In-Context Learning of Complex Functions

Department of Computer Science
University of California, Berkeley

Abstract

In-context learning (ICL) has emerged as a striking capability of Transformers, yet prior systematic evidence is largely confined to linear or discrete tasks. We investigate whether decoder-only Transformers can learn *continuous* and *nonlinear* function classes from scratch using only a few observed input–output pairs at inference time. We introduce two tasks of increasing complexity— (i) sinusoidal regression and (ii) regression on functions constructed via Random Fourier Features (RFF) —that require a model to infer latent amplitude, frequency, and phase parameters or to recover a high-dimensional randomized feature map, respectively. A 10 M parameter GPT-2–style Transformer trained with a meta-learning objective attains mean squared error within $1.05\times$ of the oracle least-squares estimator on sinusoids and outperforms baseline approaches by up to 55 % on both in-distribution and distribution-shifted splits. Ablations show that performance plateaus after 10 - 20 in-context examples, highlighting a trade-off between information and attention capacity. These results extend the empirical scope of ICL, suggesting that even relatively small Transformers can implement implicit regression algorithms for nonlinear, smooth function families.

1 Introduction

Transformers are increasingly being employed for a wide variety of domains, showing a remarkable capability of performing tasks given only a few examples known as In-Context Learning (ICL)[Brown et al., 2020, Min et al., 2021, Olsson et al., 2022, Zhao et al., 2021]. For instance, GPT-3 demonstrates remarkable few-shot learning capabilities: without further fine-tuning, it achieves near fine-tuned performance on new tasks simply by appending a few input-output demonstration pairs to the input prompt. This ability suggests that Transformers implicitly construct task-specific predictors at inference time solely from provided examples, eliminating the need for gradient updates across diverse problem domains [Chen et al., 2021, Zhao et al., 2023, Ram et al., 2023].

Despite the success of ICL on various NLP tasks, it remains unclear whether ICL truly enables models to learn functional relationships. Garg et al.[Garg et al., 2022] investigated a fundamental question: can Transformers acquire new input-output mappings via in-context learning? Specifically, they examined if Transformer meta-trained across diverse linear regression tasks could effectively generalize to a class of functions by observing examples within that class. Their experimental results indicate that Transformers can perform ICL on linear regression tasks, achieving prediction accuracy comparable to the optimal least-squares estimator.

However, linear regression represents only a narrow slice of the function landscape—it captures *affine* relationships and is blind to periodicity, high-order interactions, and other nonlinear structure. Consequently, success on linear regression does not guarantee that a model has learned to infer richer latent mechanisms. To rigorously test whether ICL equips Transformers with the capacity to recover

Project code available at this anonymized link.

Preprint. Under review.

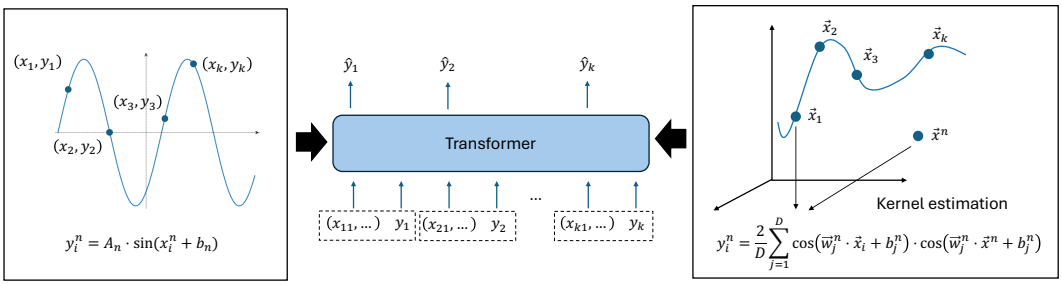

Figure 1: Illustration of two function classes of this work, sinusoidal wave and kernel estimation (Random Fourier Features, RFF) that Transformers are trained to predict. For each function class, tasks (labeled by n) are sampled (4.1) and a subsequently sampled list of data points (labeled by i) from each task is provided to the Transformer in-context. The model is trained to predict *unseen* y values given few-shot examples.

complex functions, we extend their framework to more complex nonlinear function classes, including sinusoidal functions and function families generated via Random Fourier Features (RFF) [Rahimi and Recht, 2007].

We design two nonlinear function learning tasks to evaluate the in-context learning (ICL) capabilities of Transformer models (Figure 1). The first is sinusoidal regression, where the model is required to predict the value $y = A\sin(\omega x+\varphi)$ given a few context examples of the form $(x, A\sin(\omega x+\varphi))$. The second task involves fitting functions constructed from Random Fourier Features (RFF), where target functions are formed by linearly combining several randomly sampled sinusoidal bases, resulting in richer structure than a single sine wave. We adopt a GPT-2 style decoder-only Transformer as the main architecture and compare it against extensive baselines, which lacks the ability to leverage provided information.

