Prior Forgetting and In-Context Overfitting

Sungyoon Lee
Department of Computer Science
Hanyang University
sungyoonlee@hanyang.ac.kr

Abstract

In-context learning (ICL) is one of the key capabilities contributing to the great success of LLMs. At test time, ICL is known to operate in the two modes: task recognition and task learning. In this paper, we investigate the emergence and dynamics of the two modes of ICL during pretraining. To provide an analytical understanding of the learning dynamics of the ICL abilities, we investigate the in-context random linear regression problem with a simple linear-attention-based transformer, and define and disentangle the strengths of the task recognition and task learning abilities stored in the transformer model's parameters. We show that, during the pretraining phase, the model first learns the task learning and the task recognition abilities together in the beginning, but it (a) gradually forgets the task recognition ability to recall the priorly learned tasks and (b) relies more on the given context in the later phase, which we call (a) *prior forgetting* and (b) *in-context overfitting*, respectively.

1 Introduction

Large language models (LLMs) show an emergent behavior [Wei et al., 2022], known as in-context learning (ICL) [Brown et al., 2020, Kaplan et al., 2020], that they can learn (without updating any model parameters) a new unseen task from a few demonstrations of input-output pairs given at test time. While models trained with traditional supervised learning for a given task/mapping learn task-specific features that might be spurious or irrelevant for other tasks, the ICL ability enables the models to learn stronger representations with task-agnostic architecture and task-agnostic data and to efficiently learn diverse tasks via text interaction in a flexible way. Thus, it is often considered as one of the key capabilities contributing to the great success of LLMs.

At test time, in-context learning is known to operate in the following two modes: given an in-context task, the model (i) recalls similar functions and concepts learned (priorly) in the pretraining phase and (ii) smoothly adapts to and implicitly learns the (observed) in-context task [Xie et al., 2022, Raventós et al., 2024, Pan et al., 2023, Lin and Lee, 2024]. These dual Bayesian modes of ICL are often called (i) *task recognition* and (ii) *task learning* [Pan et al., 2023]. In other words, the pretrained model (i) knows the prior task distribution and (ii) has the ability to select and perform a proper task corresponding to the given demonstrations.

The power of each mode of ICL is not the same and one often dominates the other depending on many factors. To separately investigate the effects of the task recognition and task learning abilities, it has been proposed to use the experimental setups with noisy output labels (random or semantically irrelevant labels). Min et al. [2022], Lyu et al. [2023] show that randomly replacing output labels in the demonstrations does not significantly affect ICL performance which indicates that the task recognition dominates the task learning. On the other hands, Yoo et al. [2022], Wei et al. [2023], Shi et al. [2023] show that larger models are easily distracted by wrong or irrelevant demonstrations and Pan et al. [2023] show that larger models exhibits a better task learning ability than smaller ones and

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

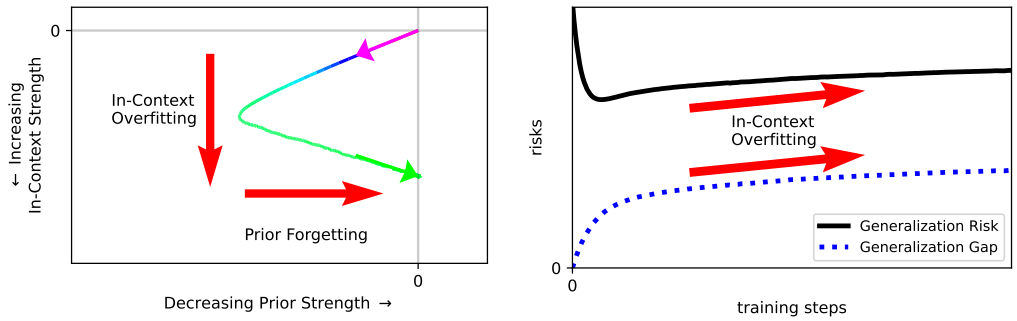

Figure 1: **Conceptual sketch of in-context overfitting and prior forgetting.** (Left) Following the gradient flow (the c-shaped trajectory (↻) from magenta to lime) in the parameter space, we observe a monotonic increase in *in-context strength* (↓) during training, which we call **in-context overfitting**, and a decrease in *prior strength* (→) in the later phase of the training, which we call **prior forgetting**. The exact meaning of the axes will become clear in Figure 2 (Bottom). (Right) We observe a monotonic increase in the *generalization gap* (blue dotted line) and a u-shaped curve of the *generalization risk* (black solid line). We also call these two phenomena **in-context overfitting** (↗) since the generalization gap is directly linked with the in-context strength.

it improves with more demonstrations. In other words, the task learning ability emerges as we scale up the model and it plays more important role than the task recognition for larger models.

As shown above, most studies attempt to analyze the behaviors of *the pretrained transformers at test time*, but to deeper understand the two modes of ICL, we need to investigate their emergence and dynamics *in the pretraining phase*. Thus, we raise the following questions:

How do in-context learning abilities emerge (and disappear) during pretraining?

To answer this question with an analytical understanding of the learning dynamics of the ICL abilities, we investigate the in-context random linear regression problem [Garg et al., 2022, Akyürek et al., 2023, Von Oswald et al., 2023a, Li et al., 2023b] with a simple linear-attention-based transformer [Schlag et al., 2021, Von Oswald et al., 2023a]. Moreover, to mathematically model a setup similar to the experimental settings considered in Min et al. [2022], Lyu et al. [2023], Yoo et al. [2022], Wei et al. [2023], Shi et al. [2023], Pan et al. [2023] and to disentangle the strengths of the task recognition and task learning abilities, we introduce new settings we call *demonstration-query task irrelevance and noncentral task model*. With these small modifications from previous works, we can derive a simple yet interesting dynamics regarding the ICL abilities.

To be specific, we show that, during the pretraining phase, the model first learns the task learning and task recognition abilities together in the beginning, but it (a) gradually forgets the task recognition ability to recall the priorly learned tasks and (b) relies more on the given context in the later phase, which we call (a) *prior forgetting* and (b) *in-context overfitting*, respectively. Figure 1 illustrates prior forgetting and in-context overfitting from the perspectives of model parameters (Left) and generalization loss (Right).

2 Related Work

After it was demonstrated that large transformers [Vaswani et al., 2017] such as GPT-3 can perform in-context learning [Brown et al., 2020, Kaplan et al., 2020], there has been a growing interest in understanding the underlying mechanisms of the ICL ability.

Garg et al. [2022] empirically show that, for in-context linear regression, the trained transformers match the performance of the optimal least squares estimator, i.e., the solution obtained by gradient flow on the in-context examples. This work is followed by Von Oswald et al. [2023a], Akyürek et al. [2023] in which the authors provide a construction of transformer that can implement gradient descent for in-context linear regression. Especially, the construction in Von Oswald et al. [2023a] only requires a single linear self-attention layer [Schlag et al., 2021] to implement a single iteration

of gradient descent. After that, many works consider linear transformer as a simple proxy for the softmax-based transformers to theoretically investigate their complex behaviors [Mahankali et al., 2024, Ahn et al., 2024a,b, Zhang et al., 2024].

Ahn et al. [2024a] show that, for a certain training objective over random instances of linear regression, the global minimizers can implement preconditioned gradient descent. Here, the parameter configuration of the minimizer is the same (up to a constant factor) as the one constructed in Von Oswald et al. [2023a]. Moreover, Zhang et al. [2024] show that such global minimizers can be achieved by running gradient flow under a certain initialization inducing the balancedness condition [Arora et al., 2019a, 2018, 2019b, Du et al., 2018] throughout the training.

Beyond the linear regression, Garg et al. [2022] also explore more complex function classes such as two-layer neural networks and decision trees, and empirically show that the trained transformer can in-context learn these function classes. Some theoretical work extend the in-context linear regression problem to exponential regression [Gao et al., 2023], softmax regression [Li et al., 2023a] or autoregressive learning [Von Oswald et al., 2023b, Sander et al., 2024, Zheng et al., 2024]. Moreover, Dai et al. [2023] empirically explore language models on real NLP tasks such as sentiment classification, topic classification, and natural language inference.

Likewise, many studies attempt to analyze the behaviors of the *pretrained* transformers at test time, e.g., the two Bayesian modes of ICL [Xie et al., 2022, Wies et al., 2023, Jiang, 2023, Wang et al., 2024b, Raventós et al., 2024, Pan et al., 2023, Lin and Lee, 2024] and their relative performance (especially when we scale the model) [Min et al., 2022, Lyu et al., 2023, Yoo et al., 2022, Wei et al., 2023, Shi et al., 2023, Pan et al., 2023, Shi et al., 2024]. On the other hand, we focus more on the emergence and rise-and-decline dynamics of the test-time performance of the two modes *during pretraining*.

To model the task distribution, Raventós et al. [2024], Lin and Lee [2024] introduce a probabilistic mixture model of multiple task groups with task-dependent input distributions, while we analyze a simple unimodal task distribution.

There are some more interesting work on the power balance between the two Bayesian modes of ICL. Wang et al. [2023] show a similar results with Min et al. [2022] for chain-of-thought prompting that invalid reasoning steps do not hurt performance on multi-step reasoning tasks. Reynolds and McDonell [2021] show that zero-shot prompts can match and even outperform few-shot prompts, which implies that the task recognition ability plays a more important role than from the task learning ability for some tasks.

Wang et al. [2024a] design a metric called competition intensity to explore the emergence of ICL and empirically show that the two modes of ICL are competitive during pretraining, while we measure the strengths of the two modes from the transformer's parameters and theoretically investigate how they emerge and disappear.

3 Settings

In this section, we first investigate the in-context random linear regression problem with a simple linear-attention-based transformer, following [Garg et al., 2022, Akyürek et al., 2023, Von Oswald et al., 2023a, Li et al., 2023b, Schlag et al., 2021], in Section 3.1. Then, we introduce some new settings and definitions to explore robustness to shift that we are given demonstrations irrelevant to the query task as similar to the empirical settings considered in Min et al. [2022], Lyu et al. [2023], Yoo et al. [2022], Wei et al. [2023], Shi et al. [2023], Pan et al. [2023] in Sections 3.2–3.3. See Appendix A for a quick reference for the notations.

