# Probing In-Context Learning: Impact of Task Complexity and Model Architecture on Generalization and Efficiency

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

We investigate in-context learning (ICL) through a meticulous experimental framework that systematically varies task complexity and model architecture. Extending beyond the linear regression baseline, we introduce Gaussian kernel regression and nonlinear dynamical system tasks, which emphasize temporal and recursive reasoning. We evaluate four distinct models: a GPT2-style Transformer, a FlashAttention-enhanced Transformer, a convolutional Hyena-based model, and the Mamba state-space model. Each model is trained from scratch on synthetic datasets and assessed for generalization during testing. Our findings highlight that model architecture significantly shapes ICL performance. The standard Transformer demonstrates robust performance across diverse tasks, while Mamba excels in temporally structured dynamics. Hyena effectively captures long-range dependencies but shows higher variance early in training, and FlashAttention offers computational efficiency but is more sensitive in low-data regimes. Further analysis uncovers locality-induced shortcuts in Gaussian kernel tasks, enhanced nonlinear separability through input range scaling, and the critical role of curriculum learning in mastering high-dimensional tasks.

## 1 Introduction

In-context learning (ICL) has emerged as a powerful paradigm in machine learning, enabling models to adapt to new tasks with minimal supervision by leveraging contextual information. Recent studies have framed ICL through the lens of meta-learning, where models learn to approximate functions from a distribution over tasks using only contextual supervision [6]. While foundational work has demonstrated the ability of transformers to internalize simple learning algorithms for tasks like linear regression [9], the scope of these investigations has often been limited to specific architectures and function classes.

This project extends the study of ICL along two critical dimensions: function complexity and model generality. First, we incorporate more complex function families, such as Gaussian kernel regression and nonlinear dynamical systems, which introduce challenges related to smoothness, locality, and temporal dependencies. These function classes push the boundaries of ICL beyond simpler tasks previously explored. Second, we evaluate ICL performance across a diverse set of models: a baseline GPT2-style transformer, a transformer variant with FlashAttention [5], a Hyena-based attention-free model [11], and Mamba, a state-space model with selective recurrence mechanisms [10].

By exploring this expanded landscape, we aim to uncover how architectural choices influence generalization in ICL settings. Our findings will provide insights into the strengths and limitations of

Submitted to CS182 PROJECT DRAFT 2025
Our code and models are available at: `https://github.com/Binwen6/CS182_PROJECT_DRAFT_2025`

different architectures when confronted with increasingly complex learning tasks, ultimately guiding the development of more robust and versatile ICL systems.

## 2 Related Work

The study of in-context learning (ICL) has been significantly shaped by the meta-learning perspective, which views ICL as a process where models learn to approximate functions from a task distribution using contextual supervision. A comprehensive survey by Dong et al. (2022) [6] outlines the definitions, techniques, and applications of ICL, emphasizing its role in enabling few-shot learning without parameter updates.

Foundational work by Garg et al. (2023) [9] established a framework for evaluating ICL using synthetic function families, such as linear regression and Fourier approximation. Their results showed that transformers can effectively internalize simple learning algorithms, but their analysis was constrained to a narrow set of architectures and function classes. Subsequent studies have expanded the scope of ICL to more diverse and complex function families. For instance, Cole et al. (2025) [4] explored ICL in linear dynamical systems, while Bhattamishra et al. (2023) [2] investigated its applicability to Boolean functions. Additionally, Sun et al. (2025) [12] and Cole et al. (2024) [3] applied ICL to nonlinear kernels and elliptic partial differential equations, respectively, highlighting the growing versatility of ICL across domains and underscoring the need to understand how different model architectures perform under these conditions.

Concurrently, the development of architectures capable of handling long sequences more efficiently than traditional transformers has gained traction. FlashAttention, introduced by Dao et al. (2022) [5], addresses the computational bottlenecks of standard attention mechanisms by implementing an IO-aware exact attention algorithm, reducing memory usage and speeding up computations. The Hyena model, proposed by Poli et al. (2023) [11], offers an alternative by replacing attention with subquadratic-time convolutional operations, providing improved efficiency for tasks involving long contexts. Mamba, developed by Gu et al. (2023) [10], employs linear-time sequence modeling with selective state spaces, achieving state-of-the-art performance on various sequence modeling tasks, including language, audio, and genomics.

The increasing diversity of ICL applications and the emergence of novel architectures motivate our work. Earlier studies investigated ICL in decision trees, sparse linear functions, and neural networks [9], while recent efforts have tackled time-dependent dynamics [4], Boolean functions [2], nonlinear kernels [12], and partial differential equations [3]. These developments highlight the importance of evaluating how architectural inductive biases, such as recurrence in Mamba or convolution in Hyena, compare to attention-based mechanisms in complex ICL settings. Our work builds on these advancements by systematically assessing the ICL capabilities of diverse architectures across an extended range of function classes, offering a comprehensive analysis of how architecture design impacts ICL effectiveness in challenging and realistic scenarios.

## 3 Approach

In this section, we formalize the in-context learning (ICL) setup and describe the synthetic function families and model architectures that that we use. Our emphasis is on evaluating how various architectures internalize different function classes.

### 3.1 Problem Setup

We adopt a standard in-context learning (ICL) framework where the model is presented with a prompt $\{(x_i, y_i)\}_{i=1}^T$ of input-output pairs followed by a query input $x_{T+1}$. The model processes the full sequence $[(x_1, y_1), \ldots, (x_T, y_T), x_{T+1}]$ as a single input and is tasked with predicting $y_{T+1}$. No parameter updates occur during inference; the model must generalize in-context from the prompt via forward computation.

This setup follows the meta-learning perspective of ICL, where the model implicitly learns a distribution over tasks and adapts to unseen functions on-the-fly, as discussed in Garg et al. [9].

