ERM4ICL: A Synthetic Framework for Understanding In-Context Learning via Empirical Risk Minimization

Anonymous Authors[1]

Abstract

In-context learning (ICL) enables large language models (LLMs), typically Transformers, to adapt to new tasks by conditioning on examples. While prior works leverage diverse synthetic datasets to investigate ICL mechanisms and Transformer properties, those datasets often neglect the critical role of instructions used for real-world LLMs. Regarding instruction as a signal to specify the hypothesis space, we propose an instruction-based ICL framework that formulates the instruction-based ICL as an empirical risk minimization problem: the provided hypothesis space serves as an instruction to guide the model, with in-context examples to identify the correct hypothesis. We perform varied explorations on such frameworks including (i) varied generalization ability to new hypothesis spaces; (ii) different model architectures; (iii) sample complexity; (iv) the effect of imbalanced in-context samples; (v) the benefit of instruction; (vi) the effect of hypothesis diversity. Our results demonstrate the significance of integrating explicit instructions into synthetic ICL, offering new insights into LLM generalization and advance the research on ICL and LLMs.

1. Introduction

LLMs and ICL. Large Language Models (LLMs) (Zhao et al., 2023), often built on Transformer architectures, have garnered widespread attention for their ability to solve complex tasks using simple text prompts. Among their many capabilities, in-context learning (ICL) (Brown et al., 2020) is particularly striking. ICL enables LLMs to adapt to new tasks by conditioning on provided examples, effectively allowing them to learn from context without explicit parameter updates. Although the notion of learning from examples is intuitive for humans, understanding how such behavior emerges in Transformer-based models remains an intriguing and challenging problem.

Synthetic Data for ICL and Transformer. To better understand the mechanisms underlying ICL and to examine Transformer properties, researchers have designed synthetic datasets to isolate and study specific aspects of ICL and Transformer. These datasets are broadly divided into two categories based on how the training sequences are structured. The first category is featured by training sequences unaligned with ICL format, namely "*unaligned sequence*" in our study. In this setting, the training sequence is generated from an underlying graphical model or function (as illustrated on the top-left panel in Fig. 1) without enforcing a specific input-output structure of the ICL format. For example, Xie et al. (2021) employs multiple Hidden Markov Models (HMMs) to represent latent language concepts and explain the ICL phenomenon from a Bayesian perspective, whereas testing sequences are formatted with a pattern such as $(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}, \boldsymbol{o}^{\text{delim}}, \boldsymbol{x}^{(2)}, \boldsymbol{y}^{(2)}, \boldsymbol{o}^{\text{delim}}, \dots)$[1], different from the training pattern. This discrepancy mirrors the gap between the unconstrained nature of the pretraining language data and the structured requirements of ICL testing scenarios encountered in practice. In contrast, the second category constrains training sequences to follow the ICL format, *i.e.*, the input-output structure $(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{y}^{(2)}, \dots)$ as shown on the top-right panel in Fig. 1, namely "*aligned sequence*" in our work. A representative example is provided by Garg et al. (2022), who investigates this framework using pretraining data derived from noiseless linear regression tasks. In detail, each input $\boldsymbol{x}$ is sampled from an isotropic Gaussian distribution $\mathcal{N}(0, \boldsymbol{I}_d)$, and each corresponding output is computed as $y_i = \langle \boldsymbol{x}_i, \boldsymbol{w} \rangle$[2], with the coefficient $\boldsymbol{w}$ also drawn from $\mathcal{N}(0, \boldsymbol{I}_d)$. During ICL inference, the prompt is constructed as $(\boldsymbol{x}^{(1)}, y^{(1)}, \dots, \boldsymbol{x}^{(k)}, y^{(k)}, \boldsymbol{x}_{\text{query}})$, where k demonstrations establish the task—allowing the model to infer the proper output for the query sample $\boldsymbol{x}_{\text{query}}$.

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1] $\boldsymbol{o}^{\text{delim}}$ is regarded as a special token to separate $(\boldsymbol{x}, \boldsymbol{y})$ pairs.

[2] $\boldsymbol{y}$ is usually a scalar rather than a vector because the linear regression task is applied on the function $\boldsymbol{y} = f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$.

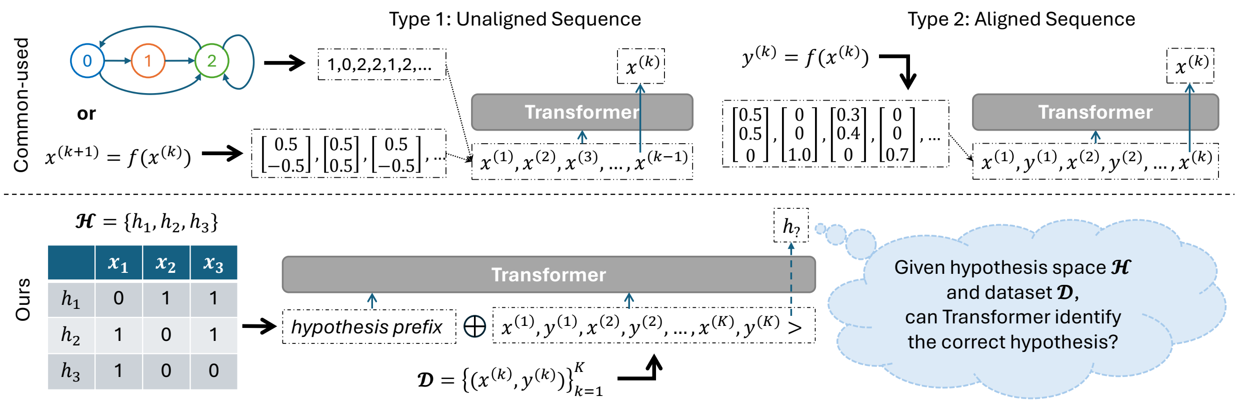

Type 1: Unaligned Sequence
Type 2: Aligned Sequence
Common-used
or
$x^{(k+1)} = f(x^{(k)})$
1,0,2,2,1,2,...
Transformer
$x^{(k)}$
$x^{(1)}, x^{(2)}, x^{(3)}, \dots, x^{(k-1)}$
$y^{(k)} = f(x^{(k)})$
$x^{(1)}, y^{(1)}, x^{(2)}, y^{(2)}, \dots, x^{(k)}$
Ours
$\mathcal{H} = \{h_1, h_2, h_3\}$
hypothesis prefix
$\oplus$
$x^{(1)}, y^{(1)}, x^{(2)}, y^{(2)}, \dots, x^{(K)}, y^{(K)} >$
$h_?$
$\mathcal{D} = \{(x^{(k)}, y^{(k)})\}_{k=1}^K$
Given hypothesis space $\mathcal{H}$ and dataset $\mathcal{D}$, can Transformer identify the correct hypothesis?

Figure 1: **Common-used frameworks *vs.* ours.** Conventional frameworks typically use either unaligned or aligned training sequences, often neglecting the role of instructions. In contrast, our approach explicitly incorporates instructions via a hypothesis prefix. We convert the hypothesis space $\mathcal{H}$ and dataset $\mathcal{D}$ into token sequences, concatenate them, and feed them to a Transformer to predict $\mathcal{D}$'s underlying hypothesis in $\mathcal{H}$. This framework allows us to assess Transformer performance across diverse configurations of $\mathcal{H}$ and $\mathcal{D}$. (TBD: revise figure)

Inspirations from Synthetic Data. Although simple, synthetic datasets have proven invaluable in advancing our understanding of ICL and Transformer behavior. Acting as controlled playgrounds, these datasets enable researchers to systematically explore diverse phenomena while also inspiring a growing number of proposals for synthetic settings. For instance, building on the setting of unaligned sequence in Xie et al. (2021), Han et al. (2023) propose a novel perspective that explains ICL via kernel regression. Akyürek et al. (2024) leverage Probabilistic Finite Automata (PFA) to model language and examine the induction head in Transformers. Makkuva et al. (2024) study how Transformers learn Markov chains. Moreover, based on the setting of *aligned sequence* in Garg et al. (2022), works such as Von Oswald et al. (2023), Ahn et al. (2023), and Fu et al. (2024) demonstrate that trained Transformers can effectively perform gradient descent using in-context samples. Lin & Lee (2024) assume that linear regression tasks are sampled from a Gaussian mixture rather than a standard Gaussian, showing that trained Transformers exhibit two distinct modes—task learning and task retrieval. To mimic the multi-step reasoning process characteristic of Chain-of-Thought (CoT) (Wei et al., 2022), Li et al. (2024a) introduce multi-step training sequences of the form $(\boldsymbol{x}_{\text{input}}^{(1)}, \boldsymbol{y}_{\text{step1}}^{(1)}, \boldsymbol{y}_{\text{step2}}^{(1)}, \boldsymbol{y}_{\text{output}}^{(1)}, \boldsymbol{x}_{\text{input}}^{(2)}, \boldsymbol{y}_{\text{step1}}^{(2)}, \boldsymbol{y}_{\text{step2}}^{(2)}, \boldsymbol{y}_{\text{output}}^{(2)}, \dots)$ to study the effects of CoT. (For additional discussions on these and other interesting studies, please refer to Sec. 2.)

