Task Generalization With AutoRegressive Compositional Structure: Can Learning From D Tasks Generalize to D^T Tasks?

Amirhesam Abedsoltan [* 1]   Huaqing Zhang [* 2]   Kaiyue Wen [3]   Hongzhou Lin [4]   Jingzhao Zhang [2]
Mikhail Belkin [1 5]

Abstract

Large language models (LLMs) exhibit remarkable task generalization, solving tasks they were never explicitly trained on with only a few demonstrations. This raises a fundamental question: When can learning from a small set of tasks generalize to a large task family? In this paper, we investigate task generalization through the lens of autoregressive compositional structure, where each task is a composition of T operations, and each operation is among a finite family of D subtasks. This yields a total class of size D^T. We first show that generalization to all D^T tasks is theoretically achievable by training on only $\tilde{O}(D)$ tasks. Empirically, we demonstrate that Transformers achieve such exponential task generalization on sparse parity functions via In-context Learning (ICL) and chain-of-thought (CoT) reasoning. We further show generalization in arithmetic and translation, beyond parity functions.

1. Introduction

Large language models (LLMs) demonstrate a remarkable ability to solve tasks they were never explicitly trained on. Unlike classical supervised learning, which typically assumes that the test data distribution follows the training data distribution, LLMs can generalize to new task distributions with just a few demonstrations—a phenomenon known as in-context learning (ICL) (Brown et al., 2020; Wei et al., 2022; Garg et al., 2022). Recent studies suggest that trained Transformers implement algorithmic learners capable of solving various statistical tasks—such as linear regression—at in-

Figure 1. We train a Transformer to learn parity functions through In-Context Learning (ICL): given a demonstration sequence $(\boldsymbol{x}_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_n, f(\boldsymbol{x}_n))$, infer the target $f(\boldsymbol{x}_{\text{query}})$ from a new input $\boldsymbol{x}_{\text{query}}$. Each function f defines a distinct learning task. In this prototype experiment, tasks are sampled from the parity function family $Parity(10, 2)$ with secret length $k = 2$ and bit length $d = 10$, totaling 45 tasks. To evaluate task generalization, we withhold a subset of tasks and train only on different subset of the remaining ones. Consistent with prior work (Bhattamishra et al., 2024), we observe that standard ICL fails to generalize across tasks. In contrast, incorporating Chain-of-Thought (CoT) reasoning significantly improves performance on unseen tasks.

ference time in context (Li et al., 2023; Bai et al., 2023). Despite their success in tasks such as learning conjunctions or linear regression, Transformers relying solely on ICL struggle with more complex problems, particularly those requiring hierarchical reasoning.

A notable case where Transformers struggle with ICL is the learning of parity functions, as examined in (Bhattamishra et al., 2024). In this setting, a Transformer is provided with a sequence of demonstrations $(\boldsymbol{x}_1, f(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_n, f(\boldsymbol{x}_n))$ and is required to predict $f(\boldsymbol{x}_{\text{query}})$ for a new input $\boldsymbol{x}_{\text{query}}$. Specifically, they focused on parity functions from the class $Parity(10, 2)$, where each function is defined by a secret key of length $k = 2$ within a length space of $d = 10$. Each function f corresponds to a distinct learning task, resulting in 45 possible tasks. To assess generalization, a subset of tasks was held out during training. Their results demonstrate that Transformers trained via ICL fail to generalize

*Equal contribution [1]Department of Computer Science and Engineering, UC San Diego. [2]Institute for Interdisciplinary Information Sciences, Tsinghua University. [3]Stanford University. [4]Amazon. This work is independent of and outside of the work at Amazon. [5]Halicioglu Data Science Institute, UC San Diego. Correspondence to: Amirhesam Abedsoltan <aabedsoltan@ucsd.edu>.
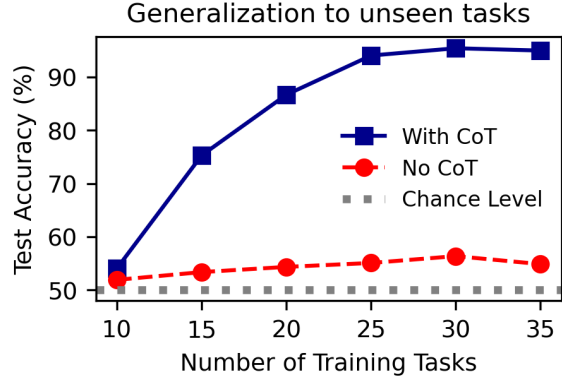
1

to unseen tasks, even when the new tasks require only a simple XOR operation. These findings, along with other empirical studies (An et al., 2023; Xu et al., 2024), suggest that standard ICL struggles with tasks requiring hierarchical or compositional reasoning.

In contrast, we found that incorporating Chain-of-Thought (CoT) reasoning—introducing intermediate reasoning steps to the model—allows Transformers to easily generalize to unseen tasks, as illustrated in Figure 1. Consistent with (Bhattamishra et al., 2024), we observe that Transformers without CoT perform only slightly better than chance level, no matter how many training tasks are presented to the model. However, as the number of training tasks increases, Transformers with CoT achieve near-perfect generalization on the held-out set of unseen tasks. We see that the extra information provided by CoT enables the model to exploit the compositional structure of the parity problem.

Motivated by this example, we aim to systematically analyze how models can leverage autoregressive compositional structures to extend their capabilities beyond the training tasks. Conventionally, learning involves approximating a target function f^* drawn from a function class $\mathcal{F}$ using examples from a training distribution over the input space $\mathcal{X}$; generalization is then measured by testing f^* on new examples. In contrast, our focus is on **task generalization**, where training is restricted to a subset of functions or "tasks" $\mathcal{F}_{\text{train}} \subset \mathcal{F}$, leaving the remaining functions, unseen during training. Our goal is to investigate whether a model trained on tasks from $\mathcal{F}_{\text{train}}$ (with inputs from $\mathcal{X}$) can generalize to *all tasks*, including *unseen* tasks.

This notion of task generalization goes beyond the standard out-of-distribution (OOD) settings (see, e.g., (Zhou et al., 2022) for review) by shifting the focus from adapting to new input distributions to learning entirely new tasks. Specifically, we ask:

How can we quantify the number of tasks a model must be trained on to generalize to the entire class $\mathcal{F}$?

To analyze task generalization, we consider a finite set of functions $\mathcal{F}$, where each function maps an input $\boldsymbol{x} \in \mathcal{X}$ to a tuple of random variables $\boldsymbol{y} = (y_1, \ldots, y_T)$. We assume each function can be characterized by a parameter tuple $\theta = (\theta_1, \theta_2, \ldots, \theta_T)$. The outputs are generated autoregressively: first, y_1 is produced from $\boldsymbol{x}$; then y_2 is generated from $\boldsymbol{x}$ and y_1; and then y_3 is generated from $\boldsymbol{x}$, y_1 and y_2; and this process continues until y_T is produced. Specifically, the sequence is generated sequentially as:

$$y_t \sim P_{\theta_t}(y_t \mid \boldsymbol{x}, \boldsymbol{y}_{<t}), \quad \text{for } t = 1, \ldots, T,$$

where $\boldsymbol{y}_{<t} = (y_1, \ldots, y_{t-1})$ denotes the previously generated outputs, and P_{θ_t} is some conditional probability distribution that is parametrized by θ_t and is conditioned on $\boldsymbol{y}_{<t}$

and $\boldsymbol{x}$. This structure can also be interpreted as a sequence of compositions,

$$\boldsymbol{x} \xrightarrow{P_{\theta_1}} y_1$$
$$\boldsymbol{x}, y_1 \xrightarrow{P_{\theta_2}} y_2$$
$$\cdots$$
$$\boldsymbol{x}, y_1, \ldots, y_{T-1} \xrightarrow{P_{\theta_{T-1}}} y_T \ .$$

We will call this function class *AutoRegressive Compositional structure*. Assuming that the cardinality of the set of possible values for each parameter θ_t is finite and is equal to D, we will use the notation $\mathcal{F} = ARC(T, D)$. The cardinality of this class is D^T.

For the sparse parity problem with k secret keys in this framework, the output sequence has length $T = k$. Given an input $\boldsymbol{x} \in \mathcal{X} = \{0, 1\}^n$, let the secret keys correspond to indices $i_1, i_2, \ldots, i_k$ (in a predetermined order). The output sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_k)$ is defined as follows,

$$y_1 = x_{i_1}, \ y_2 = x_{i_1} \oplus x_{i_2}, \ \ldots, \ y_k = x_{i_1} \oplus x_{i_2} \oplus \cdots \oplus x_{i_k}.$$

That is, each y_t recovers the XOR of the first t secret coordinates. In this example, the output distribution at each step is deterministic, assigning probability 1 to the correct XOR value and 0 to all other values.

We can now address the following fundamental question:

How many tasks in $\mathcal{F}_{\text{train}}$ must a model be trained on to generalize to all tasks in $\mathcal{F}$, including those it has not seen? In particular, can a model trained on $\tilde{O}(D)$ tasks generalize across the entire set of D^T tasks?

Our main contributions are:

- We define AutoRegressive Compositional structure and introduce a framework to quantitatively analyze task generalization when the function class follows an AutoRegressive Compositional structure. (Sections 3.2 and 3.3)

- We establish that under this structure, task generalization to all D^T tasks is theoretically achievable by training on $\tilde{O}(D)$ tasks up to logarithmic terms (Section 3.4).

- We demonstrate how the parity problem aligns with our framework and empirically show that Transformers trained on i.i.d. sampled tasks exhibit exponential task generalization via chain-of-thought (CoT) reasoning, consistent with theoretical scaling (Section 4).

- Finally, we show that the selection of training tasks significantly impacts generalization to unseen tasks. If tasks are chosen adversarially, training on even nearly all D^T of the tasks with CoT may fail to generalize to the remaining tasks (Section 5.1).

2. Related Works

2.1. Composition and Generalization

The role of composition in reasoning for language models has been widely studied. (Saparov et al., 2023) explores various out-of-distribution (OOD) generalization formats, including compositional generalization, showing that a neural network's ability to generalize compositionally is highly dependent on both architecture and task properties. Similar conclusions have been drawn in prior works (Lake & Baroni, 2018; Keysers et al., 2020). Further, (Bhattamishra et al., 2024; Dziri et al., 2023; An et al., 2023; Xu et al., 2024) examine compositional generalization in in-context learning (ICL) and find that generalization to composing multiple steps is in general hard for LLMs. One notable observation is that LLMs succeed in compositional generalization for clause satisfaction problems but not for parity problems.

