國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master's Thesis

# 拆解上下文學習：透過函數混合式訓練探討其內在機制與分布外泛化能力

# Unpacking In-Context Learning: Underlying Mechanism and Out-of-Distribution Generalization via Blended Training on Function Mixture

黃竑鈞

Hung-Chun Huang

指導教授: 林守德博士

Advisor: Shou-De Lin, Ph.D.

中華民國 114 年 7 月

July 2025

doi:10.6342/NTU202503867

# Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor, Professor Shou-De Lin, for his constant support and weekly guidance, which provided both direction and inspiration throughout my research. Even while in the United States, he never failed to check in on our progress each week, showing his deep commitment to his students. Moreover, I am especially grateful for the generous access to GPU resources in the lab, which made training the models possible.

I'm also deeply thankful to my teammates, Jason and Yun-An, whose help during our weekly discussions was invaluable. From reading numerous papers on in-context learning to writing code together, their companionship and collaboration made the process both productive and enjoyable.

Lastly, I am forever grateful to my family for their unwavering support. Their encouragement has been my greatest source of strength throughout this journey. Thank you for always being there for me.

Hung-Chun Huang

*National Taiwan University*
*July 2025*



doi:10.6342/NTU202503867

# 拆解上下文學習：透過函數混合式訓練探討其內在機制與分布外泛化能力

研究生：黃竑鈞　　指導教授：林守德　博士

國立臺灣大學資訊工程學研究所

## 摘要

當代基於 Transformer 的語言模型在各類眞實任務中展現出卓越的表現，但其內部運作機制仍未被完全理解。近期研究逐漸聚焦於「上下文學習」（in-context learning, ICL）現象，以及模型超越訓練分布進行泛化的能力。然而，多數研究是在簡化條件下進行的，訓練與評估皆以單一、明確定義的函數生成提示（Prompt），這使得模型在結構更爲多樣或模糊的情境下的表現仍不明朗。

本研究探討 ICL 透過「混合訓練」（Blended Training）後的行爲，其中每個訓練提示由多個不同類型函數隨機抽樣產生，且不提供任何明確的任務標記或結構線索。我們以標準的 ICL 任務（如線性分類或二次分類... 等）爲基礎，透過自行設計的實驗來驗證假說，並評估此訓練方式對模型行爲、抗噪性與泛化能力的影響。

實驗結果顯示，在混合訓練情境下，模型並不是用單一函數爲主軸進行函數選擇，而是展現出更具彈性的模式識別能力、對輸入雜訊的更強韌容忍度，以及更佳的異常情境泛化能力。這些發現指出，訓練中引入結構多樣性的提示，有助於提升模型在未知環境下的適應性。

關鍵詞：上下文學習、混合訓練、函數混合、函數選擇、分布外泛化



doi:10.6342/NTU202503867

# UNPACKING IN-CONTEXT LEARNING: UNDERLYING MECHANISM AND OUT-OF-DISTRIBUTION GENERALIZATION VIA BLENDED TRAINING ON FUNCTION MIXTURE

**Student: Hung-Chun Huang** **Advisor: Dr. Shou-De Lin**

**Department of Computer Science and Information Engineering**
**National Taiwan University**

## Abstract

Transformer-based language models have achieved remarkable success across a wide range of real-world tasks, yet the internal mechanisms that govern their behavior remain only partially understood. Recent research has increasingly focused on the phenomenon of in-context learning (ICL) and its ability to generalize beyond the training distribution. However, many of these studies are conducted under simplified conditions, where both training and evaluation use prompts derived from a single, clearly defined function. As a result, it remains unclear how models behave in more structurally diverse or ambiguous settings.

In this study, we examine ICL under a blended training paradigm, in which each training prompt contains examples sampled from multiple function classes, without any explicit task identifiers or structural signals. Using standard ICL benchmarks such as linear and quadratic classification, we assess how this training approach influences model behavior, robustness, and generalization.



doi:10.6342/NTU202503867

Our findings indicate that under blended training, the commonly observed function selection behavior, where the model implicitly identifies and applies a single underlying function, plays a less central role. Instead, the model demonstrates more flexible pattern recognition, improved resilience to input noise, and stronger generalization to out-of-distribution tasks. These results suggest that training on structurally mixed prompts can enhance a model's adaptability in unfamiliar scenarios.

**Keywords:** In-Context Learning, Blended Training, Function Mixture, Function Selection, OOD Generalization.



doi:10.6342/NTU202503867

# Table of Contents





doi:10.6342/NTU202503867





doi:10.6342/NTU202503867

# List of Tables





doi:10.6342/NTU202503867

# List of Figures





doi:10.6342/NTU202503867

# Chapter 1

# Introduction

In-context learning (ICL) is among the most intriguing and under-explored capabilities of large transformer-based language models (LLMs). By conditioning on input-output examples provided in a prompt, models can perform novel tasks without any parameter updates—a behavior popularized by GPT-3 [6]. This surprising ability has sparked growing interest in understanding how such behavior emerges, and whether it reflects pattern matching, implicit optimization, or generalization across tasks.

While much of the popular discussion around ICL has centered on large language models demonstrating emergent reasoning without task-specific training, our focus lies in a different but equally important setting: guiding transformer models to perform context-sensitive inference across diverse, unlabeled tasks. This distinction is crucial, rather than relying on massive pretraining to enable zero-shot generalization, we study how models can be trained to adaptively interpret contextual examples under explicit supervision. Such a framework holds strong practical potential, particularly in domains like stock market forecasting, where underlying patterns are often non-stationary and span a variety of dynamics (e.g., linear trends, regime shifts, or nonlinear cycles). A model capable of flexible adaptation across task types, without requiring parameter updates or retraining, offers a scalable and efficient alternative for real-world decision-making.



doi:10.6342/NTU202503867



To this end, early studies began to formalize ICL as a form of function learning. [8] showed that transformers trained on synthetic data can learn to represent linear (and even more complicated) functions through in-context examples. On the other hand, [9, 14] proposed Prior-Data Fitted Networks (PFNs), revealing that transformer-based models can approximate Bayesian inference when trained offline on carefully constructed task distributions. These works suggest that ICL involves structured, algorithmic behavior within the model. Subsequent analyses have examined how this behavior scales to more diverse or challenging task distributions. Several works highlight structured, phase-wise computation in transformers—showing how attention heads specialize in roles such as preprocessing, optimization, or extrapolation [2, 4, 7]. At the same time, questions remain about the generalization capabilities of ICL, particularly in out-of-distribution (OOD) settings. Some studies, using different methodologies, have pointed out that models possess generalization capabilities [4,17,20]. However, high task diversity may weaken Bayesian-like behavior [15]. Even with multi-task pretraining, generalization tends to deteriorate when tasks are highly ambiguous or misaligned with the model's inductive biases [4, 26]. These insights raise important questions about how generalization arises in ICL, and whether it necessarily depends on task abstraction, model capacity, or training distribution.