3.1 In-Context Linear Regression with Linear Transformer

We train a transformer with the training set, which consists of the input context matrices and the corresponding target responses. The input context matrix

$$Z = \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \bar{X} & \boldsymbol{x}^{(n+1)} \\ \bar{Y} & 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}^{(1)} & \boldsymbol{x}^{(2)} & \cdots & \boldsymbol{x}^{(n)} & \boldsymbol{x}^{(n+1)} \\ y^{(1)} & y^{(2)} & \cdots & y^{(n)} & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(n+1)}$$

is generated by drawing $n+1$ d-dimensional covariates $\boldsymbol{x}^{(i)}$ and an in-context task vector $\boldsymbol{w}$ representing a linear function $f_{\boldsymbol{w}} : \boldsymbol{x} \mapsto \boldsymbol{w}^\top \boldsymbol{x}$ and computing the target responses $y^{(i)}$ as follows:

$$\boldsymbol{x}^{(i)} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_{\mathcal{X}}, \boldsymbol{w} \sim \mathcal{D}_{\mathcal{W}}, y^{(i)} = \boldsymbol{w}^\top \boldsymbol{x}^{(i)} \ (i = 1, \cdots, n+1),$$

where $\boldsymbol{x}^{(i)}, \boldsymbol{w} \in \mathbb{R}^d$, $y^{(i)} \in \mathbb{R}$, $X = [\bar{X} \quad \boldsymbol{x}^{(n+1)}] \in \mathbb{R}^{d \times (n+1)}$, $\bar{X} = [\boldsymbol{x}^{(1)} \ \cdots \ \boldsymbol{x}^{(n)}] \in \mathbb{R}^{d \times n}$, $Y = [\bar{Y} \quad 0] \in \mathbb{R}^{1 \times (n+1)}$, $\bar{Y} = [y^{(1)} \ \cdots \ y^{(n)}] \in \mathbb{R}^{1 \times n}$. Here, the $\boldsymbol{x}^{(i)}$'s for $i \leq n$ and $\boldsymbol{x}^{(n+1)}$ are called the in-context covariates and the query input, respectively.

We consider a single-layer linear transformer $T_{P,Q}$ with linear self-attention (LSA) [Schlag et al., 2021, Von Oswald et al., 2023a] parametrized by two matrices P and Q and residual connection [He et al., 2016]:

$$\begin{aligned} T_{P,Q}(Z) &= -\left[Z + \frac{1}{n}\text{LSA}_{P,Q}(Z)\right]_{d+1,n+1}, \\ \text{LSA}_{P,Q}(Z) &= PZMZ^\top QZ, \end{aligned} \tag{1}$$

where

$$P = \begin{bmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_d \\ \boldsymbol{p}^\top & \kappa \end{bmatrix}, \ Q = \begin{bmatrix} \bar{Q} & \mathbf{0}_d \\ \boldsymbol{q}^\top & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}, \ M = \begin{bmatrix} I_n & \mathbf{0}_n \\ \mathbf{0}_n^\top & 0 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}.$$

Here, the first d rows $P_{1:d,:}$ of P and the last column $Q_{:,d+1}$ of Q do not affect the output $T_{P,Q}(Z)$ of the transformer (see Appendix B), i.e., during the gradient-based training they remain the same as the initial values which are usually very small, and thus we simply put $P_{1:d,:} = \mathbf{0}_{d\times(d+1)}$ and $Q_{:,d+1} = \mathbf{0}_{d+1}$.

Note that the usual self-attention (SA) can be expressed as

$$\text{SA}_{W_Q, W_K, W_V; \sigma}(Z) = W_V Z M \sigma(Z^\top W_K^\top W_Q Z),$$

where σ is usually the column-wise softmax [Bahdanau et al., 2015, Vaswani et al., 2017]. Here, the linear self-attention is a special case of the self-attention when $P = W_V, Q = W_K^\top W_Q, \sigma(Z) = Z$.

We say the transformer *performs a task* $\hat{\boldsymbol{w}}$ if $T_{P,Q}(Z) = f_{\hat{\boldsymbol{w}}}(\boldsymbol{x}^{(n+1)}) = \hat{\boldsymbol{w}}^\top \boldsymbol{x}^{(n+1)}$. For the parametrization in (1), we can express the task

$$\hat{\boldsymbol{w}} = -(\bar{Q}^\top + \boldsymbol{q}\boldsymbol{w}^\top) G_x (\boldsymbol{p} + \kappa \boldsymbol{w}), \tag{2}$$

where $G_x = \frac{1}{n}\bar{X}\bar{X}^\top = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}^{(i)}\boldsymbol{x}^{(i)\top}$. See Appendix B for details.

To learn to predict the target response $y^{(n+1)} = \boldsymbol{w}^\top \boldsymbol{x}^{(n+1)}$ for the query input, the training loss (also called the training risk) is given as

$$L_{\text{train}}(P, Q) \equiv \mathbb{E}_{\boldsymbol{w}, X}\left[\left(\boldsymbol{w}^\top \boldsymbol{x}^{(n+1)} - T_{P,Q}(Z)\right)^2\right] = \mathbb{E}_{\boldsymbol{w}, X}\left[\left((\boldsymbol{w} - \hat{\boldsymbol{w}})^\top \boldsymbol{x}^{(n+1)}\right)^2\right].$$

3.2 Demonstration-Query Task Irrelevance and Noncentral Task Model

Zhang et al. [2024] consider the following three distribution shifts:

- task shifts: the tasks provided in the pretraining phase and the tasks at test time follow different distributions.
- query shifts: the in-context covariates $\boldsymbol{x}_{\text{test}}^{(i)}$ and the query input $\boldsymbol{x}_{\text{test}}^{(n+1)}$ follow different distributions.
- covariate shifts: the in-context inputs X in the pretraining phase and the test phase follow different distributions.

We depart from these distribution shifts and the vanilla setup (without any distribution shifts) considered in previous work, and introduce another class of scenario, which we call *demonstration-query task irrelevance*, where the demonstration task $\boldsymbol{w}$ and the query task $\boldsymbol{w}_{\boldsymbol{q}}$ (both at test time) are different (independent) but sampled from the same distribution:

Assumption 3.1 (Demonstration-Query Task Irrelevance)**.**

$$\boldsymbol{w}, \boldsymbol{w_q} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_{\mathcal{W}}.$$

In other words, we are given an input context matrix

$$Z_{\text{test}} = \begin{bmatrix} X_{\text{test}} \\ Y_{\text{test}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_{\text{test}}^{(1)} & \cdots & \boldsymbol{x}_{\text{test}}^{(n)} & \boldsymbol{x}_{\text{test}}^{(n+1)} \\ \boldsymbol{w}^\top \boldsymbol{x}_{\text{test}}^{(1)} & \cdots & \boldsymbol{w}^\top \boldsymbol{x}_{\text{test}}^{(n)} & 0 \end{bmatrix}$$

with the demonstration task $\boldsymbol{w}$, but the target response is determined by another independent query task $\boldsymbol{w_q}$, not by the demonstration task $\boldsymbol{w}$:

$$y_{\text{test}}^{(n+1)} = \boldsymbol{w_q}^\top \boldsymbol{x}_{\text{test}}^{(n+1)}.$$

Note that this assumption is designed not to model a real-world scenario but to disentangle the power of the two ICL abilities because it is similar to the empirical settings considered in Min et al. [2022], Lyu et al. [2023], Yoo et al. [2022], Wei et al. [2023], Shi et al. [2023], Pan et al. [2023] that we are given demonstrations irrelevant to the query task.

To investigate the robustness and generalization to the demonstration-query irrelevance, we consider the following generalization risk (also called test risk):

$$L_{\text{test}}(P, Q) \equiv \mathbb{E}_{\boldsymbol{w_q}, \boldsymbol{w}, X_{\text{test}}} \left[\left(\boldsymbol{w_q}^\top \boldsymbol{x}_{\text{test}}^{(n+1)} - T_{P,Q}(Z_{\text{test}}) \right)^2 \right]$$

and the generalization gap between the training risk and the generalization risk defined as follows:

$$\Delta L(P, Q) \equiv L_{\text{test}}(P, Q) - L_{\text{train}}(P, Q). \tag{3}$$

If the task learning plays a more important role than the task recognition, then the model relies more on the demonstration task that is irrelevant to the query task and thus it shows a higher generalization risk and gap. Therefore, by measuring the generalization risk or gap, we can separately analyze the task learning ability from the task recognition ability.

Similarly, we also want to separately investigate the power of the task recognition ability with which, given a demonstration task, the model can recall similar concepts from prior knowledge to infer the query task. However, the two tasks are independent. Therefore, we need to consider *shared concept* in the prior knowledge between the two tasks.

To this end, we simply examine a *non-centeral* task distribution $\mathcal{D}_{\mathcal{W}}$ with a nonzero mean $\boldsymbol{\mu} \neq \mathbf{0}_d$. Here, the task center $\boldsymbol{\mu}$ explains the prior task distribution in the sense that $\boldsymbol{w} = \boldsymbol{\mu} + \boldsymbol{s}$ and $\boldsymbol{w_q} = \boldsymbol{\mu} + \boldsymbol{s_q}$ share the (non-zero) prior knowledge $\boldsymbol{\mu}$ and they have their own knowledge $\boldsymbol{s}$ and $\boldsymbol{s_q}$. By using this prior knowledge represented by a nonzero vector, we can measure how much the model knows and utilizes this prior, i.e., the power of the task recognition ability (see (7)).

Assumption 3.2 (Isotropic Covariate and Noncentral Task)**.** We assume that

(i) the covariate distribution is $\mathcal{D}_{\mathcal{X}} = \mathcal{N}(\mathbf{0}_d, I_d)$ and

(ii) the task distribution $\mathcal{D}_{\mathcal{W}} = \mathcal{N}(\boldsymbol{\mu}, \Sigma_{\mathcal{W}})$ is noncentral with $\|\boldsymbol{\mu}\| = 1$ and isotropic with $\Sigma_{\mathcal{W}} = \sigma^2 I_d$.