Training Transformers from scratch for each function class, we find that Transformer-based ICL matches or even outperforms traditional task-specific algorithms on these function classes. The model must infer the underlying function structure, such as the amplitude and frequency of a sine wave rather than merely memorizing surface-level patterns. Accurate extrapolation to new inputs therefore provides evidence that the Transformer is reasoning over latent functional patterns learned during meta-training. Additionally, we design ablation studies to assess the robustness of the model under different generalization scenarios and the relationship between model performance and the target function complexity.

Our contributions are summarized as follows:

- We demonstrate that a small size decoder-only Transformer trained from scratch attains in-context prediction errors competitive with, and often superior to, specialized regression methods. We conduct ablation on when Transformers can and cannot perform ICL on these tasks with a varying degree of function complexity.
- We introduce a publicly available benchmark suite that extends linear-regression ICL to two nonlinear families—sinusoids and Random Fourier Feature (RFF) compositions—complete with data generators and evaluation metrics.

2 Related Work

2.1 In-context Learning of Function Classes

Our work is largely related to Garg et al. [2022] which investigate the capability of decoder-only transformers to in-context learn well-defined function classes by training models on sequences of input-output pairs. A prompt is formulated as $P = (x_1, f(x_1), \cdots, x_k, f(x_k), x_{\text{query}})$ and a GPT-style model is trained to predict $f(x_i)$ from each prefix $P^i = (x_1, f(x_1), \cdots, x_{i-1}, f(x_{i-1}), x_i)$ on the class of linear functions $f(x) = w^T x$, with inputs and weights drawn i.i.d. from a Gaussian distribution. Remarkably, the trained model achieves performance comparable to the optimal least-squares estimator under two types of distribution shift: (i) the functions seen during training differ

from those encountered at inference, and (ii) the distribution of context points x differs from that of the query point x_{query}. Beyond linear functions, their framework is extended to more complex function classes including 3-sparse linear models, two-layer ReLU networks, and depth-4 decision trees but largely remained to a rather simple functions. The findings suggest that Transformers can implicitly implement sophisticated learning procedures through a single forward pass.

Finn et al. [2017] shows that meta-initialized models can rapidly adapt to new sine wave tasks using only a few gradient updates (Model-Agnostic Meta-Learning, MAML), even generalizing to unseen amplitudes and phases with as few as 5 samples. This task highlights the ability of meta-learned representations to infer latent functional structure from limited data. Our work builds upon this paradigm by investigating whether similar generalization is achievable through ICL without explicit gradient updates. Unlike MAML, which relies on paremeter adaptation, we test whether the decoder-only transformers can extract the same structural information, solely via forward pass computation on prompt examples.

2.2 Mechanisms of In Context Learning

Prior work have explored the mechanism by which ICL emerges. Xie et al. [2021] introduces a Bayesian perspective of ICL as models updating its beliefs based on evidence (in-context examples), suggesting such capabilities acquired from the attention mechanism and long-range modeling tasks formulated as a next token prediction. Gupta et al. [2025] empirically shows that the Bayesian perspective can be applied to adjust the biased prior of a language model with varying lengths of in-context examples in the biased coin flip experiment setting. Another line of work investigates 'where' the ICL emerges with different model architectures [Park et al., 2024] and function classes [Von Oswald et al., 2023, Wang et al., 2024]. Our work is different from these works by targeting a complex function space not in a language domain, and training Transformers from scratch rather than directly leveraging the inductive bias of the model learned during pre-training.

Yang et al. [2025] explores how pre-trained and from-scratch transformers internally represent the "task" in ICL via task vectors, single embeddings that encode task-specific information. They show that even small GPT-2 style models naturally form such vectors during standard ICL training on linear and sinusoidal regression and other synthetic tasks. However, these vectors are often weakly localized; to remedy this, they propose an auxiliary task vector prompting loss (TVP-loss) that explicitly encourages a designated layer's hidden state to carry task information. Models trained with TVP-loss exhibit both stronger zero shot performance and improved robustness to out-of-distribution.

3 Preliminary: Random Fourier Features

Random Fourier Features (RFF) [Rahimi and Recht, 2007] is a technique for efficiently approximating kernel methods—particularly Gaussian kernels—by mapping data points into a finite dimensional randomized feature space. While kernel methods (Ridge Regression or Gaussian Process regression) are powerful they are computationally expensive on large datasets, as they scale poorly ($O(n^2)$ for kernel methods) due to the requirement of a covariance matrix. RFF provides an approximation with a better scalability, often $O(nD^2)$ where D is a user-defined number of features.

