CAN IN-CONTEXT LEARNING REALLY GENERALIZE TO OUT-OF-DISTRIBUTION TASKS?

Anonymous authors
Paper under double-blind review

ABSTRACT

In this work, we explore the mechanism of in-context learning (ICL) on out-of-distribution (OOD) tasks that were not encountered during training. To achieve this, we conduct synthetic experiments where the objective is to learn OOD mathematical functions through ICL using a GPT-2 model. We reveal that Transformers may struggle to learn OOD task functions through ICL. Specifically, ICL performance resembles implementing a function within the pretraining hypothesis space and optimizing it with gradient descent based on the in-context examples. Additionally, we investigate ICL's well-documented ability to learn unseen abstract labels in context. We demonstrate that such ability only manifests in the scenarios without distributional shifts and, therefore, may not serve as evidence of new-task-learning ability. Furthermore, we assess ICL's performance on OOD tasks when the model is pretrained on multiple tasks. Both empirical and theoretical analyses demonstrate the existence of the **low-test-error preference** of ICL, where it tends to implement the pretraining function that yields low test error in the testing context. We validate this through numerical experiments. This new theoretical result, combined with our empirical findings, elucidates the mechanism of ICL in addressing OOD tasks.

1 INTRODUCTION

Pretrained large language models (LLMs) can perform in-context learning (ICL) (Brown, 2020), where providing a few examples of input-output pairs and a query example improves the model's ability to generate the desired output, compared to zero-shot predictions. Understanding ICL's ability to learn out-of-distribution (OOD) input-output relationships, which are unseen during training, has recently gained significant attention.

Recent studies have demonstrated that ICL can tackle seemingly new tasks. For instance, Garg et al. (2022); Raventós et al. (2023); Zhang et al. (2023a); Akyürek et al. (2023) found that ICL can learn new linear regression weights after pretraining on a large set of weight vectors. Moreover, Pan (2023); Kossen et al. (2024); Vacareanu et al. (2024) revealed that real-world LLMs like Llama-2 (Touvron et al., 2023) and GPT-4 (Achiam et al., 2023) are capable of solving artificially constructed tasks likely unseen in their pretraining data, such as a classification task with abstract labels.

However, another line of research (Yadlowsky et al., 2023; Ahuja & Lopez-Paz, 2023) has raised a contrasting view, showing that ICL struggles to generalize to OOD tasks where there are distributional shifts in either the input distribution $P(X)$ or the input-label mapping $P(Y|X)$. These findings raise several important questions:

> *Can ICL really learn new input-output mappings from the context? What underlying mechanism of ICL determines its performance on OOD tasks?*

In this work, we aim to consolidate previous findings by addressing these questions. First, we empirically show that when pretrained on a specific function class, the OOD performance of ICL approaches that of a model from the same function class optimized via gradient descent. This suggests that ICL tends to implement functions encountered during pretraining, which could explain its failure on OOD tasks that significantly deviate from the training distribution. Furthermore, we reproduce the widely observed phenomenon that ICL can perform classification with abstract labels.

We find that solving such tasks requires retrieving similar labels from the context, a capability that can be acquired through pretraining on analogous tasks. This implies that success in such tasks of ICL may not indicate an inherent ability to learn new tasks. Finally, we explore scenarios in which the model is pretrained on multiple tasks, empirically uncovering the algorithm selection mechanism for OOD tasks. Building on the work of Lin & Lee (2024), we also provide a comprehensive theoretical framework for understanding the ICL mechanism.

Our contributions are summarized as follows:

1. We empirically show that ICL tends to implement the pretraining function based on the downstream task context, highlighting its limitation in solving OOD tasks (Section 2.1).
2. We further investigate ICL's ability to perform classification with unseen abstract labels. Although this appears to be evidence of ICL learning OOD tasks, we find that such tasks can be solved by retrieving similar examples from the context. This retrieval ability can arise from training on tasks with more diverse abstract labels (Section 3.1) and only emerges when the testing function is in distribution (Section 3.2). Additionally, we find that pretrained Llama-3-8B (Dubey et al., 2024) and Llama-2-7B fails to learn OOD functions through ICL in a synthetic word classification task (Section 3.3), further confirming ICL's limitations in OOD scenarios.
3. Finally, we explore the ICL's behavior when trained on multiple tasks, and observe that the algorithm selection mechanism broadly occurs in OOD scenarios. We theoretically prove the **low-test-error** preference of ICL prediction, i.e., the ICL prediction prefers to implement the pretraining function with lower test error (Section 4.1). We also validate our theory with numerical experiments (4.2).

2 Exploring the ICL Performance on OOD Tasks

2.1 GPT-2 Implements Functions Classes Seen During ICL Pretraining

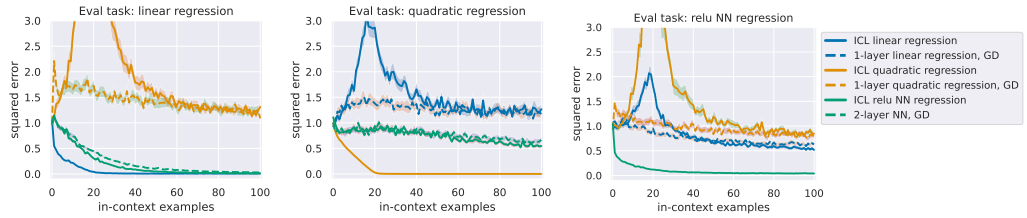
Evaluating GPT-2 on unseen mathematical function classes. To investigate the ICL performance on new tasks that are unseen during training, following Garg et al. (2022), we train a GPT-2 (Radford et al., 2019) from scratch on some simple functions and evaluate it on functions different from the training ones. Denote the Transformer model parameterized by θ as M_θ. The pretraining objective is:

$$\min_\theta \frac{1}{T}\sum_{i=1}^{T}\mathbb{E}_{f\sim\mathcal{F}}[\|M_\theta(\mathcal{S}_i \oplus \boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_{i+1})\|_2^2], \tag{1}$$

where $\mathcal{S}_i = [\boldsymbol{x}_1 \oplus y_1 \oplus \boldsymbol{x}_2 \oplus y_2 \oplus ... \oplus \boldsymbol{x}_i \oplus y_i] \in \mathbb{R}^{d\times 2i}$ is the context of length i, $\oplus$ denotes concatenation. $\boldsymbol{x}_i \in \mathbb{R}^d$ are sampled from a standard Gaussian distribution $\mathcal{N}(0,1)$ with dimension $d = 20$. Let $y_i = f(\boldsymbol{x}_i)$ represent the labels, with $\mathcal{F}$ denoting the hypothesis class to which f belongs. We train three separate GPT-2 models on three different function classes $\mathcal{F}$: linear regression, quadratic regression (element-wise square followed by linear regression), and a 2-layer ReLU network (detailed descriptions are in Appendix C.1). We then evaluate their ICL performance on these three tasks. Note that even when the testing and training functions are i.i.d. sampled from the same task, the specific instances of the testing functions remain unseen during training. For comparison, we also train models within the corresponding $\mathcal{F}$ with gradient descent (GD) using the testing in-context examples (details in Appendix C.1).

Observations. We plot the test error on the three tasks in Figure 1 and observe that: 1) (an existing finding in Garg et al. (2022)) when evaluated on the same task $\mathcal{F}$ as pretraining, ICL can reach near-zero test error. 2) (our novel finding) when evaluated on a new task, ICL performs similarly to the corresponding model of the pretraining function class optimized by GD given enough in-context examples. This indicates that the ICL prediction implicitly implements function classes seen during pretraining. 3) (our novel finding) The models trained on linear and quadratic regression exhibit a double descent error curve (Nakkiran, 2019), characterized by a high error when given exact d examples and evaluated on a new task, which has been theoretically and empirically revealed under the noisy linear regression setting by Nakkiran (2019) and Garg et al. (2022), respectively. This further demonstrates that ICL implements the linear regression pretraining task, as the double

descent curve is a distinctive phenomenon unique to linear regression models. We leave an existing theoretical result of Zhang et al. (2023a) that offers a similar insight in Appendix D.

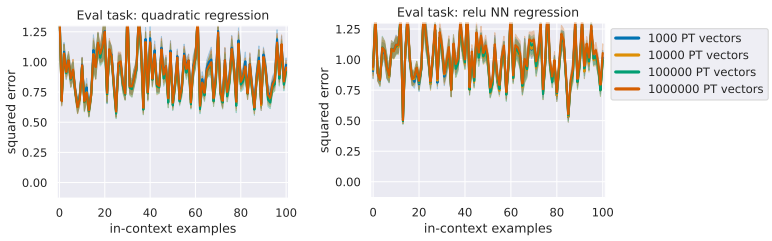


(a) (b) (c)

Figure 1: The ICL test error of Transformers trained on different function classes (solid lines) and the performance of the models from the corresponding pretraining functions classes trained by gradient descent (GD) using the in-context examples (dashed lines). Y-axis: test square error. X-axis: context length. The evaluation tasks are (a) linear regression, (b) quadratic regression, and (c) 2-layer ReLU network regression. In all sub-figures, we observe that as the test context length increases, the ICL performance of the Transformer pretrained on a particular function class closely approaches that of the model from this function class trained by GD.

2.2 Will Generalization Capabilities Emerge from Increasing the Number of Training Tasks?

Recent work by Raventós et al. (2023) empirically demonstrates that when both the training and test tasks are linear regression, and the number of training vectors exceeds a certain "task diversity threshold" (approximately $2^{14} \sim 2^{15}$), ICL can generalize from a finite training set sampled biasedly from $\mathcal{N}(0,1)$ to the test distribution $P_{\text{test}} = \mathcal{N}(0,1)$ (see Appendix A.3 for details). We investigate whether similar phenomena persist for test tasks with larger distributional shifts. We train models using varying numbers of linear regression vectors and evaluate them on quadratic and ReLU neural network regression tasks. In Figure 2, we find that training on a vast number of in-distribution (ID) functions does not yield any improvements, providing further evidence that ICL may struggle to achieve OOD generalization.



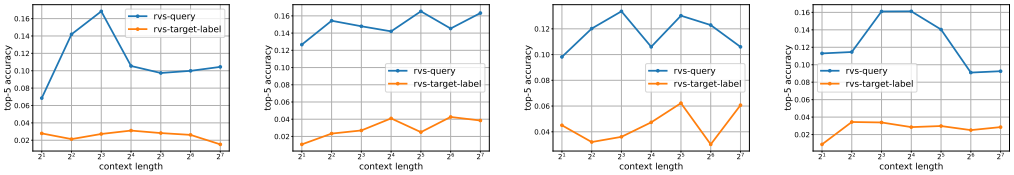
(a) quadratic regression (b) ReLU NN regression

Figure 2: The ICL test error of models trained on different numbers of linear regression vectors. Even if the number of training vectors (up to $1,000,000 \approx 2^{20}$) surpasses the threshold ($2^{14} \sim 2^{15}$) reported by Raventós et al. (2023), no model exhibits generalization to OOD function classes.