Motivation. While synthetic datasets have significantly advanced our understanding of ICL and Transformers, a crucial gap remains between these datasets and real-world ICL scenarios. In practice, ICL often relies on instructions to guide the learning process, such as "perform sentence sentiment classification on the following sentences based on examples" or "translate the following sentence based on demonstrations." Instructions have been shown to significantly enhance the accuracy of ICL (Brown et al., 2020). However, most existing synthetic frameworks overlook this crucial aspect, neglecting the role of instructions in guiding the learning process. Motivated by this limitation, we ask:

Can we build a synthetic dataset that incorporates instructions and supports diverse explorations?

Notably, two recent works (Xuanyuan et al., 2024; Huang & Ge) adopt prefix correlated with the task to implicitly hint at the task domain. In contrast, our approach explicitly provides a hypothesis space as a prefix to the Transformer, guiding the model's understanding of the intended task.

Our Synthetic Data and ERM. We propose a novel synthetic dataset framework (illustrated on the bottom panel in Fig. 1) that integrates a hypothesis space into the ICL procedure via a hypothesis prefix (more details in Fig. 2 of Sec. 3.1). A set of K samples, $\mathcal{D} = \{(x^{(k)}, y^{(k)})\}_{k=1}^K$, is provided in-context so that the Transformer can identify the hypothesis in the given space that minimizes the empirical risk on $\mathcal{D}$. This setup naturally aligns with the task of empirical risk minimization (ERM). Leveraging this framework, we explore several aspects of Transformer behavior on the ERM task: (i) We evaluate the generalization ability of trained models to new hypothesis spaces, novel hypotheses, and varying sizes of hypothesis spaces; (ii) We compare different model architectures (Transformer, Mamba, LSTM, and GRU), highlighting their distinct properties on these generalization tasks; (iii) We examine the sample complexity required for achieving space and hypothesis generalization, and discover that merely a few dozen training spaces are sufficient for near-perfect generalization. (iv) We examine the effect of imbalanced in-context samples, demonstrating that imbalance can slow down the training process; (v) We assess the benefit of incorporating a hypothesis prefix, which notably enhances the accuracy of ICL; (vi) We show pretraining hypothesis diversity can significantly improve accuracy of ICL when with instruction.

We summarize our contributions as follows:

- We introduce a novel synthetic dataset framework for ICL that explicitly incorporates instructions via a hypothesis prefix. This design casts the ICL task as an empirical risk minimization problem and provides a controlled testbed for studying LLM behavior.
- We perform extensive empirical evaluations on our framework. Most interestingly, we demonstrate that (a) Transformers can generalize to new hypothesis spaces and novel hypotheses even when trained on data from only tens of hypothesis spaces, and (b) compared with ICL without instruction, incorporating an explicit hypothesis prefix as instruction significantly boosts the accuracy of ICL, even on new tasks/hypotheses.

2. Related Works

Unaligned Pretraining Sequence for ICL. We categorize the first type of synthetic pretraining sequences as "unaligned." This type of pretraining sequence has the format $(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \ldots)$, a sequence of tokens generated from a next-token generative model, such as the Hidden Markov Model (HMM) (Xie et al., 2021), Probabilistic Finite Automata (PFA) (Akyürek et al., 2024), or simply a function $\boldsymbol{x}_{i+1} = f(\boldsymbol{x}^{(i)}) + \epsilon$ (Li et al., 2023; Sander et al., 2024). Xie et al. (2021) representing the first category proposes to use multiple HMMs to represent latent concepts in real-world language, and explain the ICL phenomenon via a Bayesian perspective. Han et al. (2023) leverages the same setting and explains ICL via kernel regression. Unaligned sequences are also generated by Akyürek et al. (2024) using Probabilistic Finite Automata (PFAs), Edelman et al. (2024) using Markov Chain, and Nichani et al. (2024) using Markov Chain with causal structure, to study the induction head (Olsson et al., 2022) of LLMs. While the abovementioned datasets only consider single-step dependence between tokens, i.e., the next token only depends on one of the previous tokens, Makkuva et al. (2024) further explore higher-order Markov chains.

Aligned Pretraining Sequence for ICL. The second type is categorized as "aligned." This type of pretraining sequence has the format $(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{y}^{(2)}, \ldots)$[3], a sequence of vectors generated from an underlying regression task. Garg et al. (2022) first examines this setting with pretraining and testing sequences generated from noiseless linear regression tasks. Specifically, Garg et al. (2022) assumes all $\boldsymbol{x}$s are sampled from an isotropic Gaussian distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Then within each sequence, $y^{(i)} = \langle \boldsymbol{w}, \boldsymbol{x}^{(i)} \rangle$, where $\boldsymbol{w}$ is sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$. When performing ICL inference during testing, prompts are designed as $(\boldsymbol{x}^{(1)}, y^{(1)}, \boldsymbol{x}^{(2)}, y^{(2)}, ..., \boldsymbol{x}^{(k)}, y^{(k)}, \boldsymbol{x}_{\text{query}})$, where k pairs of $(\boldsymbol{x}^{(i)}, y^{(i)})$ served as k demonstrations of the linear regression task of coefficient $\boldsymbol{w}$.

Noiseless Linear Regression. Based on the well-defined problem setup by Garg et al. (2022), researchers systematically study the mechanisms of ICL and properties of Transformer. For instance, there is a particular interesting line of research on connecting ICL to gradient descent, firstly hinted by Garg et al. (2022). Akyürek et al. (2023); von Oswald et al. (2023) then show that one attention layer can be exactly constructed to perform gradient descent, and empirically find similarities between in-context inference and gradient descent. Further, Ahn et al. (2023) theoretically show that under certain conditions, Transformers with one or more attention layers trained on noisy linear regression task minimizing the pretraining loss will implement gradient descent algorithm. Further, (Fu et al., 2024) show that Transformers learn to approximate second-order optimization methods for ICL, sharing a similar convergence rate as Iterative Newton's Method. Besides gradient descent, there are lots of other interesting topics on ICL and Transformers based on this linear regression setting, such as looped Transformer (Yang et al., 2024; Gatmiry et al., 2024), training dynamic (Zhang et al., 2024; Huang et al., 2024; Kim & Suzuki, 2024), generalization Panwar et al. (2024), etc.

Noisy Linear Regression. Such a simple noiseless linear regression task is very inspiring and nevertheless it is further extended to varied versions. By extending the linear regression to noisy linear regression—$y = \langle \boldsymbol{x}, \boldsymbol{w} \rangle + \epsilon$, Li et al. (2023) analyze the generalization and stability of ICL. Wu et al. (2024) and Raventós et al. (2024) analyze the effect of task diversity on the attention model's ICL risk. Via extending the noisy linear regression task sampling from Gaussian to Gaussian mixture, Lin & Lee (2024) shows pretrained Transformer exhibits two different modes including task retrieval and learning. With the tasks of $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} + \epsilon$ where $\boldsymbol{W}$ is a matrix rather than a vector, Chen et al. examine the training dynamic of multi-head attention for ICL.