Here, we name $b \equiv \operatorname{tr}(\Sigma_{\mathcal{W}}) = \sigma^2 d$ *the task dispersion*.

Under the above isotropic covariate assumption (i), we have

$$L_{\text{train}}(P, Q) = \mathbb{E}_{\boldsymbol{w}, X} \left[\left((\boldsymbol{w} - \hat{\boldsymbol{w}})^\top \boldsymbol{x}^{(n+1)} \right)^2 \right] = \mathbb{E}_{\boldsymbol{w}, \bar{X}} [\|\boldsymbol{w} - \hat{\boldsymbol{w}}\|^2] \tag{4}$$

$$L_{\text{test}}(P, Q) = \mathbb{E}_{\boldsymbol{w_q}, \boldsymbol{w}, X} \left[\left((\boldsymbol{w_q} - \hat{\boldsymbol{w}})^\top \boldsymbol{x}^{(n+1)} \right)^2 \right] = \mathbb{E}_{\boldsymbol{w_q}, \boldsymbol{w}, \bar{X}} [\|\boldsymbol{w_q} - \hat{\boldsymbol{w}}\|^2]. \tag{5}$$

Note that we assume the isotropic covariate assumption (i) for simplicity which can be easily relaxed to obtain similar equations with (4) and (5) up to a constant multiple, since $\mathbb{E}\left[\left(\boldsymbol{a}^\top \boldsymbol{x}^{(n+1)} \right)^2 \right] = \operatorname{tr}(\Sigma_{\mathcal{X}}) \mathbb{E}\left[\|\boldsymbol{a}\|^2 \right]$ if the covariate has zero-mean and covariance of $\Sigma_{\mathcal{X}}$ instead of I_d.

Moreover, with the noncentral task assumption (ii), we define the following generalization risks:

Definition 3.3 (Generalization Risk at Zero)**.** If the transformer performs the zero-task $\hat{\boldsymbol{w}} = \mathbf{0}_d$, i.e., $T_{P,Q}(\boldsymbol{x}^{(n+1)}) = 0$, then the corresponding generalization risk is called *the generalization risk at zero* defined as

$$L_0 \equiv \mathbb{E}_{\boldsymbol{w_q}}\left[\|\boldsymbol{w_q}\|^2\right] = \|\boldsymbol{\mu}\|^2 + \operatorname{tr}(\Sigma_{\mathcal{W}}) = 1 + b.$$

When the parameters are near the small initialization, the generalization risk is about L_0.

Definition 3.4 (Generalization Risk at Random)**.** If the transformer performs a random task $\boldsymbol{w'} \sim \mathcal{D}_{\mathcal{W}}$, i.e., $T_{P,Q}(\boldsymbol{x}^{(n+1)}) = \boldsymbol{w'}^{\top}\boldsymbol{x}^{(n+1)}$, then the expected generalization risk over $\boldsymbol{w'} \sim \mathcal{D}_{\mathcal{W}}$ is called *the generalization risk at random* defined as

$$L_r \equiv \mathbb{E}_{\boldsymbol{w_q},\boldsymbol{w'}}\left[\|\boldsymbol{w_q} - \boldsymbol{w'}\|^2\right] = 2\operatorname{tr}(\Sigma_{\mathcal{W}}) = 2b.$$

Definition 3.5 (Generalization Risk at Optimum)**.** When the transformer in (1) is optimally trained with the parameter (P^*, Q^*) that minimizes the training loss L_{train}, the train/test risks are called *the train/test risks at optimum* defined as

$$L^*_{\text{train}} \equiv L_{\text{train}}(P^*, Q^*), \quad L^*_{\text{test}} \equiv L_{\text{test}}(P^*, Q^*),$$

and the gap between the two is denoted by $\Delta L^* \equiv L^*_{\text{test}} - L^*_{\text{train}}$. Note that L^*_{test} is not optimal generalization risk.

3.3 Two-Parameter Transformer and Prior/In-Context Strength

Given $\boldsymbol{\mu} \in \mathbb{R}^d$, we consider the following simple transformers parametrization with two scalars α and κ:

$$P = \begin{bmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_d \\ \alpha\boldsymbol{\mu}^{\top} & \kappa \end{bmatrix}, Q = \begin{bmatrix} I_d & \mathbf{0}_d \\ \mathbf{0}_d^{\top} & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)} \tag{6}$$

which, if we put $\boldsymbol{\mu} = \mathbf{0}_d$, reduces to the parameter construction in Von Oswald et al. [2023a] to implement a single step of gradient descent. From now on, we will use the two-parameter notation $T_{\boldsymbol{\theta}}$ with $\boldsymbol{\theta} = [\alpha, \kappa]^{\top}$ instead of $T_{(P,Q)}$.

The two-parameter transformer $T_{\boldsymbol{\theta}}$ *performs* the following task $\hat{\boldsymbol{w}}$, i.e., $T_{\boldsymbol{\theta}}(\boldsymbol{x}^{(n+1)}) = \hat{\boldsymbol{w}}^{\top}\boldsymbol{x}^{(n+1)}$:

$$\hat{\boldsymbol{w}} = -G_x(\alpha\boldsymbol{\mu} + \kappa\boldsymbol{w}), \tag{7}$$

$$-\alpha : \text{prior strength}, -\kappa : \text{ in-context strength}.$$

Here α and κ are the weights of the task center $\boldsymbol{\mu}$ (independent of the demonstrations) and the in-context task $\boldsymbol{w}$, respectively. Thus, we refer to $-\alpha$ and $-\kappa$ (or just α and κ) as *the (semantic/task) prior strength* and *the in-context strength*, respectively. For example, if α is much larger than κ in magnitude, then $\hat{\boldsymbol{w}}$ ignores the demonstrations from $\boldsymbol{w}$, relying heavily on the task prior $\mathcal{D}_{\mathcal{W}}$ (task recognition) represented by $\boldsymbol{\mu}$. On the other hand, if κ is much larger than α in magnitude, then $\hat{\boldsymbol{w}}$ only relies on the demonstration task $\boldsymbol{w}$ (task learning) and the model cannot recover irrelevant query task.

This two-parameter model enables us to disentangle the two modes of ICL, the task recognition and task learning abilities, with the two distinct parameters, the prior strength α and in-context strength κ, respectively. Thus, in the next section, we will explore the evolution of these two parameters.

4 Main Results

In this section, first we show that our objective function is quadratic (Theorem 4.1) and then, for this quadratic loss, the gradient flow is a linear ODE which shows a simple dynamics (Theorem 4.2). Using this dynamics, we can analyze how the prior/in-context strengths and generalization gap/risk evolve during pretraining. Moreover, we also discuss how the value of task dispersion b affects the learning dynamics (Theorem 4.3).

4.1 Training Dynamics

The training loss function for the two-parameter transformer, from (4) and (7), can be expressed as follows:

$$L_{\text{train}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{w},\bar{X}}[\|\boldsymbol{w} - \hat{\boldsymbol{w}}\|^2] = \mathbb{E}_{\boldsymbol{w},\bar{X}}[\|\boldsymbol{w} + G_x(\alpha\boldsymbol{\mu} + \kappa\boldsymbol{w})\|^2].$$

From this equation, it can be easily shown that the training loss function is quadratic with respect to the two variables α and κ. The following theorem states the details of this quadratic loss function:

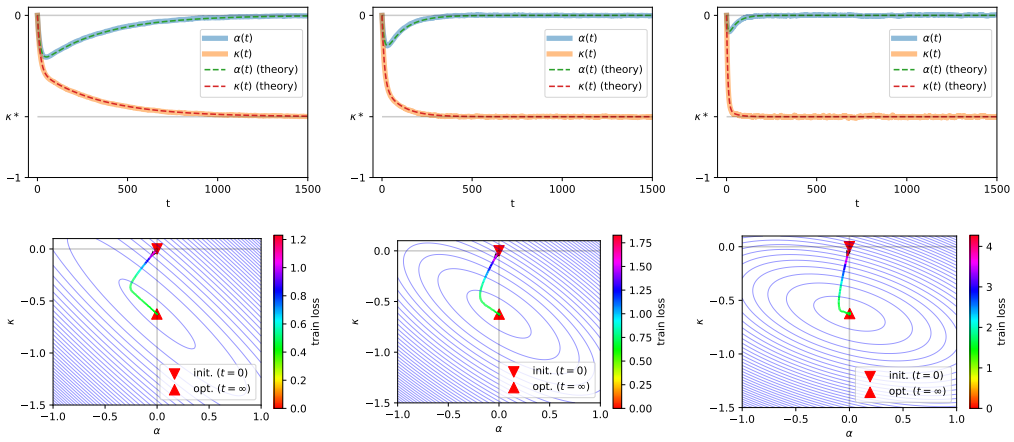

Figure 2: **Evolution of the two parameters, α and κ, for different b's.** We train the two-parameter transformer using SGD with learning rate of 0.01 and batch size of 4,000. We also use $n = 10, d = 5$, and $\sigma = 0.2, 0.4, 0.8$, i.e., the task dispersion $b = \sigma^2 d = 0.2, 0.8, 3.2$ (from Left to Right). Top: Empirical results with SGD (solid lines) and theoretical results of (11) and (12) with gradient flow (dashed lines). Bottom: Training trajectory (from initialization ▼ to the optimum ▲) drawn with quadratic loss landscape (elliptic level sets) on the $\alpha\kappa$-plane. The color of trajectory indicates the training loss value. See Figure 1 (Left) together.