Motivated by a different perspective, [13] investigated in-context function mixtures, where function classes are randomly combined at the input level. They introduced blended training, a setup in which models are trained on examples randomly sampled from multiple function families, and applied it to both Transformer and Mamba architectures. While their study reported performance results under this setting, the blended case lacked focused discussion on its impact on model behavior or generalization.

doi:10.6342/NTU202503867

In this work, we revisit and extend blended training in the context of in-context learning. We systematically evaluate its performance on standard ICL tasks such as linear and quadratic classification, and further probe its internal mechanisms using a series of diagnostic experiments. While prior work often frames ICL as a process of identifying a single underlying function from a set of candidates, i.e. function selection or lowest-error selection [3, 20, 21], our findings suggest that this view may be overly restrictive. Even in the vanilla setting, we observe cases where the model does not follow the function selection hypothesis. This effect becomes more pronounced under blended training, where the lack of task labels and structural cues encourages the model to resolve ambiguity in a more flexible, context-sensitive manner. Rather than relying on internal function selection, blended-trained models appear to integrate local patterns more fluidly, which may contribute to their improved robustness and generalization in out-of-distribution scenarios. Our contributions are as follows:

- **Performance validation:** We build upon the blended training paradigm [13] and confirm that it achieves comparable accuracy to vanilla training methods on common ICL benchmarks involving multiple function classes.
- **Mechanism analysis:** We empirically analyze the internal mechanisms of vanilla-trained and blended-trained models through controlled experiments and demonstrate that their behavior challenges commonly held assumptions about function selection and lowest-error preference.
- **Robustness and generalization:** We show that blended training enhances both noise robustness and out-of-distribution generalization, even in cases where recent literature suggests ICL tends to overfit to the training distribution.

doi:10.6342/NTU202503867

# Chapter 2

# Background

## 2.1 In-Context Learning as Function Learning

In-Context Learning (ICL) describes the surprising ability of large language models (LLMs) to perform tasks without parameter updates, by conditioning solely on a prompt containing example input-output pairs. In this setup, a model is given a context consisting of $(x_1, y_1), (x_2, y_2), ..., (x_{k-1}, y_{k-1})$ and is asked to predict the output $y_k$ corresponding to a new input $x_k$. No task identifier or training labels are provided at inference time, the model must infer the latent function F governing the relationship between inputs and outputs in-context.

A pioneering view of ICL as function learning was introduced by [8], who demonstrated that transformers trained on synthetic supervised tasks could recover various function classes purely through context. The model is viewed as approximating a function f from a hypothesis space, with the prompt acting as a support set for generalization. Subsequent work has expanded the space of tested functions with deeper behavior analysis, including linear functions [1, 15, 24], boolean functions [5], dynamical systems [12] and even neural networks [22]. These benchmarks serve as a foundation for probing ICL's inductive behavior, and provide controlled settings to evaluate mechanism, performance, and generalization.



doi:10.6342/NTU202503867

## 2.2 Multi-Function Contexts and Task Mixture

While early studies in ICL typically focused on prompts where all input-output pairs came from a single underlying function [5, 8, 12, 22, 24], recent work has explored broader task distributions involving mixtures of functions and increased task ambiguity. Some recent studies have introduced functional diversity during training, sampling from multiple function classes to analyze generalization across tasks or uncover underlying mechanisms [11, 18, 21, 26]. However, their evaluation protocols typically remain structured, with each prompt at test time derived from a single function class. This preserves a consistent functional identity within the context and implicitly encourages models to specialize or recover that function.

A more ambiguous and less structured setting is presented in blended training, introduced by [13]. In this setup, each training prompt contains examples drawn from multiple function classes, but no task identifiers, segmentation tokens, or ordering structure is provided. The model receives a mixed context of input-output pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_{k-1}, y_{k-1})$, where each $y_i$ (for $1 \leq i \leq k-1$) is generated by a function $f_j$ (for $1 \leq j \leq n$) sampled from a set of functions $\{f_1, \ldots, f_n\}$. There is no explicit information about which function generated which output, nor any positional or grouping cues. This blended training setting provides an opportunity to investigate how models handle ambiguity and whether they rely on recovering global function identity or instead adapt flexibly to local patterns in the prompt. In this work, we adopt blended training as our primary experimental setup to probe the emergence of attention structures, robustness to contextual noise, and generalization to out-of-distribution tasks.

doi:10.6342/NTU202503867



## 2.3 Attention Analysis in ICL

Attention-based interpretability has played a central role in uncovering in-context learning (ICL) mechanisms. Several specialized attention patterns have been identified across layers and heads. One important discovery is the *induction head* [2,17], a head that copies tokens forward in the prompt by focusing attention on the previous token position. Induction heads are thought to facilitate extrapolation and enable sequence pattern matching. More recently, *retrospective heads* have been proposed [4]. These heads attend backward across the prompt to identify examples similar to the current query input, acting as a kind of in-context nearest-neighbor retriever. Retrospective attention appears to play a critical role in model behavior when tasks are ambiguous or not easily classifiable by position.

In our work, we assess the importance of each attention head using a masking-based diagnostic adapted from [7]. For each attention head $j$ in layer $i$, we zero out its output and compute the resulting drop in prediction accuracy:

$$\Delta\text{Acc}^{(i,j)} = \text{Acc}_{\text{full}} - \text{Acc}_{\text{masked}(i,j)}$$

where $\text{Acc}_{\text{full}}$ is the original model accuracy, and $\text{Acc}_{\text{masked}(i,j)}$ is the accuracy when that specific head is ablated. We then normalize each head's impact within its layer as:

$$W_{i,j} = \frac{\Delta\text{Acc}^{(i,j)}}{\sum_k \Delta\text{Acc}^{(i,k)}}$$

resulting in a heatmap that quantifies the relative importance of all heads in each layer.

This diagnostic approach complements structural analysis by quantifying the functional role of each head in ICL behavior, offering insight into how different attention pathways contribute to prompt interpretation and prediction under various training regimes.

doi:10.6342/NTU202503867

## 2.4 Generalization to Out-of-Distribution Functions

The structure of this section follows prior work [20], which outlines several perspectives on how in-context learning may generalize to out-of-distribution settings. The extent to which ICL generalizes beyond its training distribution remains a topic of active debate. Several theoretical frameworks have been proposed to explain generalization behavior:

- **Bayesian inference.** The model infers a latent task concept from the prompt and makes predictions accordingly [14, 23, 25]. However, these frameworks often leave the process of task inference implicit, especially in OOD settings.
- **Gradient descent emulation.** Transformers may internally perform gradient-based optimization. Prior works [16, 19] construct architectures where ICL mimics a linear regression solver trained by gradient descent.
- **Function or algorithm selection.** The model chooses a function from a set of pre-trained routines [3, 21]. This view suggests brittle generalization when encountering functions outside the training distribution.
- **Retrieval-based reasoning.** Some studies argue that the model retrieves in-context examples with similar inputs to the query using attention [10].