2.3 Pretrained LLMs Tend to Make In-distribution Predictions during ICL

In this section, we will demonstrate how the tendency of ICL to perform ID predictions manifests in real-world LLMs. To this end, we designed a task involving predicting labels with letters reversed. In some basic tasks like outputting antonyms or translating from English to French, all the letters of the original labels are reversed (e.g., "positive"→"evitisop"). We found that in this task, a pretrained Llama-3-8B (Dubey et al., 2024) tend to output the reversed result of the query word rather than first predicting the correct label and then reversing it. Although both reversal tasks are uncommon, directly outputting the reversed version of a word is relatively more common than first reasoning

and then outputting the reversed prediction. Therefore, this result reflects to some extent that LLMs, when performing ICL, are more inclined to make predictions within the pretraining distribution. See Appendix C.2 for more details.



(a) Antonym (b) Country-capital (c) English-French (d) English-German

Figure 3: The top-1 accuracy of predicting the reversed query word and predicting the reversed target label word. The accuracy of predicting the reversed query word is higher than outputting the reversed target, indicating ICL makes ID predictions.

3 Learning Abstract Labels May Not Be A New-task-learning Ability

3.1 Classification Tasks with Unseen Abstract Labels

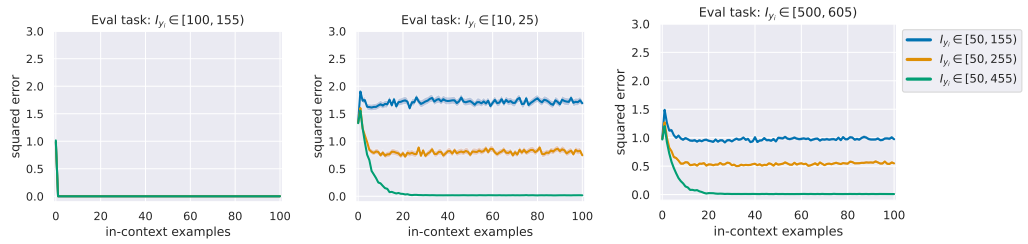
Recent works (Pan, 2023; Kossen et al., 2024) have shown that LLMs can perform classification tasks in which the labels are "abstract symbols" with no semantic meaning. For instance, in the SST-2 binary classification task, the labels "positive" and "negative" are substituted with abstract terms like "foo" and "bar", respectively. These tasks are likely not seen during pretraining. Pan (2023) refer to this ability of ICL to perform such classification as "task learning" (TL). In this section, we explore whether the TL ability really reflects a new-task-learning capability of ICL or if it merely stems from the model having learned similar tasks during pretraining.

The retrieval ability can be gained by pretraining on a retrieval task with diverse input-label mappings. The classification of abstract labels can be approached by first retrieving an example with semantics similar to the query and then outputting the label of that example, as empirically demonstrated in previous research (Wang et al., 2023; Yu & Ananiadou, 2024). Therefore, the retrieval ability is a crucial prerequisite for performing abstract-label classification. We design a retrieval task to investigate whether ICL's retrieval capability can emerge from training on similar tasks. Specifically, we generate a predefined word embedding $E \in \mathbb{R}^{N\times d}$ and randomly sample $\boldsymbol{x}_i \in \mathbb{R}^d$ from the first 5 rows of E. Each vector $\boldsymbol{x}_i$ corresponds to the $I_{\boldsymbol{x}_i}$-th row of E, i.e., $\boldsymbol{x}_i = E_{I_{\boldsymbol{x}_i}}$. To generate the labels y_i, we follow these steps: First, map the index $I_{\boldsymbol{x}_i}$ to a new one $I_{y_i} \in [N]$ using the mapping rule $I_{y_i} = I_{\boldsymbol{x}_i} + s$, where $s \in \mathbb{N}$ is randomly sampled. Second, we set $y_i = E_{I_{y_i}}$. All in-context examples in a sequence share the same mapping rule defined by s. To succeed in this task, the model must retrieve the same token as the query example from the context and output its subsequent token. All models are trained with 200,000×64 sequences, where 200,000 is the number of training steps and 64 is the batch size.

We train three models with three different ranges of s: $s \sim \mathcal{U}(50, 150)$, $s \sim \mathcal{U}(50, 250)$, and $s \sim \mathcal{U}(50, 450)$ and evaluate on $s \sim \mathcal{U}(50, 150)$, $s \sim \mathcal{U}(10, 20)$, and $s \sim \mathcal{U}(500, 600)$, where $\mathcal{U}$ denotes the uniform distribution. We plot the test error in Figure 4.

Observations. In Figure 4, all three models perform well when labels are in distribution (a). When the labels are OOD, the ICL performance improves with the number of label vectors (random mappings) encountered during training. This demonstrates that retrieval ability can emerge from training on diverse retrieval tasks. These findings may also offer new insights into how real-world LLMs develop in-context retrieval capabilities: when autoregressive pretraining includes numerous instances requiring the model to retrieve tokens from previous contexts, such abilities can emerge. We further validate this finding by observing the transition of the attention scores in Appendix B.1.

The ability to perform linear regression and then retrieval can also be gained by pretraining on a similar task. To further reproduce the emergence of the abstract label learning ability of

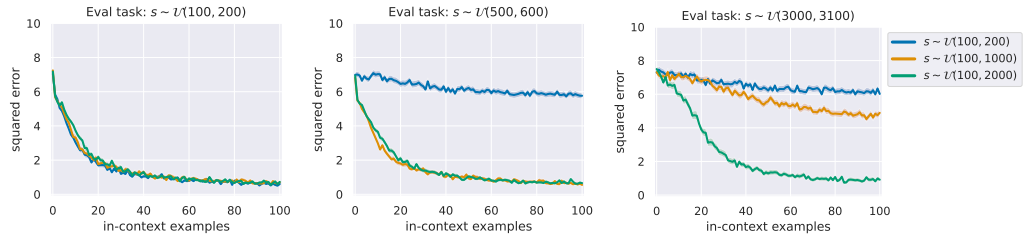


(a) Eval $s \sim \mathcal{U}(50, 150)$ (b) Eval $s \sim \mathcal{U}(10, 20)$ (c) Eval $s \sim \mathcal{U}(500, 600)$

Figure 4: The ICL test error of Transformers trained on the retrieval task with different numbers of label tokens. "Eval" denotes "evaluated on". Note that the indices of training label tokens $I_{y_i} \in [50, 455)$, so the labels in (a) are ID while (b) and (c) are OOD.

real-world LLMs, we design a task that emulates the natural language classification with abstract labels like "foo" and "bar". The task function is defined as follows: $y_i = f(\boldsymbol{x}_i) = E_{I_{\boldsymbol{x}_i}}$, where $I_{\boldsymbol{x}_i} = \text{floor}(0.4 * (\boldsymbol{w}^\top \boldsymbol{x}_i)) + s$, with E being the predefined word embedding and $s \in \mathbb{N}+$ shared in the same sequence. Here, $\boldsymbol{x}_i, \boldsymbol{w} \sim \mathcal{N}(0, 1) \in \mathbb{R}^d$. Each $\boldsymbol{x}_i$ is mapped to y_i according to $\boldsymbol{w}$ and s. [1] In this task, estimating $\boldsymbol{w}$ and calculating $\boldsymbol{w}^\top \boldsymbol{x}_i$ simulates predicting the original label ("positive" and "negative") based on the semantics in the natural language task, while retrieving the abstract labels from in-context examples that share the same $\text{floor}(0.4 * (\boldsymbol{w}^\top \boldsymbol{x}_i))$ as the query from the context resembles identifying the abstract labels ("foo" and "bar").

Again, we train three models on different ranges of mappings: $s \sim \mathcal{U}(100, 200)$, $s \sim \mathcal{U}(100, 1000)$, and $s \sim \mathcal{U}(100, 2000)$, and evaluate on $s \sim \mathcal{U}(100, 200)$, $s \sim \mathcal{U}(500, 600)$, and $s \sim \mathcal{U}(3000, 3100)$. The test error is plotted in Figure 5.



(a) Eval $s \sim \mathcal{U}(100, 200)$ (b) Eval $s \sim \mathcal{U}(500, 600)$ (c) Eval $s \sim \mathcal{U}(3000, 3100)$

Figure 5: The ICL test error of Transformers trained and tested on the linear regression + retrieval task with different numbers of label tokens. "Eval" denotes "evaluated on". Only the model trained on the largest number of tasks exhibits generalization to unseen label tokens.

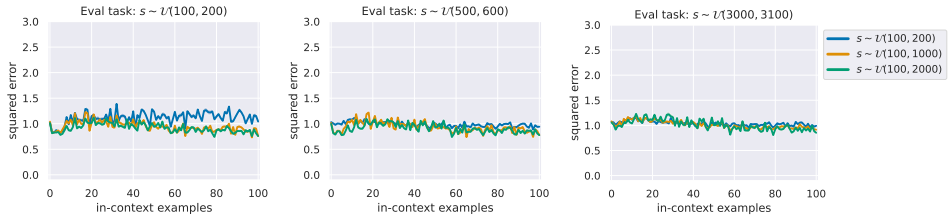
Observations. In Figure 5, the generalization ability to unseen labels also improves as the number of labels encountered during training increases. Notably, only the model trained with $s \sim \mathcal{U}(100, 2000)$ performs well on the unseen labels. This suggests that as long as the LLM has been exposed to sufficiently many similar tasks during training, it can effectively address arbitrary OOD labels retrievable from context through ICL. Therefore, the ability of ICL to perform abstract label classification may not serve as evidence of learning new tasks.

3.2 Abstract Label Classification Can Only Be Achieved on ID Tasks

A retrieval task with OOD testing functions & observations. One might question whether, once the target labels appear in the context, ICL can generalize beyond the training function class by retrieving the target label from the context. To investigate this, we conduct the same predict-then-retrieval task as in Figure 5 but replace the testing functions with quadratic regression while pre-

[1] In our experimental setup, given a sufficiently long context (≈ 50), the label of the query is highly likely to appear in the context, as the number of the possible classes is far less than the number of in-context examples.

serving linear regression as the pretraining task. The results in Figure 6 show that the generalization doesn't improve with training on more ID functions. Combining observations from Figure 5, we conclude that ICL can only solve classification with unseen labels over ID test function classes. Once the underlying task function is OOD, ICL fails even if the target label appears in the context. This finding highlights a limitation in improving an LLM's performance through in-context examples. While providing examples with shared labels may seem helpful, this approach may fail if the task is OOD and the underlying prediction rule is too complex for the LLM to learn.