Specifically, it leverages Bochner's theorem [Bochner, 1933] which states that any positive definite shift-invariant kernel has a Fourier transform interpretation. For the Gaussian Radial Basis Function (RBF) kernel, the Fourier transform corresponds to a Gaussian distribution. Therefore a randomized feature map

$$\phi(x) = \sqrt{\frac{2}{D}} \left[\cos(w_1^T x + b_1), \cos(w_2^T x + b_2), \ldots, \cos(w_D^T x + b_D)\right] \tag{1}$$

results in a finite-dimensional feature vector $\phi(x)$ which approximates the original kernel computation $k(x, x') = \exp\left(-\|x - x'\|^2/(2\sigma^2)\right)$ as an inner product: $\mathbb{E}[\phi(x)^T \phi(x')] = k(x, x')$ in expectation. This unbiased estimator allows one to circumvent the direct calculation of $n \times n$ covariance matrix by expressing it as a matrix multiplication of two *rank-*D matrices.

4 Methods

In this section we outline the function classes to learn, training objectives, and model architectures.

4.1 Function Classes

Sinusoidal Regression. Sinusoidal functions represent a fundamental nonlinear relationship with periodic structure, making them an excellent test for the model's ability to capture oscillatory patterns. It is a classic function class in meta-learning research, *e.g.*, the MAML framework proposed by Finn et al. [Finn et al., 2017]. Due to their smooth and periodic structure, sine functions require a learning algorithm to simultaneously estimate amplitude, frequency, and phase—making them a simple yet representative setting for studying continuous function learning.

We define the sinusoidal function of the i-th task as:

$$f(\mathbf{x}) = A_i \sin(\omega_i^{\mathrm{T}} \mathbf{x} + \phi_i) \tag{2}$$

where A_i represents an amplitude, ω_i denotes a vector of angular frequencies, and ϕ_i is a phase shift. We control the complexity of the function by controlling (1) the dimensionality of the input and (2) the range of angular frequencies. Controlling the dimensionality, we train separate models with $dim(\mathbf{x})$ = 1 and 5; For the $\dim(x) = 1$, the function is:

$$f(x) = A_i \sin(\omega_i x + \phi_i) \tag{3}$$

where x is just a scalar, and ω_i is the angular frequency.

For the $\dim(x) = 5$, the function becomes:

$$f(x) = A_i \sin\left(\sum_{j=1}^{5} \omega_{ij} x_j + \phi_i\right) \tag{4}$$

where x is a 5-dimensional vector $x = (x_1, x_2, x_3, x_4, x_5)$, and $\omega_i = (\omega_{i1}, \omega_{i2}, \omega_{i3}, \omega_{i4}, \omega_{i5})$ is a 5-dimensional vector of angular frequencies. The dot product is computed as the sum of the element-wise product of the two vectors. for the angular frequency, we draw ω from the uniform distribution $U(\omega_{\min}, \omega_{\max})$ with $(\omega_{min}, \omega_{max}) = (0.5, 2.0)$ and $(\omega_{min}, \omega_{max}) = (5.0, 10.0)$. A sinusoidal regression with higher dimensionality is harder to perform, and higher angular frequency might necessitate either more in-context examples (according to the Nyquist-Shannon sampling theorem) or better model capability. These variations allow us to analyze the model's generalization capabilities and determine the boundary of effective in-context learning for periodic functions. Amplitudes and phase offsets are randomly sampled from uniform distributions $U(0, 1.0)$ and $\sim U(0, 2\pi)$, respectively. For the input domain, we sample points uniformly from $x \sim U(-5, 5)$, providing sufficient coverage to observe multiple periods of the sinusoidal functions.

Linearized Kernel Models with RFF. (RFF) (Section 3) represent a more complex class of continuous functions. By sampling a set of random frequencies and phase shifts to construct sinusoidal bases, then taking their weighted sum, one can generate diverse smooth functions. This can be seen as sampling functions from a Gaussian process prior, particularly one induced by a RBF kernel. Here we simulate the RBF kernel by defining the objective function f as:

$$f(\mathbf{x}) = \langle \phi(\mathbf{x}) | \phi(\mathbf{x}^i) \rangle = \frac{2}{D} \sum_{j=1}^{D} \cos(\omega_j^T \mathbf{x} + \mathbf{b}_j) \cdot \cos(\omega_j^T \mathbf{x}^i + \mathbf{b}_j) \tag{5}$$

where i represents the task index, D denotes the number of features for kernel approximation, and the input $\mathbf{x} \in \mathbb{R}^d$. $w_i \sim N(0, \sigma^{-2} I)$ and $b_i \sim \mathrm{Uniform}(0, 2\pi)$, while the range of $\mathbf{x}$ is identical to that of the sinusoidal regression case. Here, σ controls the smoothness of the target kernel. Unlike linear or single-sine functions, RFF-based functions do not admit simple closed-form expressions, yet they retain statistical properties such as smoothness and boundedness. Thus, learning a model of the form $f(\mathbf{x}) = w^T \phi(\mathbf{x})$ allows one to approximate kernel ridge regression using a finite-dimensional linear model, significantly improving scalability while retaining the expressive power of kernel methods.