More than Linear Regression. Beyond linear regression, researchers are also interested in non-linear regression and classification. The research directions are scattered, and we

[3]In theoretical papers, the pretraining sequence usually concatenates the paired $\boldsymbol{x}$ and $\boldsymbol{y}$ to the same vector to simplify the analysis. We refer the reader to Wibisono & Wang (2023) for a discussion of this difference.

list them as follows. Bai et al. (2023) show that Transformers can perform in-context algorithm selection, *i.e.*, adaptively selecting different ICL algorithms such as gradient descent, least square, or ridge regression. Bhattamishra et al. (2024) show Transformer can learn a variety of Boolean function classes. Cheng et al. (2024) provide evidence that Transformers can learn to implement gradient descent to enable them to learn non-linear functions. Guo et al. (2024) show that trained Transformer achieves near-optimal ICL performance under $y = \langle \boldsymbol{w}, f(\boldsymbol{x}) \rangle$, where f is a shallow neural network (MLP). Examining linear and non-linear regression tasks, Fan et al. (2024) and Tripuraneni et al. (2024) show Transformer can perform ICL on composited or mixed tasks of pretrained linear or non-linear tasks, and Yadlowsky et al. (2024) examine whether trained Transformers can generalize to new tasks beyond pretraining. Park et al. (2024) examine whether Mamba can in-context learn varied synthetic tasks. Via examining regression and classification tasks, Kim et al. (2024) show task diversity helps shorten the ICL plateau pretraining. Ramesh et al. assume there are multiple functions composited to connect $\boldsymbol{x}$ and $\boldsymbol{y}$ pair, *e.g.*, $\boldsymbol{y} = f_1 \circ f_2 \circ f_3(\boldsymbol{x})$ to study the compositional capabilities of Trasnformer. Li et al. (2024b) study how non-linear Transformer learns binary classification.

Synthetic Dataset with Instruction. As far as the authors know, there are two articles on synthetic datasets with instruction. Huang & Ge append an additional vector $\boldsymbol{\mu}$ to the commonly studied sequences constructed from $(\boldsymbol{x}, \boldsymbol{y})$ pairs, which leads to the sequence with the format $(\boldsymbol{\mu}, \boldsymbol{x}^{(1)}, \boldsymbol{w}^\top \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{w}^\top \boldsymbol{x}^{(2)}, \ldots)$ and show that the trained Transformer can achieve significantly lower loss for ICL when the task descriptor $\boldsymbol{\mu}$ is provided. Xuanyuan et al. (2024) is the work most closely related to us, who develop a new synthetic dataset based on task $((a \cdot x) \circ (b \cdot y)) \bmod p = r$, where (x, y) is the input, r is the output, $\circ$ is an operation $(+, -, /)$, and each task is defined by the parameters $(a, b, \circ)$ (p is a constant). The instruction is further constructed with $(a_l, a_u, b_l, b_u, \circ)$, where a_l and a_u are the lower and upper bounds of a (similar for b), and $\circ$ is the operation. Therefore, the instruction constrains the possible tasks, *i.e.*, provide information on the underlying true task of in-context samples. With such setting, Xuanyuan et al. (2024) study how the information provided by instruction affect the accuracy of ICL.

3. Problem Definition

3.1. Empirical Risk Minimization for Binary Classification

In this work, we consider the binary classification problem where the input space is finite. Formally, let the input space be defined as $\mathcal{X} = \{x_1, x_2, \ldots, x_{|\mathcal{X}|}\}$, and the label space as $\mathcal{Y} = \{0, 1\}$. The learning task is characterized by a hypothesis space $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$, where each hypothesis $h \in \mathcal{H}$ is a deterministic function $h : \mathcal{X} \to \mathcal{Y}$. A realizable instance of the binary classification problem is specified by the tuple $(\mathcal{X}, \mathcal{Y}, P, h^*)$, where P is a probability distribution over the input space $\mathcal{X}$, and $h^* : \mathcal{X} \to \mathcal{Y}$ is the target hypothesis.

During learning, a learner A receives a training dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^K$ and it is tasked to find a hypothesis that produces least generalization risk $R(A(\mathcal{D}))$ on future test sample drawn from P and h^*. Empirical risk minimization (ERM) is one of the most popular learning algorithm. It learns by minimizing the empirical risk on the training dataset $\mathcal{D}$ with respect to a loss function $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ and is given as,

$$\hat{h}_{\mathcal{D}} = \mathrm{ERM}(\mathcal{D}, \mathcal{H}) \leftarrow \min_{h \in \mathcal{H}} \sum_{i=1}^{K} l(h(x^{(i)}), y^{(i)}).$$

In this paper, we focus on $0/1$ risk,

$$l(\hat{y}, y) = \begin{cases} 1, & \text{if } \hat{y} = y \\ 0, & \text{if } \hat{y} \neq y \end{cases}.$$

Remark 3.1. ERM focuses mainly on sample complexity, that is, determining the minimum number of samples required to identify a hypothesis $\hat{h}$ with a risk below a specified threshold. Differently, in this paper, we explore whether Transformer models can accurately pinpoint the exact hypothesis when provided with sufficient number of samples.

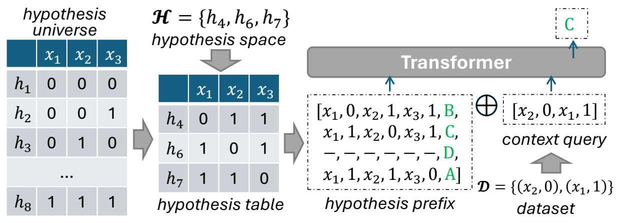


Figure 2: **The framework.** We begin by sampling a subset from the broader hypothesis universe as the hypothesis space $\mathcal{H}$. Next, we encode both the hypothesis space $\mathcal{H}$ and the dataset $\mathcal{D}$ into sequences of tokens. These token sequences are concatenated and fed into a Transformer model. Finally, we assess whether the Transformer can identify the correct underlying hypothesis $h \in \mathcal{H}$ that generated the dataset $\mathcal{D}$.

3.2. Learning ERM In-Context via Transformer

We detailed how Transformer learns ERM in this section.

Definition 3.2 (ERM-Capable)**.** Given a set of inputs $\mathcal{X} = \{x_1, x_2, \ldots, x_{|\mathcal{X}|}\}$ and a hypothesis space $\mathcal{H} = \{h_1, h_2, \ldots, h_m\}$ with m hypotheses, where each hypothesis h_i maps an input $x \in \mathcal{X}$ to binary label $y \in$

$\mathcal{Y} := \{0,1\}$, an algorithm is said to be *ERM-capable* if it can correctly identify the underlying hypothesis given a sufficient set of input-output pairs denoted by $\mathcal{D} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(K)}, y^{(K)})\}$.

We construct the framework as shown in Fig. 2 to examine whether Transformer can be trained to learn this ability. The hypothesis space is first converted to a hypothesis prefix, then concatenated with context query representing the dataset $\mathcal{D}$ with K samples[4]. The Transformer is then asked to predict the index of the underlying hypothesis of the given dataset. We list the pseudo algorithm of training and teating in the Algorithm 1, namely "Meta-ERM Training and Testing Framework" because ERM-capacity is trained and tested on varied hypothesis spaces.

