Analyzing limits for in-context learning

Omar Naim[1] Jérôme Bolte[2] Nicholas Asher[1]

[1]Institut de Recherche en Informatique de Toulouse, University of Toulouse, France
[2]Toulouse School of Economics, University of Toulouse Capitole, France
omar.naim.docs@gmail.com, jbolte@ut-capitole.fr, nicholas.asher@irit.fr

Abstract

Our paper challenges claims from prior research that transformer-based models, when learning in context, implicitly implement standard learning algorithms. We present empirical evidence inconsistent with this view and provide a mathematical analysis demonstrating that transformers cannot achieve general predictive accuracy due to inherent architectural limitations.

1 Introduction

Previous research suggests that transformers are capable of learning in-context function classes and use algorithms during inference, which would confer general, predictive accuracy on out-of-training distribution inputs. We challenge these claims by demonstrating that transformers do not employ such algorithms at inference time. We provide a mathematical analysis showing that inherent architectural restrictions prevent transformers from reaching general predictive accuracy Naim et al. [2025][1].

2 Methods/Approach

To understand the mechanisms underlying ICL, a capability acquired during training, we study two simple problems with small transformers: predicting values of polynomial functions from a context that partially specifies their graphs, and predicting the values of quantificational sentences like *every (some) number is positive* given an input sequence of numbers. These problems provide well-defined, well-studied settings using clean, structured data (see Appendix B). We draw on prior observations and experimental set up for our analysis; e.g., attention layers are necessary and sufficient to ICL these problems (see Section B), and so we concentrate on attention layer only transformers.

To perform ICL on a task or function g, the model gets a prompt of input-output examples of the form $(x_1, g(x_1), ...x_p, g(x_p), x)$, and then predicts a value for $g(x)$. We explore the ICL task on functions in $\mathbb{R}^n[X]$ for $1 \leq n \leq 6$, using a variety of training and testing distributions and a variety of transformer models.[2] For details about training and evaluation metrics, see Appendix C.

3 Results/Discussion

(1) Our experiments reveal three key findings: (i) transformers, in particular two-layer or more attention only models, achieve near-zero error when test and training distributions coincide, but (ii) fail to generalize under distribution shifts. (iii) Transformers exhibit similar error rates across a wide range of polynomial functions, even though computational difficulty varies with the polynomial degree (Table 1). These results contradict prior claims (see Section B) that transformers implement

[1]Our full paper can be found in `https://arxiv.org/pdf/2502.03503`
[2]Our code is available at `https://github.com/omyokun/icl-polynomials`

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: What Can('t) Transformers Do?.

linear regression or analogous algorithms for ICL of linear functions, since such algorithms are inherently robust to distributional shifts and typically require multiple attention layers for their implementation. For additional details, see Appendix D.2.

(2) We investigate model behavior under distribution shifts and observe the presence of *boundary values*, extremes that the model cannot exceed during prediction, thus limiting generalization beyond these values. Boundary values are a feature of all function learning tasks we looked at. Boundary values appear consistently across all function learning tasks we examined. Changing the training distribution from Gaussian to Uniform confirms that these boundaries align with the largest and smallest values seen during training. (3) and (4) emphasize the critical role of memory in ICL, rather than inference-based learning. For supplementary analysis, see Appendix D.5.

(3) To verify that the observed limitations are not due to the standard training procedure (T), we explore alternative training strategies, A_1 and A_2. For linear functions, we leveraged the basis representation $(1, x)$. Under A_1, the model was trained separately on the directions $a \cdot 1$ and $b \cdot x$, with a, b drawn from the same distribution; this approach yielded lower performance than T. For A_2, training was performed on multiple linear combinations $a \cdot e_i = a \cdot (\cos(\theta_i) \cdot 1 + \sin(\theta_i) \cdot x)$, covering various directions in the function space (see Figure 1, Appendix C). Although A_2 outperformed A_1, it still fell short of T. A similar trend was observed for polynomial functions: even with abundant training data, small transformers appear unable to capture the underlying functional structure.

(4) With an ablation study, we identify Layer Normalization as the main contributor to boundary values. However, removing Layer Normalization does not resolve the generalization problem, suggesting that these limitations stem from deeper architectural constraints. For further details, see Appendix D.6.

(5) We formalize the attention-only transformer model and show theoretically that while boundary values stem from Layer Normalization, the generalization issue is inherent to the architecture itself.

A trained transformer with L layers and H attention heads defines a deterministic function $\hat{f}^{\theta}$ for ICL: given a prompt $(x_1, g(x_1), \ldots, x_p, g(x_p), x)$, it outputs the prediction $\hat{f}^{\theta}(x_1, g(x_1), ...x_p, g(x_p), x)$:

$$LN\left[x^{(L-1)} + \left(\sum_{h=1}^{H} \gamma_h \left(\sum_{j=1}^{p} s\left(x^{(L-1)}(Q^{h,L-1}{K^{h,L-1}}^T){x_j^{(L-1)}}^T\right) x_j^{(L)} + s\left(x^{(L-1)}(Q^{h,L-1}{K^{h,L-1}}^T)\right){x^{(L-1)}}^T\right) V^{h,L-1}\right)\right] \cdot W_{dec} \quad (1)$$

with Q, K and V the Query, Key and Value matrices, s the scoring function, W_{dec} the decoding matrix, and x^l the representation of the input token x at layer l (Section E.1). This allows an analysis of the model's ICL behavior for any prompt, including out-of-distribution and boundary cases.

Lemma 1 *The multihead attention function tends to a single linear function $x \to ax + b$, as $x \to \infty$.*

Lemma 1 implies that Layer Normalization tends towards a constant for large inputs (proofs in Appendices E.2 and E.3). We thus have a mathematical derivation for boundary values.

Lemma 2 *A 1 layer AL transformer cannot ICL the class of linear functions on significantly out of distribution inputs. For the proof see Appendix E.4*

With Lemma 1, we prove Lemma 2, which furnishes a base case for an inductive proof that attention only transformers of any complexity fail to learn the class of linear functions on inputs significantly out of training distribution (see Theorem 1 Appendix E.4) thus providing a theoretical ground for our empirical findings.

Our mathematical analysis also imposes intrinsic limits on ICL. Since Proposition 1 relies only on the mathematical form of f^{θ} and the presence of high norm inputs, its generalization limits for ICL apply to all tasks and data.

Corollary 1 *Architectural constraints prevent generalization beyond training distributions.*

While removing normalization unexpectedly improves performance in our setting (Table 4), if the query input has a high embedding norm, the model's output effectively precludes ICL.