Theorem 4.1 (Quadratic Loss)**.** *For the two-parameter transformer, the training loss is quadratic with respect to the parameter $\boldsymbol{\theta} = [\alpha, \kappa]^\top \in \mathbb{R}^2$:*

$$L_{train}(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top C_2 \boldsymbol{\theta} + C_1^\top \boldsymbol{\theta} + C_0, \text{ where} \tag{8}$$

$$C_2 = 2\frac{n+d+1}{n}\begin{bmatrix} 1 & 1 \\ 1 & 1+b \end{bmatrix} \in \mathbb{R}^{2\times 2}, C_1 = 2\begin{bmatrix} 1 \\ 1+b \end{bmatrix} \in \mathbb{R}^2, C_0 = 1 + b \in \mathbb{R},$$

and $b = \mathrm{tr}(\Sigma_{\mathcal{W}}) = \sigma^2 d$. The training loss $L_{train}(\boldsymbol{\theta})$ is minimized at $\boldsymbol{\theta} = \boldsymbol{\theta}^*$, where

$$\boldsymbol{\theta}^* = \begin{bmatrix} \alpha^* \\ \kappa^* \end{bmatrix} = \begin{bmatrix} 0 \\ -\frac{n}{n+d+1} \end{bmatrix} \text{ and } L^*_{train} = \frac{d+1}{n+d+1} L_0. \tag{9}$$

Note that, when $\boldsymbol{\mu} = \mathbf{0}_d$, the global minimum θ^* reduces to the one in Ahn et al. [2024a] as a special case. The proof is deferred to Appendix D.

Figure 2 (Bottom) shows the quadratic loss landscape with elliptic level sets and a unique minimum (since the Hessian matrix C_2 is positive-definite) at $\boldsymbol{\theta}^*$ on the κ-axis (shown with ▲). The geometry of the loss landscape highly affects the learning dynamics as will be detailed in the following theorem and the later sections.

Theorem 4.2 (Training Dynamics)**.** *Under the same setting of Theorem 4.1, by solving the following linear differential equation (gradient flow) starting from $\boldsymbol{\theta}(0) = 0$:*

$$\dot{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{\theta}} L_{train}(\boldsymbol{\theta}) = -C_2\boldsymbol{\theta} - C_1,$$

we can obtain the following solution:

$$\boldsymbol{\theta}(t) = ce^{-\lambda_+ t}\boldsymbol{v}_+ - ce^{-\lambda_- t}\boldsymbol{v}_- + \boldsymbol{\theta}^*, \tag{10}$$
$$c = \frac{n}{(n+d+1)\sqrt{4+b^2}},$$

where

$$\lambda_\pm = \frac{2+b\pm\sqrt{4+b^2}}{2} \text{ and } \boldsymbol{v}_\pm = \left[1, \frac{b\pm\sqrt{4+b^2}}{2}\right]^\top$$

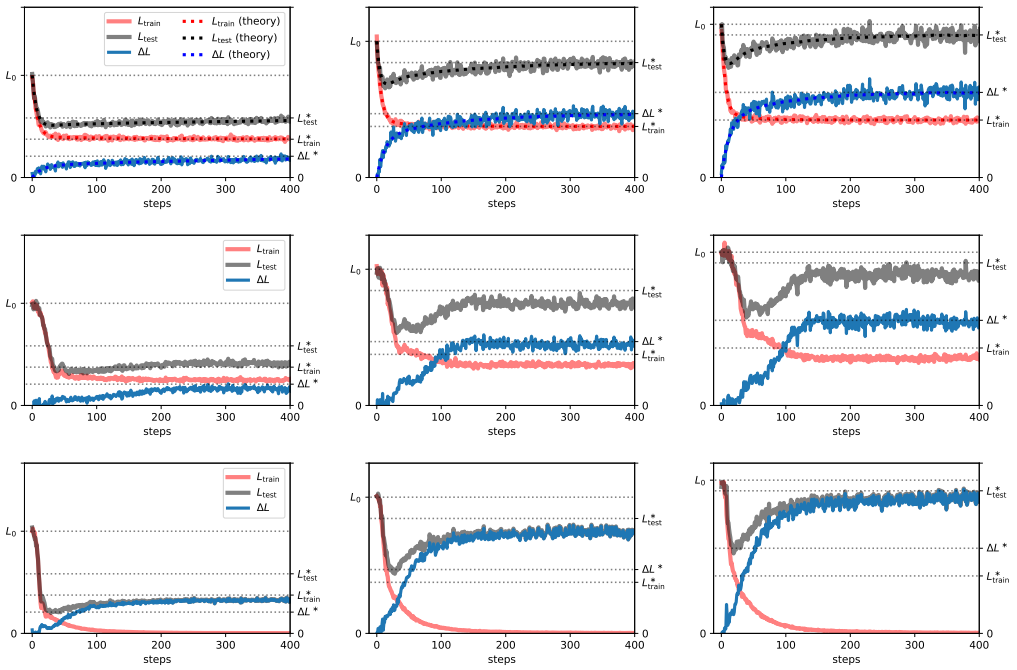

Figure 3: **ICL loss curves for different parameterizations (each row) and different b's (each column).** We train (Top) the two-parameter transformer in (6), (Middle) the full-parameter transformer in (1), and (Bottom) a practical multi-head ($h = 8$) multi-layer ($\ell = 12$) transformer with softmax attention and residual connection. We use $n = 10, d = 5$, and $\sigma^2 = 0.04, 0.12, 0.16$, i.e., $b = 0.2, 0.6, 0.8$ (from Left to Right). It shows empirical results with gradient descent (solid lines) and theoretical results with gradient flow (colored dotted lines). The horizontal dotted lines indicate the four equations, $L_0 = 1 + b, L^*_{\text{train}} = \frac{d+1}{n+d+1} L_0, L^*_{\text{test}} = \frac{(d+1)L_0 + nL_r}{n+d+1}, \Delta L^* = -2b\kappa^*$. The training loss (red) and generalization loss (black) start from L_0 and the training loss monotonically decreases to L^*_{train}. On the other hand, the generalization gap (blue) monotonically increases to ΔL^*. As a result, the generalization loss shows a u-shaped curve and converges to L^*_{test}. We use learning rate of 0.01 and batch size of 4,000. We use SGD for the two-parameter transformer, but use AdamW for the full-parameter transformer and practical models. See Figure 1 (Right) together.

are the eigenvalues and the corresponding eigenvectors of C_2, respectively. To be specific, we have

$$\alpha(t) = -2ce^{-\beta_b t} \sinh(\gamma_b t) \le 0, \tag{11}$$

$$\kappa(t) = 2ce^{-\beta_b t} \cosh(\gamma_b t - \tau_b) + \kappa^* \ge \kappa^*, \tag{12}$$

where $\beta_b = \frac{2+b}{2}, \gamma_b = \frac{\sqrt{4+b^2}}{2}$ *and* $\tau_b = \operatorname{arctanh}\left(\frac{b}{\sqrt{4+b^2}}\right)$.

Note that the context length n can only scale the dynamics through the constant c and does not affect the shape of the dynamics. The proof is deferred to Appendix D.

Figure 2 shows $\alpha(t), \kappa(t)$ (Top) and the trajectory $\boldsymbol{\theta}(t)$ on the $\alpha\kappa$-plane (Bottom). Our theory (11) and (12) with gradient flow (Top, dashed) explains the empirical results with SGD (Top, solid) well.

4.2 Prior Forgetting and In-Context Overfitting

Theorem 4.2 illustrates the analytical dynamics of the two parameters $\alpha(t)$ and $\kappa(t)$. First, from (11), we can show that $\alpha(t)$ decreases till $t \le t_0$ for some $t_0 > 0$ and then increases back to 0 as t goes to ∞ since the derivative

$$\alpha'(t) = -2ce^{-\beta_b t}(-\beta_b \sinh(\gamma_b t) + \gamma_b \cosh(\gamma_b t)) = 2ce^{-\beta_b t} \sinh(\gamma_b t - \bar{\tau}_b)$$

changes its sign (from negative to positive) at $t = t_0 = \bar{\tau}_b/\gamma_b$, i.e., $\alpha'(t_0) = 0$, where $\bar{\tau}_b = \mathrm{arctanh}(\gamma_b/\beta_b) > 0$. Here the critical point t_0 is a decreasing function of b (see Figure 5 in Appendix E), and as expected, Figure 2 (Top) shows an earlier critical point as we increase b.

Second, from (12), we can show that $\kappa(t)$ monotonically decreases to $\kappa^* = -n/(n+d+1)$ since

$$\kappa'(t) = 2ce^{-\beta_b t}\left(-\beta_b \cosh(\gamma_b t - \tau_b) + \gamma_b \sinh(\gamma_b t - \tau_b)\right) = -2ce^{-\beta_b t}\cosh(\gamma_b t - \tau_b - \bar{\tau}_b) < 0.$$

Figure 2 (Top) demonstrates the above two phenomena: (i) $\alpha(t)$ decreases in the beginning and then increases back to 0 and (ii) $\kappa(t)$ monotonically decreases to κ^*. In other words, (i) the prior strength $-\alpha$ gets stronger in the beginning, but it gets weaker, reaching 0 in the later phase, which we call *prior forgetting*, and (ii) the in-context strength $-\kappa$ increases and it leads to the increase in the generalization gap (we will show this in the next section) which we call *in-context overfitting*.

4.3 Generalization Gap

Moreover, for the generalization gap of the model during the pretraining, from (3), (4), (5), (7), we have

$$\Delta L(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{w_q},\boldsymbol{w},\bar{X}}[\|\boldsymbol{w_q} - \hat{\boldsymbol{w}}\|^2 - \|\boldsymbol{w} - \hat{\boldsymbol{w}}\|^2] = 2\mathbb{E}_{\boldsymbol{w_q},\boldsymbol{w},\bar{X}}[(\boldsymbol{w_q} - \boldsymbol{w})^\top G_x(\alpha\boldsymbol{\mu} + \kappa\boldsymbol{w})] = -2b\kappa \tag{13}$$

since $\mathbb{E}[\boldsymbol{w_q}^\top \boldsymbol{w}] - \mathbb{E}[\boldsymbol{w}^\top \boldsymbol{w}] = \|\boldsymbol{\mu}\|^2 - (\|\boldsymbol{\mu}\|^2 + b) = -b$. Thus, the generalization gap increases with the in-context strength $-\kappa$.