Algorithm 1 Meta-ERM Training and Testing Framework

```
1:  **Inputs:** a set of inputs $\mathcal{X}$, a training set of hypothesis spaces $\mathcal{S}^{\text{train}} = \{\mathcal{H}_i^{\text{train}}\}_{i=1}^{N^{\text{train}}}$, a testing set of hypothesis spaces $\mathcal{S}^{\text{test}} = \{\mathcal{H}_i^{\text{test}}\}_{i=1}^{N^{\text{test}}}$, batch size $B$, hypothesis prefix size $L$, and context query size $K$
2:  **for training epoch do**
3:    sample $\{\mathcal{H}_i\}_{i=1}^{B} \overset{\text{i.i.d.}}{\sim} \text{Uniform}(\mathcal{S}^{\text{train}})$
4:    **for** each hypothesis space $\mathcal{H} \in \{\mathcal{H}_i\}_{i=1}^{B}$ **do**
5:      sample $h \sim \text{Uniform}(\mathcal{H})$
6:      **// Generate dataset $\mathcal{D}$**
7:      sample $\{x^{(i)}\}_{i=1}^{K} \overset{\text{i.i.d.}}{\sim} \text{Uniform}(\mathcal{X})$
8:      construct dataset $\mathcal{D} = \{(x^{(i)}, h(x^{(i)}))\}_{i=1}^{K}$
9:      **// Construct sequence $s$ based on $\mathcal{H}$, $h$, and $\mathcal{D}$**
10:     construct hypothesis prefix, context query, and hypothesis index $z$ based on $\mathcal{H}$, $\mathcal{D}$, $h$, $L$, and $K$
11:     $s \leftarrow \text{concatenate}(\text{hypothesis prefix}, \text{context query}, z)$
12:     **// Cross-entropy loss for next token prediction**
13:     $\mathcal{L} \leftarrow -\sum_{t=2}^{|s|} \log P(s_t \mid s_{<t})$
14:   **end for**
15:   update model parameters using $\mathcal{L}$ of the batch
16: **end for**
17: **for testing epoch do**
18:   sample $\{\mathcal{H}_i\}_{i=1}^{B} \overset{\text{i.i.d.}}{\sim} \text{Uniform}(\mathcal{S}^{\text{test}})$
19:   **for** each hypothesis space $\mathcal{H} \in \{\mathcal{H}_i\}_{i=1}^{B}$ **do**
20:     sample $h \sim \text{Uniform}(\mathcal{H})$
21:     generate dataset $\mathcal{D}$ via:
22:       **either** the same as training
23:       **or** constructing Opt-T set based on $h$ and $\mathcal{H}$
24:     **construct sequence $s$ based on $\mathcal{H}$, $h$, and $\mathcal{D}$**
25:     **evaluate the prediction accuracy on $z$, $y$, etc**
26:   **end for**
27: **end for**
```

During training, $\mathcal{D}$ is constructed by $(x, h(x))$ pairs via uniformly randomly sampling x from $\mathcal{X}$; while during testing, we consider two sample strategies of $(x, h(x))$ pairs in the context query: (i) "i.i.d.": the same as training; (ii) "Opt-T": given a hypothesis space $\mathcal{H}$ and a chosen hypothesis $h \in \mathcal{H}$, an optimal teaching set[5] is derived and duplicated to number K to fit the context query size K.

[4]The dataset $\mathcal{D}$ may contain duplicated samples

Hypothesis Prefix.[6] Given a hypothesis space $\mathcal{H}$, its hypothesis prefix with size L is constructed as shown in Fig. 2. Blank hypothesis is used to fill the hypothesis prefix when $|\mathcal{H}| < L$. A randomly assigned hypothesis index token z is used to label each hypothesis. In the illustrated example in Fig. 2, zs are sampled from $L = 4$ hypothesis index tokens {"A","B","C","D"} without replacement[7].

Context Query. Given a dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{K}$ with K pairs of (x, y), the context query of size K is constructed to represent the dataset and trigger the prediction of the hypothesis index with token ">" as shown in Fig. 2.

With this framework, we are interested in whether a trained Transformer is able to generalized to perform ERM on new hypothesis spaces. Starting with introducing the hypothesis universe, we further deliver the four generalization cases considered in our experiments.

Hypothesis Universe $\mathcal{H}^{\text{uni}}$. Given a set of inputs $\mathcal{X} = \{x_1, \ldots, x_{|\mathcal{X}|}\}$, we define the hypothesis universe $\mathcal{H}^{\text{uni}} = \mathcal{Y}^{\mathcal{X}}$ as the collection of all possible binary classification hypotheses. This universe contains $M = 2^{|\mathcal{X}|}$ distinct hypotheses. For both training and testing purposes, we construct specific hypothesis spaces by sampling subsets from $\mathcal{H}^{\text{uni}}$. The procedures for constructing training and testing hypothesis spaces are detailed in following definitions.

Definition 3.3 (Space Generalization)**.** Given $\mathcal{H}^{\text{uni}}$ of size M, we form all $C(M, m) = \frac{M!}{m!(M-m)!}$ distinct hypothesis spaces, each containing m hypotheses. We then *subsample* these spaces into disjoint training and testing subsets, so that no testing space appears in the training set (though individual hypotheses may overlap), yet we may not use all available spaces. By training on the selected training spaces and evaluating on the unseen testing spaces, we assess generalization to new hypothesis spaces.

Definition 3.4 (Hypothesis Generalization)**.** Given $\mathcal{H}^{\text{uni}}$ of size M, we partition it into disjoint training and testing subsets of sizes M^{Train} and M^{test}, respectively. We then generate hypothesis spaces, each containing m hypotheses, exclusively from the training or testing subsets. Under the hypothesis generalization setting, we train the Transformer

[5]An optimal teaching set is a set with the minimum number of samples to identify a hypothesis from the hypothesis space.

[6]Please refer to appendix B for the full version.

[7]We use variable z to represent the hypothesis index, and create a set of L hypothesis index tokens as a pool from which each hypothesis is randomly assigned a unique index without replacement.

on the training spaces and evaluate on the testing spaces. Because no testing hypothesis appears among the training hypotheses, this setup probes how well the trained Transformer generalizes to entirely new hypotheses.

Definition 3.5 (Space&Size Generalization)**.** Building on the setting of space generalization, while maintaining non-overlapping training and testing spaces, we allow training hypothesis space to include varied number of hypotheses $m \in \mathcal{M} \subset [L]$. We investigate whether the trained Transformer can perform well on hypothesis spaces with varied sizes $m \in [L] \setminus \mathcal{M}$, where $[L] = \{1, 2, \dots, L\}$.

Definition 3.6 (Hypothesis&Size Generalization)**.** Based on the setting of hypothesis generalization, while maintaining non-overlapping training and testing hypotheses, we allow training hypothesis space to include varied number of hypotheses $m \in \mathcal{M} \subset [L]$. We investigate whether the trained Transformer can perform well on hypothesis spaces with varied sizes $m \in [L] \setminus \mathcal{M}$, where $[L] = \{1, 2, \dots, L\}$.

4. Experiments

4.1. Setting of Experiments

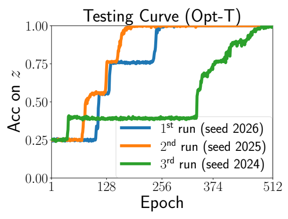
Pretraining. During pretraining, we backpropagate gradients based on next-token prediction for all tokens. Each training sequence s consists of a hypothesis prefix, a context query, and a hypothesis index token. As illustrated in Fig. 2, we feed the entire sequence s (excluding the final index token z) into the Transformer. We then compute cross-entropy loss for each subsequent token (excluding the very first). Refer to Appendix A.1 for Transformer hyperparameters.

Components of Pretraining Loss. We conducted experiments to determine the optimal components to include in the pretraining loss. Specifically, we evaluated four configurations: applying the loss (i) solely to the final hypothesis index token, (ii) exclusively to the content query, (iii) only to the labels (y) of the content query, and (iv) across all tokens. We empirically find that incorporating the loss across all tokens leads to the best performance.

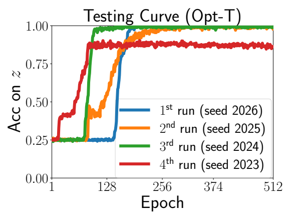
4.2. Four Types of Generalization

This section investigates whether a Transformer trained on ERM tasks can generalize to various hypothesis spaces. We explore four types of generalization scenarios, defined in Definitions 3.3, 3.4, 3.5, and 3.6. Detailed settings for the training and testing hypothesis spaces, hypothesis prefixes, and context queries are provided in Appendix A.2.

We first demonstrate that the Transformer successfully learns ERM and that this capability generalizes effectively to both space generalization and hypothesis generalization, as illustrated in Fig. 3(a) and 3(b). Furthermore, we show that the learned ERM ability extends to hypothesis spaces of varying sizes, as depicted in Fig. 4(a) and 4(b). Across all settings, the Transformer effectively generalizes to unseen hypothesis spaces. Additionally, the accuracy curves exhibit multiple plateaus, with the duration of each plateau varying across different runs.

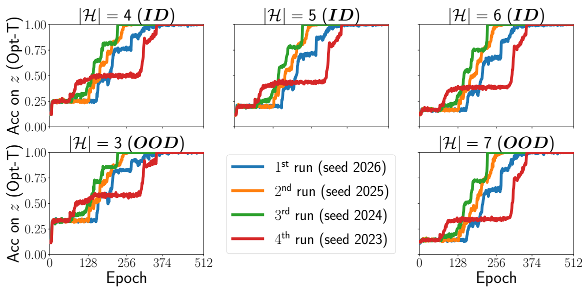


(a) space generalization.