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. Understanding in-context learning in transformers and llms by learning to learn discrete functions. *arXiv preprint arXiv:2310.03016*, 2023.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Yan Dai, Kwangjun Ahn, and Suvrit Sra. The crucial role of normalization in sharpness-aware minimization. *Advances in Neural Information Processing Systems*, 36:67741–67770, 2023.

P Kingma Diederik. Adam: A method for stochastic optimization. *(No Title)*, 2014.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Deqing Fu, Tian-Qi Chen, Robin Jia, and Vatsal Sharan. Transformers learn higher-order optimization methods for in-context learning: A study with linear models. *arXiv preprint arXiv:2310.17086*, 2023.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

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

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, 2023.

Angeliki Giannou, Liu Yang, Tianhao Wang, Dimitris Papailiopoulos, and Jason D Lee. How well can transformers emulate in-context newton's method? *arXiv preprint arXiv:2403.03183*, 2024.

Maximilian Mueller, Tiffany Vlaar, David Rolnick, and Matthias Hein. Normalization layers are all that sharpness-aware minimization needs. *Advances in Neural Information Processing Systems*, 36:69228–69252, 2023.

Omar Naim and Nicholas Asher. On explaining with attention matrices. In *ECAI 2024*, pages 1035–1042. IOS Press, 2024a.

Omar Naim and Nicholas Asher. Re-examining learning linear functions in context. arXiv:2411.11465 [cs.LG], 2024b.

Omar Naim and Nicholas Asher. Improving in-context learning with a better scoring function. *arXiv preprint arXiv:2508.14685*, 2025.

Omar Naim, Jerome Bolte, and Nicholas Asher. Analyzing limits for in-context learning. *arXiv preprint arXiv:2502.03503*, 2025.

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

Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. *arXiv preprint arXiv:2306.04891*, 2023.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. *Advances in Neural Information Processing Systems*, 36, 2024.

Juergen Schmidhuber, Jieyu Zhao, and MA Wiering. Simple principles of metalearning. *Technical report IDSIA*, 69:1–23, 1996.

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

Jingfeng Wu, Difan Zou, Zixiang Chen, Vladimir Braverman, Quanquan Gu, and Peter L Bartlett. How many pretraining tasks are needed for in-context learning of linear regression? *arXiv preprint arXiv:2310.08391*, 2023.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. Association for Computational Linguistics, 2024.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

Qinan Yu, Jack Merullo, and Ellie Pavlick. Characterizing mechanisms for factual recall in language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9924–9959, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.615. URL `https://aclanthology.org/2023.emnlp-main.615/`.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

Yufeng Zhang, Fengzhuo Zhang, Zhuoran Yang, and Zhaoran Wang. What and how does in-context learning learn? bayesian model averaging, parameterization, and generalization. *arXiv preprint arXiv:2305.19420*, 2023.

A Limitations

Our experiments use a controlled setting with synthetic mathematical functions. Although this setup enables a precise analysis of model behavior and architectural effects, it abstracts away from many complexities present in real-world NLP tasks, such as natural language variability, noise, and hierarchical structure.

Our analysis is primarily focused on decoder-only transformers with relatively modest model sizes (up to 38M parameters). While the mathematical results hold for all transformers, larger models used for instance in production NLP systems may very well exhibit less drastic limitations on generalization due to scale.

Despite these limitations, our framework and findings lay the groundwork for understanding and addressing ICL failure modes, offering a foundation for improved architectural design and training strategies in both synthetic and natural language domains.

B Related Work

Brown et al. [2020] introduced in-context learning (ICL) as a framework that occurs entirely at inference time; LLMs learn tasks by analogy based on examples presented in the prompt without updating their parameters. Despite its promise, the underlying mechanisms behind ICL remain only partially understood.

Akyürek et al. [2022], Von Oswald et al. [2023], Fu et al. [2023], Xie et al. [2021], Wu et al. [2023], Zhang et al. [2023], Panwar et al. [2023] have suggested that LLMs perform implicit gradient-based updates, higher-order optimization, or approximate Bayesian inference when prompted with in-context examples. However, these remain speculative hypotheses grounded in what the architecture could, in principle, implement, rather than direct evidence of the mechanisms at play. As noted by Dong et al. [2022], much of this analysis remains confined to simple tasks like linear regression or Boolean functions Bhattamishra et al. [2023].

Garg et al. [2022] showed that small transformers trained from scratch on synthetic data can acquire ICL abilities. Garg et al. [2022] also showed that while model performance degraded somewhat in out of distribution testing, it was relatively robust. Similarly, Raventós et al. [2024] studied how ICL performance changes as a function of the number of pretraining examples.

Olsson et al. [2022] investigated architectural components responsible for ICL, proposing that *induction heads*, a learned copying and comparison mechanism, underlie ICL. Geva et al. [2021] and Bietti et al. [2024] explored the role of memory in transformers, showing that models heavily rely on memorization via attention matrices. This is further supported by findings from Yu et al. [2023] and Geva et al. [2023], who argue that transformers prioritize memorized patterns during inference.

Xie et al. [2021], Zhang et al. [2024], Giannou et al. [2024], Naim and Asher [2024b] show that when train and inference distributions do not coincide, ICL performance on linear functions degrades. Naim and Asher [2024b] showed that transformers trained from scratch on linear functions fail to extrapolate, once we increase the range of out of distribution testing data from that examined in Garg et al. [2022]: performance degrades from accurate prediction, to constant outputs (which they call *boundary values*) or to near-random behavior. Naim and Asher [2025] similarly showed that that small transformers as well as larger LLMs failed to extrapolate on out of distribution testing data on the quantificational task mentioned above. Wu et al. [2024] use the notion of a counterfactual task to show the limits of ICL.

C Training details

Training[3] a model to perform in-context learning can be seen as meta-learning Schmidhuber et al. [1996] where the model learns to perform new tasks based on in-context examples, without any changes on its parameters. In practice, for autoregressive models, the ICL objective is implemented through standard supervised learning Brown et al. [2020], Garg et al. [2022], Akyürek et al. [2022]: the model is presented with multiple functions from a given class, each evaluated at several input points, and is trained to learn the underlying function class. Details are found in Appendix A. Our ICL tasks involve training from scratch on sequences containing in-context examples (input-output pairs) $(x_1, f(x_1), ..., x_i)$ ending with a query input x_i that is used to generate the corresponding output. We train a transformer model f^θ parameterized by θ to minimize the expected ICL loss over all the prompts:

$$\min_{\theta} \mathbb{E}_{g\sim\mathcal{D}_{\mathcal{F}}} \left[\mathbb{E}_{x_1,\ldots,x_p\sim\mathcal{D}_{\mathcal{I}}} \sum_{i=0}^{k} \ell\left(y_{i+1},\ f^{\theta}\left((x_1, g(x_1), \ldots, x_{i+1})\right)\right)\right] \quad (2)$$

where $\ell(.,.)$ represents the loss function: we use squared error for the linear function task and cross-entropy for the quantifier task. In the quantifier task, y represents the ground truth given a sequence ending with the input x. In the function task, y is the ground truth value of $f(x)$, where f is the underlying function generating the sequence up to x. We employ curriculum learning on a set S of training sequences of varying lengths, ranging from 1 to $k = 40$. Models are trained for 500k steps and use a batch size of 64, using Adam optimizer. The models saw over 1.3 billion training examples for each distribution we studied.