Another line of research investigates composition as a mechanism underlying emergent abilities in language models. (Arora & Goyal, 2023) demonstrates that language modeling can lead to learning tuples of skills, which are small compositions of fundamental capabilities. Building on this idea, (Kaur et al., 2025; Zhao et al., 2024) leverage compositional structures to generate supervised fine-tuning (SFT) data, leading to improved language model performance.

Beyond sequential composition, other forms of compositionality in neural networks have been explored. (Song et al., 2025) investigates layer-wise composition in transformers, while (Schug et al., 2024) proposes a modular neural architecture for learning hidden compositional representations. Additionally, (Wiedemer et al., 2024) examines compositional structures in image reconstruction, and (Lippl & Stachenfeld, 2025) provides a theoretical analysis of composition in kernel and linear models.

While prior work has largely focused on qualitative insights into compositional generalization, our work takes a quantitative approach: studying how many training tasks are needed to achieve task generalization over an entire function class.

2.2. Learning and Testing with Multiple Distributions

Our work aims to analyze generalization when the training and testing distributions differ. This problem has been studied from various perspectives in the statistical learning community. One approach is to frame it as learning a shared representation across multiple tasks. (Ye et al., 2021) defines variation and informativeness between different environments based on a common representation, while (Arjovsky et al., 2019) addresses the problem by designing specific training objectives. Earlier studies on linear and kernel models also explore this direction (Du et al., 2017; Lei et al., 2021).

Another perspective considers the testing environment as a distribution shift, where the model may sample during inference to achieve domain adaptation. (Mansour et al., 2009) analyzes generalization error when a model is trained on distribution P but tested on a different distribution Q, introducing an error bias dependent on the distance $d(P, Q)$. To mitigate this bias, (Cortes et al., 2010) proposes reweighting training samples when test samples from Q are available.

A related line of research investigates scenarios where both training and test samples are accessible. Notable setups include covariate shift (Kpotufe & Martinet, 2021; Ma et al., 2023) and domain adaptation (Sugiyama et al., 2007; Ben-David & Urner, 2014). When direct sampling from the test distribution is not feasible, alternative strategies focus on training robustly against worst-case shifts. This can be achieved through adversarial perturbations or min-max optimization formulations (Madry et al., 2018; Raghunathan et al., 2020; Duchi et al., 2023).

In this work we impose an AutoRegressive Compositional(ARC) structure on the function class and propose a new framework to study task generalization. This compositional structure decomposes the function class into atomic subtasks, enabling a modular approach to learning. By leveraging this structure, we establish a quantitative understanding of how many training tasks are required for generalization. Our results provide a theoretical foundation for structured learning and demonstrate how models can efficiently generalize beyond the training distribution.

3. Theoretical Framework for Task Generalization

In this section, we present a theoretical framework to study task generalization with autoregressive compositional structure. When the compositional structure holds, we show that there exists a learning algorithm that is only trained on $\tilde{O}(D)$ different tasks, but can generalize to exponentially many unseen tasks.

3.1. Preliminaries and Notations

For a positive integer n, denote $[n] = \{1, 2, \cdots, n\}$. For a finte set $\mathcal{S}$, we denote by $\Delta(\mathcal{S})$ the probability simplex over with support $\mathcal{S}$. Given t sets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_t$, their **Cartesian product** is defined as

$$\underset{i=1}{\overset{t}{\times}} \mathcal{S}_i := \{(s_1, s_2, \ldots, s_t) \mid s_i \in \mathcal{S}_i \text{ for all } i \in [t]\}.$$

We further denote $\mathcal{S}^t := \times_{i=1}^{t} \mathcal{S}$. For two probability distributions P and Q over a discrete space $\mathcal{S}$, the **total variation distance** is defined as

$$\mathrm{TV}(P, Q) := \frac{1}{2} \sum_{s \in \mathcal{S}} |P(s) - Q(s)|.$$

We let the bold letter $\boldsymbol{y}$ denote a sequence, and the subscripted $\boldsymbol{y}^j$ denote the j_{th} sequence / example. Within each sequence $\boldsymbol{y}^j = (y_1, ..., y_T)$, the regular letter y_t denote the t_{th} token in the sequence.

3.2. AutoRegressive Compositional Structure

In the following definition, we formally introduce the **AutoRegressive Compositional (ARC)** task class, which models structured sequence generation through a composition of conditional distributions.

Definition 3.1. *(AutoRegressive Compositional task class).* Let $\mathcal{X}$ and $\mathcal{Y}$ denote the finite input and output spaces, respectively. The AutoRegressive Compositional (ARC) task class consists of sequential generation processes:

$$\mathcal{F} := \{f_\theta = (P_{\theta_1}, \ldots, P_{\theta_T}) \mid P_{\theta_t} \in \mathcal{P}_{\Theta_t} \text{ for all } t \in [T]\},$$

where each task $f_\theta \in \mathcal{F}$ for any input $\boldsymbol{x} \in \mathcal{X}$ generates an output sequence $\boldsymbol{y} = (y_1, \ldots, y_T) \in \mathcal{Y}$ through an autoregressive sampling process:

$$y_t \sim P_{\theta_t}(\cdot \mid \boldsymbol{x}, \boldsymbol{y}_{<t}), \quad \text{for all } t \in [T].$$

At each step t, the conditional probability distribution P_{θ_t} is drawn from a subtask family $\mathcal{P}_{\Theta_t}$, parametrized by θ_t:

$$\mathcal{P}_{\Theta_t} := \{P_{\theta_t}(\cdot \mid \boldsymbol{x}, \boldsymbol{y}_{<t}) : \mathcal{X} \times \mathcal{Y}^{t-1} \to \Delta(\mathcal{Y}) \mid \theta_t \in \Theta_t\}.$$

Here, Θ_t represents the parameter space at step t, and the overall task parameter space is $\Theta := \bigtimes_{t=1}^T \Theta_t$. Assuming each step has a finite number of possible subtasks, i.e., $|\Theta_t| = d$ for all $t \in [T]$, the AutoRegressive Compositional task class $ARC(d, T)$ consists of $|\mathcal{F}| = |\Theta| = d^T$ tasks.

Given any input $\boldsymbol{x} \in \mathcal{X}$ and a sequence $\boldsymbol{y} \in \mathcal{Y}^T$, the joint distribution for a task $f_\theta = (P_{\theta_1}, \cdots, P_{\theta_T}) \in \mathcal{F}$ is:

$$P_\theta(\boldsymbol{x}, \boldsymbol{y}) = P(\boldsymbol{x}) \prod_{s=1}^T P_{\theta_s}(y_s \mid \boldsymbol{x}, \boldsymbol{y}_{<s}), \quad (1)$$

and for partial sequences up to any $t \in [T]$:

$$P_{\theta_{1:t}}(\boldsymbol{x}, \boldsymbol{y}_{1:t}) = P_x(\boldsymbol{x}) \prod_{s=1}^t P_{\theta_s}(y_s \mid \boldsymbol{x}, \boldsymbol{y}_{<s}). \quad (2)$$

At a high level, an AutoRegressive Compositional task class $ARC(D, T)$ is characterized by two key properties:

- **Modularity.** The generation process is decomposed into T sequential steps, each governed by an independent conditional distribution $P_{\theta_t} \in \mathcal{P}_{\Theta_t}$. This modular structure allows tasks to be constructed by combining different components at each step.

- **Exponential Growth.** The task class size grows exponentially in T as $|\mathcal{F}| = D^T$, despite each step having only D choices. This reflects the combinatorial nature of task construction, where variations at each step lead to an exponentially large set of possible tasks.

3.3. Task Generalization

Under the autoregressive task learning setup, there are two levels of generalization:

1. Generalizing to unseen inputs within a task.

2. Generalizing to unseen tasks in the class $\mathcal{F}$.

We focus on the latter one, referred as **task generalization**.

Training Phase During training, the model can only access to a small subset of tasks $\mathcal{F}_{\text{train}} = \{f_{\theta^1}, \ldots, f_{\theta^{n_\theta}}\} \subseteq \mathcal{F}$ with $n_\theta = |\mathcal{F}_{\text{train}}|$. For each task $f_{\theta^i} \in \mathcal{F}_{\text{train}}$, we observe n_x i.i.d. demonstration samples:

$$\mathcal{D}_i = \{(\boldsymbol{x}^{i,j}, \boldsymbol{y}^{i,j})\}_{j=1}^{n_x} \overset{\text{i.i.d.}}{\sim} P_{\theta^i}(\boldsymbol{x}, \boldsymbol{y}),$$

where P_{θ^i} is defined as in Equation (1). The full training dataset is the union of $\mathcal{D}_i$ denoted by $\mathcal{D}_{\text{train}} = \{\mathcal{D}_i\}_{i=1}^{n_\theta}$.

We assume the learner does not know the true subtask conditional distribution families $\{\mathcal{P}_{\Theta_t}\}_{t=1}^T$ a priori. Instead, it accesses to a **larger hypothesis class**:

$$\mathcal{P}_{\Xi_t} := \{P_{\zeta_t}(\cdot \mid \boldsymbol{x}, \boldsymbol{y}_{<t}) \mid \zeta_t \in \Xi_t\} \supseteq \mathcal{P}_{\Theta_t},$$

where Ξ_t parameterizes the learner's model class at step t. The goal of training is to *identify the true subtask families* $\mathcal{P}_{\Theta_t}$ from $\mathcal{P}_{\Xi_t}$ through $\mathcal{D}_{\text{train}}$.

Inference Phase At test time, the learner is given ℓ inference-time demonstration samples:

$$\mathcal{D}_{\text{infer}} = \{(\tilde{\boldsymbol{x}}^i, \tilde{\boldsymbol{y}}^i)\}_{i=1}^\ell \overset{\text{i.i.d.}}{\sim} P_{\tilde{\theta}}(\boldsymbol{x}, \boldsymbol{y}),$$

where $f_{\tilde{\theta}} \in \mathcal{F}$ is an unseen task. The learner must identify the true conditionals $\{P_{\tilde{\theta}_t}\}_{t=1}^T$ from $\mathcal{P}_{\Theta_t}$ for each step t. Formally, the learner $\mathcal{A}$ that is trained on $\mathcal{D}_{\text{train}}$ and given $\mathcal{D}_{\text{infer}}$ as input, produces an output sequence of conditional distributions:

$$\mathcal{A}(\mathcal{D}_{\text{infer}}; \mathcal{D}_{\text{train}}) \in \{(P_{\xi_1}, \cdots, P_{\xi_T}) \mid P_{\xi_t} \in \mathcal{P}_{\Xi_t}, t \in [T]\}.$$

3.4. Main Result: Exponential Task Generalization

We now establish our main theoretical result: with the compositional structure in Definition 3.1, a learner can achieve exponential task generalization with only $\tilde{O}(D)$ training tasks. This demonstrates how compositional structure fundamentally reduces the sample complexity of task learning from exponential to polynomial in D. Our results hold under the following mild assumptions:

Assumption 3.2 (Compositional Identifiability)**.** The autoregressive task class $\mathcal{F}$ satisfies:

1. **Finite Subtask Families.** For each $t \in [T]$, the hypothesis class $\mathcal{P}_{\Xi_t}$ is finite and the subtask conditional distribution family $\mathcal{P}_{\Theta_t} \subseteq \mathcal{P}_{\Xi_t}$ has size $|\mathcal{P}_{\Theta_t}| = D$.