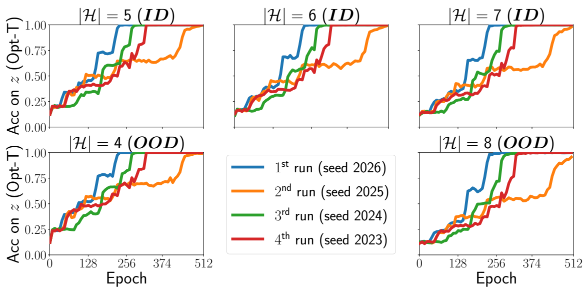


(b) hypothesis generalization.

Figure 3: **Multiple runs on space and hypothesis generalizations.** Transformer successfully learns the ERM ability, and generalize to new spaces and hypotheses. Refer to Appendix C.1, Fig. 12 and 13 for more curves including loss, training accuracy (i.i.d.) and testing accuracy (i.i.d.).



(a) testing curves of space&size generalization.

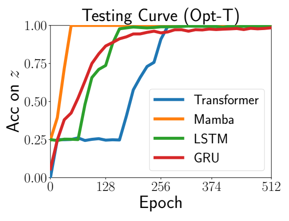


(b) testing curves of hypothesis&size generalization.

Figure 4: **Multiple runs on space&size and hypothesis&size generalizations.** For space&size generalization, the Transformer trained on hypothesis spaces with sizes $m^{\text{train}} \in \{4, 5, 6\}$ (ID, in-distribution) successfully generalizes to hypothesis spaces with sizes $m^{\text{test}} \in \{3, 7\}$ (OOD, out-of-distribution). Similarly, For hypothesis&size generalization, the Transformer trained on hypothesis spaces with sizes $m^{\text{train}} \in \{5, 6, 7\}$ (ID) successfully generalizes to hypothesis spaces with sizes $m^{\text{test}} \in \{4, 8\}$ (OOD).

4.3. Compare with Other Model Architectures

We compare Transformer with other model architectures, including Mamba (Gu & Dao, 2023), LSTM (Hochreiter, 1997), and GRU (Cho et al., 2014). We investigate whether each model can effectively fit the training dataset and, if so, generalize to the four types of unseen hypothesis spaces.


(a) space generalization.

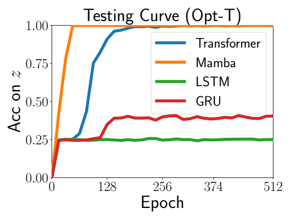
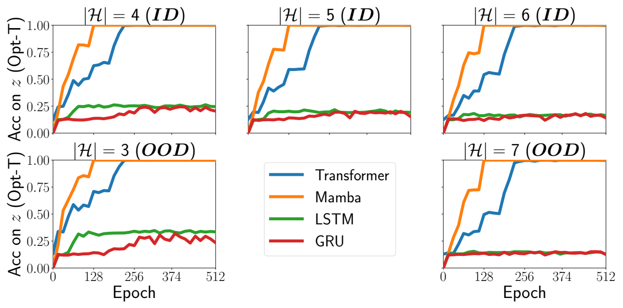

(b) hypothesis generalization.

Figure 5: **Various models on space and hypothesis generalizations.** All models successfully acquire ERM capabilities and achieve strong performance in space generalization. However, under hypothesis generalization, only Transformer and Mamba succeed. Refer to Appendix C.2, Fig. 14 and 15 for more curves including loss, training accuracy (i.i.d.) and testing accuracy (i.i.d.).

We begin by evaluating space generalization and hypothesis generalization across different model architectures. Utilizing the hyperparameter search space detailed in Appendix A.1, we found that all four models—Transformer, Mamba, LSTM, and GRU—generalize well in space generalization tasks, as illustrated in Fig. 5(a). In contrast, during hypothesis generalization tasks, the GRU model was able to fit the training set but failed to generalize to the testing set, while the LSTM model was unable to fit the training set entirely, as shown in Fig. 5(b). Furthermore, in both space&size generalization and hypothesis&size generalization scenarios (see Fig. 6(a) and 6(b)), the LSTM and GRU models were unable to fit the training set. On the other hand, the Mamba model not only generalized effectively across all four generalization setups but also demonstrated faster convergence compared to the Transformer model.

4.4. Effect of Training Space Count on Generalization

We evaluate how the number of training hypothesis spaces affects the space and hypothesis generalization abilities. In Fig. 7(a), we evaluate Mamba, Transformer, GRU, and LSTM, and we observe that with only 2^3 and 2^5 training hypothesis spaces, Mamba and Transformer can achieve nearly perfect space generalization, and LSTM and GRU require more sample to achieve good space generalization and the results have high variance. In Fig. 7(b), since GRU and LSTM fail under hypothesis generalization, we evaluate Mamba and Transformer. Results show that with 2^4 and 2^6 training hypothesis spaces, Mamba and Transformer can achieve nearly their best hypothesis generalization.


(a) testing curves of space&size generalization.

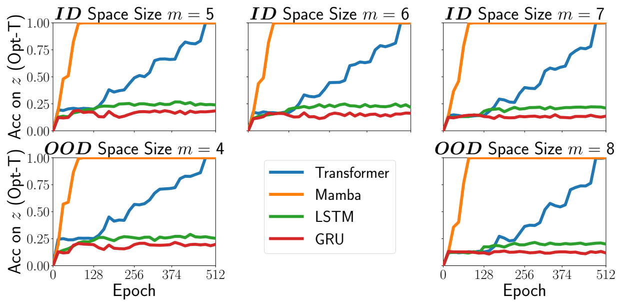
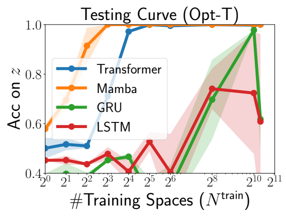

(b) testing curves of hypothesis&size generalization.

Figure 6: **Various models on space&size and hypothesis&size generalization.** In both space&size and hypothesis&size generalization settings, the Transformer and Mamba demonstrate strong performance, whereas LSTM and GRU fail to do so. Additionally, Mamba requires fewer epochs to converge compared to the Transformer. As illustrated in Fig. 16 in Appendix C.2, LSTM and GRU are unable to fit the training data.


(a) space generalization.

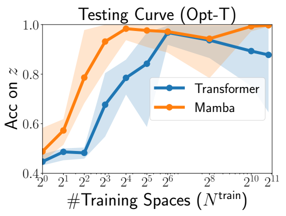

(b) hypothesis generalization.

Figure 7: **Effect of training space count.** Transformer and Mamba's ERM ability can generalize to new hypothesis spaces with only 8 to 16 training hypothesis spaces. Refer to Appendix C.3, Fig. 17 and 18 for more curves including training accuracy (i.i.d.) and testing accuracy (Opt-T).

4.5. Effect of Imbalanced In-Context Samples

In this section, we investigate how an imbalanced sample distribution in the context queries affects training procedure. Specifically, we consider a distribution over $\mathcal{X}$:

$$\text{norm}(\underbrace{\frac{1}{\sqrt{\mathrm{D}}}, \dots, \frac{1}{\sqrt{\mathrm{D}}}}_{\lfloor \frac{|\mathcal{X}|}{2} \rfloor \text{ items}}, \underbrace{1}_{(|\mathcal{X}| \bmod 2) \text{ items}}, \underbrace{\sqrt{\mathrm{D}}, \dots, \sqrt{\mathrm{D}}}_{\lfloor \frac{|\mathcal{X}|}{2} \rfloor \text{ items}}),$$

where D[8] represents the disparity among the distribution over $\mathcal{X}$, *i.e.*, $\mathrm{D} = \frac{\max_{x \in \mathcal{X}} P(x)}{\min_{x \in \mathcal{X}} P(x)}$. We evaluated how the imbalance affects the training procedure in Fig. 8 through varied D values, showing that such an imbalance lags the convergence of the training process.