(a) Eval $s \sim \mathcal{U}(100, 200)$ (b) Eval $s \sim \mathcal{U}(500, 600)$ (c) Eval $s \sim \mathcal{U}(3000, 3100)$

Figure 6: The ICL test error of Transformers evaluated on a quadratic regression + retrieval task. Different colors denote models trained on the linear regression + retrieval task with different numbers of label tokens. "Eval" denotes "evaluation". The model trained on $s \sim \mathcal{U}(100, 2000)$ doesn't generalize better than the other two models.

3.3 Real-world LLMs May Not Necessarily In-context Learn New Tasks

Evaluating Llama-3 on an OOD synthetic word classification task. In this section, we assess whether real-world LLMs can tackle OOD tasks through ICL. We select the pretrained Llama-3-8B and evaluate it on a synthetic word classification task. To ensure the task is far from the pretraining distribution, we randomly sample $\boldsymbol{x}_i \in \mathbb{R}^d$ from the word embedding of Llama-3-8B (denoted as E_{llama}) and generate random linear mappings $\boldsymbol{W} \in \mathbb{R}^{d \times C}$ as task functions (where $C = 10$). The label words are created by mapping $\boldsymbol{x}_i$ to one of the ten label vectors in E_{llama} using $\boldsymbol{W}$. Experimental details are in Appendix C.4. To complete this task, the model must learn $\boldsymbol{W}$ in context.

For comparison, we also evaluate the ICL performance of Llama-3-8B on a retrieval version of this task. Concretely, we first randomly sample $C = 10$ different vectors from E_{llama} as $\boldsymbol{x}_i$ and compute $\boldsymbol{y}_i$ in the same way as the above classification task to get $S = [\boldsymbol{x}_1, \boldsymbol{y}_1, ..., \boldsymbol{x}_C, \boldsymbol{y}_C]$. Then we repeat S 20 times to construct the input sequence $S' = [S \oplus S \oplus ... \oplus S]$, where $\oplus$ denotes concatenation operation. The goal is to predict the next token given a prefix of S'. To succeed in this task, the model has to retrieve the same token as the query token (the last $\boldsymbol{x}_i$ of S') from the context and output its subsequent token $\boldsymbol{y}_i$. The results of these two tasks are presented in Figure 7.

Observations. From Figure 7, we observe that the ICL performance on the synthetic classification task is close to random guessing (10% accuracy), while performance on the retrieval task is significantly better. Considering that the input and label distributions of the two tasks are very similar (similar results also hold for Llama-2-7B in Appendix B.3), we have reason to believe that LLMs struggle to learn new input-output mappings from context; instead, ICL appears to be more adept at retrieval tasks. To show that the failure in the synthetic word classification task is mainly due to its OOD nature instead of some other factors that make it difficult to learn, we train a GPT-2 to perform the same task in Appendix B.2 and find that the task can be well addressed after training.

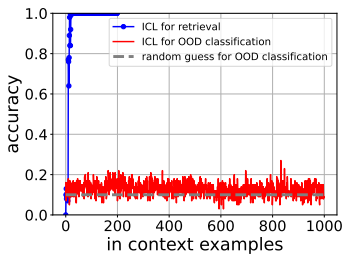
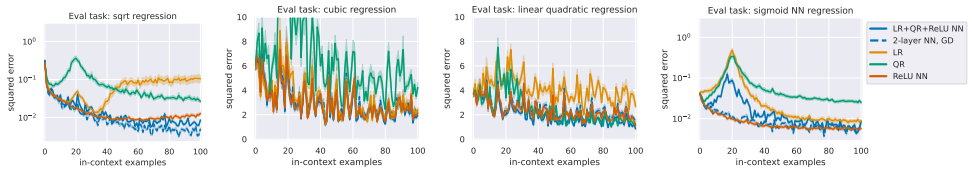


Figure 7: The ICL accuracy of Llama-3-8B on the synthetic tasks. For the retrieval task, we only plot the results within the context length 200 since the performance has saturated rapidly.

4 THE ALGORITHM SELECTION MECHANISM EXISTS BROADLY WHEN EVALUATED ON OOD TASKS

Real-world LLMs are pretrained on a huge corpus that could contain massive tasks. Bai et al. (2023); Yadlowsky et al. (2023) have empirically found that the ICL performance of Transformers trained on multiple tasks approaches the optimal pretraining function when evaluated on one of the training tasks. In this section, we will show that this algorithm-selection phenomenon of ICL persists even when evaluated on OOD tasks, regardless of the distribution of the testing functions, and provide a theoretical characterization of the algorithm-selection mechanism.

The Model pretrained on a single task vs. the model pretrained on multiple tasks. In Figure 8, we compare the performance of GPT-2 models trained on a single task—linear regression (LR), quadratic regression (QR), 2-layer ReLU network (ReLU NN) regression—against the model trained on all three tasks when encountering four kinds of OOD tasks. We also plot the error of a 2-layer ReLU NN trained by GD (dashed blue line). The results are in Figure 8.



(a) (b) (c) (d)

Figure 8: The ICL performance of models trained on the individual task: linear regression (LR), quadratic regression (QR), 2-layer ReLU network (ReLU NN) regression, and the model trained on the mixture of the three tasks (LR+QR+ReLU NN). The evaluation functions are (a) square root, (b) cubic, (c) linear+quadratic, and (d) 2-layer Sigmoid network (details in Appendix C.1). The performance of the model trained on the mixed tasks is comparable to that of the model trained on the single task that performs the best on the evaluation task.

Observations. 1) the ICL performance of the model trained on mixed tasks (LR+QR+ReLU NN) is comparable to the performance of the model trained on a single task with the lowest test error on the evaluation task. This suggests that ICL can automatically select the best pretraining functions according to the downstream context. 2) ReLU NN consistently performs the best on all four OOD test functions. Moreover, the performance of the ReLU model trained by GD aligns well with the ICL performance of the GPT-2 trained on the same function class. This demonstrates that our findings in Section 2.1 still hold when the transformer is trained on a mixture of multiple tasks.

4.1 THEORETICALLY REVEALING THE MECHANISM OF ALGORITHM SELECTION

In this section, we will provide theoretical insight into the working mechanism of the algorithm selection of ICL. We find there simultaneously exist two parallel mechanisms: the **Low-test-error preference** and the **Similar-input-distribution preference.**

A mixed Gaussian pretraining dataset of multiple tasks. In this section, we theoretically analyze the algorithm selection mechanism of ICL on OOD tasks, based on the theoretical framework of Lin & Lee (2024). Consider a noisy linear regression pretraining dataset with the inputs and task weights following the mixed Gaussian distribution:

Assumption 4.1. (Mixed Gaussian pretraining data) Input distribution: $P(\boldsymbol{x}|\boldsymbol{\mu}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \sigma_x^2\boldsymbol{I})$, label distribution: $P(y|\boldsymbol{x}, \boldsymbol{w}) = \mathcal{N}\left(y|\langle\boldsymbol{x}, \boldsymbol{w}\rangle, \sigma_y^2\right)$. The input means and task weights are sampled from a mixed Gaussian distribution: $P(\boldsymbol{\mu}, \boldsymbol{w}) = \sum_{m=1}^{M} \pi_m \mathcal{N}(\boldsymbol{\mu}; \boldsymbol{\mu}_m, \sigma_\mu^2\boldsymbol{I}) \cdot \mathcal{N}\left(\boldsymbol{w}; \boldsymbol{w}_m, \sigma_w^2\boldsymbol{I}\right)$, where $\sum_{m=1}^{M} \pi_m = 1$, $0 < \pi_m < 1$ and $\|\boldsymbol{\mu}_m\| = \|\boldsymbol{w}_m\| = 1, \forall m$. Define $\delta_\mu = \frac{\sigma_\mu^2}{\sigma_x^2}$ and $\delta_w = \frac{\sigma_w^2}{\sigma_y^2}$. Each training sequence $\mathcal{S}_T = [\boldsymbol{x}_1 \oplus y_1 \oplus ... \oplus \boldsymbol{x}_T \oplus y_T]$ is constructed by first sampling the input mean and the task weight according to $P(\boldsymbol{\mu}, \boldsymbol{w})$ and then sampling $\boldsymbol{x}_i$ and y_i according to $P(\boldsymbol{x}|\boldsymbol{\mu})$ and $P(y|\boldsymbol{x}, \boldsymbol{w})$, respectively. Denote this pretraining distribution as P_{tr}.

The lemma below states that the closed-form prediction of the model trained on the pretraining data under Assumption 4.1, given the testing context, remains a Gaussian mixture of the reweighted pretraining task weights.

Lemma 4.2. *(Corollary 2. of Lin & Lee (2024), closed-form ICL prediction of the pretrained model) Denote the model M^* that minimizes the risk on the pretraining data of Assumption 4.1, i.e., $M^* \in \arg\min \frac{1}{T}\sum_{i=0}^{T-1} \mathop{\mathbb{E}}_{S_i \sim P_{tr}} \left[\left\|M\left(\mathcal{S}_i \oplus \boldsymbol{x}_{i+1}\right) - y_{i+1}\right\|^2\right]$, then the prediction on any sequence $\mathcal{S}_i \oplus \boldsymbol{x}_{i+1}$ by M^* is as follows: $M^*\left(\mathcal{S}_i \oplus \boldsymbol{x}_{i+1}\right) = \left\langle \boldsymbol{x}_{i+1}, \sum_{m=1}^{M} \tilde{\pi}_m \tilde{\boldsymbol{w}}_m \right\rangle$. where $\tilde{\pi}_m$, and $\tilde{\boldsymbol{w}}_m$ depending on both the pretraining task and the downstream context example are given in Lemma 1 of Lin & Lee (2024).*

Based on the closed-form ICL prediction, we now analyze how the downstream context affects $\tilde{\pi}$, which determines how ICL selects the pretraining functions. First, we introduce Lemma 4.3 that characterizes the ratio of the reweighted weight of two pretraining tasks:

Lemma 4.3. *(Appendix H.1 of Lin & Lee (2024)) Consider any two different pretraining component α and β, given a testing context $\mathcal{S}_T \oplus \boldsymbol{x}_{T+1}$ and the well-pretrained model M^*, the ratio between the weights of the two task priors $\tilde{\pi}_\alpha/\tilde{\pi}_\beta$ in M^*'s ICL prediction can be decomposed into two terms: $\tilde{\pi}_\alpha/\tilde{\pi}_\beta = \frac{\pi_\alpha}{\pi_\beta}\exp\left(\Psi_{\boldsymbol{\mu}}(\alpha,\beta) + \Psi_{\boldsymbol{w}}(\alpha,\beta)\right)$, where*

$$\Psi_{\boldsymbol{\mu}}(\alpha,\beta) = \left(\sum_{i=1}^{T+1}\|\boldsymbol{\mu}_\beta - \boldsymbol{x}_i\|^2 - \sum_{i=1}^{T+1}\|\boldsymbol{\mu}_\alpha - \boldsymbol{x}_i\|^2\right) / \left(2\sigma_x^2\left(1 + (T+1)\delta_\mu\right)\right). \tag{2}$$

Further, assuming the testing in-context examples $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}^*, \tau_x^2\boldsymbol{I})$, if $\|\boldsymbol{\mu}_\beta - \boldsymbol{\mu}^*\|^2 - \|\boldsymbol{\mu}_\alpha - \boldsymbol{\mu}^*\|^2 \geq 0$ holds, then as the context length $T \to \infty$, the first term $\Psi_{\boldsymbol{\mu}}(\alpha,\beta) \to (\|\boldsymbol{\mu}_\beta - \boldsymbol{\mu}^*\|^2 - \|\boldsymbol{\mu}_\alpha - \boldsymbol{\mu}^*\|^2)/2\sigma_\mu^2 \geq 0$.