Figure 3 (blue curves) shows the increase in the generalization gap ΔL during pretraining, which we also call *in-context overfitting*. It empirically shows that the full-parameter transformer in (1) (bottom) behaves similar to our analytical model (dotted lines) for the two-parameter transformer, also demonstrating in-context overfitting. Note that, without the two-parameter restriction, the training and generalization losses become a little smaller.

4.4 Different Dynamics Depending on Task Dispersion b

For the trajectory $\boldsymbol{\theta}(t)$ in (10), we have the derivative

$$\boldsymbol{\theta}'(t) = \underbrace{-c\lambda_+ e^{-\lambda_+ t}}\, \boldsymbol{v}_+ + \underbrace{c\lambda_- e^{-\lambda_- t}}\, \boldsymbol{v}_-$$

with the two orthogonal components $\boldsymbol{v}_+$ and $\boldsymbol{v}_-$. At time t, the coefficients are $\Theta(\lambda\exp(-t\lambda))$ with respect to each λ and we have

$$\frac{\partial}{\partial\lambda}\lambda\exp(-t\lambda) = (1-t)\lambda\exp(-t\lambda)\begin{cases} > 0, & \text{if } t < 1, \\ < 0, & \text{if } t > 1 \end{cases}.$$

Thus, as shown in Figure 2 (Bottom), in the beginning ($t < 1$), the flow is mostly aligned with the sharper direction $-\boldsymbol{v}_+$ of the quadratic loss landscape because $\lambda_+ > \lambda_-$. On the other hand, in the later phase ($t > 1$), the opposite holds and the flow gets more aligned with the flatter one $\boldsymbol{v}_-$.

The following theorem provides intuitive pictures for the loss landscape and the gradient flow, and their changes due to variations in the value of task dispersion b.

Theorem 4.3 (Evolution of Task Recognition/Learning)**.** *The eigenvalues $\lambda_\pm$ and the corresponding eigenvectors $\boldsymbol{v}_\pm$ of the Hessian C_2 of the training loss L_{train} are:*
(i) for a small $b = \sigma^2 d \ll 1$,

$$\lambda_+ = 2 + \Theta(b) \approx 2, \qquad \boldsymbol{v}_+^\top = [1, 1 + \Theta(b)] \approx [1, 1]\,,$$
$$\lambda_- = \Theta(b) \ll \lambda_+, \qquad \boldsymbol{v}_-^\top = [1, -1 + \Theta(b)] \approx [1, -1]\,,$$

and, (ii) for a large $b \gg 1$,

$$\lambda_+ = \Theta(b), \qquad \boldsymbol{v}_+^\top = [1, \Theta(b)] \parallel\!\sim [0, 1]\,,$$
$$\lambda_- = 1 + \Theta(1/b) \ll \lambda_+, \qquad \boldsymbol{v}_-^\top = [1, \Theta(1/b)] \parallel\!\sim [1, 0]\,,$$

respectively, where $\boldsymbol{v} \parallel\!\sim \boldsymbol{u}$ means that $\boldsymbol{v}$ and $\boldsymbol{u}$ are approximately parallel.

The proof is deferred to Appendix D.

Figure 2 (Bottom) shows that when b is small, e.g., $b = 0.2$ (Left), the two main directions of the elliptic level sets are $\boldsymbol{v}_+ \approx [1, 1]^\top$ and $\boldsymbol{v}_- \approx [1, -1]^\top$ and the gradient flow is first aligned with the sharper direction $\boldsymbol{v}_+$ then later it follows the flatter direction $\boldsymbol{v}_-$. And as b gets larger, e.g., $b = 0.8$ (Right), the two directions becomes more like $\boldsymbol{v}_+ \mathrel{\|\sim} [1, 0]^\top$ and $\boldsymbol{v}_- \mathrel{\|\sim} [1, 0]^\top$.

Figure 2 (Top) shows that, as b gets larger (from Left to Right), $\alpha(t)$ gets to have relatively smaller dip as the sharper direction is nearly aligned with $[0, 1]^\top$ orthogonal to the direction increasing α.

4.5 Generalization Risk

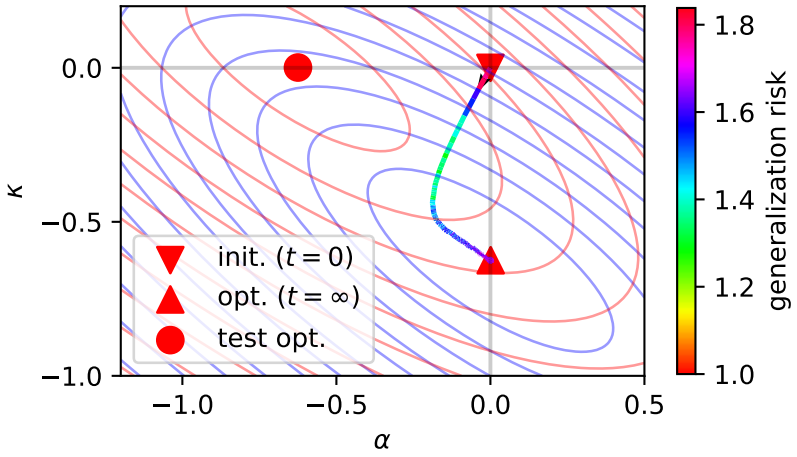

Figure 4: **Generalization risk (red ellipses) and training risk (blue ellipses).** The minimizers for the generalization risk and training risk are $[\kappa^*, 0]^\top$ (●) and $[0, \kappa^*]^\top$ (▲), respectively. The color of trajectory indicates the generalization risk value which decreases in the beginning, but increases back (in-context overfitting). See Figure 3 together.

From (13), the generalization risk $L_{\text{test}}(\boldsymbol{\theta}) = L_{\text{train}}(\boldsymbol{\theta}) - 2b\kappa$ can also be expressed as a quadratic form similar to (8):

$$L_{\text{test}}(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^\top C_2 \boldsymbol{\theta} + C_1'^\top \boldsymbol{\theta} + C_0 \tag{14}$$

$$\text{with } C_1' = C_1 + [0, -2b]^\top = 2[1, 1]^\top \text{and } \arg\min_{\boldsymbol{\theta}} L_{\text{test}}(\boldsymbol{\theta}) = -C_2^{-1} C_1' = [\kappa^*, 0]^\top.$$

In addition, $L_{\text{test}}^* = L_{\text{train}}^* - 2b\kappa^* = \frac{(d+1)L_0 + nL_r}{n+d+1}$ is in between L_0 (at zero) and L_r (at random) from (9). If we are given many demonstrations with large n, then the trained model nearly performs the demonstration task which is a random task for the irrelevant query task.

Figure 4 visually demonstrates how the generalization risk behaves on the trajectory $\boldsymbol{\theta}(t)$. From (14), the generalization risk has the minimizer (●) on the α-axis, while the training risk (▲) on the κ-axis. With the two elliptic level sets for the generalization risk (red) and training risk (blue), we can expect that the generalization risk decreases as $\boldsymbol{\theta}(t)$ flows along the sharper direction $-\boldsymbol{v}_+$, and then it increases as $\boldsymbol{\theta}(t)$ get close to $\boldsymbol{\theta}^*$ following $\boldsymbol{v}_-$.

5 Conclusion

In this paper, we investigate how the two modes of ICL emerge and disappear during pretraining. By introducing new simple settings, demonstration-query task irrelevance and noncentral task distribution, we can separately analyze the two modes of ICL and show two interesting phenomena: prior forgetting and in-context overfitting. Due to the simplicity of the analysis, we hope that our insights will motivate the future work toward understanding ICL.

Acknowledgments

We thank the anonymous reviewers for insightful reviews. This work was partially supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grants (RS-2020-II201373, Artificial Intelligence Graduate School Program (Hanyang University); RS-2023-002206284, Artificial intelligence for prediction of structure-based protein interaction reflecting physicochemical principles), the National Research Foundation of Korea (NRF) grants (RS-2023-00244896, Implicit bias of optimization algorithms for robust generalization of deep learning; the BK21 FOUR (Fostering Outstanding Universities for Research) project; NRF-2024S1A5C3A02043653, Socio-Technological Solutions for Bridging the AI Divide: A Blockchain and Federated Learning-Based AI Training Data Platform) and Korea Institute for Advanced Study (KIAS) grant funded by the Korean government (MSIT).

References

K. Ahn, X. Cheng, H. Daneshmand, and S. Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36, 2024a.

K. Ahn, X. Cheng, M. Song, C. Yun, A. Jadbabaie, and S. Sra. Linear attention is (maybe) all you need (to understand transformer optimization). In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=0uI5415ry7.

E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0g0X4H8yN4I.

S. Arora, N. Cohen, and E. Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International conference on machine learning*, pages 244–253. PMLR, 2018.

S. Arora, N. Cohen, N. Golowich, and W. Hu. A convergence analysis of gradient descent for deep linear neural networks. In *International Conference on Learning Representations*, 2019a. URL https://openreview.net/forum?id=SkMQg3C5K7.

S. Arora, N. Cohen, W. Hu, and Y. Luo. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32, 2019b.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, 2015.

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

D. Dai, Y. Sun, L. Dong, Y. Hao, S. Ma, Z. Sui, and F. Wei. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.247. URL https://aclanthology.org/2023.findings-acl.247.

S. S. Du, W. Hu, and J. D. Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in neural information processing systems*, 31, 2018.

Y. Gao, S. Mahadevan, and Z. Song. An over-parameterized exponential regression. *arXiv preprint arXiv:2303.16504*, 2023.

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

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

H. Jiang. A latent space theory for emergent abilities in large language models. *arXiv preprint arXiv:2304.09960*, 2023.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

S. Li, Z. Song, Y. Xia, T. Yu, and T. Zhou. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*, 2023a.

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

Z. Lin and K. Lee. Dual operating modes of in-context learning. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=ElVHUWyL3n.

X. Lyu, S. Min, I. Beltagy, L. Zettlemoyer, and H. Hajishirzi. Z-icl: Zero-shot in-context learning with pseudo-demonstrations. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

A. V. Mahankali, T. Hashimoto, and T. Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8p3fu56lKc.

S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.759. URL https://aclanthology.org/2022.emnlp-main.759.