[3] Our code can be found in https://anonymous.4open.science/r/icl-polynomials/

C.1 Evaluation metric

We evaluated the model's generalization capabilities across a range of testing distributions for both functions, denoted $D_{\mathcal{F}}^{test}$ and input data points, denoted $D_{\mathcal{I}}^{test}$.

For each testing scenario $(D_{\mathcal{I}}^{test}, D_{\mathcal{F}}^{test})$, we generate a set of $N = 100$ functions sampled from $D_{\mathcal{F}}^{test}$. For each function we sample $N_b = 64$ batches, where each batch contains $N_p = 41$ data points drawn from $D_{\mathcal{I}}^{test}$. Within each batch b, we predict output of prompts $(x_1^b, f(x_1^b), ..., x_{k-1}^b, f(x_{k-1}^b), x_k^b)$ with $k \geq 2$. The mean squared error (MSE) is computed over all predictions within each batch and averaged across all batches for a given function. Finally, we report the overall ICL evaluation metric as the average MSE across the entire set of test functions.

$$\epsilon_\sigma = \frac{1}{N} \sum_{i=1}^{N} \sum_{b=1}^{N_b} \frac{1}{N_b} \left(\frac{1}{N_p} \sum_{i=3}^{N_p} (\text{pred}_i^b - y_i^b)^2 \right) \tag{3}$$

To improve interpretability of squared error values, we define *error rate* $r_\epsilon = \frac{\epsilon_\sigma}{|\epsilon_* - \epsilon_0|}$ where ϵ_* is the best ϵ_σ error for a model M with $\hat{f}(x)$ calculated with Least Squares, and ϵ_0 is the worst ϵ_σ error for a model M such that $\forall x$, $\hat{f}_M(x) = 0$. In all our error calculations, we exclude the first $n + 1$ predictions of each batch from the squared error calculation for $g \in \mathbb{R}^n[X]$, since we need at least n+1 points to be able to find g.

To ensure fair and meaningful comparisons across models, we fix the random seed when generating the 100 test functions and the corresponding prompting points x_i. This guarantees that all models are evaluated on the same set of functions and input distributions. The purpose of this evaluation setup is to assess how well models generalize when progressively exposed to out-of-distribution (OOD) data. By keeping the test conditions constant while varying the models, we can isolate the effect of model architecture and training on their ability to adapt to novel inputs through in-context learning.

C.2 More training details

Additional training information: We use the Adam optimizer Diederik [2014] , and a learning rate of 10^{-4} for all models.
Computational resources: We used Nvidia A-100 GPUs to train the different versions of transformer models from scratch.

C.3 Alternative training strategies

See figure 1.

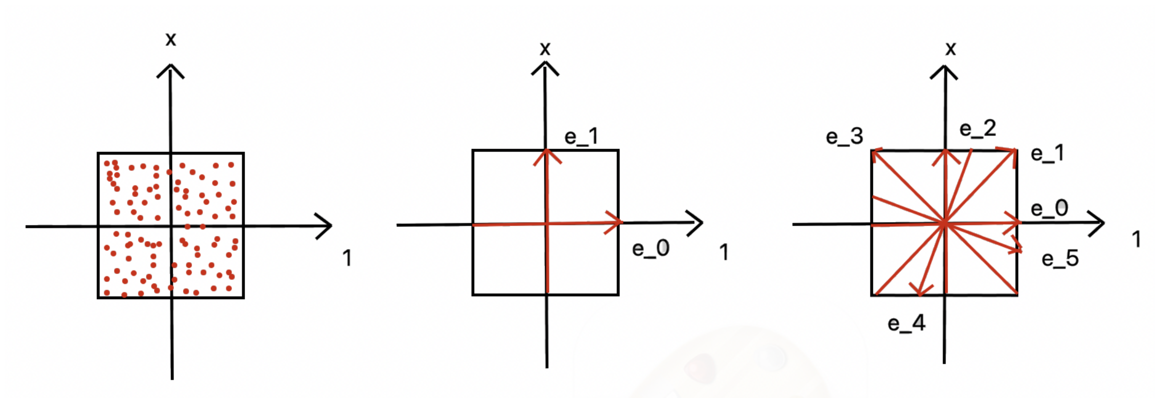

Figure 1: Representation of the different types of training, based on the polynomials of degree 1: (1,X). (Left), training on a cloud of points, (middle) on the two principal directions of the basis and (right) training on several directions. The rectangle represents the set of polynomials of degree 1 taking the weights in U(-1,1).

D Important Results:

D.1 Two Views of Learning In-Context

Learning a task in-context means that the model is able to predict the output for a new input using only a few input-output examples provided within the same sequence. Specifically, given a prompt of few input-output examples of the form $(x_1, g(x_1), ...x_p, g(x_p), x)$, a transformer is able to approximate the value of $g(x)$, regardless of which specific input samples x_i are included in the in-context examples.

We distinguish two notions of "learning a a class of functions in context". The first, call it *ICL*$_1$, states that a transformer model $\hat{f}^\theta$ can ICL a function class $\mathcal{F}$ on a given distribution sampling functions $g \in D_\mathcal{F}$ and a distribution matching inputs $x_i \in D_\mathcal{I}$ where both correspond to training distribution:

$$\begin{gathered}\forall x_i, y_i, x \in D_\mathcal{I} \; \forall g \in D_\mathcal{F}, \\ ||\hat{f}^\theta(x_1, g(x_1), ...x_p, g(x_p), x) - g(x)||^2 \approx \\ ||\hat{f}^\theta(y_1, g(y_1), ...y_p, g(y_p), x) - g(x)||^2 < \epsilon\end{gathered} \tag{4}$$

A second definition of ICL is more demanding: a model M can *ICL*$_2$ a target function class $\mathfrak{G}$ if it is capable of approximating all functions within $\mathfrak{G}$ across all possible distributions of inputs. Formally,

$$\begin{gathered}\forall x_i, y_i, x \forall g \in \mathfrak{G} ||\hat{f}_{x_1, g(x_1), ...x_p, g(x_p)}(x) - g(x)||^2 \approx \\ ||\hat{f}_{y_1, g(y_1), ...y_p, g(y_p)}(x) - g(x)||^2 < \epsilon\end{gathered} \tag{5}$$

D.2 Can transformer-based models ICL polynomial functions?

