b) Rule 30

(c) Rule 110

Figure 5: Rule icons for Rule 2, 3 and 110. Black cells represent 1 and white cells represent 0. One should ignore the cells located at bottom left corners and bottom right corners for each icon, as we are not considering boundary condition here.

C More Phase Diagrams

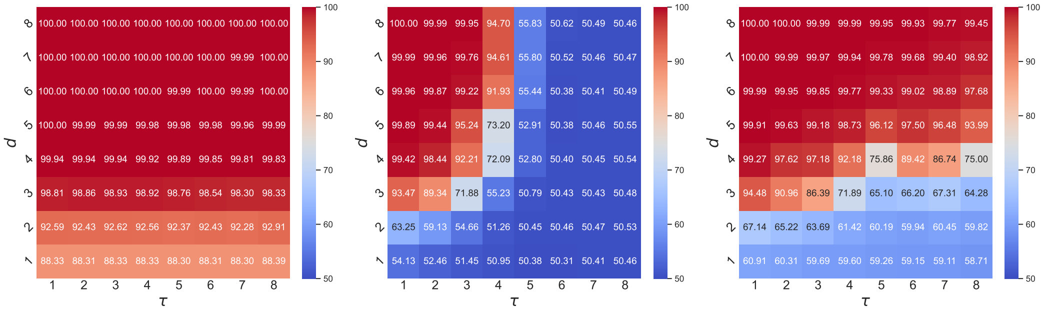
In this section, we plot more phase diagrams with number of parameters $N = 2^{23}$ and $N = 2^{24}$. We see for chaotic rule, i.e. Rule 30, larger model performs better for small τ while at the same time suffers from overfitting for larger τ as the next-token prediction accuracy drops compared to the settings with $N = 2^{22}$.

C.1 Next-token Prediction Accuracy (`Eval 1`)

See Figures 6 and 7 for next-token prediction results.

C.2 Length Generalization (`Eval 2`)

See Figures 8 and 9 for length generalization results.

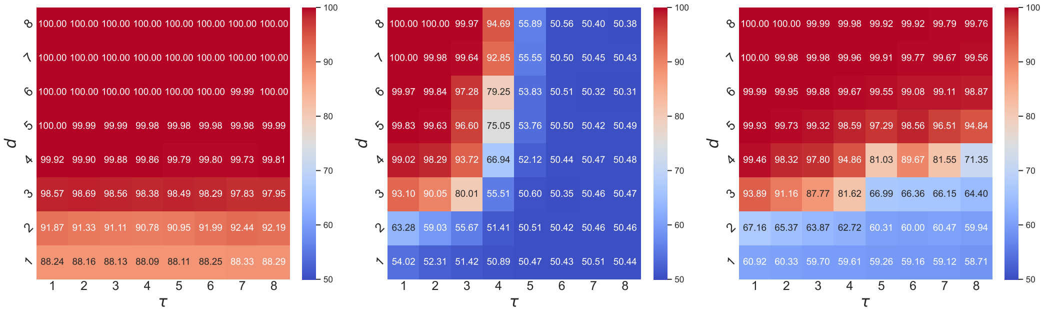


(a) Rule 2

(b) Rule 30

(c) Rule 110

Figure 6: Depth vs τ phase diagrams for selected ECA rules: Test accuracy (`Eval 1`) on states generated from unseen initial conditions. Same setting as Figure 2 while the number of parameters is $N = 2^{23}$

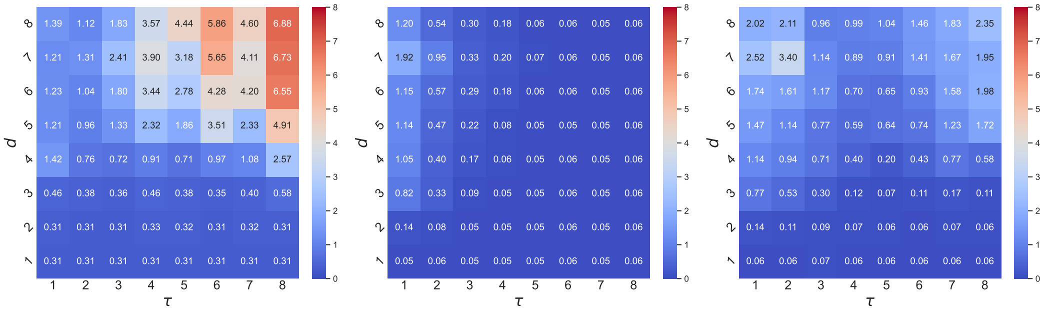


(a) Rule 2

(b) Rule 30

(c) Rule 110

Figure 7: Depth vs τ phase diagrams for selected ECA rules: Test accuracy (`Eval 1`) on states generated from unseen initial conditions. Same setting as Figure 2 while the number of parameters is $N = 2^{24}$