We focus on the function selection hypothesis here, which has been supported by several theoretical and empirical studies [20], who argue that models tend to select functions that minimize test-time error. We ask whether models trained under either vanilla or blended regime must select from learned functions, or if they can flexibly fit to in-context patterns with a more general "super function".

doi:10.6342/NTU202503867

# Chapter 3

# Task Design and Generalization Settings

To investigate the mechanisms and generalization behavior of in-context learning under different training paradigms, we design controlled synthetic tasks where the target functions are explicitly defined. This enables a precise evaluation of model behavior and internal representations.

| **Function Name** | **Category** | **Inequality** |
|---|---|---|
| Linear Classification (LC) | (1), (2) | $f_{LC}(x) = 1[w^\top x > 0]$ |
| Checkerboard Classification (CC) | (1) | $f_{CC}(x) = 1[(w_1^\top x)(w_2^\top x) > 0]$ |
| Quadratic Classification (QC) | (2) | $f_{LC}(x) = 1[x^\top Ax > 0]$ |
| Residual Classification (R) | (2), (3) | $f_R(x) = 1[x_j > \tau], j \in \{1, \ldots, d\}$ |
| General Quadratic Classification | (3) | $f_{LC}(x) = 1[x^\top Ax + w^\top x + b > 0]$ |

Table 3.1: Function types and their corresponding inequality-based decision rules used in different test settings.

## 3.1 Category 1: LC vs. CC Binary Task Mixture

In the first setting, we construct a binary function mixture consisting of two qualitatively distinct classification tasks:

- **Linear Classification (LC):** A basic binary classification task that separates



doi:10.6342/NTU202503867

data with a straight-line boundary.

- **Checkerboard Classification (CC):** A more complex binary classification task based on agreement in sign from two linear classifiers

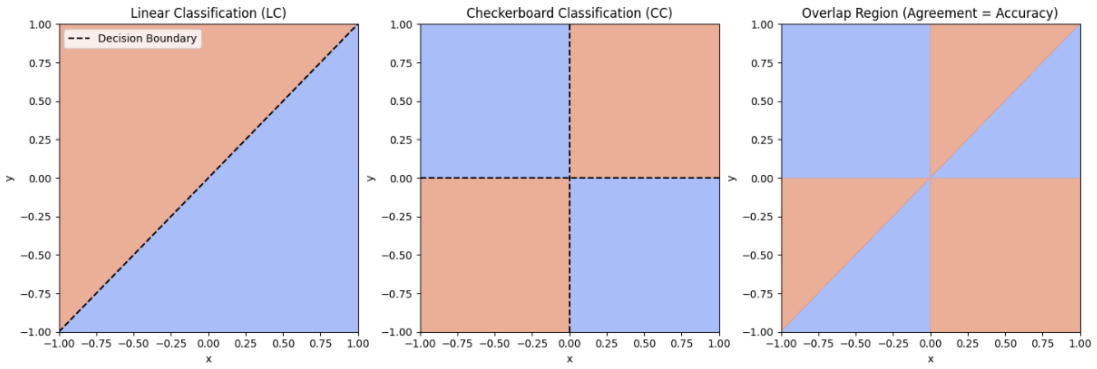



Figure 3.1: Visual illustration of task incompatibility. Left: Linear Classification (LC) with a single decision boundary. Middle: Checkerboard Classification (CC) with alternating labels based on sign agreement. Right: Overlap region showing prediction agreement between LC and CC.

The design of the LC and CC tasks ensures that their decision boundaries are fundamentally misaligned, making it impossible for a classifier trained on one to correctly predict the other. Specifically, attempting to solve the Checkerboard Classification (CC) task using a model optimized for Linear Classification (LC) results in near-random performance ($\sim 50\%$ accuracy), and vice versa.

This stems from the symmetry and structure of the CC function: it labels points as class 1 when the signs of two linear projections agree (both positive or both negative), and class 0 otherwise. Consequently, CC has alternating positive and negative regions that cannot be separated by a single hyperplane. On the other hand, LC is based on a single hyperplane decision boundary, which inherently fails to capture the XOR-like structure embedded in CC. This incompatibility provides a

doi:10.6342/NTU202503867

natural lower bound: any attempt to apply an LC decision rule to CC-labeled data (or vice versa) will produce accuracy close to 50%, reinforcing that high accuracy on both must come from task-specific generalization rather than naive function selection or averaging.

## 3.2 Category 2: QC vs. LC vs. R Multiple Task Mixture

In this setup, we evaluate whether models trained on a mixture of distinct function classes can generalize to unseen but structurally related tasks. Specifically, we train the model using three function types:

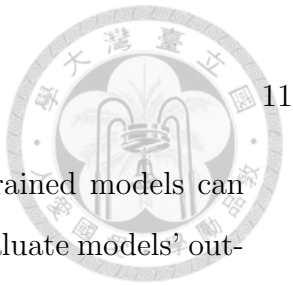
- **Quadratic Classification (QC):** A task involving a smooth, curved decision boundary, typically defined by a paraboloid surface.
- **Linear Classification (LC):** Again, a basic binary classification task that separates data with a straight-line boundary.
- **Residual Classification (R):** A simple rule-based classifier that decides based on thresholding a selected input dimension, resulting in axis-aligned vertical or horizontal boundaries.

This setting is designed to test the model's ability to unify patterns and structures from previously seen tasks and apply them to more complex, unseen contexts. It provides insight into how well blended or vanilla-trained models can leverage multi-function training for broader generalization.

## 3.3 Category 3: Functions used to test generalization

This category is designed to test the model's ability to unify patterns and structures from previously seen tasks and apply them to more complex, unseen

doi:10.6342/NTU202503867



contexts. It provides insight into how well blended or vanilla-trained models can leverage multi-function training for broader generalization. To evaluate models' out-of-distribution (OOD) generalization ability, we introduce held-out test functions that differ structurally from the training objectives.

For category 1, which involves training on Linear Classification (LC) and Checkerboard Classification (CC), we use the Residual Classification (R) as the OOD test function. This choice is motivated by the observation that CC and Quadratic Classification (QC) share certain spatial regularities, such as symmetric patterns and multiple decision boundaries, which may make QC less distinct as a generalization probe. In contrast, R introduces a simple yet orthogonal decision structure based on axis-aligned thresholding, making it a more appropriate challenge for assessing extrapolative capability.

For category 2, where the model is trained on LC, QC, and R, we construct a general QC function as the OOD target. This general QC is generated with unseen parameterizations, providing a shifted but structurally related function class. The aim is to examine whether the model can extract high-level abstractions across multiple training functions and apply them in novel configurations.

doi:10.6342/NTU202503867

# Chapter 4

# Training and Evaluation Setup

In this section, we describe the details of the model training and evaluation procedures.