We study the ICL problem for the polynomial function classes $\mathbb{R}^n[X]$, with degree $n \in \{1, \ldots, 6\}$. Both inputs and coefficients are drawn from the uniform distribution $[-1, 1]$, i.e. $D_\mathcal{I}, D_\mathcal{F} \sim \mathcal{U}(-1, 1)$. For each degree n, we train a transformer model from scratch on that distribution.

Our experiments show that all tested models can successfully *ICL*$_1$ this task: given prompts of the form $(x_1, g(x_1)), \ldots, x)$, the models predict $g(x)$ with accuracy comparable to classical polynomial regression methods such as Least Squares, which are known to achieve optimal recovery when the polynomial degree is fixed. This regression-level performance in-distribution could suggest, as argued in prior work, that transformers implement algorithmic procedures resembling linear regression during inference. As shown in Figure 2, transformers consistently attain low mean squared error for polynomials of degree up to six when $D_\mathcal{I}^{test}, D_\mathcal{F}^{test} \sim \mathcal{U}(-1, 1)$.

However, the same models fail to *ICL*$_2$: when evaluated under distribution shifts in either inputs or coefficients, prediction errors increase substantially, in stark contrast with algorithms such as Least Squares that remain robust. This observation highlights a critical limitation: while transformers can mimic regression-like behavior in-distribution, their success does not arise from implementing algorithms like linear regression or least squares in inference. If they did, their performance would be insensitive to distributional changes.

degree	models / σ	1	2	3	4	5	6	7	8	9	10
1	M1	0.0	0.03	0.55	1.37	4.0	5.17	9.04	12.07	19.28	27.85
2	M2	0.0	0.02	0.48	1.49	4.01	6.41	9.69	13.11	19.96	32.97
3	M3	0.0	0.02	0.41	1.25	3.69	6.77	9.12	12.14	19.34	31.57
4	M4	0.0	0.03	0.44	1.48	4.66	7.65	10.5	14.47	20.36	32.94
5	M5	0.0	0.02	0.46	1.51	4.52	7.9	10.52	16.4	21.84	37.66
6	M6	0.00	0.04	0.40	1.20	3.73	6.56	10.03	14.62	20.52	34.75

Table 1: Comparison to show the evolution of squared error ϵ, with $D_\mathcal{I}^t \sim \mathcal{U}(-1, 1)$, $D_\mathcal{F}^t \sim \mathcal{U}(-\sigma, \sigma)$ for models for the models M_i each trained on degree i and tested on the same degree.

Although all models with two (Table 2) or more layers performed nearly perfectly when test distributions closely matched training distributions, their performance deteriorated significantly, in contrast

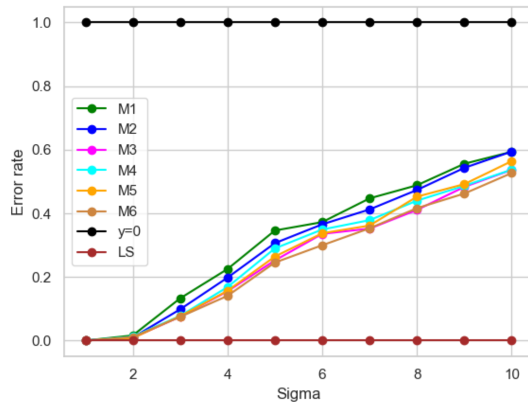

Figure 2: Evolution of error rates for various 12L8AH $d_{emb} = 256$ models with $D_{\mathcal{F}}, D_{\mathcal{I}}, D_I^t \sim \mathcal{U}(-1,1)$ and $D_F^t \sim \mathcal{U}(-\sigma,\sigma)$ for various σ, each trained from scratch on a different degree. E.g., Mn is a model trained on degree n only. The black line is a predictor that yields $f(x_n) = 0, \forall f$ and $\forall x_n$. The dark red line LS represents a perfect estimator with our clean input data.

to the modest deteriorations observed by Garg et al. [2022], once the test distributions deviated significantly from the training regime. This indicates model sensitivity when predicting values for higher order polynomials and continuous functions, like that observed by Naim and Asher [2024b] for linear functions, to shifts in both $D_{\mathcal{I}}$ and $D_{\mathcal{F}}$. Increasing model size from 22.5M to 38M parameters Naim and Asher [2024b] for linear functions did not significantly improve performance, suggesting that the observed limitations are not simply due to model capacity but may reflect deeper architectural constraints.[4]

degree	models / σ	1	2	3	4	5	6	7	8	9	10
1	M1	0.00	0.03	0.83	2.12	6.26	7.60	14.15	18.07	27.66	39.17
2	M2	0.00	0.04	1.07	2.76	7.23	10.69	15.70	20.25	28.85	44.72
3	M3	0.00	0.05	0.86	2.25	6.37	11.44	15.05	19.77	29.68	46.47

Table 2: Comparison showing the evolution of the squared error ϵ for 2L8AH models, with $D_{\mathcal{I}}^t \sim \mathcal{U}(-1,1)$ and $D_{\mathcal{F}}^t \sim \mathcal{U}(-\sigma,\sigma)$, for models M_i, each trained on degree i and tested on the same degree.

D.3 Can transformer-based models ICL quantification tasks?

We next consider the ICL problem for quantificational reasoning over real-valued sequences. Our quantification task is to predict the truth of the simple quantified sentences "every (some) number in the sequence is positive" and given a contextually given string of numbers of length 40. Our training set of strings S contain numbers chosen from a training distribution $D_{\mathcal{I}}$, which we set to the Gaussian distribution $\mathcal{N}(0,1)$.

Our experiments Naim and Asher [2025] show that models can reliably *ICL$_1$* both tasks: given a few demonstration examples of sequences paired with their quantifier truth values, they infer the quantificational property of new sequences with high in-distribution accuracy.

However, as in the polynomial setting, the same models fail to *ICL$_2$*: under distribution shifts in either the underlying value distribution or sequence length, generalization degrades significantly. This stands in contrast with classical symbolic procedures (e.g., scanning all elements), which are invariant to such shifts. Thus, transformers' apparent success in-distribution does not indicate an algorithmic implementation of quantifier reasoning, but rather a brittle pattern-matching strategy tied to training conditions.

[4]by taking $d_{emb} = 512$ instead of $d_{emb} = 256$.