2. **Task Identifiability.** For any $t \in [T]$, $\theta_{1:t-1} \in \bigtimes_{s=1}^{t-1} \Theta_s$, and $\theta_t \in \Theta_t$, $\zeta_t \in \Xi_t$, $P_{\zeta_t} \neq P_{\theta_t}$, the induced distributions stasify:

$$\mathrm{TV}\left(P_{\theta_{1:t-1},\theta_t}, P_{\theta_{1:t-1},\zeta_t}\right) > 0.$$

Furthermore, for any $t \in [T]$, $\theta_{1:t-1} \in \bigtimes_{s=1}^{t-1} \Theta_s$, and $\theta_t \neq \theta'_t \in \Theta_t$, the induced distributions satisfy:

$$\mathrm{TV}\left(P_{\theta_{1:t-1},\theta_t}, P_{\theta_{1:t-1},\theta'_t}\right) \geq c > 0.$$

Under these conditions, we establish our main theorem:

Theorem 3.3 (**Exponential Task Generalization**). *Let $\mathcal{F}$ be an AutoRegressive Compositional(ARC) task class satisfying Assumption 3.2. Then there exists a learner $\mathcal{A}$ with the following property: if during training, one samples $n_\theta \geq D \ln(100\, D\, T)$ tasks uniformly and independently from $\mathcal{F}$, each provided with n_x i.i.d. demonstration samples as the training dataset, and if at inference one observes $\ell \geq \frac{2\ln\left(100\, T\, n_\theta\right)}{c^2}$ i.i.d. demonstration samples from a previously unseen task $P_{\tilde{\theta}} \in \mathcal{F}$, then*

$$\lim_{n_x \to \infty} \Pr\left[\mathcal{A}\left(\mathcal{D}_{\mathrm{infer}}; \mathcal{D}_{\mathrm{train}}\right) \neq \left(P_{\tilde{\theta}_1}, \ldots, P_{\tilde{\theta}_T}\right)\right] \leq 0.02,$$

where $\mathcal{D}_{\mathrm{train}}$ and $\mathcal{D}_{\mathrm{infer}}$ denote the training dataset and inference-time demonstration samples respectively, and the probability is taken over the random selection of training tasks $\mathcal{F}_{\mathrm{train}} \subseteq \mathcal{F}$, the training data $\mathcal{D}_{\mathrm{train}}$, and the inference-time demonstration samples $\mathcal{D}_{\mathrm{infer}}$.

In other words, Theorem 3.3 shows that the learner can generalize from only $\tilde{O}(D)$ tasks to an exponentially large number of unseen tasks, on the order of D^T. The learning algorithm $\mathcal{A}$ operates in two stage. In the training stage, it applies a maximum-likelihood estimation (MLE) procedure to $\mathcal{D}_i$ in order to identify the subtasks of the i-th training task $(P_{\theta_1^i}, \cdots, P_{\theta_T^i})$. In the inference stage, it then uses a total-variation-based distribution discrimination test (Lemma A.1) on inference-time demonstration samples $\mathcal{D}_{\mathrm{infer}}$ to recover $f_{\tilde{\theta}} = (P_{\tilde{\theta}_1}, \cdots, P_{\tilde{\theta}_T})$. The proof is deferred to Appendix A.1.

Remark 3.4. If we additionally assume that every subtask distribution is separated from any incorrect hypothesis by a fixed total-variation margin, i.e. for all $t \in [T]$, $\theta_{1:t-1} \in \bigtimes_{s=1}^{t-1} \Theta_s$, and $\theta_t \in \Theta_t$, $\zeta_t \in \Xi_t$ with $P_{\zeta_t} \neq P_{\theta_t}$,

$$\mathrm{TV}\left(P_{\theta_{1:t-1},\theta_t},\, P_{\theta_{1:t-1},\zeta_t}\right) \geq r > 0,$$

then one can replace the MLE procedure used in the training stage with the same distribution-discrimination approach from the inference stage (Lemma A.1). Under this condition, we can derive a *non-asymptotic* bound on the n_x needed per task for accurate identification. See Appendix B for details.

3.5. Example: Sparse Parity Problem

To illustrate the role of the AutoRegressive Compositional structure, we use sparse parity problem as an example.

Sparse Parity Problem. Given d binary variables $\boldsymbol{x} = (b_1, b_2, \ldots, b_d)$, a sparse parity function selects k secret indices $S = \{i_1, i_2, \ldots, i_k\}$ and outputs 1 if the sum of the corresponding variables is odd, and 0 otherwise:

$$\mathrm{parity}_S(b_1, b_2, \ldots, b_d) = b_{i_1} \oplus b_{i_2} \oplus \cdots \oplus b_{i_k},$$

where $\oplus$ denotes the XOR (exclusive OR) operation. We define $Parity(d, k)$ as the set of all parity functions with d variables and k secret indices, yielding a total of

$$|Parity(d, k)| = \binom{d}{k} = O(d^k).$$

Representation Matters. Without Chain-of-Thought (CoT), the sparse parity problem $Parity(d, k)$ is of $ARC(\binom{d}{k}, 1)$, but with CoT, it becomes an autoregressive compositional structure of $ARC(d, k)$.

- No CoT $\to ARC\left(\binom{d}{k}, 1\right)$.

- With CoT $\to ARC(d, k)$.

Indeed, without CoT, the model maps input $\boldsymbol{x} = (b_1, \ldots, b_d)$ directly to output $y = b_{i_1} \oplus b_{i_2} \oplus \cdots \oplus b_{i_k}$ in a single step, hence we have length $T = 1$ and the breadth $D = |Parity(d, k)| = O(d^k)$. Such exponential dependency on the breadth suggests that one would need to train on $O(d^k)$ tasks, explaining what we are observing unsuccessful task generalization in the introduction figure.

In contrast, with CoT (Abbe et al., 2024; Wen et al., 2025), the parity computation is decomposed into small steps:

$$\boldsymbol{y} = (b_{i_1}, b_{i_1} \oplus b_{i_2}, \ldots, b_{i_1} \oplus \cdots \oplus b_{i_k}),$$

enabling a structured representation that reduces the breadth. More precisely, at step t, the class of subtask is exactly defined by the XOR operation with previous token and one secret index :

$$P_{(t, i_t)}(y_t | \boldsymbol{x}, \boldsymbol{y}_{<t}) = \mathbf{1}[y_t = y_{t-1} \oplus b_{i_t}].$$

where i_t is the t-th secret index, by default lying $\in [1, d]$. Therefore, the breadth D at each step is exactly d.

This said, the CoT effectively reduces the breadth from $O(d^k)$ to d. According to Theorem 3.3,

Corollary 3.5. *For the sparse parity problem described above, we can show that the parameter c in Assumption 3.2 is $\frac{1}{2}$, thus when $n_\theta \geq d \ln(100kd), \ell \geq 8 \ln(100k n_\theta)$, it holds that*

$$\lim_{n_x \to \infty} \Pr\left[\mathcal{A}\left(\mathcal{D}_{\text{infer}}; \mathcal{D}_{\text{train}}\right) \neq \left(P_{(1,i_1)}, \cdots, P_{(k,i_k)}\right)\right] \leq 0.02.$$

In other words, the family of $Parity(d, k)$ with CoT is learnable with $O(d \log(d))$ training tasks.

4. Experiments: Parity Problem Case Study

As we have shown, there exists a learning algorithm that is only trained on $\tilde{O}(d)$ different tasks to fully generalize on all the tasks in $Parity(d, k)$. However, from a practical standpoint, it is not clear whether a *Transformer* can actually match this task complexity. In this section, we present empirical evidence demonstrating that a standard Transformer can indeed learn the sparse parity function with CoT using $\tilde{O}(d)$ training tasks. The paper's GitHub repository can be found online.

4.1. Experimental Setup: In-Context Learning

Our empirical setup is a refinement of the theoretical framework presented in Section 3.3, and closely follows that of (Garg et al., 2022; Bhattamishra et al., 2024). In this setup, a sequence model M (such as Transformers) is trained using N sequences, each sequence consisting of m demonstration samples $(\boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{y}_m)$. The model is trained for the next token-prediction task, except that we only consider y_j in loss optimization: for each context $(\boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{x}_{j-1}, \boldsymbol{y}_{j-1}, \boldsymbol{x}_j)$, the model predicts $\hat{\boldsymbol{y}}_j$, and the loss is given by $\frac{1}{m} \sum_{j=1}^m \ell(\hat{\boldsymbol{y}}_j, \boldsymbol{y}_j)$. In our experiment, we use cross-entropy loss to measure the discrepancy between the predicted output $\hat{\boldsymbol{y}}_j$ and the true output $\boldsymbol{y}_j$.

When Chain of Thought (CoT) is used, each y_j is itself a sequence of length k representing intermediate reasoning steps. In this case, the loss is the average on all these intermediate steps.

Training Data Generation. We split both the task space and input space into training and testing. In other words, the parity tasks are split into $\mathcal{F}_{\text{train}}$ and $\mathcal{F}_{\text{test}}$; and the binary sequences are split into $\mathcal{X}_{\text{train}}$ and $\mathcal{X}_{\text{test}}$. The split in the input space helps us monitor the in-distribution training process while as the split in the task space aims to measure generalization to unseen parity functions.

To construct the ICL data, we sample m points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ uniformly at random from $\mathcal{X}_{\text{train}}$. Similarly, we sample a

function f uniformly at random from $\mathcal{F}_{\text{train}}$, and generate the sequence $(\boldsymbol{x}_1, f(\boldsymbol{x}_1), \ldots, \boldsymbol{x}_m, f(\boldsymbol{x}_m))$. This process is repeated N times, each time with a fresh sample of m points and a new function f.

