Understanding the Transient Nature of In-Context Learning: The Window of Generalization

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

In-Context Learning (ICL) is one of the main mechanisms driving few shot learning capabilities of large language models (LLMs). A rich literatures explores the causal factors giving rise to this mechanism, while recent studies has pointed out that this mechanism can be transient. In this work, we study ICL on a synthetic task consisting of a probabilistic mixture of Markov chains which is simple enough to allow theoretical analysis yet rich enough to reproduce multiple phenomena discussed in the ICL literature. Here, we focus on analyzing the transient nature of ICL using this setup and elucidate the role of data and model training using a mechanistic phase diagram. Our findings conclude to: 1) A certain data diversity is required for ICL 2) a non-generalizing Bayesian solution might arise later in training if its circuit complexity is higher. We conclude: *ICL, or any other generalizing solution, is subject to transience if there exists a better solution narrowly fitting the training distribution accessible by gradient descent*.

1 Introduction

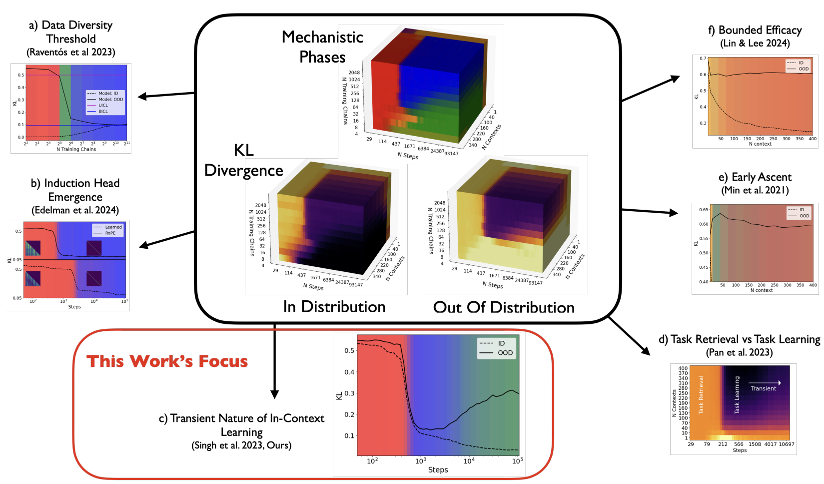

Figure 1: **Markov Mixtures data explains multiple In-Context Learning Phenomena** Autoregressive transformers trained on *Markov Mixtures* exhibits multiple phenomena of In-Context Learning. a) Data Diversity threshold for ICL [1] b) Emergence of Induction Heads for ICL [2, 3, 4, 5] **c) This work's focus: Transient Nature of ICL** [6, 7] d) Task Retrieval and Task Learning Phases of ICL [8] e) Early Ascent of ICL Accuracy [9] f) Bounded Efficacy of ICL [10]

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

In-Context Learning (ICL) is often attributed to be at the core of the impressive capabilities of LLMs[11, 12, 13]. Motivated by this, many past work has explored different aspects of ICL, including algorithms [14, 15, 16, 17, 18, 19], data dependence [1, 5], training dynamics [2, 3, 20, 21], test time behavior [9, 8, 22, 23, 10]. Recently, it has been proposed that ICL can also be transient [6, 7]. In this work, we develop a synthetic dataset which is defined in discrete token space unlike the common linear regression setup[14, 16, 21, 1] called *Markov Mixtures*. This simple dataset reproduces multiple phenomenologies explored in the literature as seen in Fig. 1. We then focus on one specific phenomena, **the transient nature of ICL**, and explain why and when ICL is transient.

Our main contributions is as follows:

1. **Markov Mixtures Setup as a dataset to study ICL**: We introduce a discrete sequence dataset, *Markov Mixture*, simple enough to postulate analytic solutions yet which reproduces many ICL phenomena.
2. **Discovering control variables for ICL**: We discover data diversity and training time as two sharp requirements for ICL to emerge.
3. **Explaining the transient nature of ICL** We provide a simple explanation of why ICL emerges: "There exists a non-generalizing solution which performs better on the training set".
4. **Elucidating the role of data and model training for generalization** We run extensive experiments to reveal the role of data and model training in controlling generalization strategies.