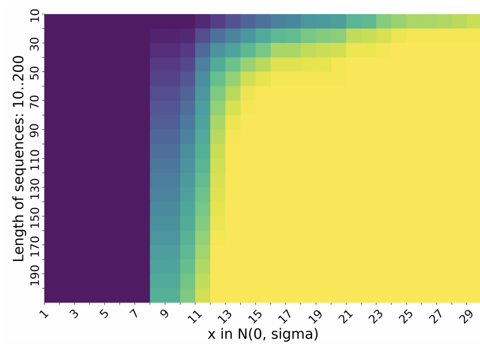

Figure 3: Heatmaps showing the evolution of errors for the 12L8AH model on the "every" task. Model was trained on data in $D_{\mathcal{I}} = \mathcal{N}(0, 1)$ for lengths from 11 to 40 and tested in $D_{\mathcal{I}}^{test} = \mathcal{N}(0, \sigma)$ for $\sigma \in \{1, ..., 10\}$ and lengths from 10 to 200 for each task. Yellow represents a much higher error rate than purple.

A heat map analysis of the attention weights[5] revealed that the models did not implement a recursive strategy to leverage autoregressive predictions, where attention would ideally focus on the query and the most recent input to generate the output . Instead, attention was distributed almost uniformly across input tokens, or at least spanned many elements, even in successful predictions (Figure 4). Notably, this pattern was consistent across all attention heads and layers for the "every" and "some" tasks, respectively, indicating that the models relied on a broad survey of the input sequence rather than stepwise, algorithmic computations.

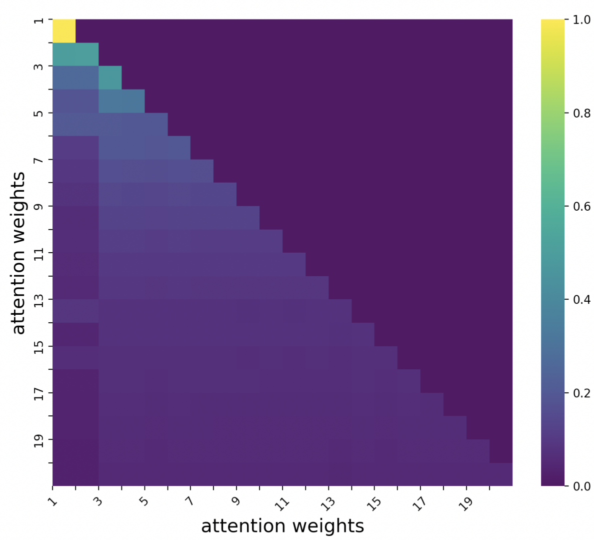

Figure 4: Heatmap showing the evolution of attention weights in an attention head of the last layer of the 12AL8AH model.

D.4 Training Distribution Width Effects on Out-of-Distribution Generalization

We evaluated the generalization capabilities of transformers trained from scratch on linear functions by testing their performance across varying distribution ranges. Table 3 reports squared errors for models trained on uniform distributions $\mathcal{U}(-a, a)$ with $a \in \{1, 5, 10, 100\}$ and tested on inputs from $\mathcal{U}(-1, 1)$ and outputs from $\mathcal{U}(-\sigma, \sigma)$ for varying σ. The results reveal a clear relationship between training distribution width and out-of-distribution generalization. Models trained on narrow distributions ($\mathcal{U}(-1, 1)$) achieve near-zero in-distribution error ($\sigma = 1$) but degrade rapidly as the test

[5]We can also use Naim and Asher [2024a] to further interpret these attention weights.

distribution widens. Models trained on moderate distributions ($\mathcal{U}(-5,5)$ and $\mathcal{U}(-10,10)$) exhibit robust performance, maintaining sub-unit errors across all test conditions, with the $\mathcal{U}(-10,10)$ model achieving particularly stable results. Training with extremely broad distributions, such as $\mathcal{U}(-100,100)$, can extend boundary values but comes at a substantial cost: performance deteriorates sharply across all testing scenarios, with errors exceeding 2000, indicating that overly wide training distributions hinder the learning of precise linear relationships. Overall, these findings suggest that moderate expansion of the training distribution improves generalization without sacrificing accuracy, but this benefit saturates and eventually reverses at extreme scales.

models \ σ	1	2	3	4	5	6	7	8	9	10
$\mathcal{U}(-1,1)$	0.0	0.03	0.55	1.37	4.0	5.17	9.04	12.07	19.28	27.85
$\mathcal{U}(-5,5)$	0.01	0.01	0.02	0.03	0.03	0.05	0.12	0.27	0.75	1.61
$\mathcal{U}(-10,10)$	0.13	0.15	0.17	0.2	0.26	0.26	0.32	0.35	0.41	0.49
$\mathcal{U}(-100,100)$	2217.84	2373.82	2494.31	2526.93	2472.45	2467.52	2317.92	2232.03	2129.0	2092.81

Table 3: Comparison showing the evolution of squared errors for models trained on different distributions $D_{\mathcal{I}}, D_{\mathcal{F}} \sim \mathcal{U}(-a,a)$, for $a = 1, 5$ or 100 sampling from $\mathcal{P}^1$ with $D_i^t \sim \mathcal{U}(-1,1)$ and $D_F^t \sim \mathcal{U}(-\sigma,\sigma)$.

D.5 Boundary values and limits of generalization

D.5.1 Boundary values : Description

Through a systematic evaluation of shifted distributions for linear functions, Naim and Asher [2024b] identified fundamental limits to generalization: each model exhibits fixed boundary values (B^-, B^+) that constrain predictions to the smallest and largest outputs observed during training. These boundary values act as hard cutoffs, beyond which the model is unable to extrapolate.

We extend this analysis to polynomial functions of degree greater than one. Our experiments show that the same phenomenon persists: regardless of polynomial degree, models maintain fixed boundary values. Importantly, this behavior is not an artifact of the task but rather a characteristic of the model itself. Even when varying training strategies, the induced boundary values remain unchanged (Figure 5).

This finding suggests that transformers do not learn open-ended functional rules, but instead interpolate within a bounded range tied to their training distribution. Boundary-induced failures thus provide further evidence against algorithmic interpretations of ICL, where methods such as linear regression would extrapolate beyond observed ranges.

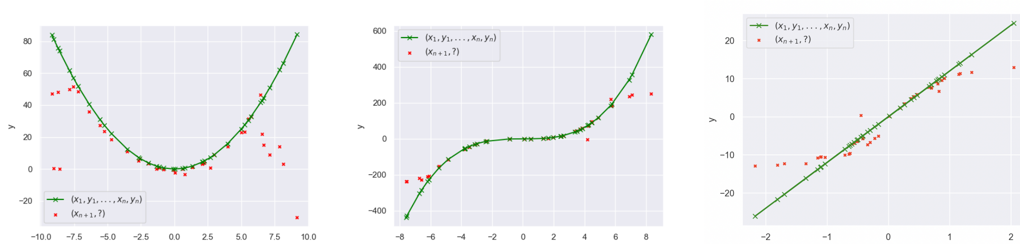

Figure 5: First two plots for models $P2$ for $f(x) = x^2$ $P3$ for $f(x) = x^3$, trained on $D_{\mathcal{I}} = D_{\mathcal{F}} = N(0,1)$ showing boundary values. Third plot shows boundary values for 2L32ah attention only model, with $d_{embedding} = 256$ to ICL the function $f(x) = 12x$.