Evaluating Task Generalization. For evaluation, we sample f randomly from the held-out tasks $\mathcal{F}_{test}$ and sample inputs uniformly at random from $\mathcal{X}_{\text{test}}$. We report the accuracy of the prediction $f(\boldsymbol{x}_m)$ given the demonstration $(\boldsymbol{x}_1, f(\boldsymbol{x}_1), \ldots, \boldsymbol{x}_{m-1}, f(\boldsymbol{x}_{m-1}), \boldsymbol{x}_m)$. This setting challenges the model to generalize to novel tasks beyond those encountered during training.

4.2. Experimental Results

As discussed in Section 3.5, introducing CoT transforms the parity problem class $Parity(d, k)$ into an AutoRegressive Compositional structure $ARC(D, T)$ with $D = d$ and $T = k$. A key empirical question is:

how does the number of training tasks scale with d and T to achieve a fixed target accuracy on unseen task sets?

To investigate this, we conduct experiments on:

1. Scaling $T(= k)$, i.e. the length of secret indices.

2. Scaling $D(= d)$, i.e. the ambient dimension of input.

Scaling T for a Fixed D. We examine how the number of training tasks affects test accuracy while keeping the ambient dimension fixed at $d = 15$. Specifically, we evaluate test accuracy for $k = \{3, 4, 5, 6, 7\}$ under varying numbers of training tasks. As k increases, the size of the parity class grows significantly—from approximately 500 for $k = 3$ to around 6500 for $k = 7$.

Remarkably, despite this increase, the test accuracy follows a similar trajectory. With just $3d \ln(d) \approx 122$ training tasks, the model generalizes to unseen cases with high accuracy ($>95\%$). For $k = 7$, this means training on 122 tasks enables generalization to about 6,400 unseen ones! This empirical results suggests that the required number of training tasks remains roughly the same, regardless of k, consistent with the theoretical scaling of $\tilde{O}(d)$ tasks.

Scaling D for a fixed T. We examine the effect of increasing $d \in \{10, 15, 20\}$ while keeping $k = 3$ fixed. For each d, we train on a total number of tasks proportional to $d \ln(d)$, up to $4 \times d \ln(d)$. Figure 2, Panel B, shows similar task generalization performance across different ambien dimension of d, providing further evidence that $\tilde{O}(d)$ i.i.d. training tasks are sufficient for generalization to unseen tasks on praity functions with CoT.

Scaling D and T together. We examine the effect of jointly increasing d and k, with $(d, k) \in \{(10, 5), (15, 7), (20, 10), (25, 12), (30, 15)\}$. For each d,
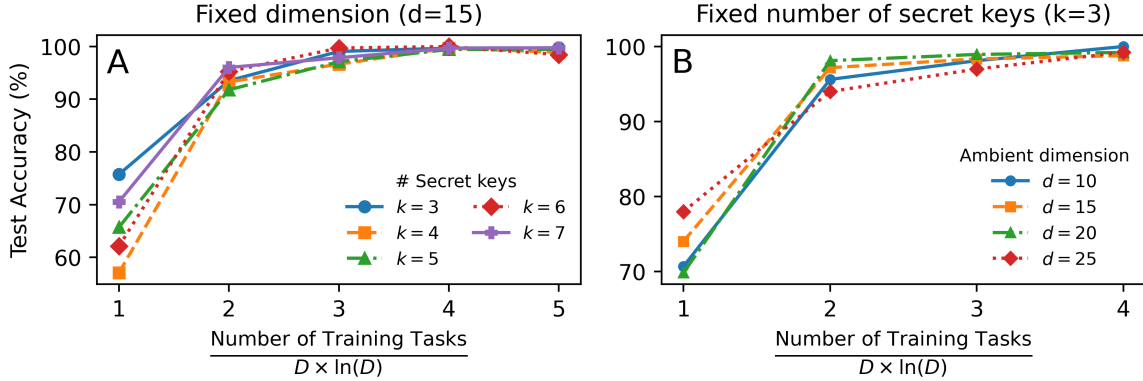
Figure 2. Test accuracy on unseen tasks. For parity task: $D = d$ as the ambient dimension and $T = k$ as the number of secret indices. We show that the empirical scaling closely follows the theoretical scaling of $D \ln(D)$. **(A)** For a fixed $D = 15$, as T increases, the test accuracy on unseen tasks remains similar, even though the total number of tasks ($\sim D^T$) grows exponentially with T. **(B)** For a fixed secret length is 3, as D increases, the number of tasks grows polynomially with D, yet the number of tasks required to generalize reasonably to unseen tasks remains in $\propto D \log D$.

the model is trained on a total of $3 \times d \ln(d)$ i.i.d. tasks. Table 1 shows that generalization performance remains consistent across these settings, providing further evidence that $\tilde{O}(d)$ training tasks are sufficient to generalize to unseen parity tasks using CoT prompting. As a concrete example, for $d = 30, k = 15$, training on just 306 tasks enables generalization to approximately 155 million unseen tasks.

d	k	#TrainingTasks	#TotalTasks	Accuracy (%)
10	5	69	252	98.51
15	7	121	6,400	99.12
20	10	180	185,000	98.67
25	12	241	3,200,000	98.60
30	15	306	155,000,000	98.10

Table 1. Task generalization performance as d and k increase. Training on only $\tilde{O}(d)$ tasks enables generalization to exponentially many unseen tasks in the parity function family.

Subtask Identification via Linear Probing. Finally, we probe the hidden representations of the Transformer (see (Alain & Bengio, 2017)) to see if it *identifies and then executes subtasks* at the inference time, consistent with the framework of Section 3. Specifically, we add a linear classifier to the final attention layer's hidden state when producing the i-th token in the Chain-of-Thought, aiming to predict the i-th secret index. Only the linear classifier is trained, while all Transformer parameters remain frozen. Table 2 shows that for $d \in \{10, 15, 20\}$, $k = 3$, the linear probe consistently achieves high accuracy at all the coordinates. This suggests that at each CoT step, the model indeed first infers the relevant subtask (the secret index) from the in-context examples and then executes that subtask to generate the output token—an ability it acquires during training. Further experimental details appear in Appendix D.

d	Token 1	Token 2	Token 3
10	100.00%	99.83%	91.08%
15	95.42%	99.97%	98.73%
20	97.38%	95.54%	91.92%

Table 2. Validation accuracy (%) of linear probes trained to predict each secret token position from the final hidden state.

5. Task Generalization Beyond i.i.d. Sampling and Parity Functions

In this section, we extend our experiments beyond i.i.d. task sampling and parity functions. We show an adversarial example where biased task selection substantially hinders task generalization for sparse parity problem. In addition, we demonstrate that exponential task scaling extends to a non-parity tasks including arithmetic and multi-step language translation.

5.1. Task Generalization Beyond i.i.d. Task Sampling

In previous sections, we focused on *i.i.d. settings*, where the set of training tasks $\mathcal{F}_{train}$ were sampled uniformly at random from the entire class $\mathcal{F}$. Here, we explore scenarios that deliberately break this uniformity to examine the effect of task selection on out-of-distribution (OOD) generalization.

How does the selection of training tasks influence a model's ability to generalize to unseen tasks? Can we predict which setups are more prone to failure?

To investigate this, we consider two cases parity problems with $d = 10$ and $k = 3$, where each task is represented by its tuple of secret indices (s_1, s_2, s_3):

1. **Generalization with a Missing Coordinate.** In this

setup, we exclude all training tasks where the second coordinate takes the value $s_2 = 5$, such as $(1, 5, 7)$. At test time, we evaluate whether the model can generalize to unseen tasks where $s_2 = 5$ appears.

2. **Generalization with Missing Pair.** Here, we remove all training tasks that contain both 4 *and* 6 in the tuple (s_1, s_2, s_3), such as $(2, 4, 6)$ and $(4, 5, 6)$. At test time, we assess whether the model can generalize to tasks where both 4 and 6 appear together.

If you had to guess. Which scenario is more challenging for generalization to unseen tasks? We provide the experimental result in Table 3.

In the first scenario, despite being trained on all tasks except those where $s_2 = 5$, which is of size $O(D^T)$, the model struggles to generalize to these excluded cases, with prediction at chance level. This is intriguing as one may expect model to generalize across position. The failure suggests that positional diversity plays a crucial role in the task generalization of Transformers.

In contrast, in the second scenario, though the model has never seen tasks with both 4 *and* 6 together, it has encountered individual instances where 4 appears in the second position (e.g., $(1, 4, 5)$) or where 6 appears in the third position (e.g., $(2, 3, 6)$). This exposure appears to facilitate generalization to test cases where both 4 *and* 6 are present.

As a result, when the training tasks are not i.i.d, an adversarial selection such as exclusion of specific positional configurations may lead to failure to unseen task generalization even though the size of $\mathcal{F}_{train}$ is exponentially large.

5.2. Task Generalization Beyond Parity Problems

5.2.1. ARITHMETIC TASK

We introduce the family of *Arithmetic* task that, like the sparse parity problem, operates on d binary inputs $b_1, b_2, \ldots, b_d$. The task involves computing a structured arithmetic expression over these inputs using a sequence of addition and multiplication operations.

Formally, we define the function:

$$\text{Arithmetic}_S \colon \{0, 1\}^d \to \{0, 1, \ldots, d\},$$

where $S = (\text{op}_1, \text{op}_2, \ldots, \text{op}_{d-1})$ is a sequence of $d - 1$ operations, each op_k chosen from $\{+, \times\}$. The function evaluates the expression by applying the operations sequentially from left-to-right order: for example, if $S = (+, \times, +)$, then the arithmetic function would compute $\text{Arithmetic}_S(b_1, b_2, b_3, b_4) = ((b_1 + b_2) \times b_3) + b_4$.

By introducing a step-by-step CoT, arithmetic class belongs to $ARC(2, d-1)$: this is because at every step, there is only

$D = |\Theta_t| = 2$ choices (either $+$ or $\times$) while the length is $T = d - 1$, resulting a total number of 2^{d-1} tasks.

Task generalization for the arithmetic task with CoT. It has $d = 2$ and $T = d - 1$ as the ambient dimension, hence $D \ln(DT) = 2 \ln(2T)$. We show that the empirical scaling closely follows the theoretical scaling.
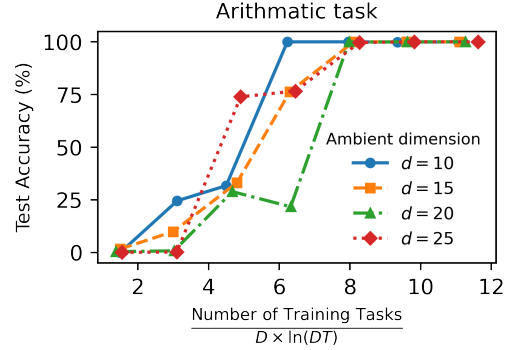
Notably, when scaling with T, we observe in the figure above that the task scaling closely follow the theoretical $O(D \log(DT))$ dependency. Given that the function class grows exponentially as 2^T, it is truly remarkable that training on only a few hundred tasks enables generalization to an exponentially larger space—on the order of $2^{25} > 33$ Million tasks. This exponential scaling highlights the efficiency of structured learning, where a modest number of training examples can yield vast generalization capability.