However, Lin & Lee (2024) didn't analyze how the second term $\Psi_{\boldsymbol{w}}(\alpha,\beta)$ would evolve given any downstream task, which we will demonstrate to play an important role in the algorithm selection mechanism. In the following theorem, we prove that $\Psi_{\boldsymbol{w}}(\alpha,\beta)$ converges to a non-negative value when the pretraining function class α performs better on the downstream context than β.

Theorem 4.4. *(ICL prediction favors the pretraining function with low error on the context) Given the context $\boldsymbol{S}_T$, if the empirical risk of implementing a function of the pretraining task α is less than that of β, i.e., $\frac{1}{T}\sum_{i=1}^{T}|\boldsymbol{w}_\beta\boldsymbol{x}_i - y_i|^2 - |\boldsymbol{w}_\alpha\boldsymbol{x}_i - y_i|^2 \geq 0$, then, under some mild Assumptions E.2 on the distribution of $\boldsymbol{S}_T$, we have $\Psi_{\boldsymbol{w}}(\alpha,\beta) \geq 0$.*

Combining Lemma 4.3, if the downstream inputs $\boldsymbol{x}_i$, $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}^*, \tau_x^2\boldsymbol{I})$ and $\|\boldsymbol{\mu}_\beta - \boldsymbol{\mu}^*\|^2 - \|\boldsymbol{\mu}_\alpha - \boldsymbol{\mu}^*\|^2 \geq 0$ hold, then as $T \to \infty$, we have $\tilde{\pi}_\alpha/\tilde{\pi}_\beta \geq \pi_\alpha/\pi_\beta$.

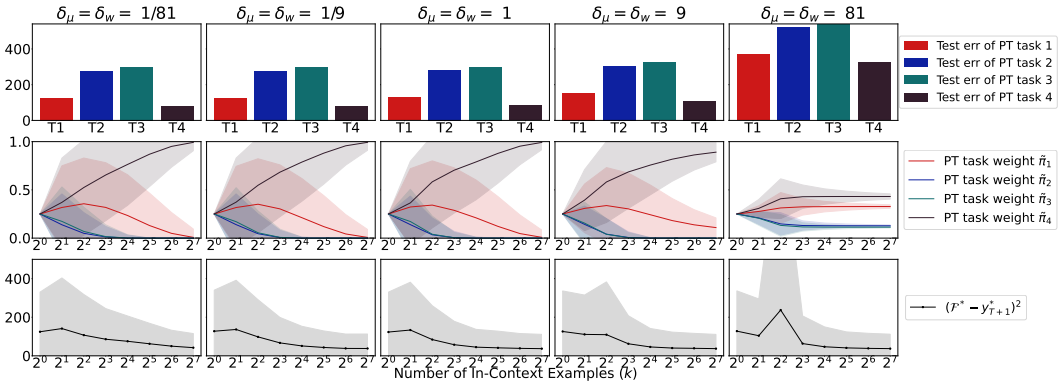
Summary of the algorithm-selection mechanism. 4.3 and Theorem 4.4 together elucidate the algorithm-selection mechanism of ICL. According to Lemma 4.2, the ICL prediction of the model pretrained on the mixed Gaussian data will be a reweighted combination of the pretraining task vectors $\boldsymbol{w}_i$. Whether the ratio between the weights of two pretraining tasks, $\tilde{\pi}_\alpha/\tilde{\pi}_\beta$, given a downstream context, exceeds the original ratio π_α/π_β depends on two factors: 1) whether the pretraining input distribution of α is closer to the downstream input distribution than that of β; 2) whether the task function of α induces lower test error in downstream context than that of β. When both conditions are met, we have $\tilde{\pi}_\alpha/\tilde{\pi}_\beta \geq \pi_\alpha/\pi_\beta$, indicating that ICL prefers α over β in its predictions.

4.2 Empirical Validation of the Algorithm-selection Mechanism of ICL

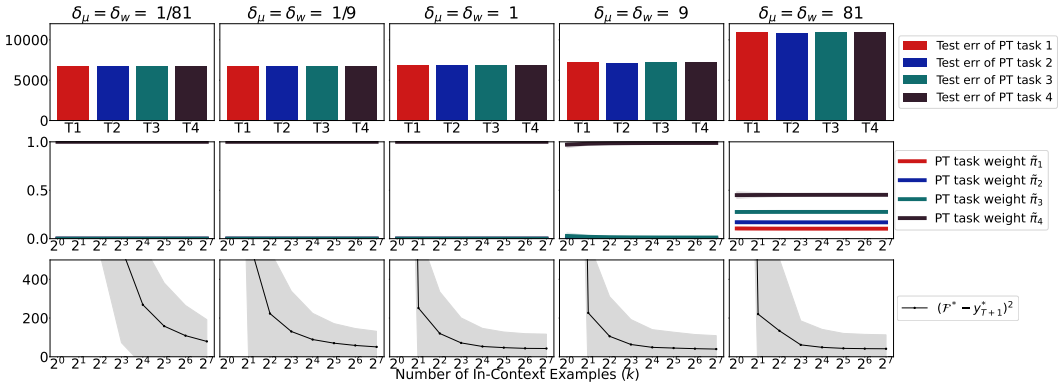
Now we validate our theoretical findings regarding ICL's algorithm-selection mechanism in OOD tasks by conducting numerical experiments following Lin & Lee (2024). In Figure 9a and 9b, the training data is a Gaussian mixture with four components (see Assumption 4.1), while the test function is a two-layer ReLU network (Appendix C.1). Both the training and the test data are in ICL format. We compute the test error of using each pretraining task function to predict the downstream function (the first row of Figure 9a and 9b), the weights for each pretraining function during ICL inference (the second row), and the test error of the pretrained ICL model with the closed form prediction derived in Lemma 4.2 (the third row). We evaluate five different noise levels ($\delta_x = \delta_w \in \{1/81, 9/1, 1, 9, 81\}$) and consider two settings described below.

Low-test-error preference of ICL. To validate Theorem 4.4, we ensure that the distributional distances between the inputs of each training task and the test data remain consistent. Specifically, all $\boldsymbol{x}_i$ in both training and test data are sampled from $\mathcal{N}([0,0,0]^\top, \sigma_x^2 \boldsymbol{I})$. The task weights for different pretraining tasks vary, as detailed in the top half of Table 1. In this setup, only the test error of the pretraining functions influences algorithm selection. From the top two rows of Figure 9a, we can observe a clear negative correlation between the ICL performance and the test error of the task weight. This result supports Theorem 4.4, confirming that ICL prefers the pretraining functions with a low test error in the downstream context. Also, it's consistent with the observations in Figure 8.

Similar-input-distribution preference of ICL. We also empirically validate Lemma 4.3 in Figure 9b. In this case, the distributional distances between the input of different pretraining tasks and that of the test context vary: the distances of different tasks are ordered from largest to smallest as $1 > 2 > 3 > 4$, while the test errors of different pretraining functions are almost the same (detailed setup is in the bottom half of Table 1). As shown in the bottom two rows in Figure 9b, the task weight $\tilde{\pi}_i$ is positively correlated with the similarity between the training and testing input distribution. This is consistent with Lemma 4.3 which demonstrates that ICL prefers to select the pretraining function whose input distribution is close to the downstream one.



(a) Numerical verification of the low-test-error preference



(b) Numerical verification of the similar-input-distribution preference

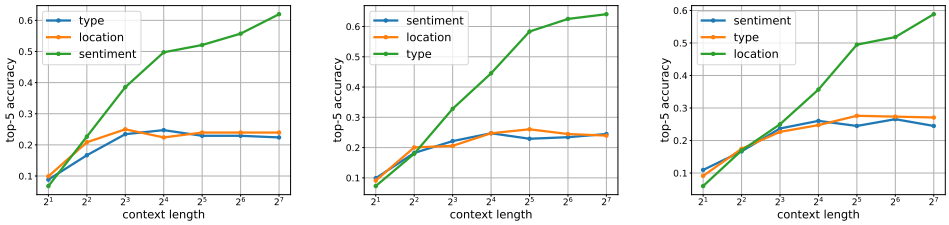
Figure 9: Empirical validation of the algorithm-selection mechanism of ICL. The first rows: the test error of the four pretraining functions. The mid rows: the weights of each pretraining function in the closed-form downstream ICL prediction (given by Lemma 4.2). The last rows: the test error of the pretrained ICL model with the closed form prediction derived in Lemma 4.2. **Observations.** 1) In the first two rows of Figure 9a, the value of the task weight $\tilde{\pi}_i$ is negatively correlated with the test error of pretraining task i. 2) In the first two rows of Figure 9b the task weights are negatively correlated with the distance between the training and testing input distribution.

Table 1: Experiment setting of Figure 9a and Figure 9b. "PT" and "DS" are short for "pretraining" and "downstream", respectively.