J. Pan, T. Gao, H. Chen, and D. Chen. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8298–8319, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.527. URL https://aclanthology.org/2023.findings-acl.527.

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

L. Reynolds and K. McDonell. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended abstracts of the 2021 CHI conference on human factors in computing systems*, pages 1–7, 2021.

M. E. Sander, R. Giryes, T. Suzuki, M. Blondel, and G. Peyré. How do transformers perform in-context autoregressive learning ? In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=kZbTkpnafR.

I. Schlag, K. Irie, and J. Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning*, pages 9355–9366. PMLR, 2021.

F. Shi, X. Chen, K. Misra, N. Scales, D. Dohan, E. H. Chi, N. Schärli, and D. Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR, 2023.

Z. Shi, J. Wei, Z. Xu, and Y. Liang. Why larger language models do in-context learning differently? In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 44991–45013. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/shi24f.html.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

J. Von Oswald, M. Schlegel, A. Meulemans, S. Kobayashi, E. Niklasson, N. Zucchet, N. Scherrer, N. Miller, M. Sandler, M. Vladymyrov, et al. Uncovering mesa-optimization algorithms in transformers. *arXiv preprint arXiv:2309.05858*, 2023b.

B. Wang, S. Min, X. Deng, J. Shen, Y. Wu, L. Zettlemoyer, and H. Sun. Towards understanding chain-of-thought prompting: An empirical study of what matters. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023.

X. Wang, X. Tang, W. X. Zhao, and J.-R. Wen. Investigating the pre-training dynamics of in-context learning: Task recognition vs. task learning. *arXiv preprint arXiv:2406.14022*, 2024a.

X. Wang, W. Zhu, M. Saxon, M. Steyvers, and W. Y. Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. *Advances in Neural Information Processing Systems*, 36, 2024b.

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

J. Wei, J. Wei, Y. Tay, D. Tran, A. Webson, Y. Lu, X. Chen, H. Liu, D. Huang, D. Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

N. Wies, Y. Levine, and A. Shashua. The learnability of in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=f3JNQd7CHM.

S. M. Xie, A. Raghunathan, P. Liang, and T. Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

K. M. Yoo, J. Kim, H. J. Kim, H. Cho, H. Jo, S.-W. Lee, S.-g. Lee, and T. Kim. Ground-truth labels matter: A deeper look into input-label demonstrations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2422–2437, 2022.

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

C. Zheng, W. Huang, R. Wang, G. Wu, J. Zhu, and C. Li. On mesa-optimization in autoregressively trained transformers: Emergence and capability. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=E2BYPreuU8.

A Some Notations

$d \in \mathbb{N}$	(covariate/task dimension)
$n \in \mathbb{N}$	(number of demonstrations)
$Z, Z_{\text{test}} \in \mathbb{R}^{(d+1)\times(n+1)}$	(input context matrix)
$X, X_{\text{test}} \in \mathbb{R}^{d\times(n+1)}, \bar{X} \in \mathbb{R}^{d\times n}$	
$Y, Y_{\text{test}} \in \mathbb{R}^{1\times(n+1)}, \bar{Y} \in \mathbb{R}^{1\times(n+1)}$	
$\boldsymbol{x}^{(i)}, \boldsymbol{x}_{\text{test}}^{(i)} \in \mathbb{R}^d$	(in-context covariate; $i = 1, 2, \cdots, n$)
$\boldsymbol{x}^{(n+1)}, \boldsymbol{x}_{\text{test}}^{(n+1)} \in \mathbb{R}^d$	(query input)
$\mathcal{D}_{\mathcal{X}} = \mathcal{N}(\mathbf{0}_d, I_d)$	(covariate distribution)
$G_x = \frac{1}{n}\bar{X}\bar{X}^\top = \sum_{i=1}^{n} \boldsymbol{x}^{(i)}\boldsymbol{x}^{(i)\top} \in \mathbb{R}^{d\times d}$	((uncentered) sample covariance matrix)
$\boldsymbol{w} \in \mathbb{R}^d$	(in-context task)
$\boldsymbol{w_q} \in \mathbb{R}^d$	(query task)
$\hat{\boldsymbol{w}} \in \mathbb{R}^d$	(task performed by the transformer)
$\mathcal{D}_{\mathcal{W}} = \mathcal{N}(\boldsymbol{\mu}, \Sigma_{\mathcal{W}})$	(task distribution)
$\boldsymbol{\mu} \in \mathbb{R}^d$	(task center)
$\Sigma_{\mathcal{W}} = \sigma^2 I_d$	(task covariance)
$b = \text{tr}(\Sigma_{\mathcal{W}}) = \sigma^2 d \in \mathbb{R}$	(task dispersion)
$y^{(i)} = \boldsymbol{w}^\top \boldsymbol{x}^{(i)} \in \mathbb{R}$	(target response; $i = 1, 2, \cdots, n+1$)
$y_{\text{test}}^{(i)} = \boldsymbol{w}^\top \boldsymbol{x}_{\text{test}}^{(i)} \in \mathbb{R}$	($i = 1, 2, \cdots, n$)
$y_{\text{test}}^{(n+1)} = \boldsymbol{w_q}^\top \boldsymbol{x}_{\text{test}}^{(n+1)} \in \mathbb{R}$	
$T_{P,Q} : R^{(d+1)\times(n+1)} \to \mathbb{R}$	(single-layer (full-parameter) linear Transformer)
$T_{\boldsymbol{\theta}} : R^{(d+1)\times(n+1)} \to \mathbb{R}$	(two-parameter Transformer)
$\text{LSA}_{P,Q}(Z) = PZMZ^\top QZ$	(linear self-attention)
$P = \begin{bmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_d \\ \boldsymbol{p}^\top & \kappa \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$	(P for the full-parameter transformer)
$P = \begin{bmatrix} \mathbf{0}_{d\times d} & \mathbf{0}_d \\ \alpha\boldsymbol{\mu}^\top & \kappa \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$	(P for the two-parameter transformer)
$Q = \begin{bmatrix} \bar{Q} & \mathbf{0}_d \\ \boldsymbol{q}^\top & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$	(Q for the full-parameter transformer)
$Q = \begin{bmatrix} I_{d\times d} & \mathbf{0}_d \\ \mathbf{0}_d^\top & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$	(Q for the two-parameter transformer)
$M = \begin{bmatrix} I_n & \mathbf{0}_n \\ \mathbf{0}_n^\top & 0 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$	(masking matrix)
$\boldsymbol{\theta} = [\alpha, \kappa]^\top \in \mathbb{R}^2$	(two-parameter parameter)
$-\alpha \in \mathbb{R}$	(prior strength)
$-\kappa \in \mathbb{R}$	(in-context strength)
$L_{\text{train}} \in \mathbb{R}$	(training loss)
$L_{\text{test}} \in \mathbb{R}$	(generalization/test loss)
$\Delta L = L_{\text{test}} - L_{\text{train}} \in \mathbb{R}$	(generalization gap)

B Transformer Parameterization and Task $\hat{w}$

As shown below, the first d rows of P and the last column of Q do not affect the output $T_{P,Q}(Z)$, so we put them $\mathbf{0}$. We also show that the transformer performs $\hat{\boldsymbol{w}} = -(\bar{Q}^\top + \boldsymbol{q}\boldsymbol{w}^{(i)\top})G_x(\boldsymbol{p} + \kappa\boldsymbol{w}^{(i)})$.