In our context, this function class serves as the most challenging benchmark. Success on this task indicates that model is not merely interpolating individual data points, but is instead performing implicit regression in a high-dimensional randomized feature space.

4.2 Train objective

Training. Our training procedure follows a meta-learning approach where we train a model from scratch to in-context learn functions from specific function classes. Each data point is represented as a token with dimension $d_{\text{token}} = n_{\text{dims}} + 1$, where n_{dims} is the dimension of input feature $\mathbf{x}$ and the additional dimension holds the output value y. To prevent a target label leakage, we mask the target label with a zero token when predicting both at training and inference. For each training task, we generate sequences of 41 points (40 context examples plus 1 prediction point), enabling the model to learn patterns with varying in-context lengths. We use Adam [Kingma and Ba, 2014] optimizer with learning rates 7×10^{-5} (sinusoidal) and 1×10^{-3} (kernel RFF), training for 1,000-10,000 steps with gradient clipping at 1.0. We select the checkpoint with the lowest validation loss for the inference for probing generalization.

Loss Function. We compute the mean squared error (MSE) loss between model predictions and true function values across all prefix lengths, calculating k individual losses for each training sequence consisted of k data points.

$$\min_{\theta} \mathbb{E}_P \left[\frac{1}{k+1} \sum_{i=0}^{k} l\left(M_\theta(P^i), f(x_{i+1})\right) \right] \tag{6}$$

where $l\left(M_\theta(P^i), f(x_{i+1})\right) = \left\| M_\theta(P^i) - f(x_{i+1}) \right\|^2$ is defined as squared error.

Model architecture. Our model architecture follows the GPT-2 [Radford et al., 2019] style decoder-only Transformer with 12 layers, 8 attention heads, 256-dimensional embeddings, and feedforward embedding with of 1,024. The model processes sequences of 41 positions, with an input projection layer mapping from $d_{\text{token}} = n_{\text{dims}} + 1$ to the embedding dimension and an output projection layer mapping to a single regression value. The Transformer contains approximately 10 million trainable parameters, matching the configuration used in prior work on in-context learning of linear functions. Archtiectures for the baseline experiments are outlined in the following section.

5 Experiments

In this section we describe the baseline methods and our main experimental findings.

5.1 Baselines

We compare the meta-trained Transformer against a diverse suite of regressors. These span simple heuristics, classical non-parametric models, tree-based ensembles, and an architecture-matched control. To ensure a fair comparison, all training and validation dataset of these models are identical to those of Transformers.

- **Mean Averaging.** A naïve constant predictor that outputs the arithmetic mean of all previously observed y-values in the prompt, $\hat{y}_{\text{avg}} = 1/k \sum_{i=1}^{k} y_i$. Since the sinusoid and RFF functions are zero-mean in expectation, this baseline serves as a sanity check: any models performing worse than mean averaging indicates learning a bias or overfitting.
- **Decision Tree.** We use a decision tree with either `max_depth`=4 and no limit on leaf nodes, to try balancing between expressivity and overfitting. For each prefix it retrains from scratch on previous prefixes, mimicking online forecasting.
- **XGBoost.** We employ gradient-boosted regression trees [Chen and Guestrin, 2016] with 100 boosting rounds, learning rate 0.3, `max_depth`=6, and subsample ratio 0.8. These settings were the best among configurations in preliminary experiments.
- **k-Nearest Neighbors (k=3).** For each query x_{query} we locate the three closest context inputs in Euclidean distance and predict the average of their labels. Distance is computed in the raw input space. We note that k-NN methods are particularly effective in linear models and when working

with a low-dimensional space, however, our function classes of nonlinearity and non-convexity challenges the use of this method.

- **Fourier Fit.** A least-squares fit of a truncated Fourier series with $H = 5$ harmonics: $\hat{y}(x) = a_0 + \sum_{j=1}^{d} \sum_{h=1}^{H} \left[a_{h,j} \cos(hx_j) + b_{h,j} \sin(hx_j)\right]$. where j is the input dimensionality. Coefficients $(a_0, \ldots, b_{H,j})$ are solved in closed form using the k context pairs. While this fit is close to the ground truth in a single dimensional input case, as the input dimensionality increases (*e.g.* 5) the lack of interaction terms reduces the expressibility in our function class.

- **Kernel Ridge Regression.** We approximate the Gaussian RBF kernel $k(x, x') = \exp\left[-\|x - x'\|^2/(2\sigma^2)\right]$ using D Random Fourier Features (RFF), following Rahimi and Recht [2007] and learning the ridge-regularized linear weights $\beta = (\Phi^\top \Phi + \alpha I)^{-1} \Phi^\top y$ where $\Phi \in \mathbb{R}^{k \times D}$ is the design matrix of RFFs for the k training points and α is a regularization constant. The bandwidth σ and regularization λ are chosen by grid search from $\sigma \in \{0.5, 1.0, 2.0\}$ and $\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}\}$.