### 3.2 Function Families

To evaluate ICL generalization, we define a set of synthetic task families $\mathcal{F}$, each representing a distribution over real-valued functions $f : \mathbb{R}^d \to \mathbb{R}$. Each sampled function generates a prompt and query for training and evaluation.

**Linear Regression.** Each task samples a weight vector $w \sim \mathcal{N}(0, I_d)$, normalized to unit norm. The output is generated via:

$$y_i = \langle w, x_i \rangle + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2),$$

where $\sigma$ is a fixed noise level.

**Gaussian Kernel Regression.** We define a radial basis kernel regression task with $C$ centers $\{c_j\}_{j=1}^C$ and weights $\beta \in \mathbb{R}^C$ per task. For each input $x_i$:

$$y_i = \sum_{j=1}^{C} \beta_j \cdot \exp\left(-\frac{\|x_i - c_j\|^2}{2h^2}\right) + \varepsilon_i.$$

Outputs are normalized to unit variance per batch and perturbed with scaled Gaussian noise. This setting introduces smooth nonlinearities and locality-aware structure.

**Nonlinear Dynamical Systems.** Each task defines a recurrence rule $x_{t+1} = F(x_t)$ and output $y_t = \langle v, x_t \rangle + \varepsilon_t$. The nonlinear transition $F$ includes:

- Polynomial:
$$F(x) = Wx + W'x^2 + b \tag{1}$$
- Tanh: $F(x) = \tanh(Wx + b)$
- Duffing Oscillator, VDP, Lorenz: structured chaotic and oscillatory systems

These tasks require the model to track latent states across time, highlighting architectural capacity for recurrence and memory.

### 3.3 Model Architectures

We evaluate four encoder-only architectures with matched parameter budgets:

- **Baseline Transformer:** GPT2-style decoder-only transformer with causal self-attention [13].
- **FlashAttention Transformer:** Variant with FlashAttention kernels [5] for IO-aware optimized attention (See Figure 1).
- **Hyena Transformer:** Replaces self-attention with Hyena operators [11], using convolutional modulation mechanisms (See Figure 2).
- **Mamba:** Selective state space model using implicit continuous-time recurrence [10] (See Figure 3).

All models are trained from scratch and evaluated under the same context-query formulation.

### 3.4 Training Procedure

We train all models using the squared error loss between predicted and target query outputs:

$$\mathcal{L} = \frac{1}{B} \sum_{b=1}^{B} \left(f_\theta(x_{T+1}^{(b)}) - y_{T+1}^{(b)}\right)^2.$$

### 3.5 Curriculum Learning

To improve convergence and stability, we adopt curriculum learning [1, 14, 7]. During training, tasks are sampled from small dimensions, gradually increasing task complexity as training proceeds. This allows faster convergence, especially for difficult function classes like chaotic dynamics.

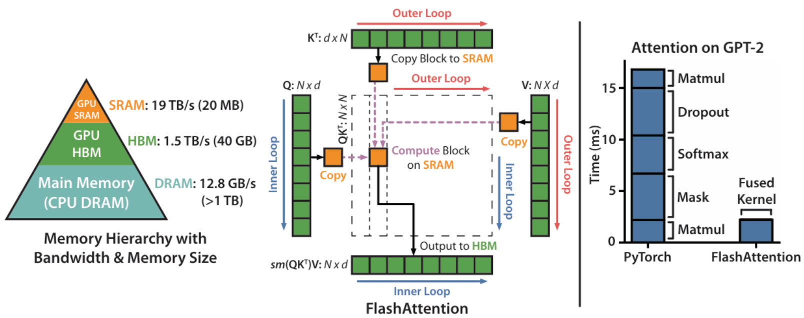



Figure 1: FlashAttention mechanism. The design tiles attention computation to avoid memory bottlenecks, achieving high throughput on modern hardware.

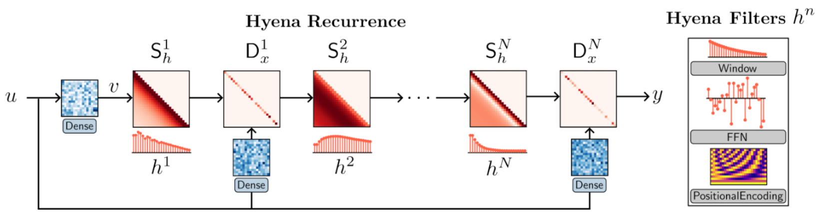



Figure 2: Hyena recurrence. Combines implicit long convolutions with multiplicative gating, allowing attention-like behavior without quadratic cost.

### 3.6 Evaluation Criteria

We evaluate the generalization ability of each model by computing:

- Mean Squared Error (MSE) over test prompts
  The Mean Squared Error (MSE) is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  where $y_i$ is the actual value, $\hat{y}_i$ is the predicted value, and $n$ is the number of observations.
- Generalization to new task instances from each function family
- Robustness under context length variation and input noise
- Scaling behavior as context length $T$ increases

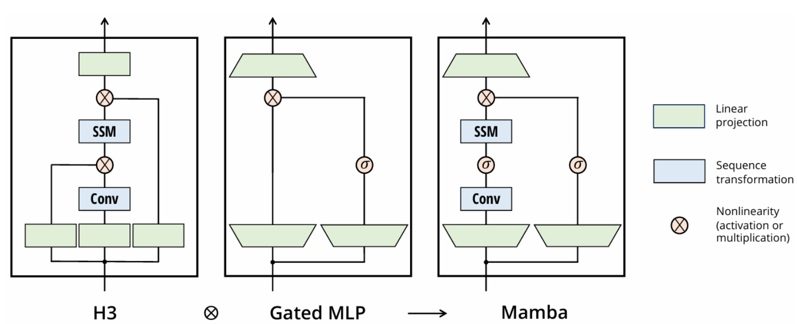



Figure 3: Mamba architecture. Uses state-space sequence modeling (SSM) with gating and convolution to replace self-attention.

Model parameters are not updated at the test time. In all cases, the model must extrapolate in-context based solely on the prompt.