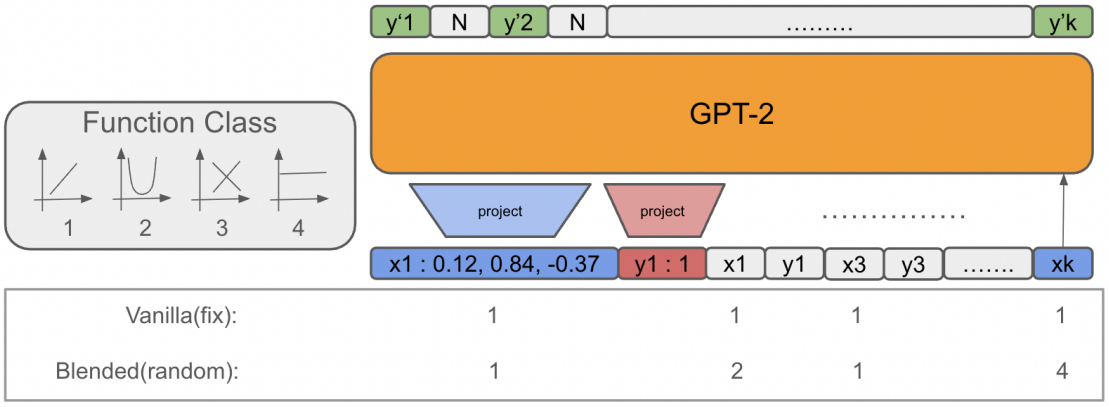



Figure 4.1: Illustration of the training diagram: GPT-2 processes input-output pairs drawn from either a single (vanilla) or a mixed (blended) set of function classes.

## 4.1 Training Detail

During training, each input sequence consists of 100 input-output pairs. The model observes 99 context points $(x_1, y_1), (x_2, y_2), ..., (x_{99}, y_{99})$ and is trained to predict the target output $y_{100}$ corresponding to the input $x_{100}$. Both the inputs x and the underlying function weights w are independently sampled from a standard



doi:10.6342/NTU202503867

| **Parameter** | **Value** |
|---|---|
| Number of points ($k$) | 100 (99 context + 1 evaluation) |
| Data size | $64 \times 300{,}000$ (batch size $\times$ epochs) |
| Input dimension (dim) | 3 |
| Learning rate (lr) | 0.001 |
| Input distribution ($x$) | $\mathcal{N}(0, 1)$ |
| Weight distribution ($w$) | $\mathcal{N}(0, 1)$ |
| GPU | NVIDIA GeForce RTX 3090 |

Table 4.1: Training configuration and data generation setup.

normal distribution $\mathcal{N}(0, 1)$. The model architecture is based on GPT-2, which processes the sequence of tokenized input-output pairs using separate embedding layers for x and y before feeding them into the transformer layers.

Two training strategies are compared: vanilla and blended. In the vanilla setting, all points within a training example are sampled from the same function class (e.g., linear, quadratic), and this function remains fixed throughout the prompt. In contrast, the blended setting introduces greater variability by randomly selecting a function class for each point in the context, with each function class chosen uniformly from a predefined set. For each training instance, the model predicts the final output $y_{100}$ based on the full context of 99 preceding points. The predicted values $\hat{y}_1, \hat{y}_2, ..., \hat{y}_{100}$ are compared with the ground truth targets to compute the loss, and the performance of the model is tracked over time for both training settings.

The model was trained with a batch size of 64 in 300000 epochs, so the total number of data sequences is $64 \times 300000$. The latent dimension $k$ is configured to 3 to facilitate both prompter training and visualization. Training was conducted using a learning rate of 0.001. All experiments were performed on an Nvidia GeForce RTX 3090 GPU.

doi:10.6342/NTU202503867

## 4.2 Evaluation Method

To evaluate the performance of the model, the following method was used. In each trial, a context of 99 points was randomly sampled, and the 100-th point was appended 2000 times to assess prediction accuracy within that context. This procedure was repeated 1000 times, and the average accuracy across all trials was reported as the final result.

For the bar chart analysis, the model that achieved the highest accuracy in each trial was identified, and its selection frequency across all trials was used to compare overall performance among models.

doi:10.6342/NTU202503867

# Chapter 5

# Experimental Results

In this section, We present several experiments designed to address the key research questions outlined in the main contributions. These experiments cover performance validation, mechanism analysis, robustness and generalization.

## 5.1 Performance Validation

| **Training Mode** | **LC (%)** | **CC (%)** |
|---|---|---|
| Blended | 98.60 | **95.90** |
| Vanilla | **98.80** | 95.65 |

Table 5.1: Accuracy (%) for LC and CC Binary Task Mixture under different training modes.

| **Training Mode** | **LC (%)** | **QC (%)** | **R (%)** |
|---|---|---|---|
| Blended | **98.70** | 95.50 | 98.15 |
| Vanilla | 98.45 | **96.60** | **99.50** |

Table 5.2: Accuracy (%) for LC, QC, and R Multiple Task Mixture under different training modes.

To evaluate whether blended training impacts in-context learning performance, we compare models trained under blended and vanilla supervision across several function classes, including linear classification (LC), quadratic classification



doi:10.6342/NTU202503867

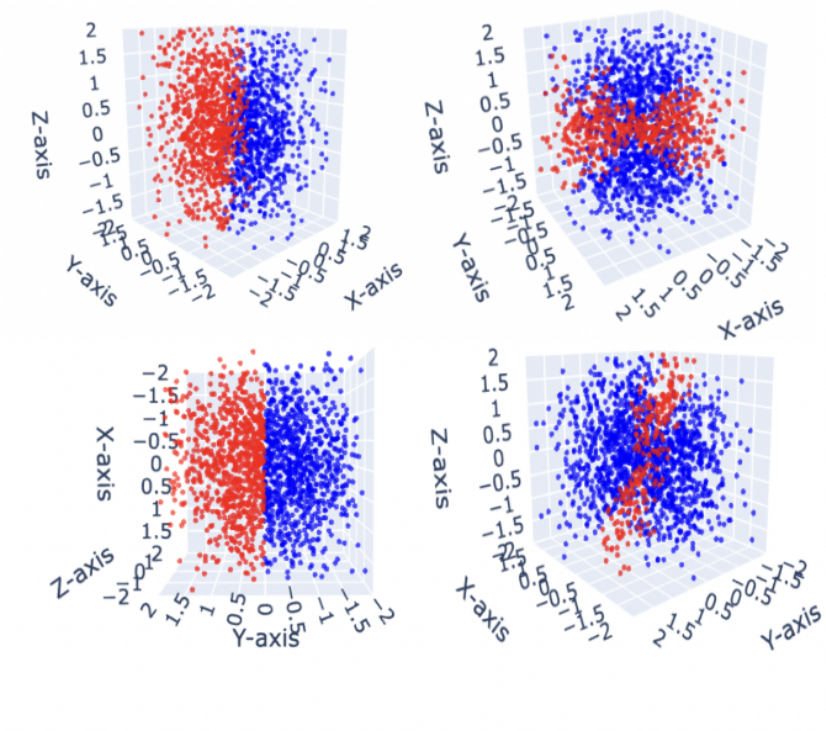



Figure 5.1: Visualization of 3D classification data from four different functions. Top-Left: LC, Top-Right: QC, Bottom-Left: R, Bottom-Right: CC

(QC), residual classification (R), and checkerboard classification (CC). As shown in Table 5.1 and Table 5.2, the blended-trained model achieves accuracy comparable to, and in some cases slightly surpasses, that of the vanilla-trained model. These results suggest that mixing function classes during training does not degrade predictive performance. It is worth noting that for evaluation, both vanilla- and blended-trained models are tested using vanilla-style contexts, where all input-output pairs within a sequence are generated from the same underlying function. This ensures a fair comparison of how well each model generalizes to individual function classes.