D.6 Layer Normalization as the source of boundary values

To investigate the origin of the boundary value phenomenon, we performed ablation studies by systematically removing components of the transformer architecture and retraining models from scratch on the polynomial ICL task. For each ablation, we evaluated both ICL performance and the presence of boundary values.

We found that removing layer normalization causes the model to produce significantly larger output values, effectively eliminating the boundary values observed previously. This behavior indicates that layer normalization acts as the principal mechanism enforcing predictive thresholds: by constraining the magnitude of internal activations, it prevents the model from extrapolating beyond the training range, thereby limiting generalization in high-input regimes.

Surprisingly, we did not observe the expected performance degradation. Prior work has emphasized the role of normalization in mitigating internal covariate shift and stabilizing optimization dynamics Ba et al. [2016], Dai et al. [2023], Mueller et al. [2023]. Yet in our controlled regime with $D_{\mathcal{I}}, D_{\mathcal{F}} \sim \mathcal{U}(-1,1)$, removing normalization *improves* performance (Table 4). We attribute this to the stability of the uniform $[-1,1]$ interval: inputs are compact, symmetric, and naturally well-scaled, making normalization largely redundant.

Without normalization, the model is able to approximate functions more accurately near the boundaries of the input domain, where normalization would otherwise constrain predictions. Thus, in settings where input distributions are inherently stable, removing normalization can increase representational capacity.

However, this modification does not resolve the broader generalization problem: even without boundary values, the models remain unable to perform *ICL*$_2$, as Table 4 shows for shifted distributions.

Observation 1 *Removing normalization eliminates boundary values and can improve in-distribution accuracy, but it does not enable out-of-distribution generalization.*

models \ σ	1	2	3	4	5	6	7	8	9	10
With LN	3.5×10^{-5}	0.03	0.55	1.37	4.0	5.17	9.04	12.07	19.28	27.85
Without LN	9.7×10^{-6}	3×10^{-3}	0.16	0.81	2.99	3.37	7.52	10.92	17.40	25.44

Table 4: Comparison showing the evolution of squared errors for models trained on degree 1 on different distributions $D_{\mathcal{I}}, D_{\mathcal{F}} \sim \mathcal{U}(-1,1)$, and tested on $D_i^t \sim \mathcal{U}(-1,1)$ and $D_F^t =\sim \mathcal{U}(-\sigma,\sigma)$ $\forall\sigma \in \{1,\cdots,10\}$, with and without layer normalization.

E Proofs

E.1 Some Basics

In this section, we unpack an attention only transformer model to provide mathematical expression of $\hat{f}^{\theta}$. This will enable us to formally show certain properties of ICL. A transformer is a neural network model that maps a sequence of input vectors $(x_1,\cdots,x_n)$ to a corresponding sequence of output vectors, through a stack of layers. Each layer in the transformer operates on a sequence of vectors $X^{(l)} = (x_1^{(l)}, x_2^{(l)}, ..., x_n^{(l)})$, which represents the sequence at layer l, and produces a new matrix $X^{(l+1)}$ for the next layer.

We focus on the case of autoregressive, decoder-only transformer model composed of L layers and H attention heads. In each layer, the input sequence is first processed by a multi-head self-attention mechanism. Each attention head computes attention weights and context vectors independently. The attention head operation is defined as:

$$(x_1^{(l)}, ..., x_n^{(l)}) \rightarrow (A^{h,(l+1)}(x_1^{(l)}), ..., A^{h,(l+1)}(x_n^{(l)}))$$

where $\forall i \in \{1,\cdots,n\}$

$$A^{h,(l+1)}(x_i^{(l)}) = \sum_{j=1}^{i} s\left(x_i^{(l)}(Q^h {K^h}^T){x_j^{(l)}}^T\right) x_j^{(l)} V^h \tag{6}$$

with $Q^h \in \mathbb{R}^{d_{model} \times d_q}$, $K^h \in \mathbb{R}^{d_{model} \times d_k}$ and $V^h \in \mathbb{R}^{d_{model} \times d_v}$ are Query, Key and Value matrices with $d_q = d_k = d_v = d_{model}/h$ and s is the scoring function.

The outputs of attention heads are concatenated then passed through a linear layer to form the output of the multi-head attention mechanism :

$$(A^{(l+1)}(x_1^{(l)}), ..., A^{(l+1)}(x_n^{(l)})) \tag{7}$$

where $\forall i \in \{1, \cdots, n\}$

$$A^{(l+1)}(x_i^{(l)}) = \sum_{h=1}^{H} A^{h,(l+1)} \gamma_h(x_i^{(l)})$$

with $\gamma_h \in \mathbb{R}^{d_v \times d}$ are the weights of the linear layer.

The output of the multi-head attention module is then passed through the *Add & Norm* operation. The result $\forall i \in \{1, \cdots, n\}$ is:

$$AN_i^{(l+1)} = LN(A^{(l+1)}(x_i^{(l)}) + x_i^{(l)})$$

The normalized output is then passed through a feedforward network:

$$W_1^{(l+1)} \sigma \left(W_2^{(l+1)} AN_i^{(l+1)} \right)$$

The output of the feedforward network is then passed through another *Add & Norm* operation to produce the final output of the layer $l + 1$:

$$LN \left(W_1^{(l+1)} \sigma \left(W_2^{(l+1)} AN_i^{(l+1)} \right) + AN_i^{(l+1)} \right)$$

The transformer, denoted by $\hat{f}^\theta$ processes the ICL input of the form $(x_1, g(x_1), \cdots, x)$, and produces a prediction $\hat{f}^\theta(x_1, g(x_1), ...x_p, g(x_p), x)$ after processing the inputs in L layers. Given the explicit form of $\hat{f}^\theta$, which operates autoregressively over the input sequence, we can in principle express the output as a deterministic function of the entire prompt. This allows us to analyze and potentially characterize the model's ICL behavior for any specific prompt configuration, making it possible to study whether and how it implicitly implements a learning algorithm over the in-context examples. We'll analyze the attention matrix and normalization separately.