# 4 Experiments

## 4.1 Task and Dataset

Same as [9], we evaluate in-context learning (ICL) capabilities by designing a synthetic experimental framework, where the model is trained to learn functions from a class $F$ via prompt-based adaptation, without updating the model parameters explicitly. The setup is grounded in distributions $D_F$ over functions and $D_X$ over inputs.

We begin by sampling a function $f \sim D_F$, and a sequence of inputs

$$x_1, x_2, \ldots, x_n \sim D_X.$$

These inputs are then stacked to construct the input prompt:

$$P = (x_1, f(x_1), \ldots, x_k, f(x_k), x_{k+1}).$$

Our goal is to let the models predict $f(x_{k+1})$, using only the preceding $k$ input-output pairs. This abstract framework enables us to evaluate and compare different model families on their intrinsic ability to learn in a context without explicit weight updates. In our experiments, we set $k = 20$ unless otherwise specified.

Our experiments focus on two complex function classes: Gaussian kernel regression and nonlinear dynamical systems. For Gaussian kernel regression, functions are defined by a weighted sum of 20 Gaussian kernels with a bandwidth of 1.5, where kernel centers and weights are sampled from a standard normal distribution $\mathcal{N}(0, 1)$, and outputs include additive Gaussian noise with a standard deviation of 0.1. Nonlinear dynamical systems are modeled with polynomial dynamics (up to degree 3), with coefficients drawn from $\mathcal{N}(0, 1)$ and no added noise, emphasizing temporal and recursive dependencies. Inputs for both tasks are sampled from Unif($[-1, 1]^d$), with dimensions tested across $d \in \{1, 10, 50, 100\}$ to explore scalability and high-dimensional challenges. Synthetic datasets are generated on-the-fly with a batch size of 64, ensuring diverse function samples per batch. Random seeds are fixed for training and varied for testing to ensure reproducibility and fair generalization assessment. Models are trained to minimize the mean squared error (MSE), as defined earlier, and evaluated on test prompts with unseen functions and inputs, focusing on the prediction accuracy for $f(x_{k+1})$.

## 4.2 Model Structure

We adopt a decoder-only Transformer architecture inspired by previous work [9] based on the GPT-2 family, consisting of 12 layers, 8 attention heads, and a 256-dimensional embedding space. The model processes a sequence of embedded vectors and predicts the subsequent vector in the same representation space, analogous to language modeling.

Each prompt sequence $(x_1, f(x_1), \ldots, x_k, f(x_k), x_{k+1})$ is embedded into the model's latent space through two learnable linear projections: one for input $x_i$ and one for output $f(x_i)$, the latter being zero-padded to match the input dimensionality. The model predicts the target $f(x_{k+1})$ based on the preceding $k$ in-context examples.

In addition, we experiment with several model variants, either by replacing the standard attention mechanism or changing the whole model architecture. Specifically, we also evaluate:

- **FlashAttention [5]**, an efficient attention implementation that greatly reduces memory consumption and runtime;
- **Hyena [11]**, a convolution-based alternative to attention for long-range dependency modeling;
- **Mamba [10]**, a structured state space model (SSM) proposed as a strong attention-free sequence learner. In our experiments, we configure the Mamba model with 0 attention heads and 24 layers in total.

### 4.3 Model Training

#### 4.3.1 Training Details

We train all models under the same training objective using the squared error loss, following the approach outlined in [9]. All experiments are conducted on NVIDIA RTX 4090 GPUs. Model parameters are initialized from scratch and updated via gradient descent on randomly sampled input batches. The batch size is selected from $\{64, 128\}$, and each model is trained for 50k steps unless otherwise specified.

The training dataset consists of 10k examples, with an additional 1k held out data for evaluation. The learning rate is chosen from $\{1 \times 10^{-4}, 5 \times 10^{-5}\}$ depending on the model type and function family. For certain architectures and function families, we adopt a cosine learning rate schedule with a linear warm-up of 3k steps.

For the nonlinear dynamical system function family, we increase the number of training steps and employ early stopping to ensure stable convergence and better generalization performance.

#### 4.3.2 Curriculum Learning

During training, we apply a curriculum strategy in both the input subspace dimension and prompt length. Specifically, for the linear-like task, we begin by sampling prompt inputs from a 5-dimensional subspace (with other coordinates set to zero), and set the initial prompt length to 11 (corresponding to 11 input-output pairs), while for the nonlinear-like task, according to the setting of Garg et al. (2023) [9], we begin by sampling prompt inputs from a 5-dimensional subspace (with other coordinates set to zero), and set the initial prompt length to 26 (corresponding to 26 input-output pairs). Every 2k training steps, we increase the input subspace dimension by 1 and the prompt length by 2 for the linear-like task, along with 5 for the nonlinear-like task. This gradual increase continues until the dimension reaches 20 and the prompt length reaches 41 for the linear-like task while 101 for the nonlinear-like task.

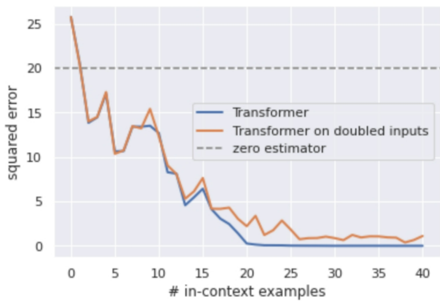
This curriculum significantly accelerates training, especially in higher-dimensional setting where convergence is slow or unstable otherwise.