2 Setup

Markov Mixtures Dataset Our data generating process (DGP) consists of a probabilistic mixture of Markov chains. n, k and l are hyperparameters of the DGP each controlling: how many transition matrices can we draw from, how big is the state space and how long is the sequence. The dataset is described in further detail in Appendix A

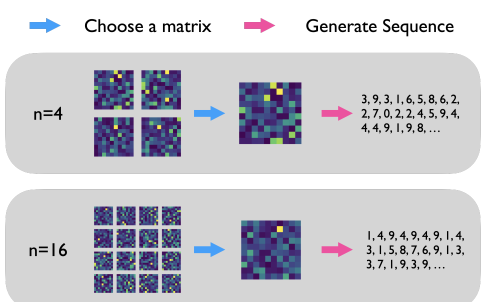

Figure 2: **Markov Mixtures Dataset** Our dataset consists of n predefined matrices from which a single matrix is chosen every time a datapoint(sequence) is drawn from the dataset.

Bayesian Solutions First, we introduce two simple Bayesian solutions which can be used to predict the next token on the *Markov Mixtures* dataset. .Since the training transition matrices are needed to compute the solution, the model needs to internalize(memorize) the training matrices to implement this solution. The first solution is **Unigram Likelihood Bayesian Averaging(ULBA)**, which observes the stationary distribution of the given context and computes the likelihood over the different transition matrices which can explain the given distribution. The second solution, conceptually very similar is the **Bigram Likelihood Bayesian Averaging(BLBA)** solution, which is similar to ULBA, but uses a bigram based likelihood. I.e., this solution computes the likelihood over *transitions* instead of states. Please refer to Appendix B for the mathematical formulation of these solutions.

Purely In-Context Solutions We discuss two purely In-Context solutions. We add the prefix "purely" to illustrate that these solutions are generalizing solutions with no performance gap between an In-Distribution (ID) evaluation and an Out-Of-Distribution (OOD) evaluation. **Unigram In-Context Learning (UICL)** infers the stationary token distribution from the context and draws new token form that distribution. **Bigram In-Context Learning (BICL)** generates the transition matrix by counting transitions from the context. This solution is the generalizing “world model" solution.

3 Results

We train a transformer[24] with a autoregressive next token cross entropy loss on our *Markov Mixtures* dataset with $k = 10$, $l = 512$ and $n \in 2^{\{2,3,4,5,6,7,8,9,10,11\}}$. Please see Appendix A for more details.

3.1 Requirements for ICL

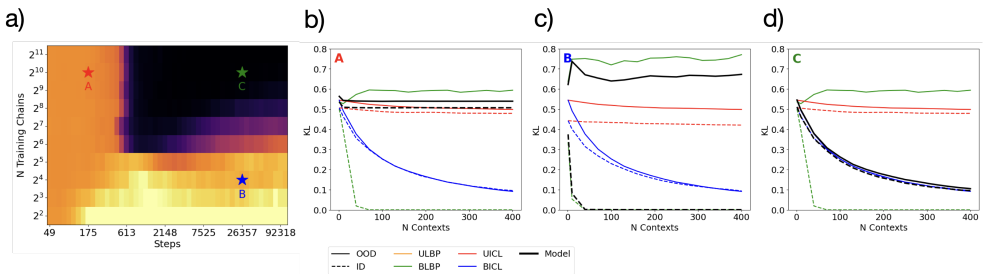

Figure 3: **Data Diversity and Enough Training is Required for ICL** a) OOD KL divergence depending on n and optimization steps. (Brighter is Higher, Worse) b,c,d) OOD and ID KL divergence of difference solutions and the model, respectively at the checkpoint corresponding to A,B,C in subpanel a).

Fig. 3 clearly demonstrates the joint data diversity and optimization step requirements for ICL. ICL emerges when, in this case, at least 600 gradient steps has been applied and when there are more than 2^6 latent transitions. It is notable that the compute requirement is extremely sharp, e.g. highly emergent behavior shows up as seen in Edelman et al. [4]. Subpanel b suggestst that checkpoint A is implementing UICL and checkpoint C is implementing BLBA, but only robustly for the ID data. Subpanel d shows that BICL has emerged and both ID and OOD data has a lower KL with more exemplars.