Experiment	DS inputs	PT task id	PT input distribution	PT task functions	PT-DS input distance
Figure 9a	$\mathcal{N}([0,0,0]^\top, \sigma_x^2 \boldsymbol{I})$	1	$\mathcal{N}([0,0,0]^\top, \sigma_x^2 \boldsymbol{I})$	$\mathcal{N}([5,5,5]^\top, \sigma_w^2 \boldsymbol{I})$	0
		2	$\mathcal{N}([0,0,0]^\top, \sigma_x^2 \boldsymbol{I})$	$\mathcal{N}([-5,5,5]^\top, \sigma_w^2 \boldsymbol{I})$	0
		3	$\mathcal{N}([0,0,0]^\top, \sigma_x^2 \boldsymbol{I})$	$\mathcal{N}([-5,5,-5]^\top, \sigma_w^2 \boldsymbol{I})$	0
		4	$\mathcal{N}([0,0,0]^\top, \sigma_x^2 \boldsymbol{I})$	$\mathcal{N}([-5,-5,-5]^\top, \sigma_w^2 \boldsymbol{I})$	0
Figure 9b	$\mathcal{N}([-4,-4,-4]^\top, \sigma_x^2 \boldsymbol{I})$	1	$\mathcal{N}([5,5,5]^\top, \sigma_w^2 \boldsymbol{I})$	$\mathcal{N}([1,1,1]^\top, \sigma_w^2 \boldsymbol{I})$	15.59
		2	$\mathcal{N}([-5,5,5]^\top, \sigma_w^2 \boldsymbol{I})$	$\mathcal{N}([1,1,1]^\top, \sigma_w^2 \boldsymbol{I})$	12.77
		3	$\mathcal{N}([-5,5,-5]^\top, \sigma_w^2 \boldsymbol{I})$	$\mathcal{N}([1,1,1]^\top, \sigma_w^2 \boldsymbol{I})$	9.11
		4	$\mathcal{N}([-5,-5,-5]^\top, \sigma_w^2 \boldsymbol{I})$	$\mathcal{N}([1,1,1]^\top, \sigma_w^2 \boldsymbol{I})$	1.73

4.3 Verifying the Algorithm-selection Mechanism on Real-world LLMs

In this section, we investigate whether real-world LLMs can perform algorithm selection through ICL. To achieve this, we design an ambiguous sentence classification task, in which each sentence can be classified based on one of three aspects: "sentiment", "type", or "location". For each ICL sequence, we select one of the aspects as the classification criterion and map the label words to meaningless strings. For instance, if we choose to classify each sentence according to its sentiment, then "positive," "neutral," and "negative" are mapped to "RqF," "IwZ," and "SdK," respectively. Detailed experimental setups are in Appendix C.5. We compute the top-5 accuracy of different classification criteria. The results in Figure 10 show that as the context length increases, the LLM finds the most appropriate criterion, exhibiting the low-test-error preference.



(a) Sentiment (b) Type (c) Location

Figure 10: The top-5 accuracy of using (a)"sentiment", (b)"type", or (c)"location" as the classification criterion for in-context examples in a test prompt. The accuracy of using the true underlying criterion to predict is significantly higher than the other two. This suggests that LLMs can perform algorithm selection in natural language tasks.

5 Conclusion

In this work, we empirically find that Transformers struggle to generalize beyond the pretraining function classes when given downstream in-context examples of OOD tasks. Instead, ICL tries to seek a near-optimal solution within the pretraining function classes. However, ICL performs well in retrieval tasks where the shift in the input-label mapping is only caused by replacing the in-context label tokens with new ones while the underlying function distribution retains. We also examine ICL's performance on OOD tasks after pretraining on multiple tasks. Our theoretical and empirical analysis reveals ICL's preference for low-test-error functions, i.e., ICL tends to implement pretraining function classes with low test error in the test context. This finding, alongside previous work (Lin & Lee, 2024), highlights two key factors that determine how ICL will implement the prediction function based on the testing context and pretraining tasks: the distance between the training and testing input distributions, and the ability of a pretraining function to solve the test task.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical

report. *arXiv preprint arXiv:2303.08774*, 2023.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. In *NeurIPS*, 2023.

Kartik Ahuja and David Lopez-Paz. A closer look at in-context learning under distribution shifts. *arXiv preprint arXiv:2305.16704*, 2023.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *ICLR*, 2023.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *NeurIPS*, 2023.

Tom B Brown. Language models are few-shot learners. In *NeurIPS*, 2020.

Liam Collins, Advait Parulekar, Aryan Mokhtari, Sujay Sanghavi, and Sanjay Shakkottai. In-context learning with transformers: Softmax attention adapts to function lipschitzness. In *NeurIPS*, 2024.

Joy Crosbie and Ekaterina Shutova. Induction heads as an essential mechanism for pattern matching in in-context learning. *arXiv preprint arXiv:2407.07011*, 2024.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In *ICLR*, 2022.

Yu Huang, Yuan Cheng, and Yingbin Liang. In-context convergence of transformers. *arXiv preprint arXiv:2310.05249*, 2023.

Jannik Kossen, Yarin Gal, and Tom Rainforth. In-context learning learns label relationships but is not conventional learning. In *ICLR*, 2024.

Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning. In *ICML*, 2024.

Preetum Nakkiran. More data can hurt for linear regression: Sample-wise double descent. *arXiv preprint arXiv:1912.07242*, 2019.

Jane Pan. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of ACL*, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario AmodeiW, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. In *NeurIPS*, 2023.

Aaditya K Singh, Ted Moskovitz, Felix Hill, Stephanie CY Chan, and Andrew M Saxe. What needs to go right for an induction head? a mechanistic study of in-context learning circuits and their formation. In *ICML*, 2024.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function vectors in large language models. In *ICLR*, 2024.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *JMLR*, 2023.

Robert Vacareanu, Vlad-Andrei Negru, Vasile Suciu, and Mihai Surdeanu. From words to numbers: Your large language model is secretly a capable regressor when given in-context examples. In *COLM*, 2024.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. Label words are anchors: An information flow perspective for understanding in-context learning. In *EMNLP*, 2023.

Zhijie Wang, Bo Jiang, and Shuai Li. In-context learning on function classes unveiled for transformers. In *ICML*, 2024.

Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning. In *NeurIPS*, 2024.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *ICLR*, 2022.

Steve Yadlowsky, Lyric Doshi, and Nilesh Tripuraneni. Pretraining data mixtures enable narrow model selection capabilities in transformer models. *arXiv preprint arXiv:2311.00871*, 2023.

Zeping Yu and Sophia Ananiadou. How do large language models learn in-context? query and key matrices of in-context heads are two towers for metric learning. In *EMNLP*, 2024.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *JMLR*, 2023a.

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*, 2023b.

A Comparison with Related Works

A.1 The Capability of ICL to Learn New Tasks

Besides studies indicating that ICL can learn new weights of linear regression (Garg et al., 2022; Raventós et al., 2023; Zhang et al., 2023a; Akyürek et al., 2023), other research has found that LLMs can tackle tasks that are unlikely to have been encountered during pretraining. For example, Pan (2023) showed that LLMs perform better than random guessing on classification tasks with meaningless labels. Kossen et al. (2024) demonstrate that ICL can identify authorship based on writing style in private communication messages not included in the pretraining corpus. Additionally, Vacareanu et al. (2024) found that large-scale LLMs can learn various linear and non-linear functions from context. We argue that these findings do not contradict our work. While the LLMs may not have seen exactly the same tasks, there is no guarantee that they haven't encountered tasks from a similar distribution in their pretraining corpus. For instance, the LLMs could have been pretrained on a corpus containing authorship identification tasks or on statistical data encompassing different functions. Our work does not claim that ICL cannot generalize to new task instances; rather, it highlights the limitation in generalizing to an unseen input-label distribution. Additionally, Yadlowsky et al. (2023) finds that ICL struggles to generalize to testing function classes that are unseen during training (e.g., convex combinations or extreme versions of the pretraining functions). They didn't delve into how ICL behaves on OOD data, while we reveal that it implements the pretraining functions.

A.2 The Algorithm-selection Mechanism of ICL

Recent works by Bai et al. (2023); Wang et al. (2024) have uncovered the algorithm selection phenomenon, demonstrating that Transformers pretrained on both linear regression and classification tasks perform well when presented with the context of either task during ICL inference. Theoretically, they show that a Transformer with specific parameters can achieve algorithm selection. Yadlowsky et al. (2023) empirically found that ICL selects the optimal pretraining function class after observing in-context examples from a function class present in the pretraining data mixture. However, the algorithm selection experiments in these studies are limited to scenarios where the test functions are among the training functions. In this work, we empirically and theoretically demonstrate that the algorithm selection phenomenon broadly occurs when given downstream context from arbitrary function classes. To the best of our knowledge, we are the first to reveal the factors that determine the selection process.

A.3 The Bayesian-optimal Perspective for Understanding ICL

Many studies have found that ICL makes Bayes-optimal predictions (Xie et al., 2022; Wies et al., 2024; Zhang et al., 2023b; Lin & Lee, 2024). However, these works have certain limitations that may reduce their practical applicability in predicting ICL behavior in general scenarios. 1) Limited empirical verification. Wies et al. (2024) and Zhang et al. (2023b) lack empirical verification of their theory on real deep transformer models; 2) Limited theoretical settings: in-distribution tasks. Wies et al. (2024) assumes the downstream tasks are components of the pretraining distribution; Xie et al. (2022) assumes that the latent concept of the test task θ^* is within the pretraining task set Θ; In Lin & Lee (2024), the training and testing tasks are all linear regression with weights sampled from Gaussian distribution. 3) Limited implications of the theoretical results: although Xie et al. (2022); Zhang et al. (2023b) prove that ICL can infer a task concept θ based on the downstream test context S_{test}, they don't reveal how S_{test} concretely affects the posterior distribution $P(\theta|S_{test})$ of the latent task concept θ inferred by the model that determines the downstream ICL prediction, especially when the true downstream task θ^* is OOD. Our work verifies and extends previous findings to a more general setting by using real deep Transformers and evaluating ICL on OOD tasks that significantly differ from the training tasks. For the first time, we also reveal how the interaction between the downstream distribution and the pretraining distribution affects ICL predictions (see Section 4).

In contrast, Raventós et al. (2023) claim that ICL can exhibit non-Bayesian properties. They empirically demonstrate that when given sufficiently diverse pretraining tasks (linear regression vectors), ICL can outperform the Bayesian estimator on a new test distribution. However, the distributional shift in their setup might not be substantial enough to show that ICL can truly adapt to a new down-

stream distribution, which is considered to be "non-Bayesian" by Raventós et al. (2023). In their setting, both the test and training vectors are sampled from the standard Gaussian distribution, and the only source of "distributional shift" comes from the finite size of the training set, which can only partially reflect the test distribution. Our work refutes their findings by showing that when the test distribution is significantly shifted, increasing the number of ID tasks may not help ICL generalize to it.