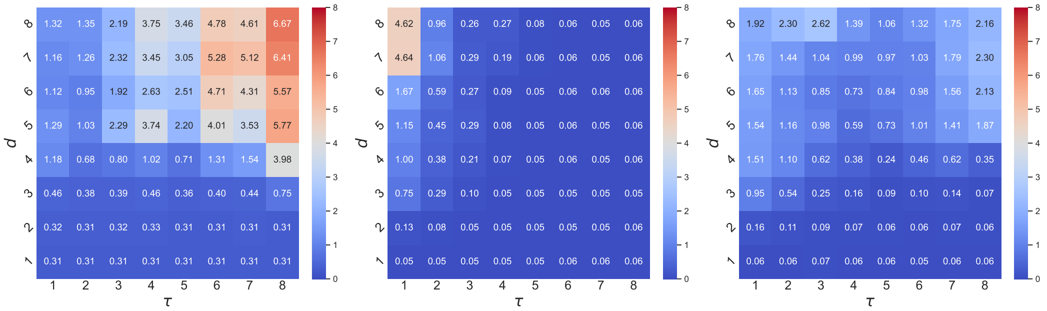


(a) Rule 2

(b) Rule 30

(c) Rule 110

Figure 8: Depth-τ phase diagrams for selected ECA rules with length generalization (`Eval 2`). Same setting as Figure 3 while the number of parameters is $N = 2^{23}$



(a) Rule 2

(b) Rule 30

(c) Rule 110

Figure 9: Depth-τ phase diagrams for selected ECA rules with length generalization (`Eval 2`). Same setting as Figure 3 while the number of parameters is $N = 2^{24}$

D More Attention Scores

D.1 $d = 5$

We plot all attention score for all heads in Figure 10.