3.2 Why Can ICL be Transient

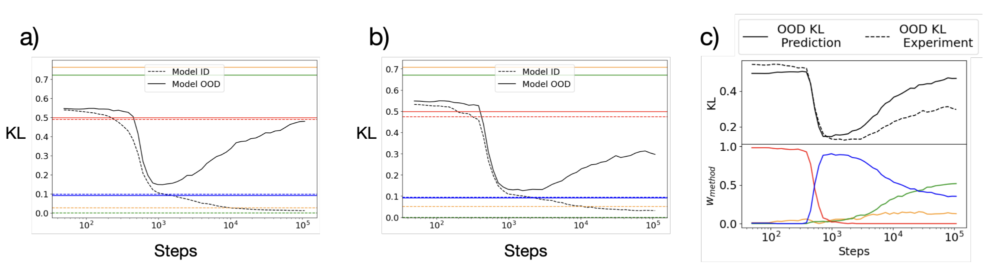

Figure 4: **The implementation BLBP drives the transience of ICL** a) ID, OOD model KL and different solution KLs. n=64 b) same for n=128 c) Mechanistic Decomposition of model outputs and corresponding test performance prediction.

Fig. 4 demonstrates why ICL can be transient. Fig. 4(a,b) shows that across different data diversity, the BICL solution starts fading away and the OOD KL divergence starts increasing when BLBP is implemented. The model does not optimize for test data, so the very mechanistic change which reduces the training data KL harms the OOD performance driving the transience of ICL. Fig. 4(c) shows that using a *mechanistic decomposition approach*(Explained in Appendix C), we can decompose the model into the 4 suggested mechanisms and get their weights, only using the training distribution. These weights are then used to predict the OOD performance as seen in the upper panel. This suggests that mechanistic decomposition on ID data and proper modelling of different mechanisms can explain/predict non-monotonic OOD performance.

3.3 The Role of Data and Model Training

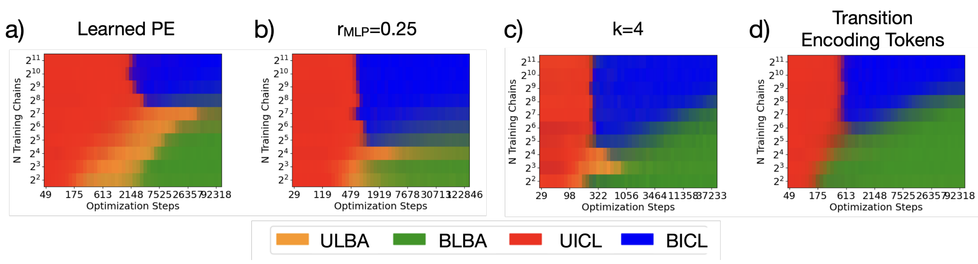

Figure 5: **Model Variants and Data Variants induces a phase diagram Change.** a) Using learned positional encodings b) Using thinner MLP layers c) using a smaller state space. d) Using transition encoding tokens.

Using the *mechanistic decomposition* seen above, we can draw a phase diagram of what solution the model implements. Here, we show that we can analyze the role of data and model training using these phase diagrams. Over different data and architectural changes, the general shape of the phase diagram remains. Notably we find that a thinner MLP suppresses the BLBP solution which intuitivelty makes sense as BLBP requires memorization of the training matrices. We also find that using tokens which embed states AND transitions, the BLBP solution emerges faster and to higher data diversity. We conclude: *Data defines the optimality of different solution mechanisms and model training determines how these solutions are navigated.*

4 Discussion

World Models v.s. Stochastic Parrots Recently, researchers have been interested in taking a position on the question of whether neural networks acquire world models or learn surface statistics (commonly said stochastic parrots).[1] Some researchers have pointed out the possibility of a parrot[25, 26] while others found evidence towards world models[27, 28, 29]. This work suggests a simple answer: “You get a parrot if the parrot works better, you get a world model if the world model works better", however, depending on your training setup, your world model *or your parrot* might not have arrived yet.