To better understand the model's behavior, we visualize the prediction pattern from the blended-trained model given an input prompt in Figure 5.1. Specifically, we construct a context of 99 input-output pairs and sequentially test the

doi:10.6342/NTU202503867

100-th query point. Each predicted point is then plotted in 3D space, colored according to the model's output: blue for class 1 and red for class 0. As shown in the figure, the model's predictions reflect clear and structured separation, suggesting that it successfully extrapolates the correct decision boundary from the observed examples. In particular, the predicted decision structure aligns well with the underlying task, indicating that the model has internalized not just individual examples but the generative rule behind them.

## 5.2 Mechanism Analysis

To further investigate whether the function selection hypothesis holds under different training regimes, we conduct a series of experiments, each accompanied by two explicitly formulated hypotheses. These experiments are designed to probe the underlying mechanisms behind model behavior and examine whether the model truly performs function selection or instead adapts dynamically based on the presented context.

### 5.2.1 (1) Function Mixture Test

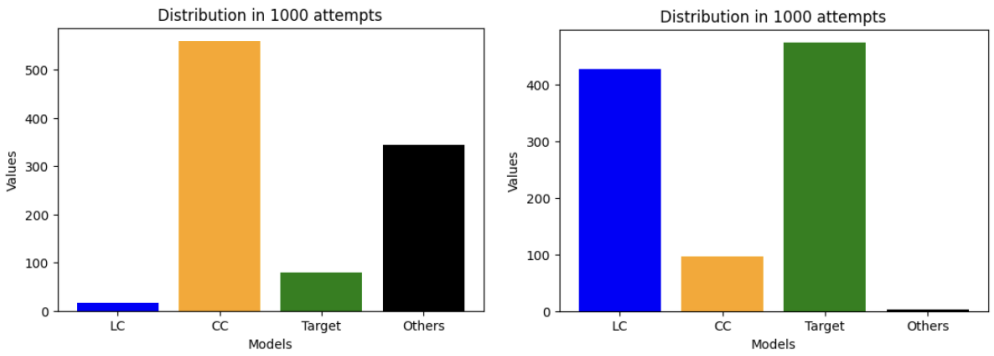



Figure 5.2: Distribution of model's choices over 1000 attempts. Left: Vanilla, Right: Blended

doi:10.6342/NTU202503867

In this experiment, we examine the model's behavior under the binary task mixture setup (Category 1), where linear classification (denoted as Function A) and checkerboard classification (Function B) are presented within the same prompt. As mentioned in Section 3.1, these two tasks are structurally incompatible: attempting to solve one using the decision boundary of the other yields accuracy close to random guessing (approximately 50%). This property allows us to probe whether the model is rigidly selecting between known functions or flexibly adapting to contextual cues. We then formulate two hypotheses:

- **H1**: The model has memorized how to solve either A or B and, when given a mixed context, simply falls back on one of the known routines.
- **H2**: Although trained to solve A and B, the model does not explicitly internalize the functions themselves. Instead, it learns how to fit examples based on contextual patterns, preserving flexibility to adapt to novel or ambiguous mixtures.

To test these hypotheses, we construct prompts containing a mixture of A and B examples (99 points), and evaluate the model's prediction on a $100^{\text{th}}$ point sampled from either function. If the model achieves accuracy above 60% on both A and B tasks within the same prompt, this suggests that it is not merely selecting between known routines, but is capable of interpolating behavior based on the input distribution. We evaluate this behavior over 2000 samples and categorize each prompt into one of four groups based on model performance:

- **LC (blue)**: The model shows strong accuracy on the linear task ($acc_{lc} > acc_{cc} + 0.2$ and $acc_{cc} < 0.6$), indicating preference for solving A.

doi:10.6342/NTU202503867

- **CC (orange)**: The model favors the checkerboard task ($acc_{cc} > acc_{lc} + 0.2$ and $acc_{lc} < 0.6$), indicating preference for solving B.
- **Target (green)**: The model achieves non-trivial performance on both functions ($acc_{lc} > 0.6$ and $acc_{cc} > 0.6$), suggesting successful adaptation to both tasks.
- **Others (black)**: Prompts that do not meet any of the above conditions, including borderline or ambiguous cases.

As shown in the bar plots (see Figure 5.2), in the blended training condition, over 400 of the tested prompts fall into the **Target** category - contexts in which the model solves both A and B with accuracy exceeding 60%. This outcome supports H2, indicating that the model does not rigidly select a single function but instead adapts dynamically based on the prompt, even in the absence of explicit function labels.

### 5.2.2 (2) Out-Of-Distribution Function Test

| | **Vanilla** | **Blended** | **LC** | **CC/QC** | **R** | **Mix** |
|---|---|---|---|---|---|---|
| **Setting 1** | 0.8495 | **0.8905** | 0.7381 | 0.6985 | – | 0.8214 |
| **Setting 2** | 0.8312 | **0.8637** | 0.6265 | 0.7909 | 0.6657 | 0.8144 |

Table 5.3: Accuracy of different models tested with binary task mixture (Setting 1) and multiple task mixture (Setting 2)

In this experiment, we evaluate whether models can generalize to out-of-distribution (OOD) function types that were not seen during training. Specifically, we compare vanilla-trained and blended-trained models to a baseline formed by individually trained models, that is, models trained on a single function type without exposure to other task types. Two evaluation settings are considered:

doi:10.6342/NTU202503867

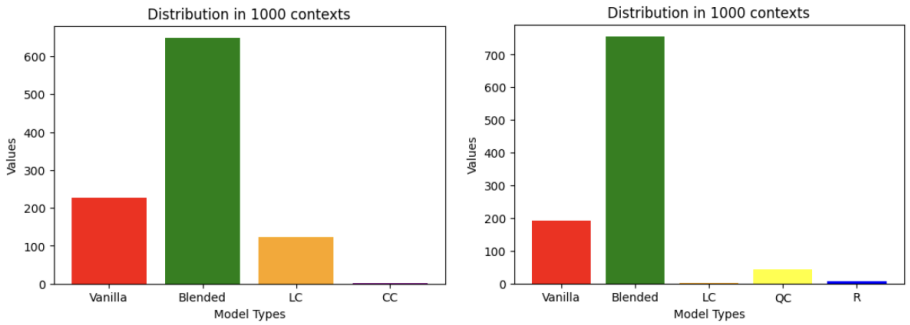



Figure 5.3: Distribution of test performance across 1000 prompts in two settings.

- **Setting 1:** The model is trained on Category 1 (LC and CC), and tested on the residual classification (R) task.
- **Setting 2:** The model is trained on Category 2 (LC, QC and R), and tested on a general quadratic classification task composed of mixed distributions (LC + QC + R).