## 5 Results

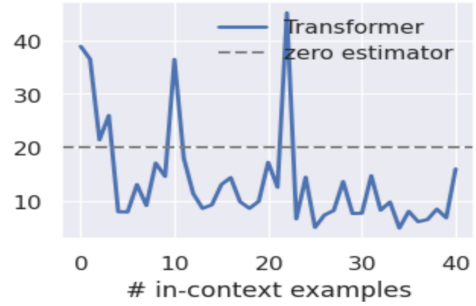
### 5.1 Tasks

**Gaussian Kernel Regression** Based on GPT 2 architecture, the performance of Gaussian kernel regression task(4b) exhibits a fluctuating pattern with an overall mild downward trend, yet lacks clear stability compared with the linear regression task(4a). Despite outperforming the zero estimator on average, the Transformer displays noticeable instability, with several spikes exceeding the baseline error. This suggests that the model's ability to utilize in-context examples effectively is limited in this setting. However, applying doubled inputs(4c) significantly reduces the squared error and yields a more stable performance compared to the standard Transformer. Continuing to grow the amount of in-context examples, the model achieved better results on Gaussian kernel regression, showing a more stable trend of deceasing(4d).

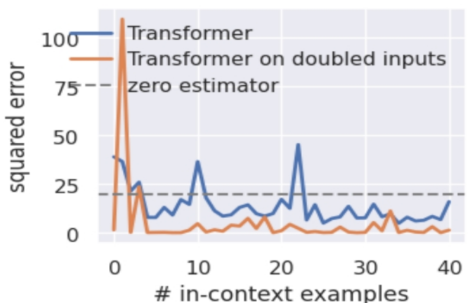
**Nonlinear Dynamics** During the training process of nonlinear dynamical systems, the loss generally increases as the dimensionality, the number of data points, and task difficulty grow. However, within certain intervals, the loss decreases, indicating that the model benefits from gradually increasing complexity. Among the evaluated dynamical systems, they all showed a trend of decreasing.(5) Functions such as *tanh* and *poly* exhibit fast and smooth convergence as the number of in-context examples increases, as they have relatively lower complexity and higher compatibility with in-context learning. In contrast, the *Lorenz* system shows a significantly higher initial error and slower convergence, which is consistent with its known chaotic behavior and intrinsic complexity. The *duffling* system demonstrates a sharp decline in error with only a few examples, highlighting its strong sensitivity to the number of in-context samples. Meanwhile, *logistic* and *vdp* systems present intermediate patterns in both convergence speed and final error, reflecting their moderate learning difficulty.



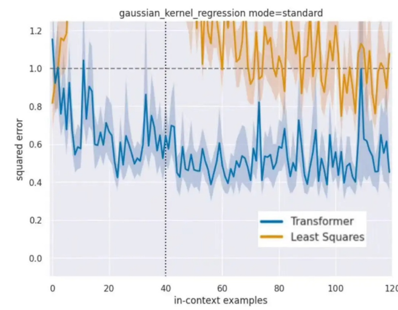

(a) Performance of linear regression tasks with GPT-2.




(b) GPT-2 vs. zero estimator on Gaussian kernel regression.

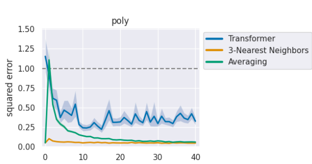



(c) Standard vs. input-doubled GPT-2 on Gaussian regression.

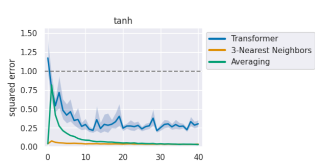



(d) Effect of increasing in-context examples on GPT-2.

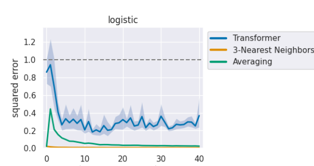
Figure 4: Summary of GPT-2 results on linear and Gaussian kernel regression tasks.



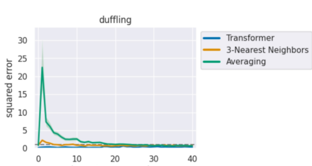

(a) polynomial functions



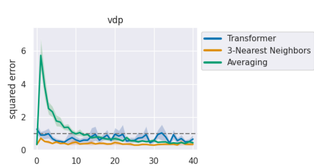

(b) tanh functions



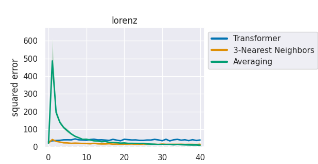

(c) logistic functions




(d) duffling functions




(e) vdp functions




(f) lorenz functions

Figure 5: Results of Nonlinear Dynamics Trained with GPT-2 Architecture

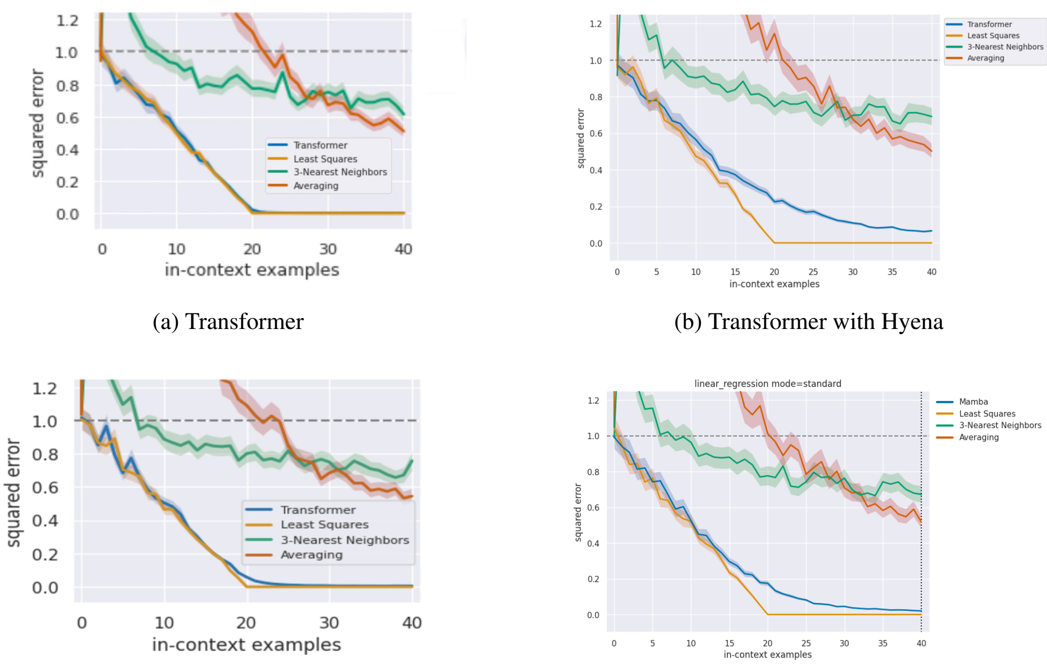
Both tasks are more complex than linear regression, and although their results are less ideal, they still demonstrate that the model has, to some extent, acquired knowledge of these functions through in-context learning.