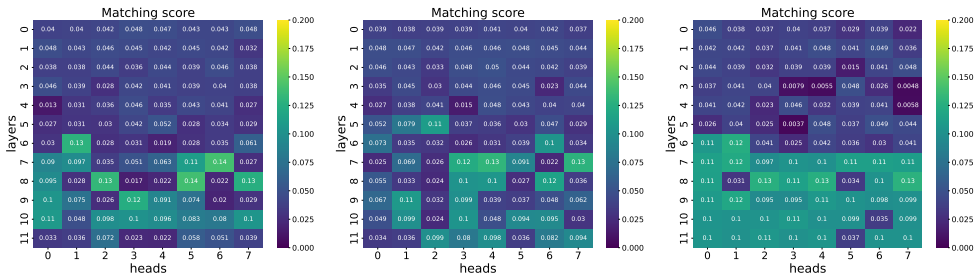
A.4 Discussion of the Setup of Our Theory

Notably, our theoretical result in Section 4.1 doesn't assume a Transformer architecture, while previous theoretical works of understanding ICL often adopt Transformers with oversimplified assumptions on their parameters or structures (Ahn et al., 2023; Zhang et al., 2023a; Huang et al., 2023; Collins et al., 2024). Additionally, our analysis shows that models pretrained on the ICL tasks can implement algorithm selection during ICL inference following Lin & Lee (2024). In contrast, prior work on algorithm selection (Bai et al., 2023) only shows that a specific set of parameters in a simplified ReLU Transformer can enable algorithm selection. However, the parameter construction is complex and somewhat tricky, and there is no theoretical or experimental guarantee that Transformers exhibiting algorithm selection will necessarily implement these parameters.

B Additional Experimental Results

B.1 Understanding the Effect of Training on More Diverse Retrieval Tasks from the Attention Scores

To further validate that the retrieval ability is evoked after trained on more random mappings, following Crosbie & Shutova (2024), we construct another retrieval task and visualize the *prefix matching score* of all attention heads of the three pretrained models in Figure 11. The prefix matching score is calculated by averaging the attention values from each token t_i to the tokens after the same token as t_i in earlier positions in the sequence, which correlates positively with the retrieval performance (Singh et al., 2024). In Figure 11, we observe that the model best at the retrieval task in Figure 4 exhibits more heads with high matching scores, further demonstrating it gains the retrieval ability by training on more retrieval sequences.



(a) PT $s \sim \mathcal{U}(50, 150)$ (b) PT $s \sim \mathcal{U}(50, 250)$ (c) PT $s \sim \mathcal{U}(50, 450)$

Figure 11: The matching score of all attention heads of models trained on the retrieval task. "PT" denotes "pretrained on". Each subfigure corresponds to a different pretrained model. The model of (c) exhibits more heads with high matching scores, which is also the most performant model in the retrieval task in Figure 4.

B.2 The Synthetic Word Classification is Not That Hard to Solve If It's In Distribution

To show the failure in the synthetic word classification in Section 3.3 is mainly due to its OOD nature rather than it's intrinsically too hard to learn, we train a GPT-2 to perform the same task as in Section 3.3. In this task, the $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are generated in the same way as Section 3.3. The only modification

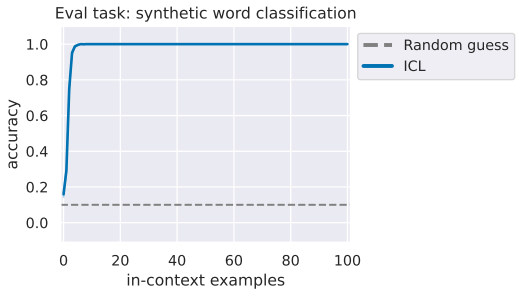
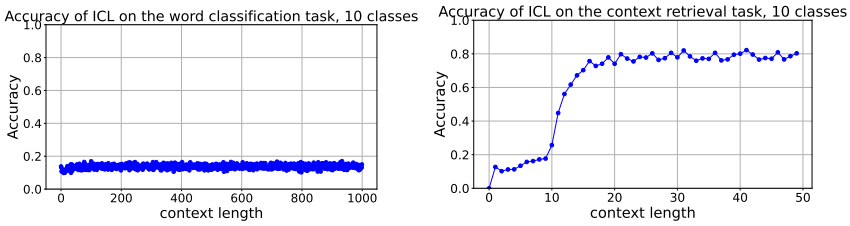


Figure 12: Test error of the GPT-2 trained and evaluated on the same synthetic OOD word classification task as in Section 3.3.

is that we use a smaller predefined vector embedding $E' \in \mathbb{R}^{10000\times 20}$ ($E_{llama} \in \mathbb{R}^{32000\times 4096}$ in the experiment in Section 3.3). The results in Figure 12 show that when $\boldsymbol{W}$ has been encountered during pretraining, ICL can well address this task.

B.3 Evaluating the Synthetic OOD Classification Task on Llama-2-7B

We also evaluate Llama-2-7B on the same OOD word classification task and the retrieval task as in Section 3.3. Figure 13 shows the same observations as in Figure 7 that the LLM can well address the retrieval task but fails to learn the OOD function $\boldsymbol{W}$. In this experiment, we set the length of the repeating sequence to be 10. We can observe that the accuracy of retrieval rapidly increases after seeing 10 in-context examples. This demonstrates that learning novel functions from the context is challenging for real-world pretrained LLMs, but the LLMs are good at retrieving.



(a) (b)

Figure 13: The ICL accuracy of Llama-2-7B on the synthetic tasks. (a) the synthetic word classification task. (b) the synthetic word retrieval task.

C Experimental Details

C.1 Experimental Details in Section 2.1 and Section 4.2

Definitions of the function classes. The function classes in Figure 1 and Figure 8 are:

- Linear regression: $y_i = \boldsymbol{w}^\top \boldsymbol{x}_i$, where $\boldsymbol{w}, \boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{w}, \boldsymbol{x}_i \sim \mathcal{N}(0,1)$.
- Quadratic regression: $y_i = \boldsymbol{w}^\top (\boldsymbol{x}_i)^2$, where $\boldsymbol{w}, \boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{w}, \boldsymbol{x}_i \sim \mathcal{N}(0,1)$, $(\boldsymbol{x}_i)^2$ denotes the element-wise square of $\boldsymbol{x}_i$.
- 2-layer ReLU network regression: $y_i = \boldsymbol{w}_1^\top \mathrm{ReLU}(\boldsymbol{w}_2 \boldsymbol{x}_i)$, where $\boldsymbol{w}_1 \in \mathbb{R}^{d'}$, $\boldsymbol{w}_2 \in \mathbb{R}^{d'\times d}$, and $\boldsymbol{x}_i \in \mathbb{R}^d$. $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{x}_i \sim \mathcal{N}(0,1)$.
- Square root linear regression: $y_i = \boldsymbol{w}^\top \sqrt{\boldsymbol{x}_i}$, where $\boldsymbol{w}, \boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{w}, \boldsymbol{x}_i \sim \mathcal{N}(0,1)$, $(\boldsymbol{x}_i)^2$ denotes the element-wise square root of $\boldsymbol{x}_i$.
- Cubic linear regression: $y_i = \boldsymbol{w}^\top (\boldsymbol{x}_i)^3$, where $\boldsymbol{w}, \boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{w}, \boldsymbol{x}_i \sim \mathcal{N}(0,1)$, $(\boldsymbol{x}_i)^2$ denotes the element-wise cube of $\boldsymbol{x}_i$.

- Linear+quadratic regression: $y_i = \boldsymbol{w}_1^\top (\boldsymbol{x}_i)^2 + \boldsymbol{w}_2^\top \boldsymbol{x}_i$, where $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, $\boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, $\boldsymbol{x}_i \sim \mathcal{N}(0, 1)$.
- 2-layer Sigmoid network: $y_i = \boldsymbol{w}_1^\top \text{Sigmoid}(w_2 \boldsymbol{x}_i)$, where $\boldsymbol{w}_1 \in \mathbb{R}^{d'}$, $\boldsymbol{w}_2 \in \mathbb{R}^{d' \times d}$, and $\boldsymbol{x}_i \in \mathbb{R}^d$. $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, $\boldsymbol{x}_i \sim \mathcal{N}(0, 1)$.

Baseline models in Figure 1. The models of each pretraining hypothesis class are implemented by training a neural network that yields functions of that hypothesis class. For example, a linear regression weight w can be implemented by a single linear layer. The models are optimized using SGD with learning rate 1e-3 for 1000 steps.

C.2 Experimental Details for Section 2.3

For the reversed-label experiment, we choose four tasks: Antonym, Capital-country, English-French, and English-German. The original datasets are adopted from Todd et al. (2024). The top-1 accuracy is computed as follows: compute the top-1 accuracy for each token predicted by the model, based on the token length of the ground-truth label word. For each context length, we compute the average accuracy over 128 test examples.

C.3 Experimental Details for Section 3.1

We now provide additional details regarding the experiments of Figure 11 . Following Crosbie & Shutova (2024), we generated a dataset consisting of 100 sequences of random tokens, each containing repeated sub-sequences. The task is to predict the next token that follows the last token in each sequence. This task can only be completed by retrieving the last token from the context and predicting its subsequent token.

C.4 Experimental Details for Section 3.3

We uniformly sample 1000 word vectors $\boldsymbol{x}_i \in \mathbb{R}^d$ from the word embedding $E \in \mathbb{R}^{N \times d}$ of the pretrained Llama-3-8B, where $N = 128256$ and $d = 4096$. Then we sample a task weight $\boldsymbol{W} \in \mathbb{R}^{d' \times C}$ from standard Gaussian distribution that only takes the first d' dimensions of $\boldsymbol{x}_i$ (denoted as $\boldsymbol{x}_i[:d']$) to compute a probability distribution over C classes: $p_i = \boldsymbol{x}_i[:d']^\top \boldsymbol{W} \in \mathbb{R}^C$. Next, we set the label vectors $\boldsymbol{y}_i = E_{\arg\max_j p_i[j]+s} \in \mathbb{R}^d$, where $s = 10000$ is a offset. We set $d' = 30 \ll d = 4096$ to reduce the complexity of the task. Hence, $\boldsymbol{x}_i$ are classified into C labels words $E[s : s+C]$. The predicted token of $\boldsymbol{x}_i$ is computed as: $\arg\max_j \hat{p}_i[j],\ j \in \{s, s+1, ..., s+C-1\}$, where $\hat{p}_i$ is the output of the last linear layer of Llama-3-8B given $\boldsymbol{x}_i$.

C.5 Experimental Details for Section 4.3

In this section, we present some details about the setups for the ambiguous classification task. The label mapping rule is presented in Table 2. For each context length, we compute the average accuracy over 128 test examples.

Table 2: Experiment setting of Figure 9a and Figure 9b. "PT" and "DS" are short for "pretraining" and "downstream", respectively.

Classification criterion	Original labels	Labels presented in the context
sentiment	"positive" "neutral" "negative"	"RqF" "IwZ" "SdK"
type	"science" "sports" "arts"	"RqF" "IwZ" "SdK"
location	"Asia" "Europe" "Africa"	"RqF" "IwZ" "SdK"

Prompt examples. Here we present some in-context examples of the input prompt of using different classification criteria.