To benchmark performance, we introduce a **Mix baseline**, defined as the maximum accuracy across individually trained models in the same round (i.e., $\max(\text{LC}, \text{QC}, \text{R})$ or $\max(\text{LC}, \text{CC})$). If a model merely memorizes function-specific routines, its performance should not exceed the Mix baseline. We consider two hypotheses:

- **H1:** The model internalizes specific function classes during training. Consequently, its generalization should be no better than the best of the singly trained models.
- **H2:** The model does not encode functions internally but instead adapts to contextual information. This allows it to go beyond the known function classes and generalize to new combinations.

doi:10.6342/NTU202503867

As shown in Table 5.3, both the vanilla and blended models consistently achieve higher accuracy than the Mix baseline in both settings. This performance gap suggests that the models are not merely selecting among pre-learned functions, but are instead adapting based on the presented context. These results support H2 and call into question the function selection hypothesis.

The accompanying bar plots (Figure 5.3) show the distribution of test performance across 1000 prompts. Compared to the individually trained models, both vanilla and blended models contribute substantially more high-performing cases, further supporting the claim that in-context learning extends beyond static function retrieval.

### 5.2.3 (3) Model Bias Test

| Model | Replace 0 pts | Replace 2 pts | Replace 5 pts | Replace 10 pts |
|---|---|---|---|---|
| Blended | 99 — 1 | 89 — 11 | 57 — 43 | 2 — 98 |
| Vanilla | 100 — 0 | 59 — 41 | 19 — 81 | 4 — 96 |

Table 5.4: Comparison under input point replacement (X — Y indicates number of LC — CC being selected out of 100 attempts)

This experiment investigates whether the model exhibits bias when interpreting ambiguous prompts. We construct contexts that resemble both linear classification (LC) and checkerboard classification (CC), then incrementally replace a small number of points to lean toward CC. We test 100 such prompts per setting, comparing the model's classification accuracy on LC and CC. Whichever function achieves higher accuracy is considered the one "selected" by the model. We evaluate two hypotheses:

- **H1**: The model selects the function that minimizes expected error.

doi:10.6342/NTU202503867

- **H2**: The model does not explicitly evaluate error but relies on internal biases or heuristics.

As shown in Table 5.4, both models initially prefer LC (Blended: 99/100; Vanilla: 100/100). As more points are replaced toward CC, the vanilla model shifts its preference earlier (e.g., 81 CC selections at 5-point replacement), while the blended model remains more committed to LC (57 LC vs. 43 CC). This suggests that the blended model may exhibit a stronger bias toward LC.

These findings contradict the lowest-error selection hypothesis (H1). If the model were simulating error, it should be indifferent under ambiguity and shift decisively once the evidence favors one function. Instead, the model shows a preference-driven response, supporting H2. While this does not conclusively rule out broader function selection, it highlights that "lowest-error selection" is not the strategy the model follows in ambiguous contexts.

### 5.2.4 (4) Attention Head Analysis

| Method | Layer | Head | Ratio (top-5) | Ratio (top-10) |
|---|---|---|---|---|
| Blended | 4 | 4 | 80% | 90% |
| Blended | 8 | 4 | 100% | 80% |
| Blended | 8 | 8 | 80% | 80% |
| Vanilla | 4 | 4 | 80% | 100% |
| Vanilla | 8 | 4 | 60% | 90% |
| Vanilla | 8 | 8 | 100% | 80% |

Table 5.5: Overlap ratios of top-k influential attention heads across tasks.

This experiment investigates whether certain attention heads specialize in solving specific tasks—namely, linear classification (LC) or checkerboard classification (CC). To evaluate this, we adopt an ablation-based approach: systematically