E.2 Proof for Lemma 1

Consider a prompt $(x_1, \cdots, x_p, x)$ where $\forall i \in \{1, \cdots, p\}$ x_i are fixed, and the only variable is x. The proof will be done by induction on the number of layers l of the attention only transformers, we will show it first for $l = 1$

Call $\tilde{x_i} = x_i W$ the linear embedding corresponding to x_i, used in the training. The output of multi-head attention for 1 layer H attention heads is:

$$Attn(x1, \cdots, x_p, x) = \sum_{h=1}^{H} \left(\sum_{j=1}^{p} s \left(x x_j (W Q^h {K^h}^T W^T) \right) x_j + s \left(x^2 (W Q^h {K^h}^T W^T) \right) x W V^h \gamma_h \right)$$

By replacing the scoring function s in Equation 6, we have:

$$\begin{aligned} Attn(x1, \cdots, x_p, x) = \sum_{h=1}^{H} \Bigg(& \sum_{j=1}^{p} \frac{x_j e^{x x_j (W Q^h {K^h}^T W^T)}}{e^{x^2 (W Q^h {K^h}^T W^T)} + \sum_{k=1}^{p} e^{x x_k (W Q^h {K^h}^T W^T)}} \\ & + \frac{x e^{x^2 (W Q^h {K^h}^T W^T)}}{e^{x^2 (W Q^h {K^h}^T W^T)} + \sum_{k=1}^{p} e^{x x_k (W Q^h {K^h}^T W^T)}} \Bigg) W V^h \gamma_h \end{aligned}$$

To simplify, let's call $\alpha_h = W Q^h {K^h}^T W^T \in \mathbb{R}$ and $\zeta_h = W V^h \gamma_h \in \mathbb{R}^d$

We then have:

$$Attn_p(x) = \sum_{h=1}^{H} \left(\sum_{j=1}^{p} \frac{x_j e^{x x_j \alpha_h}}{e^{x^2 \alpha_h} + \sum_{k=1}^{p} e^{x x_k (\alpha_h}} + \frac{x e^{x^2 \alpha_h}}{e^{x^2 \alpha_h} + \sum_{k=1}^{p} e^{x x_k \alpha_h}}) \right) \zeta_h \tag{8}$$

Let's call $\mu_j^h : x \to \frac{x_j e^{xx_j\alpha_h}}{e^{x^2\alpha_h}+\sum_{k=1}^{p} e^{xx_k(\alpha_h}}$ and $\beta^h : x \to \frac{xe^{x^2\alpha_h}}{e^{x^2\alpha_h}+\sum_{k=1}^{p} e^{xx_k\alpha_h}}$

So,

$$Attn(x1, \cdots, x_p, x) = \sum_{h=1}^{H} \left(\sum_{j=1}^{p} \mu_j^h(x) + \beta^h(x) \right) \zeta_h \tag{9}$$

to see the behavior of the function at infinity, we define the following sets $\mathbb{H}^- = \{h \in \{1, ..., H\} : \alpha_h < 0\}$, $\mathbb{H}^+ = \{h \in \{1, ..., H\} : \alpha_h > 0\}$ and $\mathbb{H}^0 = \{h \in \{1, ..., H\} : \alpha_h = 0\}$

$\mathbb{X}^+ = \{j \in \{1, ..., p\} : x_j > 0\}$, $\mathbb{X}^- = \{j \in \{1, ..., p\} : x_j < 0\}$ and $\mathbb{X}^0 = \{j \in \{1, ..., p\} : x_j = 0\}$

We have then:

$$Attn_p(x) = \sum_{h \in \mathbb{H}^+ \cup \mathbb{H}^- \cup \mathbb{H}^0} \left(\sum_{j \in \mathbb{X}^+ \cup \mathbb{X}^- \cup \mathbb{X}^0} \mu_j^h(x) + \beta^h(x) \right) \zeta_h$$

$$\begin{aligned} Attn_p(x) = &\sum_{h \in \mathbb{H}^+} \left(\sum_{j \in \mathbb{X}^+} \mu_j^h(x) + \beta^h(x) + \sum_{j \in \mathbb{X}^-} \mu_j^h(x) + \beta^h(x) + \sum_{j \in \mathbb{X}^0} \mu_j^h(x) + \beta^h(x)) \right) \zeta_h \cdot L \\ &+ \sum_{h \in \mathbb{H}^-} \left(\sum_{j \in \mathbb{X}^+} \mu_j^h(x) + \beta^h(x) + \sum_{j \in \mathbb{X}^-} \mu_j^h(x) + \beta^h(x) + \sum_{j \in \mathbb{X}^0} \mu_j^h(x) + \beta^h(x)) \right) \zeta_h \cdot L \\ &+ \sum_{h \in \mathbb{H}^0} \left(\sum_{j \in \mathbb{X}^+} \mu_j^h(x) + \beta^h(x) + \sum_{j \in \mathbb{X}^-} \mu_j^h(x) + \beta^h(x) + \sum_{j \in \mathbb{X}^0} \mu_j^h(x) + \beta^h(x)) \right) \zeta_h \cdot L \end{aligned}$$

When $x \to +\infty$, the first sum $S_1 \to_{x \to +\infty} x \sum_{\mathbb{H}^+} \zeta_h$, the second $S_2 \to_{x \to +\infty} \sum_{\mathbb{H}^-} (\sum_{j \in \mathbb{X}^-} \frac{x_j}{p} + x \sum_{j \in \mathbb{X}^0}) \zeta_h$ and the third sum: $S_3 \to_{x \to +\infty} \sum_{\mathbb{H}^0} (\sum_{j=1}^{p} \frac{x_j}{p+1} + x \sum_{j=1}^{p} \frac{1}{p+1}) \zeta_h$

Finally $Attn(x1, \cdots, x_p, x) \to_{x \to +\infty} Ax + B$

When $x \to -\infty$, the same reasoning shows that the attention function will tend asymptotically towards a linear function too.