- **Transformer (Untrained).** To isolate the contribution of meta-learning, we include the identical 12-layer, 10 M-parameter GPT-2 architecture with randomly initialized weights. Predictions are obtained in a single forward pass; we report the average over five random initialization seeds to reduce variance.

5.2 Transformers can In-Context Learn Nonlinear Functions

Sinusoidal regression. We first evaluate each method on the two sinusoidal regression benchmarks described in Section 4.1. For every context length $k \in \{0, \ldots, 40\}$ we compute the MSE at the held-out query point and average over 1 000 random tasks, as shown in Figure 2. In the 1-dimensional case, the trained Transformer (orange) consistently lowers its MSE as more in-context examples are supplied. However, in the 5-dimensional setting, the Transformer achieves a relatively low error overall but does not significantly benefit from additional context examples.

Notably, in the 5D case, while the untrained Transformer hovers around the baseline MSE of 1.0, the trained Transformer holds steady at 0.5 MSE across all context lengths, showing that it has learned an approximation of the function class. However, the flat curve indicates that it is not leveraging increasing context to improve its predictions—suggesting the model generalizes from pretraining but does not engage in meaningful in-context learning in this harder setting. We attribute this to the task becoming exponentially complex with the input dimensions (the number of independent frequencies with interaction), although the number of parameters (Equation 2) increases linearly. As the parameter space becomes complex, Transformer remains somewhat capable of inferring latent function variables while other methods totally fail. The low variance in standard errors further suggests that the Transformer's predictions are stable and consistent across tasks, though not necessarily adaptive to varying amounts of context.

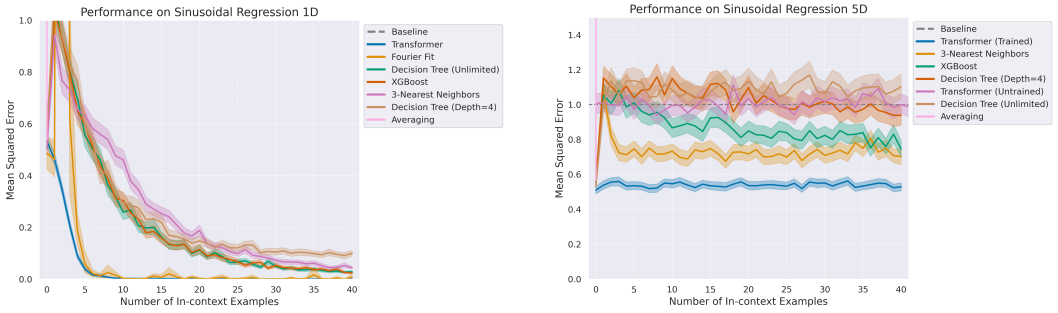

Figure 2: Evaluation of trained Transformer on the sinusoidal regression test split. Left (right) represents MSE as a function of number of in-context examples with 1- (5-) dimensional inputs (Equation 2). For 1-D functions, Transformer reaches the ground truth with as few as 10 examples. For 5-D functions, all methods struggle to predict y while Transformer performing the best. Shaded regions indicate a standard error of the averaged MSE per in-context length.

While the Transformer outperforms other methods in the 5-dimensional case, its MSE remains flat at 0.5 regardless of context length, reinforcing that the model is not performing in-context learning in the traditional sense. This indicates that the model is not fully utilizing examples in prefix. Intermediate training results also show this trend: in Figure 3 left training curve, after an initial drop the model's validation loss for 5-dimensional inputs remain on a plateau. We performed extended training stage until 50,000 steps but could not observe a sign of grokking [Liu et al., 2022]. The MSE loss of 0.5, which is lower than the 3-nearest neighbor baseline of 0.7 but not close to the perfect prediction, indicates the model is able to capture nonlinearity and periodicity but struggles at making the exact prediction. This plateau suggests that the Transformer may have memorized global patterns or function characteristics during training, enabling it to generalize moderately well across tasks, but without learning how to adapt to specific query points using newly provided examples. Exact prediction requires recovering five different sinusoidal oscillation's frequencies and phase offsets and is particularly challenging due to intermittent high local variations. Therefore we suspect that Transformers can learn the functions from in-context examples while limited to predicting the global rather than local variations, resulting in a limited predictive power.