doi:10.6342/NTU202503867

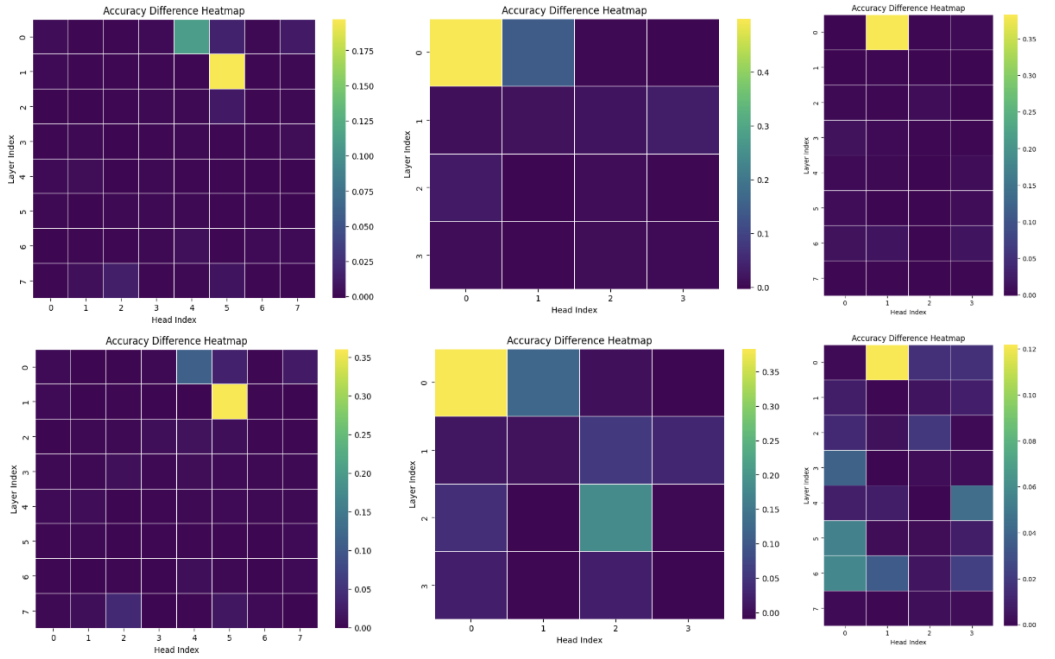



Figure 5.4: Vanilla-trained model accuracy difference heatmaps

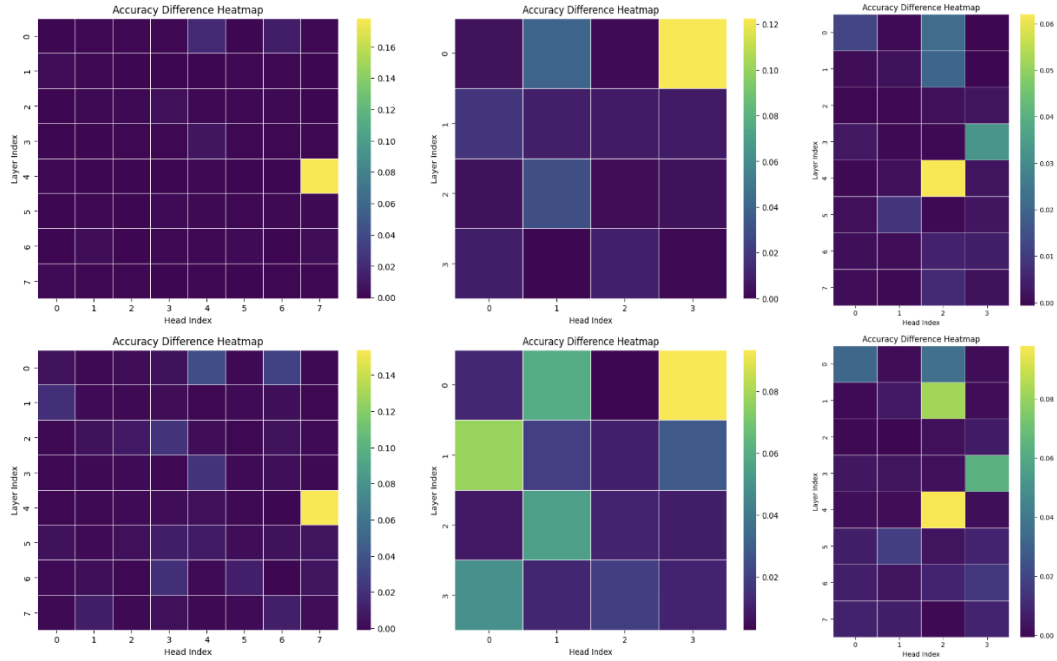



Figure 5.5: Blended-trained model accuracy difference heatmaps

Figure 5.6: Accuracy drops from attention head ablation across different transformer layers and heads. Left: Model with 8 heads of 8 layers, Middle: Model with 4 heads of 8 layers, Left: Model with 4 heads of 4 layers

doi:10.6342/NTU202503867

zeroing out each attention head and measuring the corresponding drop in model accuracy. If the model performs function selection, we would expect different heads to specialize for different functions. Conversely, if the model learns to fit contextual patterns without abstracting function identities, then the same heads should support multiple functions.

For each model variant, we generate an accuracy difference heatmap, where each cell represents the drop in accuracy after ablating a specific head. We evaluate four transformer configurations with varying numbers of layers and heads. In each heatmap, rows correspond to layers and columns to head indices. Larger values indicate heads that are more critical to model performance. We consider two competing hypotheses:

- **H1**: The model encodes abstract functions. Specific heads contribute exclusively to Function A or B, allowing the model to select the appropriate function based on the prompt.
- **H2**: The model does not explicitly encode function identity. Instead, attention heads serve as general-purpose mechanisms that support multiple functions simultaneously.

As shown in the heatmaps, ablating top-performing heads leads to accuracy drops across both LC and CC tasks. Notably, the top-$k$ influential heads tend to overlap between tasks, especially in the blended-trained models—suggesting that attention heads are not function-specific modules, but rather shared components used for flexible contextual fitting.

To quantify this overlap, we calculate the proportion of shared attention heads in the top-5 and top-10 influential positions across LC and CC. As shown

doi:10.6342/NTU202503867

in Table 5.5, blended-trained models exhibit high consistency, with top-5 overlap ratios reaching 100% in some configurations (e.g., Layer 8, Head 4), and top-10 overlaps ranging from 70% to 90%. Even in vanilla-trained models, considerable overlap exists, for example: a 100% top-5 match in Layer 8, Head 8, though with slightly more variance (e.g., 60% overlap in Layer 8, Head 4).

In sum, these findings support **H2**: attention heads contribute to flexible pattern fitting, rather than acting as modular components dedicated to specific functions. The shared influence of top heads across LC and CC undermines the function-selection hypothesis and suggests that the model processes prompts holistically: dynamically adapting its computation based on input structure, rather than invoking fixed routines aligned to specific function classes.

## 5.3 Generalization and Robustness

To assess the effectiveness of different training strategies beyond accuracy alone, we further evaluate the models' generalization and robustness. Specifically, we design experiments that separately target these two aspects: the ability to extrapolate to unseen functional compositions, and the resilience to noisy input conditions. By comparing blended, vanilla, and noise-augmented models under controlled settings, we aim to determine whether the benefits of blended training extend meaningfully into these challenging scenarios.

### 5.3.1 OOD Generalization Comparison with Noise-Augmented Model

We further assess the advantage of blended training by introducing a noise-augmented baseline. The setup follows that of (2) Out-Of-Distribution Function Test in Section 5.2.2, with the addition of a model trained on noisy contexts, where a random subset of values is flipped (0 to 1 or vice versa) with probability 0.3. This

doi:10.6342/NTU202503867

| | Vanilla | Blended | Noise |
|---|---|---|---|
| Setting 1 | 0.8551 | **0.8960** | 0.8863 |
| Setting 2 | 0.8312 | **0.8620** | 0.8270 |

Table 5.6: Generalization accuracy across different models.

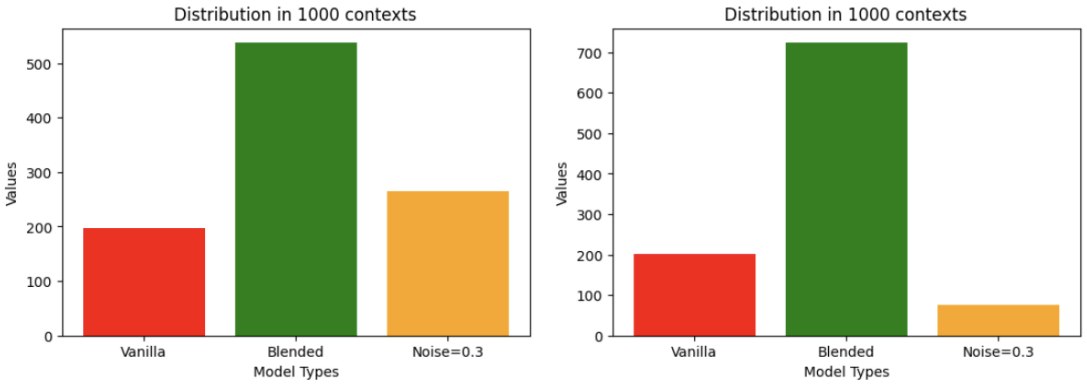



Figure 5.7: Distribution of predictions in 1000 contexts. Left: Setting 1, Right: Setting 2

noise-augmented model serves as a control, testing whether blended training merely benefits from noise-based regularization.

We evaluate two generalization settings as previously defined: **Setting 1** (train on LC and CC, test on R) and **Setting 2** (train on LC, QC, and R, test on general quadratic classification).

As shown in Figure 5.7 and Table 5.6, the blended model achieves the highest accuracy in both settings, outperforming the vanilla and noise-augmented (noted as **Noise** in the table) models across more than half of the 1000 contexts. In Setting 1, the blended model achieves an accuracy of 0.8960, outperforming both the vanilla model (0.8551) and the noise-augmented model (0.8863). In Setting 2, the noise-augmented model (0.8270) performs slightly worse than the vanilla model (0.8312), while both are still notably outperformed by the blended model (0.8620),

doi:10.6342/NTU202503867

highlighting the consistent advantage of blended training.

These results confirm that the gains from blended training cannot be attributed to noise alone. While the noise-augmented model shows some improvement in simpler settings, it fails to match blended training in structurally diverse scenarios. This indicates that blended training provides a stronger inductive bias, allowing the model to better generalize to ambiguous or unseen function compositions.