5.2.2. MULTI-STEP LANGUAGE TRANSLATION TASK

In this task, we study a sequential translation process across multiple languages (Garg et al., 2022). Given a set of D languages, we construct a translation chain by randomly sampling a sequence of T languages **with replacement**: $L_1, L_2, \ldots, L_T$, where each L_t is a sampled language. Starting with a word, we iteratively translate it through the sequence:

$$L_1 \to L_2 \to L_3 \to \cdots \to L_T.$$

For example, if the sampled sequence is EN $\to$ FR $\to$ DE $\to$ FR, translating the word "butterfly" follows:

butterfly $\to$ papillon $\to$ schmetterling $\to$ papillon.

This task follows an *AutoRegressive Compositional* structure by itself, specifically $ARC(D, T - 1)$, where at each step, the conditional generation only depends on the target language, making D as the number of languages and the total number of possible tasks is D^{T-1}. This example illustrates that autoregressive compositional structures naturally arise in real-world languages, even without explicit CoT.

We examine task scaling along D (number of languages) and T (sequence length). As shown in Figure 4, empirical D-scaling closely follows the theoretical $O(D \ln DT)$. However, in the T-scaling case, we observe a linear dependency on T rather than the logarithmic dependency $O(\ln T)$.

Table 3. Generalization Results for Scenarios 1 and 2 for $d = 10, k = 3$.

Scenario	Tasks excluded from training	Generalization accuracy
Generalization with Missing Pair	$\{4, 6\} \subseteq \{s_1, s_2, s_3\}$	96.2%
Generalization with Missing Coordinate	$s_2 = 5$	45.6%
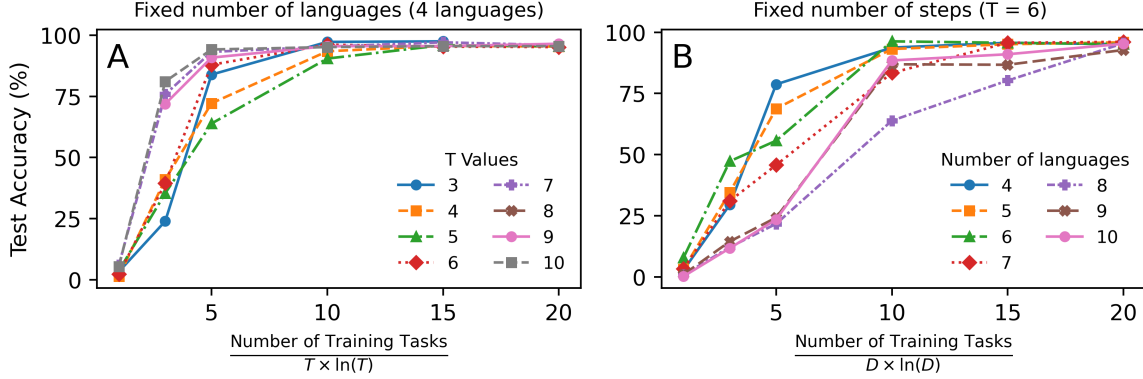
Figure 4. Task generalization for language translation task: D is the number of languages and T is the length of steps.

A possible explanation is error accumulation across sequential steps—longer sequences require higher precision in intermediate steps to maintain accuracy. This contrasts with our theoretical analysis, which focuses on asymptotic scaling and does not explicitly account for compounding errors in finite-sample settings.

Despite this, the task scaling is still remarkable — training on a few hundred tasks enables generalization to $4^{10} \approx 10^6$ tasks!

6. Conclusions

In this work, we quantitatively investigated task generalization under the autoregressive compositional structure, demonstrating both theoretically and empirically that exponential task generalization to D^T tasks can be achieved with training on only $\tilde{O}(D)$ tasks. To summarize:

- **Theoretical Framework for Task Generalization.** We introduced the *AutoRegressive Compositional* (ARC) framework to model structured task learning, demonstrating that a model trained on only $\tilde{O}(D)$ tasks can generalize to an exponentially large space of D^T tasks.

- **Formal Sample Complexity Bound.** We established a fundamental scaling law that quantifies the number of tasks required for generalization, proving that exponential generalization is theoretically achievable with only a logarithmic increase in training samples.

- **Empirical Validation on Parity Functions.** We showed that Transformers struggle with standard in-context learning (ICL) on parity tasks but achieve exponential generalization when Chain-of-Thought

(CoT) reasoning is introduced. Our results provide the first empirical demonstration of structured learning enabling efficient generalization in this setting.

- **Scaling Laws in Arithmetic and Language Translation.** Extending beyond parity functions, we demonstrated that the same compositional principles hold for arithmetic operations and multi-step language translation, confirming that structured learning significantly reduces the task complexity required for generalization.

- **Impact of Training Task Selection.** We analyzed how different task selection strategies affect generalization, showing that adversarially chosen training tasks can hinder generalization, while diverse training distributions promote robust learning across unseen tasks.

We introduce a framework for studying the role of compositionality in learning tasks and how this structure can significantly enhance generalization to unseen tasks. Additionally, we provide empirical evidence on learning tasks, such as the parity problem, demonstrating that transformers follow the scaling behavior predicted by our compositionality-based theory. Future research will explore how these principles extend to real-world applications such as program synthesis, mathematical reasoning, and decision-making tasks.

By establishing a principled framework for task generalization, our work advances the understanding of how models can learn efficiently beyond supervised training and adapt to new task distributions. We hope these insights will inspire further research into the mechanisms underlying task generalization and compositional generalization.

Acknowledgements

Impact Statement

This paper advances the theoretical understanding of task generalization, investigating how learning on a small subset of tasks enables generalization to a much larger set of unseen tasks. Our study is primarily theoretical, supported by synthetic experiments designed to isolate key structural properties of task composition. By analyzing autoregressive compositional structures, we provide insights into sample complexity and task transferability. While our work lays foundational groundwork for broader applications in machine learning, its immediate impact remains theoretical, with no direct societal implications requiring emphasis.

References

Abbe, E., Bengio, S., Lotfi, A., Sandon, C., and Saremi, O. How far can transformers reason? the locality barrier and inductive scratchpad. *Advances in Neural Information Processing Systems*, 2024.

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

An, S., Lin, Z., Fu, Q., Chen, B., Zheng, N., Lou, J.-G., and Zhang, D. How do in-context examples affect compositional generalization? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11027–11052, 2023.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Arora, S. and Goyal, A. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*, 2023.

Bai, Y., Chen, F., Wang, H., Xiong, C., and Mei, S. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=liMSqUuVg9.

Ben-David, S. and Urner, R. Domain adaptation–can quantity compensate for quality? *Annals of Mathematics and Artificial Intelligence*, 70:185–202, 2014.

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*, 2024. URL https://openreview.net/forum?id=ekeyCgeRfC.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Cortes, C., Mansour, Y., and Mohri, M. Learning bounds for importance weighting. *Advances in neural information processing systems*, 23, 2010.

Du, S. S., Koushik, J., Singh, A., and Póczos, B. Hypothesis transfer learning via transformation functions. *Advances in neural information processing systems*, 30, 2017.

Duchi, J., Hashimoto, T., and Namkoong, H. Distributionally robust losses for latent covariate mixtures. *Operations Research*, 71(2):649–664, 2023.

Dziri, N., Lu, X., Sclar, M., Li, X. L., Jiang, L., Lin, B. Y., Welleck, S., West, P., Bhagavatula, C., Le Bras, R., et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36: 70293–70332, 2023.

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.

Kaur, S., Park, S., Goyal, A., and Arora, S. Instruct-skillmix: A powerful pipeline for llm instruction tuning. In *The Thirteenth International Conference on Learning Representations*, 2025.

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., van Zee, M., and

Bousquet, O. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kpotufe, S. and Martinet, G. Marginal singularity and the benefits of labels in covariate-shift. *The Annals of Statistics*, 49(6):3299–3323, 2021.

Lake, B. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pp. 2873–2882. PMLR, 2018.

Lei, Q., Hu, W., and Lee, J. Near-optimal linear regression under distribution shift. In *International Conference on Machine Learning*, pp. 6164–6174. PMLR, 2021.

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.

Lippl, S. and Stachenfeld, K. When does compositional structure yield compositional generalization? a kernel theory. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ma, C., Pathak, R., and Wainwright, M. J. Optimally tackling covariate shift in rkhs-based nonparametric regression. *The Annals of Statistics*, 51(2):738–761, 2023.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation: Learning bounds and algorithms. In *Annual Conference on Learning Theory*, 2009.

Raghunathan, A., Xie, S. M., Yang, F., Duchi, J., and Liang, P. Understanding and mitigating the tradeoff between robustness and accuracy. In *International Conference on Machine Learning*, pp. 7909–7919. PMLR, 2020.

Saparov, A., Pang, R. Y., Padmakumar, V., Joshi, N., Kazemi, M., Kim, N., and He, H. Testing the general deductive reasoning capacity of large language models using ood examples. *Advances in Neural Information Processing Systems*, 36:3083–3105, 2023.

Schug, S., Kobayashi, S., Akram, Y., Wolczyk, M., Proca, A. M., Von Oswald, J., Pascanu, R., Sacramento, J., and Steger, A. Discovering modular solutions that generalize compositionally. In *International Conference on Learning Representations*, 2024.

Song, J., Xu, Z., and Zhong, Y. Out-of-distribution generalization via composition: a lens through induction heads in transformers. *Proceedings of the National Academy of Sciences*, 122(6):e2417182122, 2025.

Sugiyama, M., Krauledat, M., and Müller, K.-R. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8(5), 2007.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=yzkSU5zdwD. Survey Certification.

Wen, K., Zhang, H., Lin, H., and Zhang, J. From sparse dependence to sparse attention: Unveiling how chain-of-thought enhances transformer sample efficiency. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wiedemer, T., Mayilvahanan, P., Bethge, M., and Brendel, W. Compositional generalization from first principles. *Advances in Neural Information Processing Systems*, 36, 2024.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

Xu, Z., Shi, Z., and Liang, Y. Do large language models have compositional ability? an investigation into limitations and scalability. In *Conference on Language Modeling*, 2024.