Using "sentiment" as the classification criterion.

Q: The groundbreaking discovery made by Japanese scientists has revolutionized renewable energy.
A: RqF *# Original label: "positive"*

Q: A chess championship occurred in Russia, featuring players from around the continent.
A: IwZ *# Original label: "neutral"*

Using "type" as the classification criterion.

Q: A regional basketball league was formed in Kenya to promote the sport locally.
A: IwZ *# Original label: "sports"*

Q: The breathtaking architectural exhibition in Dubai left visitors absolutely awestruck.
A: SdK *# Original label: "arts"*

Using "location" as the classification criterion.

Q: A scientific paper from Finland explores new methodologies in data analysis.
A: IwZ *# Original label: "Europe"*

Q: An astronomy workshop was conducted in Ethiopia for students interested in space.
A: SdK *# Original label: "Africa"*

Accuracy computation. For a given label, the method to calculate top-5 accuracy is as follows: compute the top-5 accuracy for each token predicted by the model, based on the token length of the ground-truth label word. For a classification criterion other than the one selected in the current sequence, to verify whether the model's prediction distribution across all test samples approaches the label distribution under that criterion, we select the permutation among all possible mappings between original labels and meaningless strings that yields the highest model prediction accuracy to compute the accuracy.

D Existing Theoretical Evidence Supporting that ICL Makes ID Predictions

One recent work (Zhang et al., 2023a) theoretically proved that a one-layer linear self-attention model (LSA, defined in Appendix D) pretrained on a linear regression task will still implement the linear predictor given downstream in-context examples of arbitrary new function classes, under some assumptions on the initialization of the Transformer weight matrices. We restate the Theorem 4.2 of Zhang et al. (2023a) as Lemma D.1 below:

Lemma D.1. *(Theorem 4.2 of Zhang et al. (2023a), informal) Let $\mathcal{D}$ be a distribution over $(\boldsymbol{x}, y) \in \mathbb{R}^d \times \mathbb{R}$, whose marginal distribution on x is $\mathcal{D}_x = \mathcal{N}(0, \Lambda)$. Assume the test prompt is of the form $P = (\boldsymbol{x}_1, y_1, \ldots, \boldsymbol{x}_T, y_T, \boldsymbol{x}_{query})$, where $(\boldsymbol{x}_i, y_i), (\boldsymbol{x}_{query}, y_{query}) \overset{i.i.d.}{\sim} \mathcal{D}$. The prediction risk on the test query y_{query} of an arbitrary task satisfies:*

$$\mathbb{E}\left(\widehat{y}_{query} - y_{query}\right)^2 = \underbrace{\min_{\boldsymbol{w} \in \mathbb{R}^d} \mathbb{E}\left(\langle \boldsymbol{w}, \boldsymbol{x}_{query} \rangle - y_{query}\right)^2}_{\textit{Error of best linear predictor}} + const,$$

where const is a constant depending on the downstream context, and the expectation is over $(\boldsymbol{x}_i, y_i), (\boldsymbol{x}_{query}, y_{query}) \overset{i.i.d.}{\sim} \mathcal{D}$.

Lemma D.1 serves as a shred of theoretical evidence that ICL can just implement the pretraining function class, while the role of the context examples is to optimize the model within the pretraining hypothesis space.

Below, we provide the necessary details of the theoretical setting of Zhang et al. (2023a).

The linear self-attention (LSA) model considered in the Theorem 4.2 of Zhang et al. (2023a) (Lemma D.1) is defined as follows:

$$f_{\text{LSA}}(E;\theta) = E + W^{PV}E \cdot \frac{E^\top W^{KQ} E}{N}, \tag{3}$$

where E is the input embedding defined as follows:

$$E = E(P) = \begin{pmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N & \boldsymbol{x}_{\text{query}} \\ y_1 & y_2 & \cdots & y_N & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(N+1)}. \tag{4}$$

W^{PV} is obtained by merging the projection and value matrices into a single matrix, and W^{KQ} is attained by merging the query and key matrices into a single matrix. N is the context length.

Now we present the assumption on the attention weights of the linear-attention model in Lemma D.1.

Assumption D.2. (Assumption 3.3 in Zhang et al. (2023a), initialization). Let $\sigma > 0$ be a parameter, and let $\Theta \in \mathbb{R}^{d\times d}$ be any matrix satisfying $\left\|\Theta\Theta^\top\right\|_F = 1$ and $\Theta\Lambda \neq 0_{d\times d}$. We assume

$$W^{PV}(0) = \sigma \begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix}, \quad W^{KQ}(0) = \sigma \begin{pmatrix} \Theta\Theta^\top & 0_d \\ 0_d^\top & 0 \end{pmatrix}$$

The training objective is to minimize the population risk of the linear regression task:

$$L(\theta) = \lim_{B\to\infty} \widehat{L}(\theta) = \frac{1}{2}\mathbb{E}_{w_\tau, \boldsymbol{x}_{\tau,1}, \cdots, \boldsymbol{x}_{\tau,N}, \boldsymbol{x}_{\tau,\text{ query}}} \left[\left(\widehat{y}_{\tau,\text{ query}} - \langle w_\tau, \boldsymbol{x}_{\tau,\text{ query}} \rangle\right)^2 \right], \tag{5}$$

where $w_\tau \sim \mathcal{N}(0, I_d)$, $\boldsymbol{x}_{\tau,i}$, $\boldsymbol{x}_{\tau,query} \sim \mathcal{N}(0, \Lambda)$, $\widehat{y}_{\tau,\text{ query}}$ is the prediction of the LSA model.

E Proof of Theorem 4.4

We restate Theorem 4.4 as the Theorem E.1 below, and present the assumption it depends on.

Theorem E.1. *(ICL prediction favors the pretraining function with low error on the context) Given the context S_k, if the empirical risk of implementing a function of the pretraining task α is less than that of β, i.e., $\frac{1}{T}\sum_{t=1}^T |\boldsymbol{w}_\beta \boldsymbol{x}_i - y_i|^2 - |\boldsymbol{w}_\alpha \boldsymbol{x}_i - y_i|^2 \geq 0$, then, under some mild Assumptions E.2, we have $\Psi_{\boldsymbol{w}}(\alpha,\beta) \geq 0$.*

Combining Lemma 4.3, if the downstream inputs $\boldsymbol{x}_i$, $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}^*, \tau_x^2 \boldsymbol{I})$ and $\|\boldsymbol{\mu}_\beta - \boldsymbol{\mu}^*\|^2 - \|\boldsymbol{\mu}_\alpha - \boldsymbol{\mu}^*\|^2 \geq 0$ hold, then as $T \to \infty$, we have $\tilde{\pi}_\alpha/\tilde{\pi}_\beta \geq \pi_\alpha/\pi_\beta$.

Assumption E.2. (Assumption on the distribution of the downstream context examples.) Assume that: the minimum eigenvalue of the covariance matrix of any in-context example $\boldsymbol{x}_i$ satisfies $\lambda_{\min}(\boldsymbol{x}_i\boldsymbol{x}_i^\top) \geq 1$; $(\boldsymbol{I} + T\delta_w \boldsymbol{I})(\boldsymbol{I} + \delta_w \sum_{i=1}^T \boldsymbol{x}_i\boldsymbol{x}_i^\top)^{-1} = \boldsymbol{I}$; $\frac{1}{T}\sum_{i=1}^T 2(\boldsymbol{w}_\alpha - \boldsymbol{w}_\beta)^\top \boldsymbol{x}_i y_i \frac{1}{T}\sum_{j=1}^T \left(\boldsymbol{x}_j^\top \boldsymbol{x}_i \frac{y_j}{y_i} - \boldsymbol{x}_i^\top \boldsymbol{x}_i\right) \geq 0$

Proof. According to Lemma 1 of Lin & Lee (2024),

$$r(\alpha,\beta) = \frac{\tilde{\pi}_\alpha}{\tilde{\pi}_\beta} = \frac{\pi_\alpha C_0 c_\alpha^\mu c_\alpha^w}{\pi_\beta C_0 c_\beta^\mu c_\beta^w} = \frac{\pi_\alpha}{\pi_\beta} \exp\left(\Psi_\mu(\alpha,\beta) + \Psi_{\boldsymbol{w}}(\alpha,\beta)\right). \tag{6}$$

In the Appendix H.1 of Lin & Lee (2024), they have proved that when the context length $T \to \infty$, under the first condition in Assumption E.2, $\lim_{T\to\infty} \Psi_\mu(\alpha,\beta) =\geq 0$.

Now we prove that when the empirical risk on the in-context example of pretraining task function α is no more than that of β, the second term $\Psi_{\boldsymbol{w}}(\alpha,\beta) \geq 0$.

$$
\begin{aligned}
&\Psi_{\boldsymbol{w}}(\alpha,\beta)\\
&= \log\left(\frac{\exp\left(-\frac{\|\boldsymbol{w}_\alpha\|^2-\|\boldsymbol{w}_\alpha+T\delta_w\overline{\boldsymbol{w}}\|^2_{(\boldsymbol{I}+T\delta_w\boldsymbol{\Sigma}_{\boldsymbol{w}})^{-1}}}{2\sigma_w^2}\right)}{\exp\left(-\frac{\|\boldsymbol{w}_\beta\|^2-\|\boldsymbol{w}_\beta+T\delta_w\boldsymbol{w}\|^2_{(\boldsymbol{I}+T\delta_w\Sigma_{\boldsymbol{w}})^{-1}}}{2\sigma_w^2}\right)}\right)\\
&= \frac{\|\boldsymbol{w}_\beta\|^2-\|\boldsymbol{w}_\beta+T\delta_w\overline{\boldsymbol{w}}\|^2_{\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}}{2\sigma_w^2}-\frac{\|\boldsymbol{w}_\alpha\|^2-\|\boldsymbol{w}_\alpha+T\delta_w\overline{\boldsymbol{w}}\|^2_{\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}}{2\sigma_w^2}\\
&= \frac{\|\boldsymbol{w}_\beta\|^2-\left\|\boldsymbol{w}_\beta+\delta_w\sum_{i=1}^T\boldsymbol{x}_iy_i\right\|^2_{\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}}{2\sigma_w^2}-\frac{\|\boldsymbol{w}_\alpha\|^2-\left\|\boldsymbol{w}_\alpha+\delta_w\sum_{i=1}^T\boldsymbol{x}_iy_i\right\|^2_{\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}}{2\sigma_w^2}\\
&= \frac{\|\boldsymbol{w}_\beta\|^2-\left\|(\boldsymbol{w}_\beta-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T})+(\boldsymbol{I}+T\boldsymbol{I}\delta_w)\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\right\|^2_{\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}}{2\sigma_w^2}\\
&- \frac{\|\boldsymbol{w}_\alpha\|^2-\left\|(\boldsymbol{w}_\alpha-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T})+(\boldsymbol{I}+T\boldsymbol{I}\delta_w)\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\right\|^2_{\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}}{2\sigma_w^2}\\
&\overset{(a)}{=} \|\boldsymbol{w}_\beta-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\|^2_{\boldsymbol{I}-\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}-\|\boldsymbol{w}_\alpha-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\|^2_{\boldsymbol{I}-\left(\boldsymbol{I}+T\delta_w\overline{\boldsymbol{\Sigma}}_{\boldsymbol{w}}\right)^{-1}}\\
&\overset{(b)}{=} \|\boldsymbol{w}_\beta-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\|^2_{\frac{\delta_w\sum_{i=1}^T\boldsymbol{x}_i\boldsymbol{x}_i^\top}{1+\delta_w\sum_{i=1}^T\boldsymbol{x}_i^\top\boldsymbol{x}_i}}-\|\boldsymbol{w}_\alpha-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\|^2_{\frac{\delta_w\sum_{i=1}^T\boldsymbol{x}_i\boldsymbol{x}_i^\top}{1+\delta_w\sum_{i=1}^T\boldsymbol{x}_i^\top\boldsymbol{x}_i}}
\end{aligned}
\tag{7}
$$