Now that we have shown the result for $l = 1$ we assume that it is true for $l \in \mathbb{N}$, let's show that it is true for $l + 1$. To do this, we just need to consider the output of layer $l + 1$ as a function of layer l by using the formula defined below and then apply the same method as for $l = 1$

E.3 Proof for LN:

Proposition 1 *Layer Normalization is responsible for boundary values*

The output after Multi-head attention $Attn(x1, \cdots, x_p, x)$ is passed through the *Add & Norm* to yield $LN(Attn(x1, \cdots, x_p, x) + xW)$, which is equal to :

$$\frac{(Attn(x1, \cdots, x_p, x) + xW) - \text{mean}((Attn(x1, \cdots, x_p, x) + xW))}{\sqrt{Var(Attn_{x_1, \cdots, x_p(x)} + xW)}} \rho + \epsilon \tag{10}$$

We call $\hat{\zeta}_h = \zeta_h - \text{mean}(\zeta_h)$ and $\hat{W} = W - mn(W)$

On the one hand,

$$((Attn_p + xW)) - \text{mean}((Attn_p + xW)) = \sum_{h=1}^{H} \left(\sum_{j=1}^{p} \mu_j^h(x) + \beta^h(x) \right) \hat{\zeta}_h + x\hat{W}$$

On the other hand, $Var(Attn_p + xW) = \frac{1}{d} \sum_{i=1}^{d} [Attn_p + xW)_i - \text{mean}(Attn_{xp} + xW)]^2$

$= \frac{1}{d} \sum_{i=1}^{d} \left[\sum_{h=1}^{H} \left(\sum_{j=1}^{p} (\mu_j^h(x) + \beta^h(x) \right) ((\zeta_h)_i - \text{mean}(\zeta_h)) + x(W_i - \text{mean}(W)) \right]^2$

By using similar reasoning as the previous section, the variance $Var(Attn_{x_1,\cdots,x_p(x)} + xW) \rightarrow_{x\rightarrow\infty} c|x|$. As the nominator tends asymptotically towards a linear function at infinity, the ratio tends towards a constant that we have called boundary values. □

Although the mathematical reasoning proves the presence of a boundary value in the limit, boundary values appear empirically quite quickly.

E.4 Proof for: AL cannot ICL_2 the class of linear functions

Lemma 3 *A 1 layer AL transformer cannot ICL$_2$ the class of linear functions*

Recalling the ICL$_2$ formulation in Equation 5, we pick a very simple target: $f(x) = ax$

To prove this, it is useful to make explicit the equation that the model computes for the query on number inputs. Call $\tilde{x_i} = x_i W$ the linear embedding corresponding to x_i, used in the training. Now, putting together equations 6 and 7, we can write an equation that determines the output of multi-head attention,, $Attn_{x1,\cdots,x_p}(x)$, for the query x, which we abbreviate by $Attn_p$:

$$\sum_{h=1}^{H} \left(\sum_{j=1}^{p} s\left(x x_j (W Q^h {K^h}^T W^T) \right) x_j + s\left(x^2 (W Q^h {K^h}^T W^T) \right) x W V^h \gamma_h \right) \tag{11}$$

Lemma 4 *Given an ICL context $(x_1, \cdots, f(x_p), x)$ and a target $f(x) = ax$ for a 1 layer AL only transformer, Equation 5 simplifies to:*

$$\left\| \left(\sum_{j=1}^{p} x_j \right) \left[(a+1) W_1 V H \cdot L \right] \right\|^2 < \epsilon \left(1 + (a+1) \sum_{j=1}^{p} x_j \right)^2. \tag{12}$$

Which implies that

$$||W_1 V H \cdot L|| < \frac{\epsilon}{\sum_{j=1}^{p} x_j (a+1)} + \epsilon$$

Now pick $x_j \leq \frac{\epsilon}{||W_1 V H \cdot L||.p}$, which entails that $||W_1 V H \cdot L|| < \frac{||W_1 V H \cdot L||}{a+1} + \epsilon$. A suitable choice of a yields a contradiction. □

Theorem 1 *An N layer AL transformer cannot ICL$_2$ the class of linear functions.*

We prove this by induction on the number of layers. Lemma 1 gives the base case where we take large enough x_j and a to ensure that the predictor diverges from a suitable target function for large choices of x_p.

Our inductive hypothesis is that Equation 5 is false for level n by picking a sufficiently high input x and coefficients a. We abbreviate the matrices in the condition in Lemma 4 for level n as $B^n = W_1^n V^n H^n \cdot L^n$. Thus, we assume that for some $c_1^n, ..., c_p^n$ and a,
$|| \left(\sum_{j=1}^{p} c_j^n \right) [((a+1)) B^{n+1}] ||^2 \not< \epsilon(1 + (a+1) \left(\sum_{j=1}^{p} c_j^n \right)))^2$.
We show it also fails for attention level $n+1$. Arguing by contradiction, we assume

$|| \left(\sum_{j=1}^{p} c_j^{n+1}\right) \left[((a+1))\, B^{n+2}\right] ||^2 < \epsilon(1 + (a+1)\left(\sum_{j=1}^{p} c_j^{n+1}\right)))^2$
From the analysis of attention, we have that

$$c_j^{n+1} = AN(c_j^n, B^{n+1}c_j^n)$$

If $||B^{n+1}|| = 0 \ or \ \geq 1$ our result follows from the inductive hypothesis. Now suppose $0 < ||B^{n+1}|| \leq 1$. Given that B^{n+1} is fixed, if $||B^{n+1}||$ is sufficiently small, then for sufficient small values of x_p $||\hat{f}_{x_1,g(x_1),\ldots x_p,g(x_p)}(x)|| \not< \epsilon$. If $0 << ||B^{n+1}|| < 1$, then given Lemma 1, it suffices to take x_p large enough so that Add and Norm at level $n+1$ diverges from the target function. □

Lemma 1 also imposes important constraints on ICL_1 if we remove layer normalization. Eliminating layer normalization removes the boundary value effect, but our model then cannot effectively ICL when the query input x has a large embedding norm. Given Lemma 1, as inputs get large, the model's output asymptotically converges: $\hat{f}^{\theta}(x_1, g(x_1), \ldots, x) \approx_{||x|| \to \infty} ax + b$, where a and b are set by the model's parameters fixed during pretraining and are minimally affected by the in-context examples. Thus, in the absence of layer normalization, the model's predictions become insensitive to the prompt content as $||x||$ increases, thus precluding ICL.

Layer normalization constrains a model's outputs to a bounded range to provide good performance, but it introduces the pathology of boundary value behavior. Normalization may not be the only limitation on transformer ICL_1 performance. The softmax function used in the attention mechanism in Equation 6 also limits in-context learning Naim and Asher [2025].

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: Results can be found in section 3 and to their related parts in the Appendix.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The limitation section is in Appendix A.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: Proofs are detailed in Appendix E.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Empirical details are in section C in the Appendix, in addition the full code is provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: The full code and steps are available in the github link.

   Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Empirical details are given in the appendix C.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Distributions, metrics and all statistical information are given in the paper.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide computational resouces in section C.2 in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Code of ethics are respected.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [NA]

    Justification: The paper is studying architectural limitations of transformers to perform ICL.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: We are generating randomly values from specific distributions as explained in the paper for the training.

    Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [NA]

    Justification: We use open-source code and libraries.

    Guidelines:

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

13. **New assets**

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

    Answer: [NA]

    Justification: We are generating data from random distributions as specified.

    Guidelines:

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

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

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

    Answer: [NA]

Justification: Paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

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

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

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.