Ye, H., Xie, C., Cai, T., Li, R., Li, Z., and Wang, L. Towards a theoretical framework of out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34:23519–23531, 2021.

Zhao, H., Kaur, S., Yu, D., Goyal, A., and Arora, S. Can models learn skill composition from examples? *Advances in Neural Information Processing Systems*, 37:102393–102427, 2024.

Zhou, K., Liu, Z., Qiao, Y., Xiang, T., and Loy, C. C. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4396–4415, 2022.

A. Proofs in Section 3

Notations For a distribution over finite class $\mathcal{S}$, denote $\mathrm{supp}(P) \subseteq \mathcal{S}$ as the support of P. For two distributions P, Q over finite set $\mathcal{S}$, denote the cross entropy as

$$H(P, Q) = -\sum_{x \in \mathcal{S}} P(x) \log Q(x),$$

and denote the KL divergence as

$$KL(P, Q) = -\sum_{x \in \mathcal{S}} P(x) \log \frac{Q(x)}{P(x)}.$$

A.1. Proof of Theorem 3.3

To identify the true distribution $\tilde{\theta}$ from Θ at inference time, we need the following lemma to provide a non-asymptotic bound for distribution discrimination.

Lemma A.1 (Non-Asymptotic Discrimination of Two Distributions). *Let $\mathcal{S}$ be any finite set and $P, Q \in \Delta(\mathcal{S})$ be two distributions with total variation distance $\mathrm{TV}(P, Q) = c > 0$. Suppose we observe n independent samples $X_1, \ldots, X_n$ from an unknown distribution Y, where Y is either P or Q. Then, there exists a testing algorithm that identifies Y with probability at least $1 - \delta$ provided that*

$$n \geq \frac{2 \ln(1/\delta)}{c^2}.$$

Proof. **Testing Procedure.** Define the testing statistic ϕ as follows:

$$\phi = \frac{1}{n} \sum_{i=1}^{n} (-1)^{\mathbf{1}[P(X_i) < Q(X_i)]}.$$

Equivalently, ϕ can be written in terms of the empirical distribution $\hat{Y}_n$:

$$\phi = \sum_{x \in \mathcal{S}} \hat{Y}_n(x) \cdot (-1)^{\mathbf{1}[P(x) < Q(x)]},$$

where $\hat{Y}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}[X_i = x]$.

Compute the expected values of ϕ under $Y = P$ and $Y = Q$:

$$\mu_P = \sum_{x \in \mathcal{S}} P(x) \cdot (-1)^{\mathbf{1}[P(x) < Q(x)]}, \quad \mu_Q = \sum_{x \in \mathcal{S}} Q(x) \cdot (-1)^{\mathbf{1}[P(x) < Q(x)]}.$$

The algorithm reports $\hat{Y} = P$ if $|\phi - \mu_P| < |\phi - \mu_Q|$, and $\hat{Y} = Q$ otherwise.

Proof of Correctness. Without loss of generality, assume $Y = P$. We analyze the probability of error:

$$\Pr\left(\hat{Y} \neq P\right) = \Pr\left(|\phi - \mu_P| \geq |\phi - \mu_Q|\right).$$

Note that $\mathbb{E}[\phi] = \mu_P$ under $Y = P$. From the definition of total variation distance:

$$\mu_Q - \mu_P = \sum_{x \in \mathcal{S}} (Q(x) - P(x)) \cdot (-1)^{\mathbf{1}[P(x) < Q(x)]} = 2 \cdot d_{\mathrm{TV}}(P, Q) = 2c.$$

Thus, the error probability can be bounded as:

$$\Pr\left(|\phi - \mu_P| \geq |\phi - \mu_Q|\right) \leq \Pr\left(\phi \geq \mu_P + c\right).$$

Since ϕ is the average of n independent random variables taking values in $\{-1, +1\}$, by Hoeffding's inequality:

$$\Pr\left(\phi \geq \mu_P + c\right) \leq \exp\left(-\frac{c^2 n}{2}\right).$$

Setting $\exp\left(-\frac{c^2 n}{2}\right) \leq \delta$ gives the required sample complexity $n \geq \frac{2 \ln(1/\delta)}{c^2}$. $\square$

To prove Theorem 3.3, we introduce the following lemma:

Lemma A.2. *Consider a compositional task class $\mathcal{F}$ satisfying Assumption 3.2. Suppose the training tasks $\mathcal{F}_{\text{train}} = \{f_{\theta^i}\}_{i=1}^{n_\theta}$ satisfy the **per-component coverage** condition: for every timestep $t \in [T]$,*

$$\{P_{\theta_t^i}\}_{i=1}^{n_\theta} = \mathcal{P}_{\Theta_t}.$$

Then there exists a learner $\mathcal{A}$ such that when trained on $\mathcal{D}_{\text{train}} = \{\mathcal{D}_i\}_{i=1}^{n_\theta}$ with $\mathcal{D}_i \overset{\text{i.i.d.}}{\sim} P_{\theta^i}$ where $\mathcal{D}_i$ consists of n_x i.i.d. samples, and given $\ell \geq \frac{2 \ln(100 T n_\theta)}{c^2}$ inference-time demonstration samples i.i.d. sampled from unseen task $P_{\tilde{\theta}}$, denoted by $\mathcal{D}_{\text{infer}}$, then it holds that:

$$\lim_{n_x \to \infty} \Pr\left[\mathcal{A}(\mathcal{D}_{\text{infer}}; \mathcal{D}_{\text{train}}) \neq (P_{\tilde{\theta}_1}, \ldots, P_{\tilde{\theta}_T})\right] \leq 0.01,$$

where the probability is over the randomness in $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{infer}}$.

Proof.

Step 1: Training Stage.

We construct a learning algorithm that recovers

$$f_{\theta^i} = \left(P_{\theta_1^i}, \ldots, P_{\theta_T^i}\right)$$

from

$$\mathcal{D}_i = \{(\boldsymbol{x}^{i,j}, \boldsymbol{y}^{i,j})\}_{j=1}^{n_x} \overset{\text{i.i.d.}}{\sim} P_{\theta^i}(\boldsymbol{x}, \boldsymbol{y}),$$

which is part of the training set $\mathcal{D}_{\text{train}}$. The procedure is shown below:

Algorithm 1 Training Stage

Require: Training set $\mathcal{D}_{\text{train}} = \{\mathcal{D}_i\}_{i=1}^{n_\theta}$
1: **for** $i = 1$ to n_θ **do**
2:     **for** $t = 1$ to T **do**
3:         Observed_Support$_t \leftarrow \{(\boldsymbol{x}^{i,j}, \boldsymbol{y}_{1:t}^{i,j})\}_{j=1}^{n_x}$
4:         $\mathcal{P}_{\Xi_t'} \leftarrow \left\{P_{\xi_t} \in \mathcal{P}_{\Xi_t} : \text{supp}(P_{\hat{\theta}_{1:t-1}^i, \xi_t}) = \text{Observed\_Support}_t\right\}$
5:         $\hat{\theta}_t^i \leftarrow \arg\max_{\xi_t \in \Xi_t'} \sum_{j=1}^{n_x} \log P_{\hat{\theta}_{1:t-1}^i, \xi_t}(\boldsymbol{x}^{i,j}, \boldsymbol{y}_{1:t}^{i,j})$
6:     **end for**
7: **end for**
8: **return** $\mathcal{P}_{\hat{\Theta}_t} = \{P_{\hat{\theta}_t^i}\}_{i=1}^{n_\theta}$ for each $t \in [T]$.

We now show that, as $n_x \to \infty$,

$$\Pr[\mathcal{P}_{\hat{\Theta}_t} = \mathcal{P}_{\Theta_t} \ \forall t \in [T]] \to 1.$$

Assume $\hat{\theta}_{1:t-1}^i = \theta_{1:t-1}^i$. As $n_x \to \infty$,

$$\Pr[\text{Observed\_Support}_t \neq \text{supp}(P_{\theta_{1:t}^i})] \leq \sum_{(\boldsymbol{x}, \boldsymbol{y}_{1:t}) \in \text{supp}(P_{\theta_{1:t}^i})} \Pr[(\boldsymbol{x}^{i,j}, \boldsymbol{y}_{1:t}^{i,j}) \neq (\boldsymbol{x}, \boldsymbol{y}_{1:t}) \ \forall j \in [n_x]] \to 0.$$

Conditioned on Observed_Support$_t = \text{supp}(P_{\theta_{1:t}^i})$, any P_{ξ_t} in $\mathcal{P}_{\Xi_t}'$ must share the same support:

$$\text{supp}(P_{\theta_{1:t-1}^i, \xi_t}) = \text{supp}(P_{\theta_{1:t-1}^i, \theta_t^i}).$$

Thus the cross-entropy is finite. By the law of large numbers and Pinsker's inequality, for $P_{\xi_t} \neq P_{\theta_t^i}$ we have

$$\lim_{n_x \to \infty} \left(\frac{1}{n_x} \sum_{j=1}^{n_x} \log P_{\theta^i_{1:t-1}, \theta^i_t}(\boldsymbol{x}^{i,j}, \boldsymbol{y}^{i,j}_{1:t}) - \frac{1}{n_x} \sum_{j=1}^{n_x} \log P_{\theta^i_{1:t-1}, \xi_t}(\boldsymbol{x}^{i,j}, \boldsymbol{y}^{i,j}_{1:t}) \right)$$

$$\to KL(P_{\theta^i_{1:t-1}, \theta^i_t}, P_{\theta^i_{1:t-1}, \xi_t})$$

$$\geq 2\mathrm{TV}(P_{\theta^i_{1:t-1}, \theta^i_t}, P_{\theta^i_{1:t-1}, \xi_t})^2$$

$$> 0$$

almost surely. Thus,

$$\Pr\left[P_{\hat{\theta}^i_t} \neq P_{\theta^i_t} \right] \leq \sum_{\xi_t \in \Xi'_t, \, P_{\xi_t} \neq P_{\theta^i_t}} \Pr\left[\frac{1}{n_x} \sum_{j=1}^{n_x} \log P_{\theta^i_{1:t-1}, \xi_t}(\boldsymbol{x}^{i,j}, \boldsymbol{y}^{i,j}_{1:t}) \geq \frac{1}{n_x} \sum_{j=1}^{n_x} \log P_{\theta^i_{1:t-1}, \theta^i_t}(\boldsymbol{x}^{i,j}, \boldsymbol{y}^{i,j}_{1:t}) \right] \to 0.$$

Recursively applying this argument from $t = 1$ to T and taking a union bound over all $i \in [n_\theta]$ and $t \in [T]$,

$$\lim_{n_x \to \infty} \Pr\left[\exists (t, i) : P_{\hat{\theta}^i_t} \neq P_{\theta^i_t} \right] = 0.$$

Finally, by the assumption that $\{P_{\theta^1_t}, \ldots, P_{\theta^{n_\theta}_t}\} = \mathcal{P}_{\Theta_t}$ for each t, we conclude

$$\Pr\left[\mathcal{P}_{\hat{\Theta}_t} = \mathcal{P}_{\Theta_t} \ \forall t \in [T] \right] \to 1 \quad \text{as} \quad n_x \to \infty.$$

Step 2: Inference Stage

Assume that, in the training phase, the learner identifies the true distribution sets

$$\mathcal{P}_{\hat{\Theta}_t} = \{P_{\hat{\theta}^1_t}, \ldots, P_{\hat{\theta}^{n_\theta}_t}\} = \mathcal{P}_{\Theta_t} \quad \text{for all } t = 1, \ldots, T.$$

We now construct an algorithm that, given $\ell \geq \frac{2 \ln\left(100 \, T \, n_\theta \right)}{c^2}$ independent samples from the unseen composite task

$$f_{\tilde{\theta}} = \left(P_{\hat{\theta}_1}, \ldots, P_{\hat{\theta}_T} \right),$$

outputs the same distribution tuple $\left(P_{\tilde{\theta}_1}, \ldots, P_{\tilde{\theta}_T} \right)$ with probability at least 0.99.