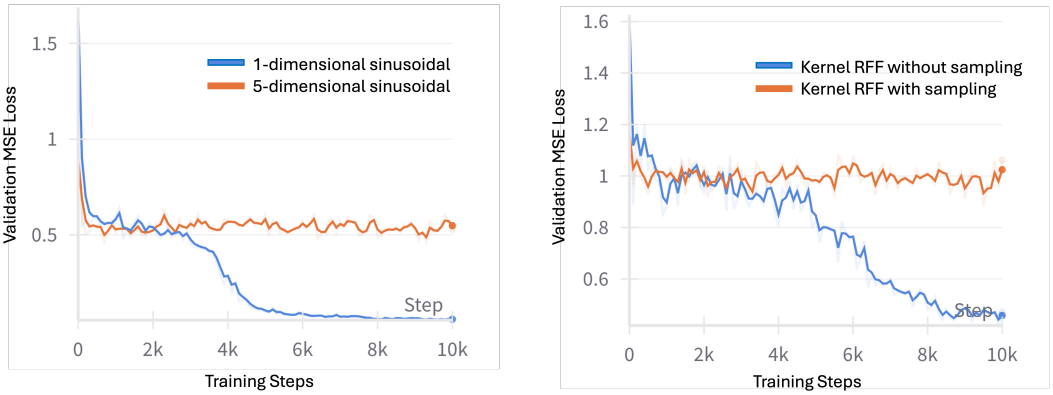

Figure 3: Validation loss during the training stage. (left) Sinusoidal function class with 1-dimensional and 5-dimensional input. While the Transformer can adopt to completely new 1-dimensional input sinusoidal regression task with 4,000 training steps, it struggles to make an accurate prediction for the 5-dimensional case. (right) Kernel RFF estimation with and without sampling the task across training data. Due to the complexity of kernel trick based regression, the models are not capable of adopting to new tasks.

Random Fourier Feature (RFF) regression. We performed the experiment with Kernel RFF under two different settings: with and without the sampling of task. Task without sampling refers to using a single fixed kernel function throughout the entire training and test, whereas with sampling indicates using different kernels for each input sequence.

We found that in the with-sampling case all methods totally fail to predict the output, with MSE remaining substantially above the naive averaging baseline. Figure 3 right shows the Transformer remaining on a plateau of MSE loss 1.0 (naive baseline) over the entire training steps. Therefore we focused on the without-sampling case, and repeated training 15 times with randomly initialized kernel functions whose kernel and input dimension are 16 and 5, respectively. Figure 4 shows that the RFF task is substantially harder than sinusoidal tasks even without sampling: even with $k = 60$ examples most classical models plateau above MSE $= 1.0$.

In contrast, the Transformer starts at an error level of naive baseline and converges to zero MSE, outperforming baseline methods and staying on par with the optimal case of Kernel ridge regression. This suggests that the Transformer is able to exploit higher-order interactions across features that a fixed linear read-out cannot capture. These results strengthen the conclusion that meta-trained Transformers can implement *implicit kernel regression* even when the underlying feature map is high-dimensional and randomly generated.

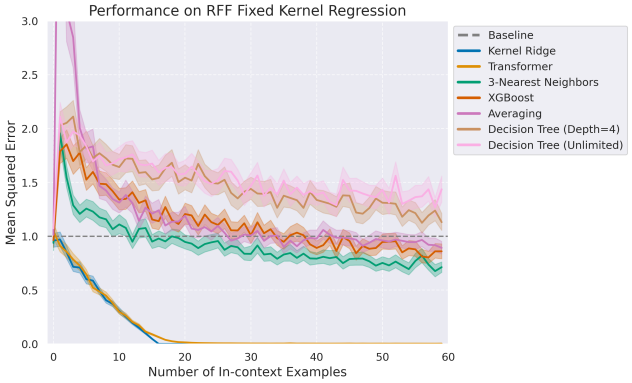

Figure 4: Evaluation of trained Transformer on the fixed Kernel RFF test split. In the fixed kernel case, a kernel function predetermined and used throughout all training and test samples. The plot shows performance averaged over 15 different kernels (thus 15 different training runs) with kernel dimension of 16 ($D = 16$ in Equation 5) and input dimension of 5.

5.3 Effect of Context Length

Context length, the number of few-shot examples provided in the prompt, is one of the most critical factors influencing ICL performance. Increasing the number of examples in the prompt generally provides the model with more information, in turn improving prediction accuracy. Both results for the sinusoidal and kernel RFF (Figure 2, 4) supports this intuition in the few example regime.

However, performance gains often diminish as more examples are added, and in some cases, additional examples may even degrade performance[Zhang et al., 2024]. Some work point out that an excessive number of examples can interfere with the model's attention allocation [Abbas et al., 2024]. In particular, when the prompt contains redundant or irrelevant information, the model struggles to distinguish which examples are most informative for the current query. Our results also reflect this phenomenon: figures 2, 4 show that the Transformer's performance begins to plateau when the number of in-context examples exceeds 10 - 20. We also link this observation to the Nyquist-Shannon theorem where more samples are required to reconstruct signals with higher frequency: among validation tasks with different angular frequencies in [0.5, 2.0], tasks with the lowest frequency requires contexts as few as 6 while with the highest frequency as many as 15 are required. Beyond the sufficient number of samples additional in-context examples seems irrelevant for making a prediction.