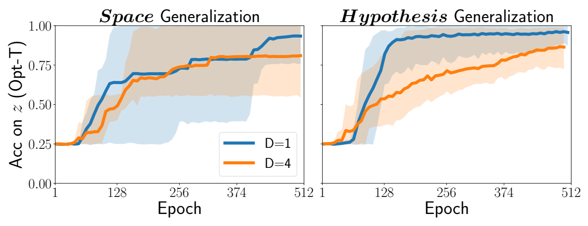


Figure 8: **The effect of sample imbalance.** Sample imbalance leads to lower convergence speed.

4.6. The Benefit of Hypothesis Prefix

In this section, we demonstrate how hypothesis prefix influences the performance of ICL. As shown in Fig. 9, under the space generalization setting, the hypothesis prefix enhances the training and testing accuracy of ICL.

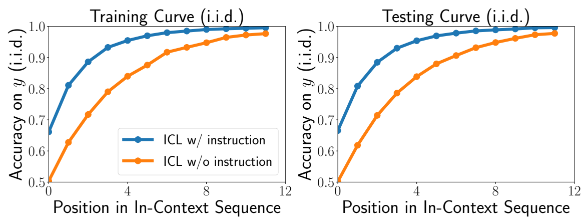


Figure 9: **The effect of instruction.** Under space generalization, instruction (hypothesis prefix) significantly boosts the performance of ICL, especially when k is small.

4.7. The Effect of Pretraining Hypothesis Diversity

In this section, we investigate the impact of hypothesis diversity combined with instruction (hypothesis prefix) on the accuracy of ICL. We conduct experiments with an input space size of $|\mathcal{X}| = 6$, resulting in a hypothesis universe $\mathcal{H}^{\text{uni}}$ comprising $2^{|\mathcal{X}|} = 64$ hypotheses. For training, we sample $M^{\text{train}} \in \{4, 8, 16, 32\}$ hypotheses from $\mathcal{H}^{\text{uni}}$, and for testing, we select $M^{\text{test}} = 16$ hypotheses from the remaining pool. As illustrated in Fig. 10, under the hypothesis generalization setting, the Transformer trained with instructions achieves higher ICL accuracy compared to models trained without instructions. Notably, the testing ICL samples are derived from unseen hypotheses, indicating that incorporating instructions can enhance ICL performance even when dealing with novel hypotheses.

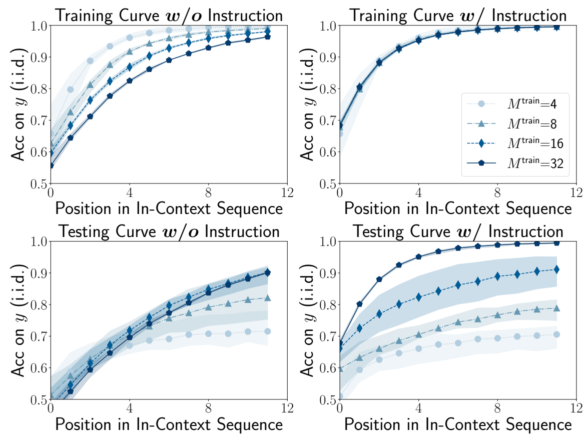


Figure 10: **The effect of hypothesis diversity.** Under hypothesis generalization, increasing the diversity of pretraining hypotheses significantly boosts the performance of ICL when instructions are provided. However, without instructions, this effect is limited.

5. Conclusion

In this paper, we introduced a novel instruction-based in-context learning (ICL) framework that explicitly integrates a hypothesis space as the instruction, effectively framing ICL as an Empirical Risk Minimization (ERM) problem. Through a series of diverse experiments, we demonstrated that trained Transformers can generalize to novel spaces and unseen hypotheses, even when trained on only a few hypothesis spaces. Moreover, we show that the incorporation of explicit instructions significantly enhances the accuracy of ICL, thereby bridging the gap between synthetic ICL studies and real-world applications. We also examined the effect of hypothesis diversity within this framework and found that increased hypothesis diversity substantially improves ICL accuracy when instructions are provided. Our framework serves as a versatile platform for diverse explorations, offering new insights and serving as a playground for research on ICL and large language models (LLMs).

We conclude our paper by acknowledging the limitations of our current framework: (i) Our study is confined to finite

[8]The notation D is distinguished from token "D" by color and dataset $\mathcal{D}$ by format.

hypothesis binary classification problems, which can be extended to more complex scenarios; (ii) The hypothesis prefix is assumed to provide an explicit hypothesis space, differing from the more implicit instructions used in real-world LLM applications.

References

Ahn, K., Cheng, X., Daneshmand, H., and Sra, S. Transformers learn to implement preconditioned gradient descent for in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=LziniAXEI9.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? Investigations with linear models. In *International Conference on Learning Representations (ICLR)*, 2023.

Akyürek, E., Wang, B., Kim, Y., and Andreas, J. In-context language learning: Architectures and algorithms. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=3Z9CRr5srL.

Bai, Y., Chen, F., Wang, H., Xiong, C., and Mei, S. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Bhattamishra, S., Patel, A., Blunsom, P., and Kanade, V. Understanding in-context learning in transformers and llms by learning to learn discrete functions. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, pp. 1877–1901, 2020.

Chen, S., Sheen, H., Wang, T., and Yang, Z. Training dynamics of multi-head softmax attention for in-context learning: Emergence, convergence, and optimality (extended abstract). In *The Thirty Seventh Annual Conference on Learning Theory, June 30 - July 3, 2023, Edmonton, Canada*. PMLR.

Cheng, X., Chen, Y., and Sra, S. Transformers implement functional gradient descent to learn non-linear functions in context. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=ah1BlQcLv4.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Edelman, B. L., Edelman, E., Goel, S., Malach, E., and Tsilivis, N. The evolution of statistical induction heads: In-context learning markov chains. *arXiv preprint arXiv:2402.11004*, 2024.

Fan, Y., Yadlowsky, S., Papailiopoulos, D., and Lee, K. Transformers can learn meta-skills for task generalization in in-context learning. In *NeurIPS 2024 Workshop on Compositional Learning: Perspectives, Methods, and Paths Forward*, 2024. URL https://openreview.net/forum?id=53dFaE1tFd.

Fu, D., Chen, T.-Q., Jia, R., and Sharan, V. Transformers learn to achieve second-order convergence rates for in-context linear regression. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

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

Gatmiry, K., Saunshi, N., Reddi, S. J., Jegelka, S., and Kumar, S. Can looped transformers learn to implement multi-step gradient descent for in-context learning? In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=o8AaRKbP9K.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Guo, T., Hu, W., Mei, S., Wang, H., Xiong, C., Savarese, S., and Bai, Y. How do transformers learn in-context beyond simple functions? A case study on learning with representations. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

Han, C., Wang, Z., Zhao, H., and Ji, H. Explaining emergent in-context learning as kernel regression. *arXiv preprint arXiv:2305.12766*, 2023.

Hochreiter, S. Long short-term memory. *Neural Computation MIT-Press*, 1997.

Huang, R. and Ge, R. Task descriptors help transformers learn linear models in-context. In *ICML 2024 Workshop on In-Context Learning*.

Huang, Y., Cheng, Y., and Liang, Y. In-context convergence of transformers. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, 2024.

Kim, J. and Suzuki, T. Transformers learn nonlinear features in context: Nonconvex mean-field dynamics on the attention landscape. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=xm2lU7tteQ.

Kim, J., Kwon, S., Choi, J. Y., Park, J., Cho, J., Lee, J. D., and Ryu, E. K. Task diversity shortens the icl plateau. *arXiv preprint arXiv:2410.05448*, 2024.

Li, H., Wang, M., Lu, S., Cui, X., and Chen, P.-Y. How do nonlinear transformers acquire generalization-guaranteed cot ability? In *High-dimensional Learning Dynamics 2024: The Emergence of Structure and Reasoning*, 2024a. URL https://openreview.net/forum?id=8pM8IrT6Xo.

Li, H., Wang, M., Lu, S., Cui, X., and Chen, P.-Y. How do nonlinear transformers learn and generalize in in-context learning? In *Forty-first International Conference on Machine Learning*, 2024b. URL https://openreview.net/forum?id=I4HTPws9P6.

Li, Y., Ildiz, M. E., Papailiopoulos, D., and Oymak, S. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, pp. 19565–19594. PMLR, 2023.

Lin, Z. and Lee, K. Dual operating modes of in-context learning. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=5H4nJIGqmK.