$$
\begin{aligned}
Z + \frac{1}{n}\mathrm{LSA}_{P,Q}(Z) &= Z + \frac{1}{n}PZMZ^\top QZ \\
&= \begin{bmatrix} X \\ Y \end{bmatrix} + \frac{1}{n}\begin{bmatrix} \dots & \dots \\ \boldsymbol{p}^\top & \kappa \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix}\begin{bmatrix} I_n & \mathbf{0}_n \\ \mathbf{0}_n^\top & 0 \end{bmatrix}\begin{bmatrix} X^\top & Y^\top \end{bmatrix}\begin{bmatrix} \bar{Q} & \dots \\ \boldsymbol{q}^\top & \dots \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix} \\
&= \begin{bmatrix} X \\ Y \end{bmatrix} + \frac{1}{n}\begin{bmatrix} \dots & \dots \\ \boldsymbol{p}^\top\bar{X} + \kappa\bar{Y} & \boldsymbol{p}^\top\boldsymbol{x}^{(n+1)} \end{bmatrix}\begin{bmatrix} I_n & \mathbf{0}_n \\ \mathbf{0}_n^\top & 0 \end{bmatrix}\begin{bmatrix} X^\top & Y^\top \end{bmatrix}\begin{bmatrix} \bar{Q} & \dots \\ \boldsymbol{q}^\top & \dots \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix} \\
&= \begin{bmatrix} X \\ Y \end{bmatrix} + \frac{1}{n}\begin{bmatrix} \dots & \mathbf{0}_d \\ \boldsymbol{p}^\top\bar{X} + \kappa\bar{Y} & 0 \end{bmatrix}\begin{bmatrix} \bar{X}^\top & \bar{Y}^\top \\ \boldsymbol{x}^{(n+1)\top} & 0 \end{bmatrix}\begin{bmatrix} \bar{Q} & \dots \\ \boldsymbol{q}^\top & \dots \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix} \\
T_{P,Q}(Z) &= -\left[Z + \frac{1}{n}\mathrm{LSA}_{P,Q}(Z)\right]_{d+1,n+1} \\
&= -Y\boldsymbol{e}_{n+1} - \frac{1}{n}\begin{bmatrix} \boldsymbol{p}^\top\bar{X} + \kappa\bar{Y} & 0 \end{bmatrix}\begin{bmatrix} \bar{X}^\top & \bar{Y}^\top \\ \boldsymbol{x}^{(n+1)\top} & 0 \end{bmatrix}\begin{bmatrix} \bar{Q} & \dots \\ \boldsymbol{q}^\top & \dots \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix}\boldsymbol{e}_{n+1} \\
&= -\frac{1}{n}\begin{bmatrix} \boldsymbol{p}^\top\bar{X} + \kappa\bar{Y} & 0 \end{bmatrix}\begin{bmatrix} \bar{X}^\top & \bar{Y}^\top \\ \boldsymbol{x}^{(n+1)\top} & 0 \end{bmatrix}\begin{bmatrix} \bar{Q} & \dots \\ \boldsymbol{q}^\top & \dots \end{bmatrix}\begin{bmatrix} \boldsymbol{x}^{(n+1)} \\ 0 \end{bmatrix} \\
&= -\frac{1}{n}\begin{bmatrix} \boldsymbol{p}^\top\bar{X} + \kappa\bar{Y} & 0 \end{bmatrix}\begin{bmatrix} \bar{X}^\top\bar{Q} + \bar{Y}^\top\boldsymbol{q}^\top & \dots \\ \boldsymbol{x}^{(n+1)\top}\bar{Q} & \dots \end{bmatrix}\begin{bmatrix} \boldsymbol{x}^{(n+1)} \\ 0 \end{bmatrix} \\
&= -\frac{1}{n}\boldsymbol{x}^{(n+1)\top}\left[(\bar{Q}^\top\bar{X} + \boldsymbol{q}\bar{Y})(\bar{X}^\top\boldsymbol{p} + \kappa\bar{Y}^\top)\right] \\
&= -\frac{1}{n}\boldsymbol{x}^{(n+1)\top}\left[\bar{Q}^\top\bar{X}\bar{X}^\top\boldsymbol{p} + \kappa\bar{Q}^\top\bar{X}\bar{Y}^\top + \boldsymbol{q}\bar{Y}\bar{X}^\top\boldsymbol{p} + \kappa\boldsymbol{q}\bar{Y}\bar{Y}^\top\right] \\
&= \boldsymbol{x}^{(n+1)\top}\left[-\frac{1}{n}\sum_{i=1}^n \bar{Q}^\top\bar{\boldsymbol{x}}^{(i)}\boldsymbol{x}^{(i)\top}\boldsymbol{p} + \kappa y^{(i)}\bar{Q}^\top\boldsymbol{x}^{(i)} + y^{(i)}\boldsymbol{q}\boldsymbol{x}^{(i)\top}\boldsymbol{p} + \kappa\boldsymbol{q}y^{(i)2}\right] \\
&= \boldsymbol{x}^{(n+1)\top}\left[-\frac{1}{n}\sum_{i=1}^n \bar{Q}^\top\bar{\boldsymbol{x}}^{(i)}\boldsymbol{x}^{(i)\top}\boldsymbol{p} + \kappa\boldsymbol{w}^\top\boldsymbol{x}^{(i)}\bar{Q}^\top\boldsymbol{x}^{(i)} + \boldsymbol{w}^\top\boldsymbol{x}^{(i)}q\boldsymbol{x}^{(i)\top}\boldsymbol{p} + \kappa\boldsymbol{q}\boldsymbol{w}^\top\boldsymbol{x}^{(i)}\boldsymbol{x}^{(i)\top}\boldsymbol{w}\right] \\
&= \boldsymbol{x}^{(n+1)\top}\left[-\frac{1}{n}\sum_{i=1}^n (\bar{Q}^\top\bar{\boldsymbol{x}}^{(i)} + \boldsymbol{q}\boldsymbol{w}^\top\boldsymbol{x}^{(i)})(\boldsymbol{x}^{(i)\top}\boldsymbol{p} + \kappa\boldsymbol{x}^{(i)\top}\boldsymbol{w})\right] \\
&= \boldsymbol{x}^{(n+1)\top}\left[-\frac{1}{n}\sum_{i=1}^n (\bar{Q}^\top + \boldsymbol{q}\boldsymbol{w}^\top)\boldsymbol{x}^{(i)}\boldsymbol{x}^{(i)\top}(\boldsymbol{p} + \kappa\boldsymbol{w})\right] \\
&= \boldsymbol{x}^{(n+1)\top}\underbrace{\left[-(\bar{Q}^\top + \boldsymbol{q}\boldsymbol{w}^\top)G_x(\boldsymbol{p} + \kappa\boldsymbol{w})\right]}_{\hat{\boldsymbol{w}}}
\end{aligned}
$$

which proves the equation (2). Moreover, for the two-parameter transformer with $\bar{Q} = I_d, \boldsymbol{q} = \mathbf{0}_d, \boldsymbol{p} = \alpha\boldsymbol{\mu}$, we have

$$
\begin{aligned}
\hat{\boldsymbol{w}} &= -(\bar{Q}^\top + \boldsymbol{q}\boldsymbol{w}^\top)G_x(\boldsymbol{p} + \kappa\boldsymbol{w}) \\
&= -G_x(\alpha\boldsymbol{\mu} + \kappa\boldsymbol{w})
\end{aligned}
$$

which proves the equation (7).

C Higher Moments of Multivariate Gaussian

Lemma C.1. *For $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 I_d)$ and a matrix $A \in \mathbb{R}^{d\times d}$, we have the first four moments as follows:*

$$\begin{aligned}
\mathbb{E}[\boldsymbol{w}] &= \boldsymbol{\mu},\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top] &= \boldsymbol{\mu}\boldsymbol{\mu}^\top + \sigma^2 I_d =: M_\sigma,\\
\mathbb{E}[\|\boldsymbol{w}\|^2] &= \|\boldsymbol{\mu}\|^2 + d\sigma^2,\\
\mathbb{E}[\boldsymbol{w}^\top A\boldsymbol{w}] &= \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \sigma^2 \operatorname{Tr}(A),\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}] &= (2\sigma^2 + d\sigma^2 + \|\boldsymbol{\mu}\|^2)\boldsymbol{\mu},\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top A\boldsymbol{w}] &= \sigma^2(A + A^\top)\boldsymbol{\mu} + (\sigma^2 \operatorname{Tr}(A) + \boldsymbol{\mu}^\top A\boldsymbol{\mu})\boldsymbol{\mu},\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}\boldsymbol{w}^\top] &= 2M_\sigma^2 - 2m^2 + (d\sigma^2 + \|\boldsymbol{\mu}\|^2)M_\sigma,\\
\mathbb{E}[\boldsymbol{w}^\top\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}] &= 2\sigma^2(2\|\boldsymbol{\mu}\|^2 + d\sigma^2) + (d\sigma^2 + \|\boldsymbol{\mu}\|^2)^2,\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top A\boldsymbol{w}\boldsymbol{w}^\top] &= M_\sigma(A + A^\top)M_\sigma - m(A + A^\top)m + \operatorname{Tr}(M_\sigma A)M_\sigma,
\end{aligned}$$

where $m = \boldsymbol{\mu}\boldsymbol{\mu}^\top$.

If $\sigma^2 = 1$, then we have

$$\begin{aligned}
\mathbb{E}[\boldsymbol{w}] &= \boldsymbol{\mu},\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top] &= \boldsymbol{\mu}\boldsymbol{\mu}^\top + I_d =: M,\\
\mathbb{E}[\|\boldsymbol{w}\|^2] &= \|\boldsymbol{\mu}\|^2 + d,\\
\mathbb{E}[\boldsymbol{w}^\top A\boldsymbol{w}] &= \boldsymbol{\mu}^\top A\boldsymbol{\mu} + \operatorname{Tr}(A),\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}] &= (2 + d + \|\boldsymbol{\mu}\|^2)\boldsymbol{\mu},\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top A\boldsymbol{w}] &= (A + A^\top)\boldsymbol{\mu} + (\operatorname{Tr}(A) + \boldsymbol{\mu}^\top A\boldsymbol{\mu})\boldsymbol{\mu},\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}\boldsymbol{w}^\top] &= 2M^2 - 2m^2 + (d + \|\boldsymbol{\mu}\|^2)M,\\
\mathbb{E}[\boldsymbol{w}^\top\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}] &= 2(2\|\boldsymbol{\mu}\|^2 + d) + (d + \|\boldsymbol{\mu}\|^2)^2,\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top A\boldsymbol{w}\boldsymbol{w}^\top] &= M(A + A^\top)M - m(A + A^\top)m + \operatorname{Tr}(MA)M.
\end{aligned}$$

If we further assume $\boldsymbol{\mu} = \mathbf{0}_d$, then we have

$$\begin{aligned}
\mathbb{E}[\boldsymbol{w}] &= \mathbf{0}_d,\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top] &= I_d,\\
\mathbb{E}[\|\boldsymbol{w}\|^2] &= d,\\
\mathbb{E}[\boldsymbol{w}^\top A\boldsymbol{w}] &= \operatorname{Tr}(A),\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}] &= \mathbf{0}_d,\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top A\boldsymbol{w}] &= \mathbf{0}_d,\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}\boldsymbol{w}^\top] &= (2 + d)I_d,\\
\mathbb{E}[\boldsymbol{w}^\top\boldsymbol{w}\boldsymbol{w}^\top\boldsymbol{w}] &= 2d + d^2,\\
\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^\top A\boldsymbol{w}\boldsymbol{w}^\top] &= A + A^\top + \operatorname{Tr}(A)I_d.
\end{aligned}$$

D Proofs

Proof of Theorem 4.1. From (4), we have

$$\begin{aligned} L_{\text{train}}(P,Q) &= \mathbb{E}_{\boldsymbol{w},\bar{X}}[\|\boldsymbol{w}-\hat{\boldsymbol{w}}\|^2] \\ &= \mathbb{E}_{\boldsymbol{w},\bar{X}}[\|\boldsymbol{w}\|^2+\|\hat{\boldsymbol{w}}\|^2-2\boldsymbol{w}^\top\hat{\boldsymbol{w}}] \\ &= \underbrace{\mathbb{E}_{\boldsymbol{w},\bar{X}}[\|\hat{\boldsymbol{w}}\|^2]}_{\text{1st}}\underbrace{-2\mathbb{E}_{\boldsymbol{w},\bar{X}}[\boldsymbol{w}^\top\hat{\boldsymbol{w}}]}_{\text{2nd}}+\underbrace{\mathbb{E}_{\boldsymbol{w}}[\|\boldsymbol{w}\|^2]}_{\text{3rd}}. \end{aligned}$$