To achieve this, we leverage the distribution discrimination technique of Lemma A.1 to distinguish between candidates in $\mathcal{P}_{\hat{\Theta}_t}$, given sufficiently many demonstration samples.

Algorithm 2 Inference Stage

Require: Inference-time demonstration samples $\mathcal{D}_{\text{infer}} = \{(\tilde{\boldsymbol{x}}^j, \tilde{\boldsymbol{y}}^j)\}_{j=1}^{\ell}$ i.i.d. from $P_{\tilde{\theta}}$, and the identified sets $\mathcal{P}_{\hat{\Theta}_t} = \{P_{\hat{\theta}_t^1}, \ldots, P_{\hat{\theta}_t^{n_\theta}}\}$ for each $t \in [T]$.

1: **for** $t = 1$ to T **do**
2:     Initialize $P_{\zeta_t} \leftarrow P_{\hat{\theta}_t^1}$.
3:     **for** $i = 2$ to n_θ **do**
4:         Compute

$$\phi \leftarrow \frac{1}{\ell} \sum_{j=1}^{\ell} (-1)^{\mathbf{1}\left[P_{\zeta_{1:t-1}, \zeta_t}(\tilde{\boldsymbol{x}}^j, \tilde{\boldsymbol{y}}_{1:t}^j) < P_{\zeta_{1:t-1}, \hat{\theta}_t^i}(\tilde{\boldsymbol{x}}^j, \tilde{\boldsymbol{y}}_{1:t}^j)\right]}.$$

5:         **if**

$$\left| \phi - \sum_{(\boldsymbol{x}, \boldsymbol{y}_{1:t}) \in \mathcal{X} \times \mathcal{Y}^t} P_{\zeta_{1:t-1}, \hat{\theta}_t^i}(\boldsymbol{x}, \boldsymbol{y}_{1:t}) (-1)^{\mathbf{1}\left[P_{\zeta_{1:t-1}, \zeta_t}(\boldsymbol{x}, \boldsymbol{y}_{1:t}) < P_{\zeta_{1:t-1}, \hat{\theta}_t^i}(\boldsymbol{x}, \boldsymbol{y}_{1:t})\right]} \right|$$

$$< \left| \phi - \sum_{(\boldsymbol{x}, \boldsymbol{y}_{1:t}) \in \mathcal{X} \times \mathcal{Y}^t} P_{\zeta_{1:t-1}, \zeta_t}(\boldsymbol{x}, \boldsymbol{y}_{1:t}) (-1)^{\mathbf{1}\left[P_{\zeta_{1:t-1}, \zeta_t}(\boldsymbol{x}, \boldsymbol{y}_{1:t}) < P_{\zeta_{1:t-1}, \hat{\theta}_t^i}(\boldsymbol{x}, \boldsymbol{y}_{1:t})\right]} \right|.$$

        then
6:             Update $P_{\zeta_t} \leftarrow P_{\hat{\theta}_t^i}$.
7:         **end if**
8:     **end for**
9: **end for**
10: **return** $(P_{\zeta_1}, \ldots, P_{\zeta_T})$.

Error Analysis. Let P_{ζ_t} be the chosen distribution at step t. Given $P_{\zeta_{1:t-1}} = P_{\tilde{\theta}_{1:t-1}}$, Lemma A.1 ensures that any incorrect candidate can be detected with high probability, as long as $P_{\tilde{\theta}_t} \in \mathcal{P}_{\hat{\Theta}_t}$. Specifically,

$$\Pr\left[P_{\zeta_t} \neq P_{\tilde{\theta}_t} \mid P_{\zeta_{1:t-1}} = P_{\tilde{\theta}_{1:t-1}} \right]$$
$$\leq \sum_{i=2}^{n_\theta} \Pr\left[\text{distribution testing fails at step } i \mid \zeta_t = \tilde{\theta}_t \text{ or } \hat{\theta}_t^i = \tilde{\theta}_t \right] \leq n_\theta \exp\left(-\frac{c^2 \ell}{2}\right).$$

A union bound over $t = 1, \ldots, T$ then implies

$$\Pr\left[(P_{\zeta_1}, \ldots, P_{\zeta_T}) \neq (P_{\tilde{\theta}_1}, \ldots, P_{\tilde{\theta}_T}) \right] \leq T n_\theta \exp\left(-\frac{c^2 \ell}{2}\right).$$

Since $\ell \geq \frac{2 \ln\left(100 \, T \, n_\theta\right)}{c^2}$, we get

$$T n_\theta \exp\left(-\frac{c^2 \ell}{2}\right) \leq 0.01.$$

Step 3: Combining Training and Inference

From the results of Step 1 and Step 2, we have

$$\lim_{n_x \to \infty} \Pr\left[\mathcal{A}(\mathcal{D}_{\text{infer}}; \mathcal{D}_{\text{train}}) \neq (P_{\tilde{\theta}_1}, \ldots, P_{\tilde{\theta}_T}) \right]$$
$$\leq \lim_{n_x \to \infty} \left(\Pr\left[(P_{\zeta_1}, \ldots, P_{\zeta_T}) \neq (P_{\tilde{\theta}_1}, \ldots, P_{\tilde{\theta}_T}) \mid \mathcal{P}_{\hat{\Theta}_t} = \mathcal{P}_{\Theta_t} \, \forall t \right] + \Pr\left[\exists t \in [T] : \mathcal{P}_{\hat{\Theta}_t} \neq \mathcal{P}_{\Theta_t} \right] \right)$$
$$= 0.01 + 0 = 0.01.$$

$\square$

Proof of Theorem 3.3. By Lemma A.2, it suffices to verify that per-component coverage condition holds for each timestep with high probability when $n_\theta = D \ln(100\, D\, T)$.

For each timestep $t \in [T]$, observe

$$\Pr\big[\{P_{\theta_t^i}\}_{i=1}^{n_\theta} \neq \mathcal{P}_{\Theta_t}\big] \leq \sum_{P_{\theta_t} \in \mathcal{P}_{\Theta_t}} \Pr\big[P_{\theta_t^i} \neq P_{\theta_t} \ \forall i \in [n_\theta]\big]$$

$$= D\left(1 - \frac{1}{D}\right)^{n_\theta}.$$

Hence,

$$\Pr\big[\exists t \in [T] : \{P_{\theta_t^i}\}_{i=1}^{n_\theta} \neq \mathcal{P}_{\Theta_t}\big] \leq \sum_{t=1}^{T} \Pr\big[\{P_{\theta_t^i}\}_{i=1}^{n_\theta} \neq \mathcal{P}_{\Theta_t}\big]$$

$$\leq D\,T\left(1 - \frac{1}{D}\right)^{n_\theta} < D\,T\,e^{-n_\theta/D} = 0.01.$$

This completes the proof. $\square$

A.2. Proof of Corollary 3.5

Proof of Corollary 3.5. It suffices to verify that for the sparse parity problem, the constant c in Assumption 3.2 is $\frac{1}{2}$.

We denote $P_{i_1,\cdots,i_t}(\boldsymbol{x}, \boldsymbol{y}_{<t}) = \frac{1}{2^d} \times \mathbf{1}[y_1 = x_{i_1}] \prod_{s=2}^{t} \mathbf{1}[y_s = y_{s-1} \oplus x_{i_s}]$. For any $i_1, \cdots, i_{t-1} \in [d]$, and $i_t \neq i'_t \in [d]$ and it holds that

$$\mathrm{TV}(P_{i_1,\cdots,i_{t-1},i_t}, P_{i_1,\cdots,i_{t-1},i'_t}) = \frac{1}{2^n} \sum_{x \in \{0,1\}^d} \mathbf{1}[x_{i_1} \oplus \cdots \oplus x_{i_{t-1}} \oplus x_{i_t} \neq x_{i_1} \oplus \cdots \oplus x_{i_{t-1}} \oplus x_{i'_t}]$$

$$= \frac{1}{2^n} \sum_{x \in \{0,1\}^d} \mathbf{1}[x_{i_t} \neq x_{i'_t}]$$

$$= \frac{1}{2}.$$

$\square$

This completes the proof.

B. Non-Asymptotic Analysis of Training-Time Demonstration Sample Complexity

In Theorem 3.3, we established only an *asymptotic* result, showing that as the number of demonstration samples per task at training time $n_x \to \infty$, the probability of correctly identifying subtask families $\mathcal{P}_{\Theta_t}$ tends to one. However, by imposing an additional assumption on the total variation gap between the true distributions and any other hypotheses, it is possible to derive a *non-asymptotic* guarantee on how large n_x must be for accurate subtask identification.

Although maximum-likelihood estimation (MLE) does not directly yield such a non-asymptotic bound in this setting, we can use the same distribution discrimination approach introduced in the inference stage (Lemma A.1).

Assumption B.1 (Compositional Identifiability with fixed tv margin)**.** The autoregressive task class $\mathcal{F}$ satisfies:

1. **Finite Subtask Families**: For each $t \in [T]$, the hypothesis class $\mathcal{P}_{\Xi_t}$ is of size at most H and the subtask conditional distribution family $\mathcal{P}_{\Theta_t} \subseteq \mathcal{P}_{\Xi_t}$ has size $|\mathcal{P}_{\Theta_t}| = D$.