Makkuva, A. V., Bondaschi, M., Girish, A., Nagle, A., Jaggi, M., Kim, H., and Gastpar, M. Attention with markov: A framework for principled analysis of transformers via markov chains. *CoRR*, 2024.

Nichani, E., Damian, A., and Lee, J. D. How transformers learn causal structure with gradient descent. *arXiv preprint arXiv:2402.14735*, 2024.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Panwar, M., Ahuja, K., and Goyal, N. In-context learning through the bayesian prism. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=HX5ujdsSon.

Park, J., Park, J., Xiong, Z., Lee, N., Cho, J., Oymak, S., Lee, K., and Papailiopoulos, D. Can mamba learn how to learn? a comparative study on in-context learning tasks. In *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2024. URL https://openreview.net/forum?id=xvr0Hctddy.

Ramesh, R., Lubana, E. S., Khona, M., Dick, R. P., and Tanaka, H. Compositional capabilities of autoregressive transformers: A study on synthetic, interpretable tasks. In *Forty-first International Conference on Machine Learning*.

Raventós, A., Paul, M., Chen, F., and Ganguli, S. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. *Advances in Neural Information Processing Systems*, 36, 2024.

Sander, M. E., Giryes, R., Suzuki, T., Blondel, M., and Peyré, G. How do transformers perform in-context autoregressive learning ? In *Forty-first International Conference on Machine Learning*, 2024.

Tripuraneni, N., Doshi, L., and Yadlowsky, S. Can transformers in-context learn task mixtures? In *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models*, 2024. URL https://openreview.net/forum?id=HpY9tkX3Ui.

von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning (ICML)*, 2023.

Von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pp. 35151–35174. PMLR, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Wibisono, K. C. and Wang, Y. On the role of unstructured training data in transformers' in-context learning capabilities. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023. URL https://openreview.net/forum?id=aBeZ3jid9i.

Wu, J., Zou, D., Chen, Z., Braverman, V., Gu, Q., and Bartlett, P. L. How many pretraining tasks are needed for in-context learning of linear regression? In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

Xuanyuan, M., Yang, T., Fu, J., and Wang, Y. On task description of in-context learning: A study from information perspective, 2024. URL https://openreview.net/forum?id=TFR0GrzERG.

Yadlowsky, S., Doshi, L., and Tripuraneni, N. Can transformer models generalize via in-context learning beyond pretraining data? In *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models*, 2024. URL https://openreview.net/forum?id=oW7oQHas3m.

Yang, L., Lee, K., Nowak, R. D., and Papailiopoulos, D. Looped transformers are better at learning learning algorithms. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

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

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

A. Experiment Setup

A.1. Hyperparameters

We list the hyperparameter searching spaces used for Transformer, LSTM, GRU, and Mamba. The best hyperparameter is searched using the setting of Space Generalization with $\|\mathcal{X}\| = 4$, and then used for all other settings.

Table 1: **Hyperparamer searching spaces for different model architectures.** The optimal hyperparameters are bolded if multiple possibilities are provided.

Model Architecture	#layers	#hidden dimensions	#learning rate	#weight decay	#batch size
Transformer	2,**8**	128	0.00001, **0.00002**, 0.00005, 0.00010	0.0005	16
Mamba	**2**,8	128	0.00020, **0.00050**, 0.00100, 0.00200	0.0005	16
GRU	**2**,8	128	0.00020, 0.00050, **0.00100**, 0.00200	0.0005	16
LSTM	**2**,8	128	0.00020, 0.00050, **0.00100**, 0.00200	0.0005	16

A.2. Setup of Generalization

We list the experimental setup in the following Table 2.

Table 2: **Experimental setup of different generalization.**

Generalization Setup	Space	Hypothesis	Space&Size	Hypothesis&Size
size of input space ($\|\mathcal{X}\|$)	4	5	5	6
size of label space ($\|\mathcal{Y}\|$)	2	2	2	2
size of training hypothesis spaces ($m^{\text{train}} = \|\mathcal{H}^{\text{train}}\|$)	4	4	4,5,6	5,6,7
size of testing hypothesis spaces (($m^{\text{test}} = \|\mathcal{H}^{\text{test}}\|$)	4	4	3,4,5,6,7	4,5,6,7,8
size of hypothesis prefix (L)	4	4	8	8
size of context query (K)	4	5	5	6
#all hypotheses (M)	$2^4 = 16$	$2^5 = 32$	$2^5 = 32$	$2^6 = 64$
#training hypotheses (M^{train})	/	16	/	32
#testing hypotheses (M^{test})	/	16	/	32
#total hypothesis spaces	C(16,4)	/	C(32,3∼7)	/
#total training hypothesis spaces	/	C(16,4)	/	C(32,5∼7)
#total testing hypothesis spaces	/	C(16,4)	/	C(32,4∼8)
#sampled training hypothesis space (N^{train})	1274	1820	3000 for all	3000 for all
#sampled testing hypothesis space (N^{test})	546	1820	1500 for all	1500 for all

B. Implementation Detail of Hypothesis Prefix

Hypothesis Prefix Given a hypothesis space $\mathcal{H}$ and its hypothesis table, the correspongding hypothesis prefix with hypothesis prefix's content length L is constructed as shown in Fig. 11. The token "P" serves as the padding token to separate hypotheses, the token ";" serves as the separation token to separate (x, y) pairs, the token "N" serves as the separation token to separate (x, y) pairs, and the token ">" is used to connect (x, y) pairs of the hypothesis to a randomly assigned hypothesis index z[9]. In the illustrated example in Fig. 11, the randomly assigned indexes zs are sampled from $M = 4$ hypothesis index tokens {"A","B","C","D"} without replacement[10].

[9]We use variable z to represent the hypothesis index.

[10]A set of L hypothesis index tokens are created serve as the pool from which the hypothesis indexes are randomly sampled without replacement.

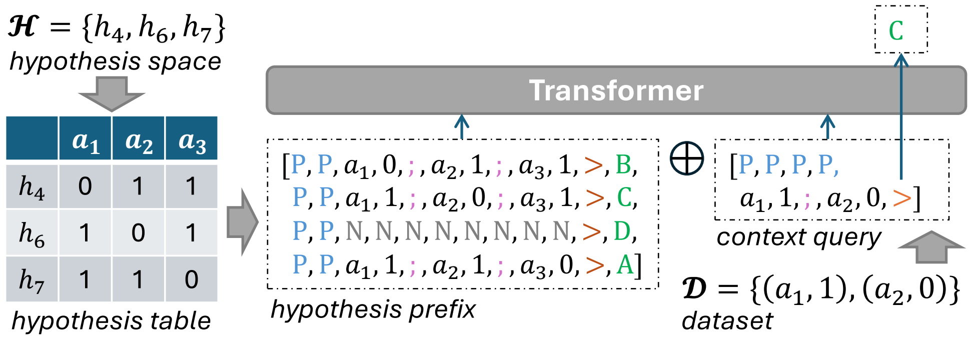

$\mathcal{H} = \{h_4, h_6, h_7\}$
hypothesis space
a_1 a_2 a_3
h_4 0 1 1
h_6 1 0 1
h_7 1 1 0
hypothesis table
Transformer
C
[P, P, a_1, 0, ;, a_2, 1, ;, a_3, 1, >, B,
P, P, a_1, 1, ;, a_2, 0, ;, a_3, 1, >, C,
P, P, N, N, N, N, N, N, N, N, >, D,
P, P, a_1, 1, ;, a_2, 1, ;, a_3, 0, >, A]
hypothesis prefix
$\oplus$
[P, P, P, P, a_1, 1, ;, a_2, 0, >]
context query
$\mathcal{D} = \{(a_1, 1), (a_2, 0)\}$
dataset

Figure 11: **The framework.** We convert hypothesis space $\mathcal{H}$ and dataset $\mathcal{D}$ to sequences of tokens, concatenate them and input to Transformer. Then we examine whether Transformer can identify the correct underlying $h \in \mathcal{H}$ of $\mathcal{D}$.