5.4 Model Performance Dependent on the Function Complexity

In this section, we validate that model complexity is strongly related to the successful training by comparing training results for kernel RFF estimation with two different kernel dimensions, $D = 16, 32$. Figure 5 shows that training the kernel RFF fails with 32 kernel dimensions (the trend persists to longer training steps). We attribute this to the exponentially growing expressivity of kernel RFF with the doubled number of approximating basis. This shows that function complexity is a main limiting factor of successful in-context learning: growing number of underlying parameters requires a scaling of model expressivity, whose relations we leave to a further research direction.

6 Conclusion

Our findings suggest that Transformers can internalize implicit learning procedures and are capable of recovering latent functional structure via a single forward pass. We construct a sampling and evaluation pipeline of two nonlinear function classes —sinusoidal regression and Kernel RFF. By training Transformers on predicting the target y based on previous (x, y) pairs in the prefix, we showed that they can predict the target y with the lowest MSE over all methods, which would be impossible without capturing the latent function dynamics. Transformers achieve consistent improvements over different seeds as seen by predictions falling within a small range of standard error, statistically significant superior performance across different training runs.

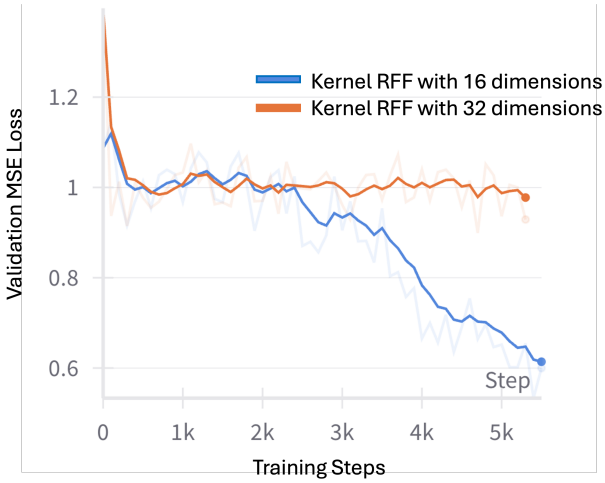

Figure 5: Training curves of kernel RFF estimation with different kernel dimensions, 16 and 32. Note that the main result in Figure 4 is performed with $D = 16$. The model cannot learn the function in high dimension: although the number of parameters it must infer is doubled, the complexity is far beyond it due to the expressivity scaling of the RFF with the number of approximating basis.

Still, many open questions remain: why is the model capable of inferring nonlinear functions, and under what conditions on function classes might it fail to do so? Would Transformers can make predictions beyond a output mapping to a single scalar value such as vector outputs, which would have a broader use case beyond a theoretical interest? How would different model architectures, especially state-space models and hybrid architectures, would perform in these tasks? Answering these questions would further connect empirical observations with principled guarantees.

7 Ethics Statement

We utilized generative AI models (`ChatGPT`, `Grok-3`, Cursor) for helping with understanding the codebase and grammar correction during writing. The artifact for this project was developed based on the training framework proposed by Garg et al. [2022] and built on top of it by forking the repository.

References

Momin Abbas, Yi Zhou, Parikshit Ram, Nathalie Baracaldo, Horst Samulowitz, Theodoros Salonidis, and Tianyi Chen. Enhancing in-context learning via linear probe calibration. In *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 238, pages 1–15. PMLR, 2024.

Salomon Bochner. Monotone funktionen, stieltjessche integrale und harmonische analyse. *Mathematische Annalen*, 108(1):378–410, 1933.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 1126–1135. PMLR, 2017.

Ankit Garg, Kaushal Mehta, Ethan Chi, Song Du, Surya Ganguli, and Benjamin Recht. What can transformers learn in-context? a case study of simple function classes. In *Advances in Neural Information Processing Systems*, volume 35, pages 24898–24911, 2022.

Ritwik Gupta, Rodolfo Corona, Jiaxin Ge, Eric Wang, Dan Klein, Trevor Darrell, and David M Chan. Enough coin flips can make llms act bayesian. *arXiv preprint arXiv:2503.04722*, 2025.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Ziming Liu, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems*, 35:34651–34663, 2022.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*, 2021.

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.

Jongho Park, Jaeseung Park, Zheyang Xiong, Nayoung Lee, Jaewoong Cho, Samet Oymak, Kangwook Lee, and Dimitris Papailiopoulos. Can mamba learn how to learn? a comparative study on in-context learning tasks. *arXiv preprint arXiv:2402.04248*, 2024.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, volume 20, pages 1177–1184, 2007.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023.

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.

Zhijie Wang, Bo Jiang, and Shuai Li. In-context learning on function classes unveiled for transformers. In *Forty-first International Conference on Machine Learning*, 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.