Optimization and Simplicity Bias Past work has explored competing solutions to a task with a simplicity perspective[30, 31]. From the simplicity perspective, we expect a transient generalizing solution when there exists a solution which is 1) better performing with respect to the training distribution and 2) complex enough to take a lot of optimization to learn. Our BLBA solution is indeed such case, as one can see from its attention matrix, which requires additional diagonal weights on top of an induction head. Our experiments with transition encoding tokens reveals that making this solution more accessible accelerates(in training steps) and severes(in data diversity) the transient nature of the BICL solution.

Conclusion We conclude our work with a hypothesis which we name *The Obvious Hypothesis*, as it might be indeed obvious for some readers:

The Obvious Hypothesis: *Model training simply optimizes it on the training data, generalization happens when the mechanism it currently implements on the training data turns out to be a generalizing one. A discovery of a better performing algorithm on the training set could lead to a deactivation of the generalizing mechanism.*

[1]The authors believe this is not a question one can answer with rigor and generality, but more of a nuance/position survey.

References

[1] Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression, 2023. URL https://arxiv.org/abs/2306.15063.

[2] 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.

[3] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

[4] Benjamin L. Edelman, Ezra Edelman, Surbhi Goel, Eran Malach, and Nikolaos Tsilivis. The evolution of statistical induction heads: In-context learning markov chains, 2024. URL https://arxiv.org/abs/2402.11004.

[5] Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. In-context language learning: Architectures and algorithms, 2024. URL https://arxiv.org/abs/2401.12973.

[6] Aaditya K. Singh, Stephanie C. Y. Chan, Ted Moskovitz, Erin Grant, Andrew M. Saxe, and Felix Hill. The transient nature of emergent in-context learning in transformers, 2023. URL https://arxiv.org/abs/2311.08360.

[7] Jesse Hoogland, George Wang, Matthew Farrugia-Roberts, Liam Carroll, Susan Wei, and Daniel Murfet. The developmental landscape of in-context learning, 2024. URL https://arxiv.org/abs/2402.02364.

[8] Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. What in-context learning "learns" in-context: Disentangling task recognition and task learning, 2023. URL https://arxiv.org/abs/2305.09731.

[9] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*, 2022.

[10] Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning, 2024. URL https://arxiv.org/abs/2402.18819.

[11] 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.

[12] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[13] OpenAI. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

[14] Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes, 2023. URL https://arxiv.org/abs/2208.01066.

[15] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models, 2023. URL https://arxiv.org/abs/2211.15661.

[16] Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent, 2023. URL https://arxiv.org/abs/2212.07677.

[17] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection, 2023. URL https://arxiv.org/abs/2306.04637.

[18] Yingcong Li, M. Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning, 2023. URL https://arxiv.org/abs/2301.07067.

[19] Tianyu Guo, Wei Hu, Song Mei, Huan Wang, Caiming Xiong, Silvio Savarese, and Yu Bai. How do transformers learn in-context beyond simple functions? a case study on learning with representations, 2023. URL https://arxiv.org/abs/2310.10616.

[20] Benjamin L Edelman, Surbhi Goel, Sham Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In *International Conference on Machine Learning*, pages 5793–5831. PMLR, 2022.

[21] Ruiqi Zhang, Spencer Frei, and Peter L. Bartlett. Trained transformers learn linear models in-context, 2023. URL https://arxiv.org/abs/2306.09927.

[22] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[23] Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. Larger language models do in-context learning differently, 2023. URL https://arxiv.org/abs/2303.03846.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[25] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623, 2021.

[26] Keyon Vafa, Justin Y. Chen, Jon Kleinberg, Sendhil Mullainathan, and Ashesh Rambachan. Evaluating the world model implicit in a generative model, 2024. URL https://arxiv.org/abs/2406.03689.

[27] Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=DeG07_TcZvT.

[28] Yida Chen, Fernanda Viégas, and Martin Wattenberg. Beyond surface statistics: Scene representations in a latent diffusion model. *arXiv preprint arXiv:2306.05720*, 2023.

[29] Adam Karvonen. Emergent world models and latent variable estimation in chess-playing language models, 2024. URL https://arxiv.org/abs/2403.15498.

[30] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks, 2020. URL https://arxiv.org/abs/2006.07710.

[31] Ekdeep Singh Lubana, Eric J Bigelow, Robert P Dick, David Krueger, and Hidenori Tanaka. Mechanistic mode connectivity. In *International Conference on Machine Learning*, pages 22965–23004. PMLR, 2023.