C. Additional Details of Experiments

C.1. Four Types of Generalization

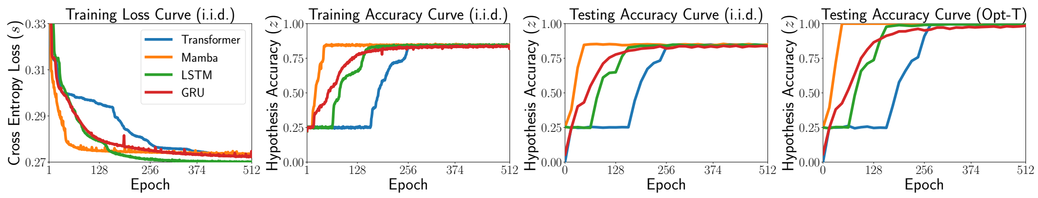


Figure 12: **Multiple runs for space generalization.**

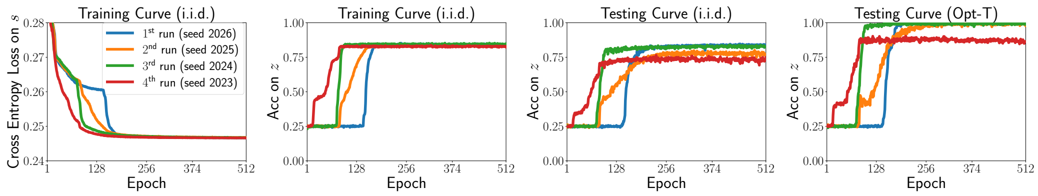


Figure 13: **Multiple runs for hypothesis generalization.**

C.2. Compare with Other Model Architectures

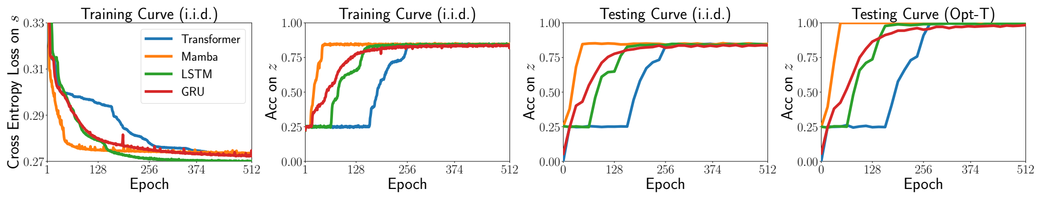


Figure 14: **Varied models on space generalization.**

C.3. Effect of Training Space Count on Generalization

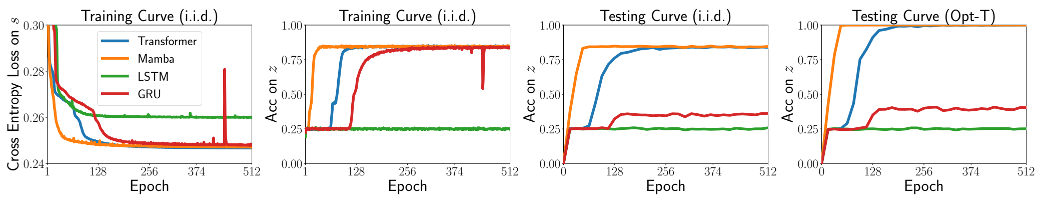
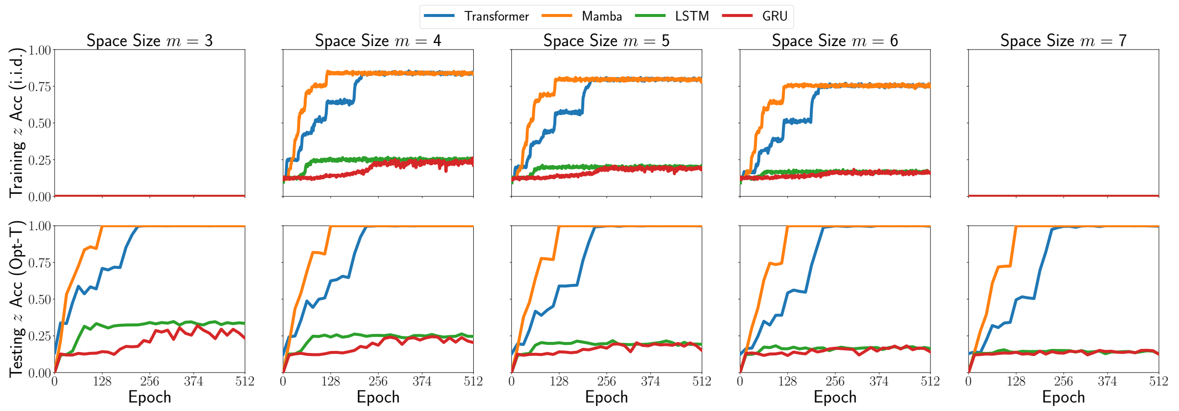
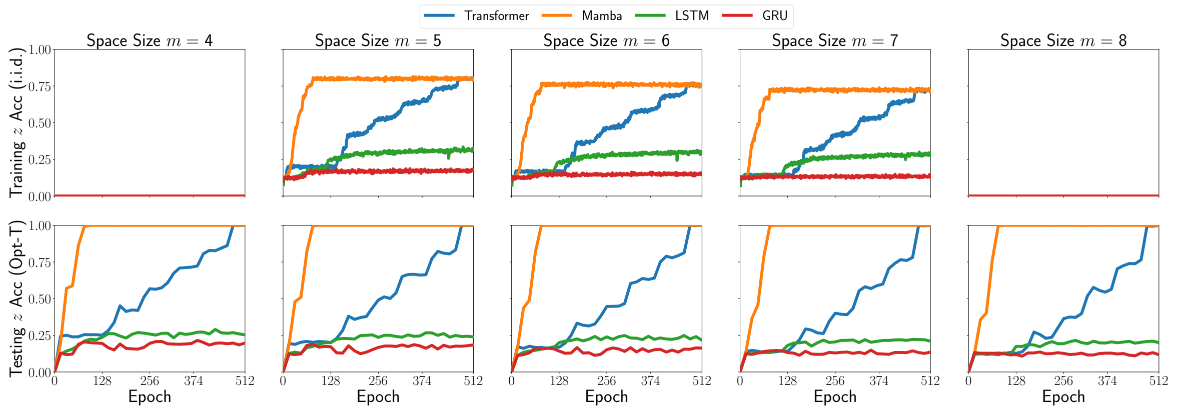


Figure 15: **Varied models on hypothesis generalization.**



(a) testing curves of space&size generalization.



(b) testing curves of hypothesis&size generalization.

Figure 16: **Varied models on space&size generalization, and hypothesis&size generalization.** For both settings of generalization, Transformer and Mamba perform well while LSTM and GRU fails. Mamba needs fewer epoch to converge. LSTM and GRU fail to fit training data.

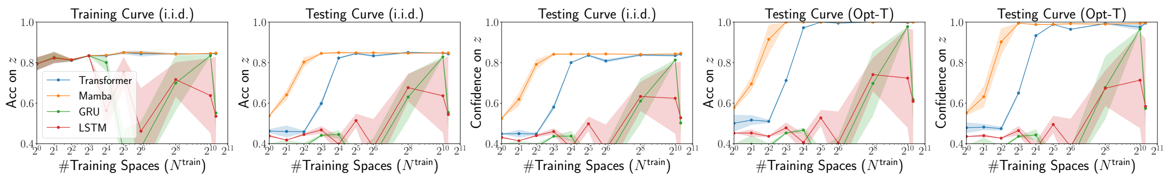


Figure 17: **Effect of training-space count on space generalization.**

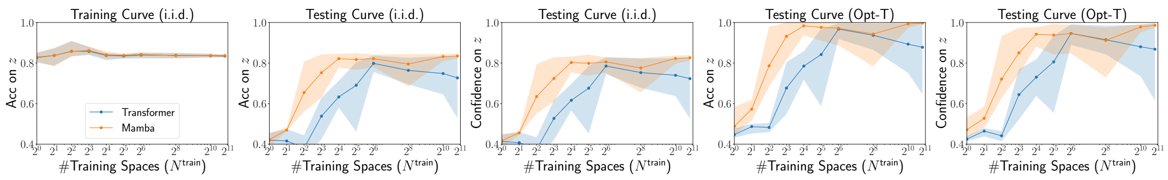


Figure 18: **Effect of training-space count on hypothesis generalization.**