### 5.2 Architecture

**Hyena** We compare the performance of a standard Transformer baseline and a Transformer augmented with Hyena(6b) filters on the same linear regression task. Although the Hyena-augmented model starts with higher initial error and greater early-stage variability, it exhibits a consistent downward trend and eventually achieves comparable performance. This progression indicates that the model is actively learning from context, not merely memorizing, and that the Hyena filters offer sufficient representational capacity for in-context learning despite their non-attentional nature.

**Flash Attention** Evaluating the GPT 2 model with flash attention on the linear regression task(6c), while the Transformer equipped with Flash Attention achieves results that are generally consistent with the baseline Transformer, its performance is marginally lower. The model performs poorly on Gaussian kernel regression, with an error peak around 20 examples, while it shows lower and decreasing errors on Nonlinear Dynamics.

**Mamba** In the linear regression task, Mamba(6d) shows a consistent reduction in error with more in-context examples, outperforming the zero estimator and approaching the performance of the Transformer. This indicates that the model is not guessing but indeed learning from context. In comparison, the results on Gaussian kernel regression are moderate, better than Least Squares but not very well, while performance on Nonlinear Dynamics is acceptable despite some initial fluctuations.




(a) Transformer

(b) Transformer with Hyena

(c) Transformer with Flash Attention

(d) Mamba

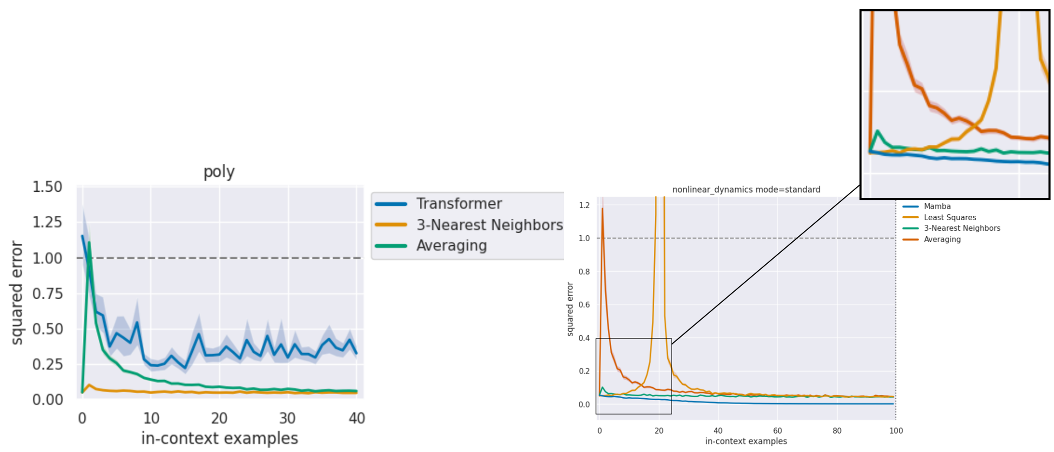
Figure 6: Results of 4 architectures on linear regression task

Among the four architectures, the standard Transformer exhibits the most stable learning behavior, with smooth error reduction and strong final convergence. Mamba shows consistent and reliable performance throughout training, with error curves closely aligned with the Least Squares baseline, albeit with a slower learning rate in the early stages. Hyena demonstrates efficient learning and strong accuracy, though its initial performance can be more sensitive to sample size. Flash Attention achieves rapid convergence as the number of in-context examples increases, but exhibits larger fluctuations in the early phase, especially under limited data conditions.

# 6 Discussion

## 6.1 Architectural Adaptation on Function Properties

Our comparative study across four architectural paradigms—GPT-2-style Transformers, FlashAttention-enhanced Transformers, Hyena, and Mamba—reveals that **model architecture strongly biases performance across different function families** in in-context learning (ICL). Transformer-based models exhibit relatively stable performance across all evaluated tasks, reflecting their general-purpose inductive bias and full-context attention mechanism [13]. However, they are constrained by quadratic scaling in compute and limited context lengths, even with optimizations like FlashAttention [5]. In contrast, **Mamba excels in tasks involving recursive structure and temporal dependencies**, such as nonlinear dynamics(7), achieving strong performance at significantly lower computational cost. This advantage stems from Mamba's structured state-space design [10], which enables efficient sequential reasoning and localized integration of information without full prompt attention. Hyena [11] falls between these extremes, leveraging long-range convolutions, but its hybrid nature may diffuse its inductive alignment with any particular function class. These findings support the view that **architectural alignment with the target function's structure is critical to ICL success**, especially for tasks with algorithmic or dynamical properties.




(a) Transformer (b) Mamba

Figure 7: Comparison between the capability on nonlinear dynamics of Transformer and Mamba

## 6.2 Localization Effect Caused by Gaussian Kernel

Initial experiments on Gaussian kernel regression revealed that **naively applying a bare Gaussian kernel formulation leads to trivial solutions**(4), with models achieving near-zero evaluation error regardless of training. This occurs because such kernels act as local interpolators: when support points are densely clustered, the model can exploit local smoothing to produce accurate outputs without needing to extract or generalize from the structure of in-context examples. To counteract this, we reframed the task by applying a **linear readout layer on top of the Gaussian similarity features**, turning the model's objective into one of learning weighted combinations of localized kernels. While this adjustment made the task more representative and challenging—restoring error curves to expected behavior—it also introduced high **variance across evaluation runs**. We attribute this to the **sensitivity of Gaussian kernels to input distribution geometry**, particularly under small bandwidths or uneven spacing of support points. These results suggest that kernel-based ICL tasks must be carefully framed to balance local smoothness with global compositional reasoning.