[32] 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.

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

A Experimental Details

A.1 Data

n determines the number of pre-drawn transition matrices of the training set. k is the number of states, here equivalent to the number of tokens. The transition matrix is thus $\mathbb{R}^{(k,k)}$. When drawing a new data point from the dataset, a transition matrix is chosen from the n matrices, say T_i. Then a start token is drawn from π_i, the stationary distribution corresponding to T_i. The Markov chain is continued using the same transition matrix, until a length of l is reached. k is fixed to 10 except in Fig 5(c), and l is always fixed to 512 during training.

A.2 Model

We use a standard transformer[24] architecture with a Rotational Positional Embedding(RoPE)[32] unless otherwise specified. The transformer has 2 layers, 4 heads and an embedding dimension of 64. The MLP layers upsample to 256 dimensions(a ratio of 4) expect in Fig 5(b) where the ratio is set to 0.25. The number of steps are adjusted to match the same total compute FLOPs.

A.3 Training

We train the model with the DGP using a batch size of 128. We have tried a batch size of 64 and 256 and observed the same results. We use the AdamW[33] optimizer with a learning rate of $1e-3$. Model parameters are initialized to $\mathcal{N}(\mathbf{0}, 0.02\mathbf{\Sigma})$.

B Additional Details for Solutions

Eq. 1 and Eq. 2 respectively defines the unigram and bigram likelihood over the training set transition matrices. π_i is the stationary distribution corresponding to T_i, which satisfies $\pi T = \pi$.

$$\text{Unigram Likelihood: } \mathcal{L}_U(T_i|\mathbf{x}_{0:t}) = \Pi_{j=1}^{t} \pi_{i[x_j]} \tag{1}$$

$$\text{Bigram Likelihood: } \mathcal{L}_B(T_i|\mathbf{x}_{0:t}) = \Pi_{j=1}^{t-1} T_{i[x_{j-1},x_j]} \tag{2}$$

$$\text{Bayesian Solution: } p(x_t|\mathbf{x}_{0:t-1}) = \sum_i p_i \; \mathcal{L}(T_i|\mathbf{x}_{0:t-1}) \; T_{i[x_{t-1},x_t]} \tag{3}$$

Eq. 3 describes the Bayesian averaging process, which takes either of the unigram or bigram likelihood and computes the next token probability. The next token probability can be simply understood as a weighted average over hypotheses of what transition matrix might be underlying the data.

Eq. 4 and Eq. 5 respectively defines the transition matrix inferred by the UICL solution and the BICL solution. Note that the transition matrix elements inferred from UICL does not depend on the indice i, i.e. it does not depend on the current state as it is a unigram solution.

$$\text{Unigram ICL: } \hat{T}^{U}_{[i,j]}(\mathbf{x}_t) = \hat{\pi}_{[j]}(\mathbf{x}_t) = \frac{\sum_{k=1}^{t} \delta_{x_k,j}}{t} \tag{4}$$

$$\text{Bigram ICL: } \hat{T}^{B}_{[i,j]}(\mathbf{x}_t) = \frac{1 + \sum_{k=1}^{t-1} \delta_{x_k,i}\delta_{x_{k+1},j}}{k + \sum_{k=1}^{t-1} \delta_{x_k,i}} \tag{5}$$

C Mechanistic Decomposition

We decompose the model output logits linearly as:

$$M(x_t|\mathbf{x}_{0:t-1}) = w_{ULBA}ULBA(x_t|\mathbf{x}_{0:t-1}) + w_{BLBA}BLBA(x_t|\mathbf{x}_{0:t-1}) \tag{6}$$

$$+ w_{UICL}UICL(x_t|\mathbf{x}_{0:t-1}) + w_{BICL}BICL(x_t|\mathbf{x}_{0:t-1}) \tag{7}$$

This system might seem under determined just from this equation, however $ULBA, BLBA, UICL, BICL$ has no hyperparameters and there is an infinite amount of $\mathbf{x}_{0:t}$ we can sample, so there exists an unique optimal solution. In practice, we draw 300 chains to determine the weights at each checkpoint. The phase diagrams in Fig 5 are constructed by assining each weight a color.