Liu Yang, Ziqian Lin, Kangwook Lee, Dimitris Papailiopoulos, and Robert Nowak. Task vectors in in-context learning: Emergence, formation, and benefit, 2025. URL `https://arxiv.org/abs/2501.09240`.

Yuxuan Zhang, Zhen Li, Yifan Wang, Yuxuan Chen, Yifan Zhang, Yuxuan Li, and Yifan Wang. Secokd: Aligning large language models for in-context learning with fewer shots. *arXiv preprint arXiv:2406.14208*, 2024. URL `https://arxiv.org/abs/2406.14208`.

Siyan Zhao, John Dang, and Aditya Grover. Group preference optimization: Few-shot alignment of large language models. *arXiv preprint arXiv:2310.11523*, 2023.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*, pages 12697–12706. PMLR, 2021.

A RBF/RFF Kernel Discussion

A.1 RFF Explanation

Without Sampling:

In this case, all of the $\mathbf{x}_i$ use the same W and b to get $\phi(\mathbf{x}_i)$.

With Sampling:

Each $\mathbf{x}_i$ uses an arbitrary pair W_i and b_i to get $\phi(\mathbf{x}_i)$.

A.2 Plot Explanation

In the Figure 6The first plot shows the similarity pattern of $\phi(\mathbf{x}_i)$ when using W_1 and b_1. Similarly, the second and third plots represent the similarity patterns obtained using W_2, b_2 and W_3, b_3, respectively.

The fourth plot demonstrates the with sampling strategy, where the first three plots are overlaid to create a combined pattern. During testing, we do not know which W_i, b_i combination the input $\mathbf{x}_j$ is using, so we assume that W_1, b_1, W_2, b_2, and W_3, b_3 are equally likely to be its real transformation. In this context, the In-Context Learning (ICL) model needs to disentangle the true W_i, b_i from this combined pattern.

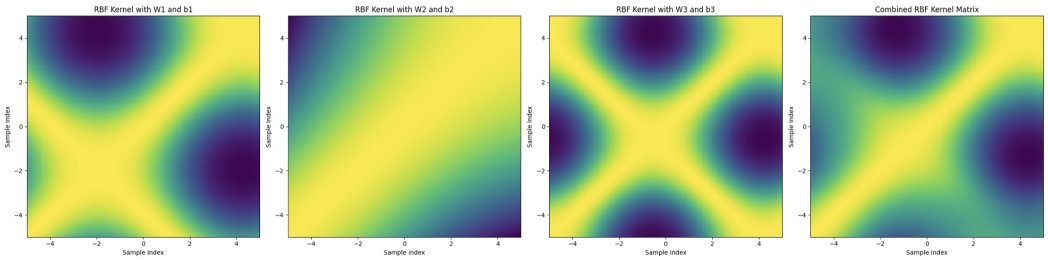

Figure 6: Visualization of the Kernel RFF with different sets of parameters W_i and b_i.

The learning complexity difference has been shown in Figure 7 The left plot shows shows the kernel similarity pattern when a fixed set of parameters (W, b) is used across all data points. This produces a clearer, more consistent pattern since all points share the same transformation.

The right plot represents the overlay of multiple similarity patterns that result from using different sets of W and b. This is because, during testing, we do not know which transformation (i.e., which pair of W_i and b_i) each sample $\mathbf{x}_j$ uses. Therefore, we assume each transformation has an equal probability of being the true one.

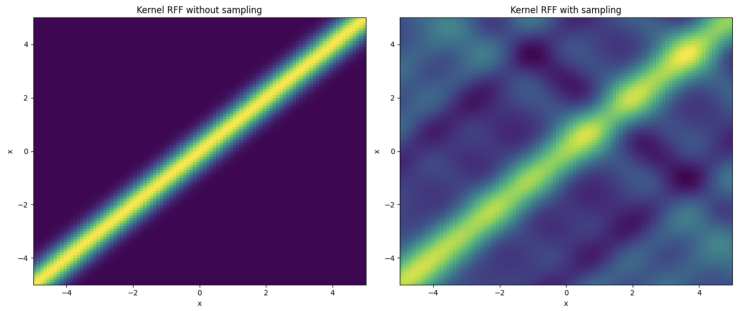

Figure 7: Visualization of Kernel RFF under two different settings: (a) without sampling and (b) with sampling. On the left, the kernel function remains consistent across the entire training and testing process, leading to a smooth, predictable structure. On the right, the use of different kernels for each input sequence results in more complex and irregular patterns, causing a significant challenge in learning.

Thus, the plot shows a combined similarity pattern from all possible transformations. The overlapping of these patterns causes the resulting image to be more complex and blurred compared to the fixed transformation shown in the left plot. This represents the uncertainty and variability introduced during the testing phase, as the model must account for all possible transformations and attempt to disentangle the true transformation from the combined pattern.