## 6.3 Exploitation on Nonlinear Terms for Geometric Separability

In robustness experiments inspired by [8], we evaluated model behavior under doubled input domains. Surprisingly, models often **performed better when the input range was expanded**, especially on nonlinear dynamics tasks. We interpret this phenomenon through the lens of **geometric separability in representation space**(1). When input $x$-values are confined to $[-1, 1]$, higher-order terms such as $x^2$ and $x^3$ exhibit minimal variation, making it difficult for the model to distinguish between support and query points. Doubling the input domain to $[-2, 2]$ amplifies local variation, especially in nonlinear terms, thereby **enhancing representational contrast**. Additionally, when outputs are normalized post-scaling, the transformation effectively injects sharper curvatures and larger gradients into the same output range. These changes make derivative patterns more salient and **easier to detect by local mechanisms** like Mamba's convolutional state updates or attention weights in Transformers. In this sense, input scaling can serve as a form of **implicit feature amplification**, improving sample efficiency and generalization on complex nonlinear functions.

### 6.4 Mechanism Behind Curriculum Alignment

We also identify a deeper structure underlying the curriculum learning strategy proposed by [8]. Their method incrementally increases the input dimension and context length in synchronized stages. Upon analysis, we observe that the **context length scaling ratio differs based on the complexity of the target function class**: for linear regression tasks, the context length grows modestly to $2d + 1$, while for more expressive function families such as decision trees and two-layer neural networks, it expands more aggressively to $5d + 1(4.3.2)$. This scaling ensures that more complex models observe sufficiently rich prompts to recover global structure, without overshooting the optimization budget. We further connect this to **gradient starvation and symmetry breaking in non-curriculum training**: starting with high-dimensional prompts leads to negligible gradient signals due to orthogonality and uniform input influence, causing models to stagnate until a mechanism is discovered. In contrast, curriculum learning offers a warm start in low-dimensional settings, progressively expanding task complexity while preserving training signal strength. This results in **earlier mechanism discovery and faster convergence**, especially for architectures like Mamba and Transformers that rely on stable subspace attention or state transitions.

## 7 Conclusion

In this work, we presented a evaluation framework for studying in-context learning (ICL) behaviors across a diverse set of function families and model architectures. Our experiments demonstrate that the architectural choices can have a rather strong impact on ICL performance, particularly under tasks with recursive or nonlinear temporal dependencies. We find that Mamba, a structured state space model, excels on nonlinear dynamical systems, while Transformers exhibit robust generality. Furthermore, we reveal the subtle phenomena such as the localization bias in Gaussian kernels, implicit feature amplification through input scaling, and convergence benefits from curriculum learning.

There are a few directions that can be explored next. First, we saw that different model architectures behave differently depending on the type of function they're working with. The function types may be broken down more carefully to investigate which models are best suited for which class, which could help us better understand the kinds of problems each model is naturally good at. Since Mamba seems to do well with time-related tasks, a natural step is to try mixing it with Transformers to build a model that handles both long-range and step-by-step reasoning.

Also, we noticed that when we made the input range larger, the models actually learned better, especially for non-linear tasks. This might be because the differences between input values became more noticeable, making it easier for the model to pick up patterns. This can be studied more carefully, with smarter ways coming up to scale or reshape the inputs so that the important features stand out more and learning becomes easier.

Despite these findings, our study has several limitations. First, the evaluation framework primarily focuses on synthetic tasks with well-defined function families, such as polynomial (1) and chaotic systems. While these tasks provide controlled settings to study ICL, they may not fully capture the complexity of real-world applications, where data distributions are often noisier and less structured. Second, the curriculum learning strategy (4.3.2) was tailored to specific dimensional and prompt length progressions, which may not generalize optimally across all model architectures or task types. Finally, our analysis of architectural performance, while comprehensive, is limited by the computational resources available, restricting the scale of models and the breadth of hyperparameter tuning. These constraints suggest caution when extrapolating our findings to larger models or diverse domains, motivating further investigation in the directions outlined above.

## References


[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pages 41–48. ACM, 2009.

[2] Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. Understanding in-context learning in transformers and llms by learning to learn discrete functions, 2023.

[3] Frank Cole, Yulong Lu, Riley O'Neill, and Tianhao Zhang. Provable in-context learning of linear systems and linear elliptic pdes with transformers, 2024.

[4] Frank Cole, Yulong Lu, Tianhao Zhang, and Yuxuan Zhao. In-context learning of linear dynamical systems with transformers: Error bounds and depth-separation, 2025.

[5] Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *arXiv preprint arXiv:2205.14135*, 2022.

[6] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, Baobao Chang, Xu Sun, and Zhifang Sui. A survey on in-context learning, 2022.

[7] Jeffrey L. Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

[8] Ankit Garg, Xueguang Liu, Weihua Hu, Fan Yang, Kevin Zhang, Percy Liang, et al. Can transformers learn in-context reinforcement learning? *arXiv preprint arXiv:2206.02080*, 2022.

[9] Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes, 2023.

[10] Albert Gu, Tri Dao, Daniel Y Fu, Nikita Buehler, Zexue Wang, Yung-Sen Chen, Alexander M Rush, and Christopher Re. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[11] Michael Poli, Alexander Lialin, Rohan Mehta, Xuechen Zhai, Tri Dao, David Luan, Pavel Izmailov, et al. Hyena hierarchy: Towards larger convolutional language models. *arXiv preprint arXiv:2302.10866*, 2023.