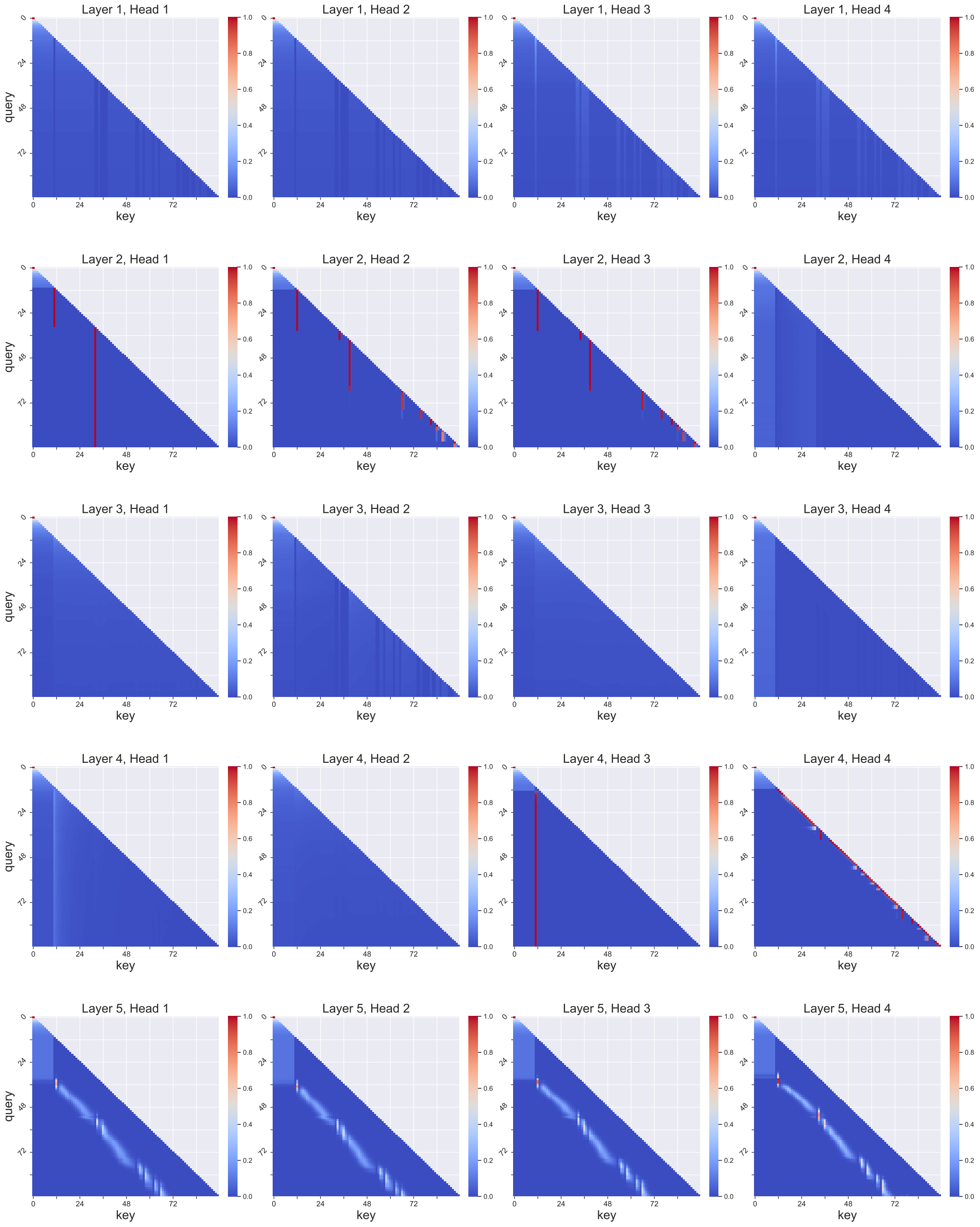


Figure 10: Attention scores for $d = 5$ models trained on Rule 30 with $\tau = 3$. Most heads that were not included in Figure 4 are doing nothing or playing similar roles to those that were selected. Only the heads 3 and 4 in layer 4 seem to be different. However, most likely, they are not playing an essential role as the former one is only looking at the first 0 after 1 for the whole sequence while the latter one is only checking very local information. Note that there is a chance that layer 4 head 4 is helping building a different algorithm.

D.2 $d = 3$

We plot all attention score for all heads in Figure 11.

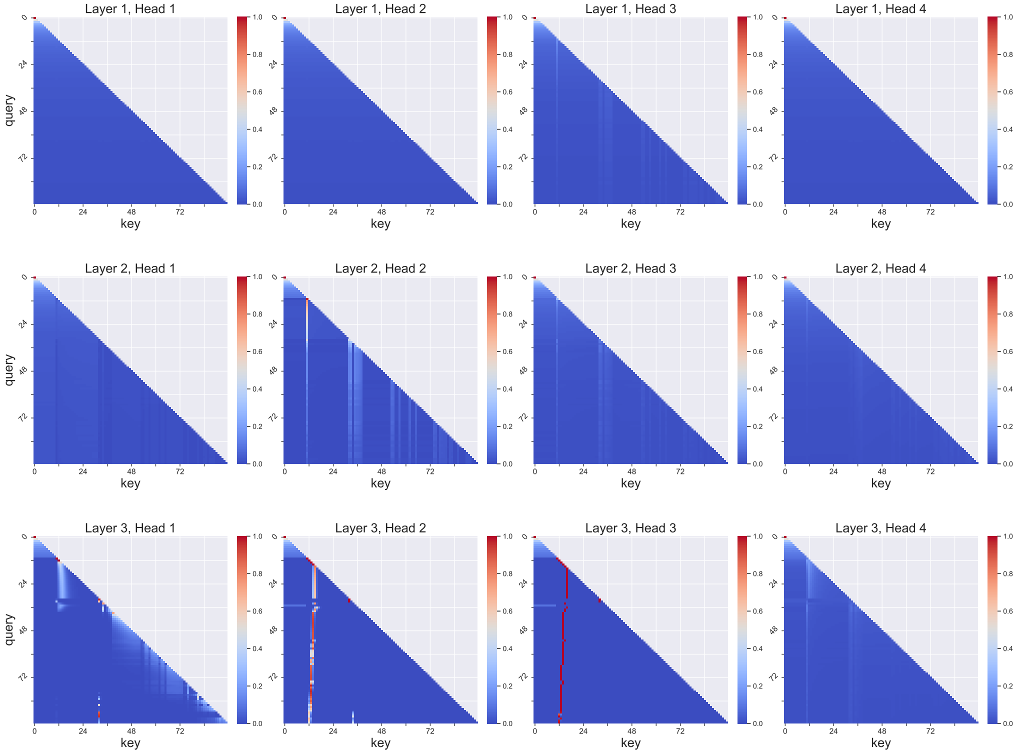


Figure 11: Attention scores for $d = 3$ models trained on Rule 30 with $\tau = 3$. Most heads that were not included in Figure 4 are doing nothing or playing similar roles to those that were selected. We find heads 2 and 3 in layer 3 similar to the head 4 in layer 4 of the $d = 5$ model, which again suggests that the model needs a larger depth to perform well.