### 5.3.2 Robustness Under Noisy Inference

| | | **Noise Level = 0.1** | | | **Noise Level = 0.2** | | | **Noise Level = 0.3** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Setting** | **Task** | Vanilla | Blended | Noise | Vanilla | Blended | Noise | Vanilla | Blended | Noise |
| Setting 1 | LC′ | 0.81 | **0.98** | 0.97 | 0.53 | 0.92 | **0.93** | 0.50 | **0.68** | 0.64 |
| | CC′ | 0.85 | **0.93** | **0.93** | 0.74 | 0.83 | **0.86** | 0.61 | 0.65 | **0.69** |
| Setting 2 | QC′ | 0.88 | **0.94** | **0.94** | 0.77 | 0.87 | **0.89** | 0.61 | 0.66 | **0.69** |
| | LC′ | 0.89 | **0.97** | **0.97** | 0.63 | **0.93** | **0.93** | 0.53 | **0.73** | 0.70 |
| | R′ | 0.87 | **0.98** | **0.98** | 0.70 | 0.94 | **0.96** | 0.58 | **0.76** | 0.72 |

Table 5.7: Accuracy under different noise levels (0.1, 0.2, 0.3) for each task, comparing Vanilla-, Blended-, and Noise-augmented models.

This experiment evaluates the robustness of different training strategies under varying levels of inference-time noise. We compare vanilla, blended, and noise-augmented models across five tasks in two settings, introducing random flip during inference at noise levels of 0.1, 0.2, and 0.3. Each cell in the accuracy table reports performance at these three levels, respectively. We consider two training configurations:

- **Setting 1**: Train on LC and CC → Test on LC′ and CC′ under noise
- **Setting 2**: Train on LC, QC, and R → Test on LC′, QC′, and R′ under noise

As summarized in Table 5.7, both the blended and noise-augmented models outperform the vanilla baseline across all noise levels and tasks. Notably, the blended

doi:10.6342/NTU202503867

model achieves comparable robustness to the noise-augmented model despite not being exposed to label noise during training. For instance, in the "LC′ (Setting 2)" setting, the blended model achieves 0.97, 0.93, and 0.73 at the three respective noise levels — matching or even better than the noise-augmented model with 0.97, 0.93, and 0.70.

These findings indicate that blended training inherently supports robustness against input noise, likely due to its exposure to diverse functional patterns during training. Unlike stochastic regularization, this diversity promotes the development of stable decision boundaries that generalize reliably even under degraded or ambiguous inputs.

doi:10.6342/NTU202503867

# Chapter 6

# Conclusion

In this work, we investigated the effects of blended training and its implications for in-context learning mechanisms. Our results demonstrate that blended training achieves comparable accuracy to vanilla training, suggesting that incorporating functional diversity does not compromise predictive performance.

We further examined the function selection hypothesis through four targeted experiments. In Experiment (1), only the blended model showed evidence against function selection. However, in Experiments (2), (3), and (4), both blended and vanilla models exhibited behaviors inconsistent with the hypothesis. These findings indicate that function selection may not adequately explain model behavior under different training strategies.

Finally, our comparison with models trained under random noise reveals that blended training offers stronger generalization and similar robustness to noisy inputs, despite not relying on explicit noise injection. This suggests that blended training provides benefits beyond noise-based regularization, promoting more stable and adaptive inference in ambiguous or degraded conditions.



doi:10.6342/NTU202503867

# Bibliography


[1] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023.

[2] Anonymous. Induction heads as a primary mechanism for pattern matching in in-context learning. In *Submitted to ACL Rolling Review - June 2024*, 2024. under review.

[3] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[4] Harmon Bhasin, Timothy Ossowski, Yiqiao Zhong, and Junjie Hu. How does multi-task training affect transformer in-context capabilities? investigations with function classes, 2024.

[5] Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. Understanding in-context learning in transformers and LLMs by learning to learn discrete functions. In *The Twelfth International Conference on Learning Representations*, 2024.

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry,




doi:10.6342/NTU202503867



Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[7] Xingwu Chen, Lei Zhao, and Difan Zou. How transformers utilize multi-head attention in in-context learning? a case study on sparse linear regression. In *ICML 2024 Workshop on Theoretical Foundations of Foundation Models*, 2024.

[8] Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[9] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *NeurIPS 2022 First Table Representation Workshop*, 2022.

[10] Hongkang Li, Meng Wang, Songtao Lu, Xiaodong Cui, and Pin-Yu Chen. How do nonlinear transformers learn and generalize in in-context learning? In *ICML*, 2024.

[11] Hongkang Li, Meng Wang, Songtao Lu, Hui Wan, Xiaodong Cui, and Pin-Yu Chen. Transformers as multi-task feature selectors: Generalization analysis of in-context learning. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023.

doi:10.6342/NTU202503867



[12] Yingcong Li, M. Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning, 2023.

[13] Yingcong Li, Xupeng Wei, Haonan Zhao, and Taigao Ma. Can mamba in-context learn task mixtures? In *ICML 2024 Workshop on In-Context Learning*, 2024.

[14] Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2022.

[15] Allan Raventos, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[16] Lingfeng Shen, Aayush Mishra, and Daniel Khashabi. Do pre-trained transformers really learn in-context by gradient descent?, 2024.

[17] Jiajun Song, Zhuoyan Xu, and Yiqiao Zhong. Out-of-distribution generalization via composition: a lens through induction heads in transformers. In *The Second Conference on Parsimony and Learning (Recent Spotlight Track)*, 2025.

[18] Nilesh Tripuraneni, Lyric Doshi, and Steve Yadlowsky. Can transformers in-context learn task mixtures? In *NeurIPS 2023 Workshop on Distribution Shifts: New Frontiers with Foundation Models*, 2024.

[19] Jonas von Oswald, Elias Niklasson, Eric Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Maxim Vladymyrov. Transformers

doi:10.6342/NTU202503867



learn in-context by gradient descent. In *International Conference on Learning Representations (ICLR)*, pages 35151–35174, 2023.

[20] Qixun Wang, Yifei Wang, Xianghua Ying, and Yisen Wang. Can in-context learning really generalize to out-of-distribution tasks? In *The Thirteenth International Conference on Learning Representations*, 2025.

[21] Zhijie Wang, Bo Jiang, and Shuai Li. In-context learning on function classes unveiled for transformers. In *Forty-first International Conference on Machine Learning*, 2024.

[22] Zhijie Wang, Bo Jiang, and Shuai Li. Transformers perform in-context learning through neural networks, 2024.

[23] Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[24] Jingfeng Wu, Difan Zou, Zixiang Chen, Vladimir Braverman, Quanquan Gu, and Peter Bartlett. How many pretraining tasks are needed for in-context learning of linear regression? In *The Twelfth International Conference on Learning Representations*, 2024.

[25] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022.

[26] Steve Yadlowsky, Lyric Doshi, and Nilesh Tripuraneni. Pretraining data mixtures enable narrow model selection capabilities in transformer models, 2023.

doi:10.6342/NTU202503867