[12] Haoyuan Sun, Ali Jadbabaie, and Navid Azizan. In-context learning of polynomial kernel regression in transformers with glu layers, 2025.

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

[14] Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work?, 2020.

# A Self Review

This section provides a self-review of the paper, addressing key aspects of the project, claims, experiments, and overall contributions in a question-and-answer format.

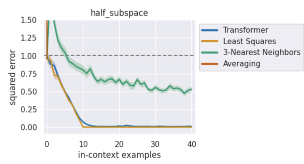
- **What is the main goal of the project?**
  The primary objective of this project is to extend the in-context learning (ICL) setting established in Garg's work by exploring a broader range of function classes and architectural variations. Specifically, we incorporate more complex tasks such as Gaussian kernel regression and nonlinear dynamical systems, and evaluate their performance under different model architectures. This allows for a deeper investigation into the mechanisms and factors that influence ICL behavior.
- **What are the main claims?**
  We make three key claims based on our empirical findings. First, all models consistently achieve the best performance on the linear regression task, suggesting its relative simplicity. Second, model size positively correlates with ICL performance, reinforcing the capacity-driven nature of in-context learning. Third, while architectural differences matter for complex tasks, model performance tends to converge for simpler tasks, indicating that task complexity amplifies architectural distinctions.
- **What are the experiments?**
  Our experimental setup involves replicating Garg's original ICL experiments and extending them by training four distinct model architectures — a standard Transformer, a FlashAttention-enhanced Transformer, a Hyena-based model, and Mamba — on three synthetic datasets. We then evaluate and compare their generalization abilities under a unified testing protocol.
- **What is the evaluation protocol?**
  Models are evaluated primarily using Mean Squared Error (MSE), with fixed random seeds and consistent hyperparameter configurations to ensure fair comparison. We also monitor training variance and convergence speed, especially in tasks with temporal or nonlinear structure, to better understand each model's strengths and weaknesses.
- **What is the data?**
  The datasets consist of three types of synthetic tasks, each containing 10,000 samples: (1) linear regression, (2) Gaussian kernel regression, and (3) nonlinear dynamical systems. All datasets are generated from the same Gaussian-distributed inputs, with training and testing sets independently sampled to ensure no overlap or information leakage.
- **What is the task?**
  The task is to evaluate how different neural network architectures perform in in-context learning scenarios, where models are required to predict the output of a new input solely based on a sequence of input-output examples, without any parameter updates. We evaluate this across three function tasks: linear regression (baseline), Gaussian kernel regression, and nonlinear dynamics.
- **How do the experiments support the goal/claims of the paper?**
  The experiments involve applying each model to all three tasks in a unified in-context learning setup. First, we evaluate all models on the same task, particularly the linear regression task, which serves as a common baseline to assess relative model capacity and generalizability. This comparison supports our first and second claims: that linear regression is the simplest task and that larger models tend to perform better across the board. Second, we compare its performance across tasks of increasing complexity with a fixed model to analyze how task difficulty interacts with architectural properties. This second dimension supports our third claim: that architectural differences become more pronounced as task complexity increases, while models tend to converge in performance on simpler tasks.
- **Are any of the limitations discussed in the paper?**
  Yes. A key limitation is the unstable performance of models on the Gaussian kernel regression task. Although the overall error tends to decrease with more in-context examples, the curve often shows small spikes and fluctuations. We suspect this is due to the localized nature of the Gaussian kernel, which makes model predictions highly sensitive to the distribution of support points.

- **What are the strengths of the paper?**
  The strengths include moving beyond the usual linear regression task to include Gaussian kernel regression and nonlinear dynamical systems, and evaluating four different sequence models under a single and carefully controlled framework. By training all models from scratch on synthetic data points and turning raw performance numbers into concrete insights, the paper serves as a valuable reference for ICL benchmarking and model design.
- **What are the weaknesses of the paper?**
  Firstly, all the data points used in training and testing is synthetic, and the study does not test whether the conclusions can be transferred to real-world sequence problems. Also, the evaluation focus entirely on MSE, without explanation of mechanistic interpretability or the potential downstream use of ICL.
- **Provide a suggestion for improving the paper.**
  A suggestion for improvement is to repeat every experiment with 5-10 random seeds, and report the 95% confidence interval for each model architecture-task pair, which could help decide whether certain dips are systematic or just due to luck.
- **What is the relevant related work?**
  The relevant related work includes Garg et al(2023) which gives the Foundational ICL benchmarks and theory. Also for the architectural innovations, we mainly refer to FlashAttention (DAO 2022) which reduces the quadratic memory bottleneck, Hyena (Poli et al., 2023) which replaces attention with filtered convolutions, and Mamba (Gu et al., 2023) that uses selective state-space recurrence.
- **Is the paper reproducible?**
  The paper is reproducible because the full training and testing codes are available on public GitHub repositories, and we have random seed settings to make sure the results can be rerun and regenerated.
- **Can you rerun the experiments?**
  Yes, the experiments can be rerun, as the datasets are synthetically generated using the provided codebase, and the computational resources required are accessible with standard GPU hardware. All model configurations and training settings are reproducible under the same environment.
- **Can you reproduce the results in the paper?**
  The results are reproducible, as we have saved the pretrained model in the file. By using the same parameter settings and following the same procedures, one should be able to replicate our results.
- **Are all the plots in the paper clearly interpretable with well-defined and explained axes, with the methodology clearly explained in the paper text?**
  Yes, the plots are clearly interpretable, as the axes are properly labeled and the methodologies are clearly presented. However, it is still necessary to standardize the scales and criteria across the graphs to enable meaningful comparison.
- **Is the English in the paper correct and clear?**
  The English used in the paper is clear and grammatically correct, as we have carefully reviewed and polished the writing. We also compared our manuscript with relevant reference papers to ensure clarity and consistency in presentation.
- **Do you have any feedback on any TODOs that the authors have left at this stage?**
  The paper includes TODOs such as conducting experiments with alternative parameter settings to improve performance, as well as running parallel experiments. We also plan to revise and standardize some of the figures to ensure proper comparability and to perform additional experiments to draw more comprehensive conclusions.