2. **Task Identifiability**: For any $t \in [T]$, $\theta_{1:t-1} \in \bigtimes_{s=1}^{t-1} \Theta_s$, and $\theta_t \in \Theta_t$, $\zeta_t \in \Xi_t$, $P_{\zeta_t} \neq P_{\theta_t}$, the induced distributions stasify:

$$\mathrm{TV}\left(P_{\theta_{1:t-1},\theta_t}, P_{\theta_{1:t-1},\zeta_t}\right) \geq r > 0.$$

Furthermore, for any timestep $t \in [T]$, $\theta_{1:t-1} \in \bigtimes_{s=1}^{t-1} \Theta_s$, and $\theta_t \neq \theta'_t \in \Theta_t$, the induced distributions satisfy:

$$\mathrm{TV}\left(P_{\theta_{1:t-1},\theta_t}, P_{\theta_{1:t-1},\theta'_t}\right) \geq c > 0.$$

Theorem B.2 (Exponential Task Generalization)**.** *Let $\mathcal{F}$ be an autoregressive compositional task class satisfying Assumption 3.2. Then there exists a learner $\mathcal{A}$ with the following property: if during training, one samples n_θ tasks uniformly and independently from $\mathcal{F}$, each provided with n_x i.i.d. demonstration samples as the training dataset, and if at inference one observes ℓ i.i.d. demonstration samples from a previously unseen task $P_{\tilde{\theta}} \in \mathcal{F}$, then*

$$\Pr\left[\mathcal{A}(\mathcal{D}_{\text{infer}};\mathcal{D}_{\text{train}}) \neq (P_{\hat{\theta}_1},\ldots,P_{\hat{\theta}_T})\right] \leq DTe^{-n_\theta/D} + n_\theta T e^{-c^2\ell/2} + n_\theta T H e^{-r^2 n_x/2}.$$

where $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{infer}}$ denote the training dataset and inference-time demonstration samples respectively, and the probability is taken over the random selection of training tasks $\mathcal{F}_{\text{train}} \subseteq \mathcal{F}$, the training data $\mathcal{D}_{\text{train}}$, and the inference time demonstration samples $\mathcal{D}_{\text{infer}}$.

Proof. Denote the hypothesis class $\mathcal{P}_{\Xi_t} = \{P_{\xi_{t,1}}, \cdots, P_{\xi_{t,|\Xi_t|}}\}$, we present the training stage of the learner.

Algorithm 3 Training Stage with Distribution Dislimination

Require: Training set $\mathcal{D}_{\text{train}} = \{\mathcal{D}_i\}_{i=1}^{n_\theta}$
1: **for** $i = 1$ to n_θ **do**
2:     **for** $t = 1$ to T **do**
3:         Initialize $P_{\hat{\theta}_t^i} \leftarrow P_{\xi_{t,1}}$.
4:         **for** $k = 2$ to $|\Xi_t|$ **do**
5:             Compute

$$\phi \leftarrow \frac{1}{n_x} \sum_{(\boldsymbol{x}^{i,j},\boldsymbol{y}^{i,j}) \in \mathcal{D}_i} (-1)^{\mathbf{1}\left[P_{\hat{\theta}_{1:t-1},\hat{\theta}_t^i}(\boldsymbol{x}^{i,j},\boldsymbol{y}_{1:t}^{i,j}) < P_{\hat{\theta}_{1:t-1},\xi_{t,k}}(\boldsymbol{x}^{i,j},\boldsymbol{y}_{1:t}^{i,j})\right]}.$$

6:             **if**

$$\left| \phi - \sum_{(\boldsymbol{x},\boldsymbol{y}_{1:t}) \in \mathcal{X} \times \mathcal{Y}^t} P_{\hat{\theta}_{1:t-1},\xi_{t,k}}(\boldsymbol{x},\boldsymbol{y}_{1:t}) (-1)^{\mathbf{1}\left[P_{\hat{\theta}_{1:t-1},\hat{\theta}_t^i}(\boldsymbol{x},\boldsymbol{y}_{1:t}) < P_{\hat{\theta}_{1:t-1},\xi_{t,k}}(\boldsymbol{x},\boldsymbol{y}_{1:t})\right]} \right|$$
$$< \left| \phi - \sum_{(\boldsymbol{x},\boldsymbol{y}_{1:t}) \in \mathcal{X} \times \mathcal{Y}^t} P_{\hat{\theta}_{1:t-1},\hat{\theta}_t^i}(\boldsymbol{x},\boldsymbol{y}_{1:t}) (-1)^{\mathbf{1}\left[P_{\hat{\theta}_{1:t-1},\hat{\theta}_t^i}(\boldsymbol{x},\boldsymbol{y}_{1:t}) < P_{\hat{\theta}_{1:t-1},\xi_{t,k}}(\boldsymbol{x},\boldsymbol{y}_{1:t})\right]} \right|.$$

            then
7:                 Update $P_{\hat{\theta}_t^i} \leftarrow P_{\xi_{t,k}}$.
8:             **end if**
9:         **end for**
10:     **end for**
11: **end for**
12: **return** $\mathcal{P}_{\hat{\Theta}_t} = \{P_{\hat{\theta}_t^i}\}_{i=1}^{n_\theta}$ for each $t \in [T]$.

Using the same approach as in Step 2 of the proof of Theorem 3.3,

$$\Pr[(P_{\hat{\theta}_1^i},\cdots,P_{\hat{\theta}_T^i}) \neq (P_{\theta_1^i},\cdots,P_{\theta_T^i})] \leq THe^{-r^2 n_x/2}.$$

By union bound,

$$\Pr[\exists\, t \in [T] : \mathcal{P}_{\hat{\Theta}_t} \neq \mathcal{P}_{\Theta_t}] \leq \Pr[\,\exists (t,i) : \ P_{\hat{\theta}_t^i} \neq P_{\theta_t^i}] \leq n_\theta T H e^{-r^2 n_x/2}.$$

The remainder of the proof then proceeds exactly as in Theorem 3.3.

$\square$

C. Extra experiments

Effect of Context Length. The theory assumes access to an infinite number of examples for each training task but does not require infinite demonstrations during inference. However, in practice, we cannot train on an infinite number of examples. Figure 5 shows that providing sufficient context length during both training and inference is crucial for strong performance. Empirically, we observed that a context length of 40 works reasonably well across all experiments with dimensions up to $d = 20$.
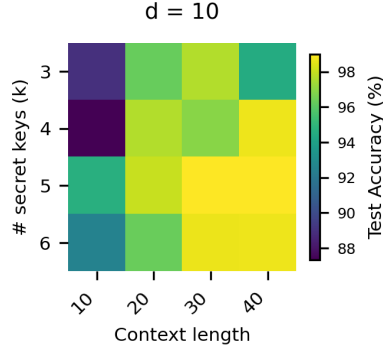

Figure 5. The effect of context length on performance.

ICL with no CoT fails in even in-distribution generalziation. We observe in Figure 6 that transformers with ICL and no CoT struggle to generalize even in simpler in-distribution settings as the number of tasks increases. In the parity task, we refer to in-distribution generalization as a setting where the model is trained on $\mathcal{F}_{train}$ tasks and $\mathcal{S}_{train}$ sequences, and then evaluated on the same set of tasks $\mathcal{F}_{train}$ but with entirely new sequences $\mathcal{S}_{test}$ that were not seen during training.

Here, the setting is the same as in (Bhattamishra et al., 2024) for Parity$(10, 2)$, but we used the same tasks during both training and testing. We trained on half of the total sequences, 2^9 and tested on unseen sequences while keeping the tasks unchanged.
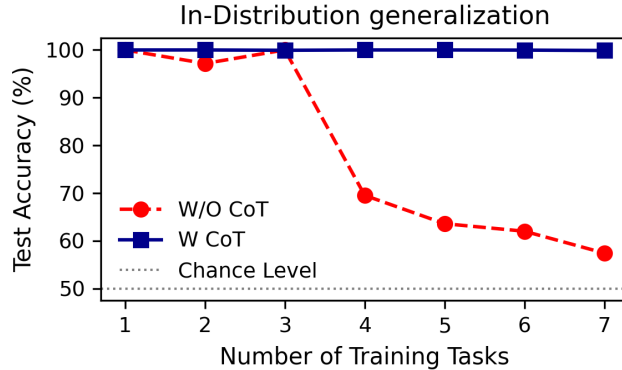

Figure 6. ICL without CoT even fails to generalize in distribution.

D. Experiment Details

Model and optimization. We used the transformers library from Hugging Face (Wolf et al., 2020) to instantiate and train our GPT-2 model from scratch. In all experiments, we used a 3-layer, 1-head configuration. We used the Wadam optimizer (Kingma & Ba, 2015) with a learning rate of 8×10^{-5} and a batch size of 64.

Parity and arithmetic. In all experiments shown in Figures 2 and 3 for both parity and arithmetic tasks, we used a context length of 40.

For the arithmetic problem, across all dimensions, we used a total of 25,000 training examples, equally distributed across the training tasks.

For the parity problem, we used 20,000 training samples, equally distributed across the training tasks for dimensions up to 15. For dimension 20, we increased the total number of training samples to 50,000.

At testing time, we always randomly select the minimum between 200 subsets and all remaining tasks, each containing 500 different sequences with the same context length of 40.

Language experiments. For the translation experiments, we train a 2-layer Transformer with 3 heads and embedding dimension 768. We use an Adam optimizer with betas being $0.9, 0.95$ and learning rate 3e-4. We will keep the number of total training samples to be $1e6$ and train for 1 pass for 6250 steps. We choose the languages randomly from the following set $\{English, French, Spanish, Chinese, German, Italian, Japanese, Russian,$
$Portuguese, Arabic\}$ and meanings (in English) from $\{cat, dog, house, apple, sky, car, road$
$, tree, bed, water, sun, moon\}$. We use a GPT-2 tokenizer and in our demonstrations, we will prepend the language of the corresponding word before each word in the following format like "English: cat".

Linear Probing We append a linear classifier to the checkpoints of models of "Increasing D for a fixed T" tasks, trained on the hidden states of the final attention layer when generating the i-th token in the Chain-of-Thought, with the goal of predicting the i-th "secret index." The models are trained on a total number of of $20,000, 20,000$, and $50,000$ training samples for $d = 10, 15$, and 20, respectively. The tasks used for training and validation are disjoint. Only the linear classifier is trained, while the parameters of the transformer are frozen. We use the Adam optimizer with a learning rate of 4×10^{-5}, and the batch size is set to be 32.