where equation (a) is due to the third condition in Assumption E.2, equation (b) is by applying the Sherman–Morrison formula. Since $\frac{\delta_w}{1+\delta_w\sum_{i=1}^T}\geq 0$, to prove that $\Psi_{\boldsymbol{w}}(\alpha,\beta)\geq 0$, we only need to show that

$$
\|\boldsymbol{w}_\beta-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\|^2_{\sum_{i=1}^T\boldsymbol{x}_i\boldsymbol{x}_i^\top}-\|\boldsymbol{w}_\alpha-\frac{\sum_{i=1}^T\boldsymbol{x}_iy_i}{T}\|^2_{\sum_{i=1}^T\boldsymbol{x}_i\boldsymbol{x}_i^\top}\geq 0. \tag{8}
$$

Further, we can derive that the term $\frac{1}{T}\sum_{i=1}^T\|\boldsymbol{w}_\beta-\boldsymbol{x}_iy_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}-\|\boldsymbol{w}_\alpha-\boldsymbol{x}_iy_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}$ below is non-negative by using the condition 2 in Assumption E.2:

$$
\begin{aligned}
&\frac{1}{T}\sum_{i=1}^T\|\boldsymbol{w}_\beta-\boldsymbol{x}_iy_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}-\|\boldsymbol{w}_\alpha-\boldsymbol{x}_iy_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}\\
&=\frac{1}{T}\sum_{i=1}^T(\boldsymbol{w}_\beta-\boldsymbol{x}_iy_i)^\top\boldsymbol{x}_i\boldsymbol{x}_i^T(\boldsymbol{w}_\beta-\boldsymbol{x}_iy_i)-(\boldsymbol{w}_\alpha-\boldsymbol{x}_iy_i)^\top\boldsymbol{x}_i\boldsymbol{x}_i^T(\boldsymbol{w}_\alpha-\boldsymbol{x}_iy_i)\\
&=\frac{1}{T}\sum_{i=1}^T(\boldsymbol{w}_\beta+\boldsymbol{w}_\alpha-2\boldsymbol{x}_iy_i)^\top\boldsymbol{x}_i\boldsymbol{x}_i^T(\boldsymbol{w}_\beta-\boldsymbol{w}_\alpha)\\
&\underset{(c)}{\geq}\frac{1}{T}\sum_{i=1}^T(\boldsymbol{w}_\beta+\boldsymbol{w}_\alpha-2\boldsymbol{x}_iy_i)^\top(\boldsymbol{w}_\beta-\boldsymbol{w}_\alpha)\\
&=\frac{1}{T}\sum_{i=1}^T\|\boldsymbol{w}_\beta^\top\boldsymbol{x}_i-y_i\|^2-\|\boldsymbol{w}_\alpha^\top\boldsymbol{x}_i-y_i\|^2\underset{(d)}{\geq}0
\end{aligned}
\tag{9}
$$

where the inequality (c) holds since according to the condition 2 in Assumption E.2, $\boldsymbol{x}_i\boldsymbol{x}_i^T-\boldsymbol{I}$ is positive semi-definite, and the inequality (d) holds since the population downstream risk of α is lower than β. Therefore, to prove inequality (8), we just need to prove that the l.h.s. of inequality

(8) multiplying $\frac{1}{T}$ is not less than $\frac{1}{T}\sum_{i=1}^{T}\|\boldsymbol{w}_\beta - \boldsymbol{x}_i y_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}$ in Equation (9):

$$\frac{1}{T}\left(\|\boldsymbol{w}_\beta - \frac{\sum_{i=1}^{T}\boldsymbol{x}_i y_i}{T}\|^2_{\sum_{i=1}^{T}\boldsymbol{x}_i\boldsymbol{x}_i^\top} - \|\boldsymbol{w}_\alpha - \frac{\sum_{i=1}^{T}\boldsymbol{x}_i y_i}{T}\|^2_{\sum_{i=1}^{T}\boldsymbol{x}_i\boldsymbol{x}_i^\top}\right) \geq \frac{1}{T}\sum_{i=1}^{T}\|\boldsymbol{w}_\beta - \boldsymbol{x}_i y_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T} - \|\boldsymbol{w}_\alpha - \boldsymbol{x}_i y_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}. \tag{10}$$

First, let's simplify the l.h.s of inequality (10):

$$\begin{aligned}
&\frac{1}{T}\left(\|\boldsymbol{w}_\beta - \frac{\sum_{i=1}^{T}\boldsymbol{x}_i y_i}{T}\|^2_{\sum_{i=1}^{T}\boldsymbol{x}_i\boldsymbol{x}_i^\top} - \|\boldsymbol{w}_\alpha - \frac{\sum_{i=1}^{T}\boldsymbol{x}_i y_i}{T}\|^2_{\sum_{i=1}^{T}\boldsymbol{x}_i\boldsymbol{x}_i^\top}\right)\\
&= \frac{1}{T}\sum_{i=1}^{T}(\boldsymbol{w}_\beta - \frac{\sum_{j=1}^{T}\boldsymbol{x}_j y_j}{T})^\top \boldsymbol{x}_i\boldsymbol{x}_i^\top(\boldsymbol{w}_\beta - \frac{\sum_{j=1}^{T}\boldsymbol{x}_j y_j}{T}) - (\boldsymbol{w}_\alpha - \frac{\sum_{j=1}^{T}\boldsymbol{x}_j y_j}{T})^\top \boldsymbol{x}_i\boldsymbol{x}_i^\top(\boldsymbol{w}_\alpha - \frac{\sum_{j=1}^{T}\boldsymbol{x}_j y_j}{T})\\
&= \frac{1}{T}\sum_{i=1}^{T}\|\boldsymbol{w}_\beta^\top\boldsymbol{x}_i - \frac{1}{T}\sum_{j=1}^{T}\boldsymbol{x}_j^\top\boldsymbol{x}_i y_j\|^2 - \|\boldsymbol{w}_\alpha^\top\boldsymbol{x}_i - \frac{1}{T}\sum_{j=1}^{T}\boldsymbol{x}_j^\top\boldsymbol{x}_i y_j\|^2\\
&= \frac{1}{T}\sum_{i=1}^{T}(\boldsymbol{w}_\beta^\top\boldsymbol{x}_i)^2 - (\boldsymbol{w}_\alpha^\top\boldsymbol{x}_i)^2 + 2(\boldsymbol{w}_\alpha - \boldsymbol{w}_\beta)^\top\boldsymbol{x}_i\frac{1}{T}\sum_{j=1}^{T}\boldsymbol{x}_j^\top\boldsymbol{x}_i y_j.
\end{aligned} \tag{11}$$

Then we simplify the r.h.s. of inequality (10):

$$\begin{aligned}
&\frac{1}{T}\sum_{i=1}^{T}\|\boldsymbol{w}_\beta - \boldsymbol{x}_i y_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T} - \|\boldsymbol{w}_\alpha - \boldsymbol{x}_i y_i\|^2_{\boldsymbol{x}_i\boldsymbol{x}_i^T}\\
&= \frac{1}{T}\sum_{i=1}^{T}(\boldsymbol{w}_\beta^\top\boldsymbol{x}_i)^2 - (\boldsymbol{w}_\alpha^\top\boldsymbol{x}_i)^2 + 2(\boldsymbol{w}_\alpha - \boldsymbol{w}_\beta)^\top\boldsymbol{x}_i\boldsymbol{x}_i^\top\boldsymbol{x}_i y_i
\end{aligned} \tag{12}$$

Subtracting Equation (12) from Equation (11), we get

$$\begin{aligned}
&\frac{1}{T}\sum_{i=1}^{T}2(\boldsymbol{w}_\alpha - \boldsymbol{w}_\beta)^\top\boldsymbol{x}_i\frac{1}{T}\sum_{j=1}^{T}\boldsymbol{x}_j^\top\boldsymbol{x}_i y_j - 2(\boldsymbol{w}_\alpha - \boldsymbol{w}_\beta)^\top\boldsymbol{x}_i\boldsymbol{x}_i^\top\boldsymbol{x}_i y_i\\
&= \frac{1}{T}\sum_{i=1}^{T}2(\boldsymbol{w}_\alpha - \boldsymbol{w}_\beta)^\top\boldsymbol{x}_i y_i\frac{1}{T}\sum_{j=1}^{T}\left(\boldsymbol{x}_j^\top\boldsymbol{x}_i\frac{y_j}{y_i} - \boldsymbol{x}_i^\top\boldsymbol{x}_i\right).
\end{aligned} \tag{13}$$

applying the condition 4 in Assumption E.2, we get the final conclusion.

□

F LIMITATIONS

1) Most experimental results are based on a GPT-2 model pretrained on a limited set of mathematical functions. It is challenging to assess whether modern large-scale language models like GPT-4 and Claude 3 Opus face similar difficulties in generalizing beyond their pretraining corpus, given the vast range of tasks and content in their pretraining data. Nevertheless, our findings highlight the limitations of ICL in solving challenging tasks for smaller models like Llama-2-7B and Llama-3-8B. 2) The models are trained on ICL data, while real-world LLMs are trained autoregressively. However, the ICL pretraining objective is also next-token prediction, so there is no clear gap between these two pretraining objectives.

G REPRODUCIBILITY

In the main text and Appendix C, we've stated all setups for reproducing our experimental results. For the theoretical part, we've included the assumptions (Assumption E.2) and proofs in Appendix E.