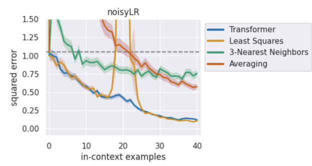
## B OOD: Out-of-Distribution Experiments

This section of appendix is a supplement to the result of out-of-distribution experiments with abundant visualization.

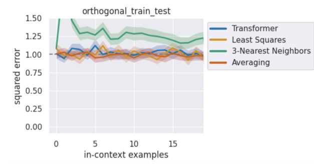
## B.1 Transformer with Flash Attention



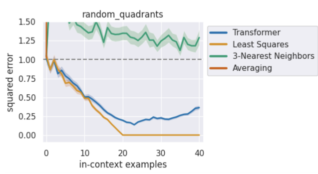

(a) Half-subspace




(b) NoisyLR




(c) Orthogonal Sampling



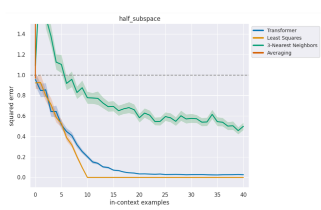

(d) Random Quadrants


(e) Scaled Distribution

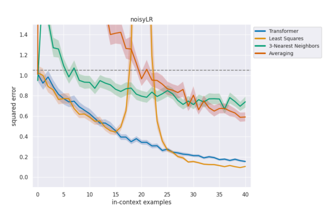

(f) Skewed Distribution

Figure 8: Results of OOD Sampling on GPT2 with Flash Attention

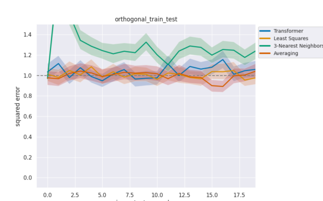
## B.2 Hyena



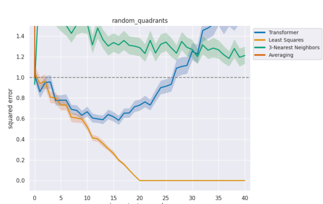

(a) Half-subspace



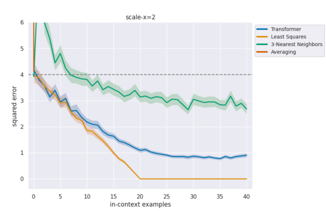

(b) NoisyLR




(c) Orthogonal Sampling




(d) Random Quadrants

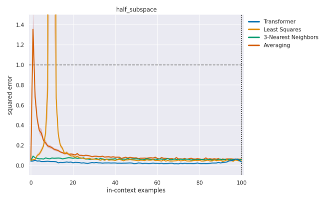



(e) Scaled Distribution

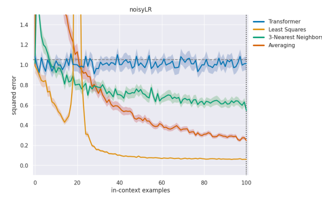

(f) Skewed Distribution

Figure 9: Results of OOD Sampling on Hyena

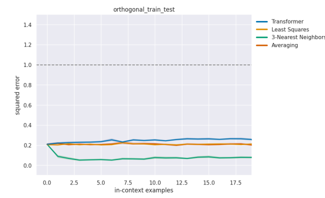
## B.3 Mamba



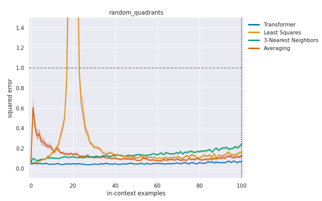
(a) Half-subspace



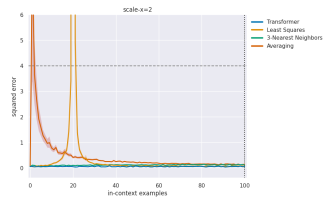
(b) NoisyLR



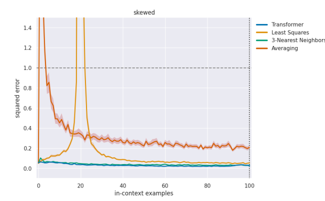
(c) Orthogonal Sampling



(d) Random Quadrants

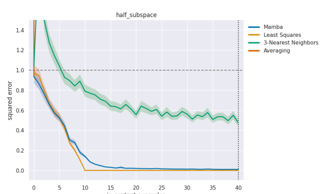


(e) Scaled Distribution

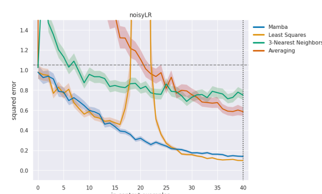


(f) Skewed Distribution

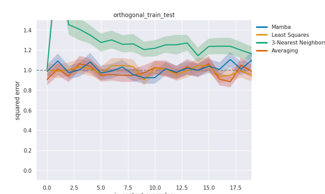
Figure 10: Results of OOD Sampling on Mamba (Gaussian Kernel Regression)



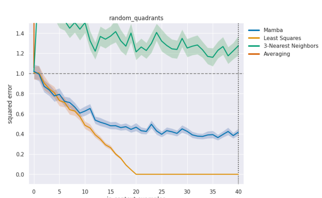
(a) Half-subspace



(b) NoisyLR

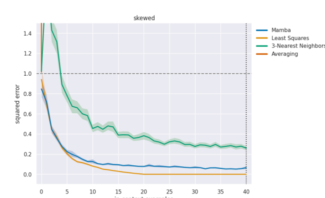


(c) Orthogonal Sampling



(d) Random Quadrants

(e) Scaled Distribution



(f) Skewed Distribution

Figure 11: Results of OOD Sampling on Mamba (Nonlinear Dynamics)