From (7), we can get the first term as follows:

$$\begin{aligned} \hat{\boldsymbol{w}} &= -G_x(\alpha\boldsymbol{\mu}+\kappa\boldsymbol{w}), \\ \|\hat{\boldsymbol{w}}\|^2 &= (\alpha\boldsymbol{\mu}+\kappa\boldsymbol{w})^\top G_x^2(\alpha\boldsymbol{\mu}+\kappa\boldsymbol{w}) \\ &= \kappa^2\boldsymbol{w}^\top G_x^2 w+2\kappa\alpha\boldsymbol{\mu}^\top G_x^2\boldsymbol{w}+\alpha^2\boldsymbol{\mu}^\top G_x^2\boldsymbol{\mu}, \\ \mathbb{E}_{\bar{X}}[\|\hat{\boldsymbol{w}}\|^2] &= \kappa^2\boldsymbol{w}^\top\mathbb{E}_{\bar{X}}[G_x^2]\boldsymbol{w}+2\kappa\alpha\boldsymbol{\mu}^\top\mathbb{E}_{\bar{X}}[G_x^2]\boldsymbol{w}+\alpha^2\boldsymbol{\mu}^\top\mathbb{E}_{\bar{X}}[G_x^2]\boldsymbol{\mu}, \\ \mathbb{E}_{\boldsymbol{w},\bar{X}}[\|\hat{\boldsymbol{w}}\|^2] &= \kappa^2(G_\mu+\operatorname{tr}(G)\sigma^2)+2\kappa\alpha G_\mu+\alpha^2 G_\mu \qquad \text{(Lemma C.1)} \\ &= (\kappa+\alpha)^2 G_\mu+\kappa^2\operatorname{tr}(G)\sigma^2 \\ &= \left((\kappa+\alpha)^2+\kappa^2\sigma^2 d\right)\frac{n+d+1}{n} \\ &= \frac{n+d+1}{n}\begin{bmatrix}\alpha & \kappa\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & 1+b\end{bmatrix}\begin{bmatrix}\alpha\\ \kappa\end{bmatrix} \\ &= \frac{1}{2}\boldsymbol{\theta}^\top C_2\boldsymbol{\theta}, \end{aligned}$$

where

$$\begin{aligned} G=\mathbb{E}_{\bar{X}}[G_x^2] &= \frac{1}{n}\mathbb{E}_{\bar{X}}[\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{x}\boldsymbol{x}^\top+(n-1)\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{x}'\boldsymbol{x}'^\top] \\ &= \frac{1}{n}[(d+2)I_d+(n-1)I_d] \qquad \text{(Lemma C.1)} \\ &= \frac{n+d+1}{n}I_d, \\ \operatorname{tr}(G) &= \frac{n+d+1}{n}d, \\ G_\mu\equiv\boldsymbol{\mu}^\top G\boldsymbol{\mu} &= \frac{n+d+1}{n}. \end{aligned}$$

The second term is

$$\begin{aligned} -\boldsymbol{w}^\top\hat{\boldsymbol{w}} &= \boldsymbol{w}^\top G_x(\boldsymbol{p}+\kappa\boldsymbol{w}) \\ &= \boldsymbol{w}^\top G_x\boldsymbol{p}+\kappa\boldsymbol{w}^\top G_x\boldsymbol{w} \\ &= \alpha\boldsymbol{w}^\top G_x\boldsymbol{\mu}+\kappa\boldsymbol{w}^\top G_x\boldsymbol{w}, \\ -\mathbb{E}_{\bar{X}}[\boldsymbol{w}^\top\hat{\boldsymbol{w}}] &= \alpha\boldsymbol{w}^\top\mathbb{E}[G_x]\boldsymbol{\mu}+\kappa\boldsymbol{w}^\top\mathbb{E}[G_x]\boldsymbol{w} \\ &= \alpha\boldsymbol{w}^\top\boldsymbol{\mu}+\kappa\boldsymbol{w}^\top\boldsymbol{w}, \\ -\mathbb{E}_{\boldsymbol{w},\bar{X}}[\boldsymbol{w}^\top\hat{\boldsymbol{w}}] &= \alpha+\kappa(1+b) \\ &= \begin{bmatrix}1 & 1+b\end{bmatrix}\begin{bmatrix}\alpha\\ \kappa\end{bmatrix} \\ &= \frac{1}{2}C_1^\top\boldsymbol{\theta}. \end{aligned}$$

The last term is $\mathbb{E}_{\boldsymbol{w}}[\|\boldsymbol{w}\|^2]=1+b=C_0$.

Next, we want to calculate the minimizer $\boldsymbol{\theta}^*$ and the corresponding minimum value. First, as $C_2 \succ 0$, the training loss has only one critical point which is the minimizer:

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} L_{\text{train}}(\boldsymbol{\theta}^*) &= C_2\boldsymbol{\theta}^* + C_1 = 0, \\
\boldsymbol{\theta}^* &= -C_2^{-1}C_1 \\
&= -\frac{n}{n+d+1}\begin{bmatrix}1 & 1\\ 1 & 1+b\end{bmatrix}^{-1}\begin{bmatrix}1\\ 1+b\end{bmatrix} \\
&= -\frac{n}{n+d+1}\frac{1}{b}\begin{bmatrix}1+b & -1\\ -1 & 1\end{bmatrix}\begin{bmatrix}1\\ 1+b\end{bmatrix} \\
&= -\frac{n}{n+d+1}\frac{1}{b}\begin{bmatrix}0\\ b\end{bmatrix} \\
&= -\frac{n}{n+d+1}\begin{bmatrix}0\\ 1\end{bmatrix}.
\end{aligned}$$

We can calculate the corresponding minimum value at the minimizer by plugging in $\boldsymbol{\theta}^* = -C^{-2}C_1$ and $\boldsymbol{\theta}^* = -[0, -n/(n+d+1)]^\top$ into $L_{\text{train}}(\boldsymbol{\theta}^*)$ as follows:

$$\begin{aligned}
L_{\text{train}}(\boldsymbol{\theta}^*) &= \frac{1}{2}\boldsymbol{\theta}^{*\top}C_2\boldsymbol{\theta}^* + C_1^\top\boldsymbol{\theta}^* + C_0 \\
&= -\frac{1}{2}C_1^\top C_2^{-1}C_1 + C_0 \\
&= \frac{1}{2}C_1^\top\boldsymbol{\theta}^* + 1 + b \\
&= -(1+b)\frac{n}{n+d+1} + 1 + b \\
&= \frac{(1+b)(d+1)}{n+d+1} \\
&= \frac{d+1}{n+d+1}L_0.
\end{aligned}$$

□

Proof of Theorem 4.2.

$$\begin{aligned}
\dot{\boldsymbol{\theta}} &= -\nabla_{\boldsymbol{\theta}} L_{\text{train}}(\boldsymbol{\theta}) = -C_2\boldsymbol{\theta} - C_1, \\
\boldsymbol{\theta}(t) &= c_1 e^{\lambda_1 t}\boldsymbol{v}_1 + c_2 e^{\lambda_2 t}\boldsymbol{v}_2 + \boldsymbol{\theta}^* \\
&= c_1 e^{\lambda_1 t}\begin{bmatrix}1\\ \frac{b+\sqrt{4+b^2}}{2}\end{bmatrix} + c_2 e^{\lambda_2 t}\begin{bmatrix}1\\ \frac{b-\sqrt{4+b^2}}{2}\end{bmatrix} + \begin{bmatrix}0\\ -\frac{n}{n+d+1}\end{bmatrix},
\end{aligned}$$

where λ_1, λ_2 are eigenvalues of $-C_2$, i.e., $\lambda_1 = -\lambda_+, \lambda_2 = -\lambda_-$, and c_1, c_2 are determined by the initial condition

$$\boldsymbol{\theta}(0) = c_1\begin{bmatrix}1\\ \frac{b+\sqrt{4+b^2}}{2}\end{bmatrix} + c_2\begin{bmatrix}1\\ \frac{b-\sqrt{4+b^2}}{2}\end{bmatrix} + \begin{bmatrix}0\\ -\frac{n}{n+d+1}\end{bmatrix} = \begin{bmatrix}0\\ 0\end{bmatrix}.$$

Therefore, solving the following system of linear equations

$$\begin{cases} c_1 + c_2 & = 0 \\ c_1\frac{b+\sqrt{4+b^2}}{2} + c_2\frac{b-\sqrt{4+b^2}}{2} - \frac{n}{n+d+1} & = 0 \end{cases},$$

we have

$$c_1 = -c_2 = \frac{n}{(n+d+1)\sqrt{4+b^2}}.$$

Therefore, we have

$$\begin{aligned}
\boldsymbol{\theta}(t) &= \frac{n}{n+d+1}\left(\frac{1}{\sqrt{4+b^2}}e^{\lambda_1 t}\begin{bmatrix}1\\ \frac{b+\sqrt{4+b^2}}{2}\end{bmatrix} - \frac{1}{\sqrt{4+b^2}}e^{\lambda_2 t}\begin{bmatrix}1\\ \frac{b-\sqrt{4+b^2}}{2}\end{bmatrix} + \begin{bmatrix}0\\ -1\end{bmatrix}\right),\\
\alpha(t) &= \frac{n}{n+d+1}\frac{1}{\sqrt{4+b^2}}e^{-\frac{2+b}{2}t}\left(e^{-\frac{\sqrt{4+b^2}}{2}t} - e^{\frac{\sqrt{4+b^2}}{2}t}\right)\\
&= -\frac{n}{n+d+1}\frac{2}{\sqrt{4+b^2}}e^{-\frac{2+b}{2}t}\sinh\left(\frac{\sqrt{4+b^2}}{2}t\right),\\
\kappa(t) &= \frac{n}{n+d+1}\frac{1}{\sqrt{4+b^2}}\left(\frac{b}{2}e^{-\frac{2+b}{2}t}\left(e^{-\frac{\sqrt{4+b^2}}{2}t} - e^{\frac{\sqrt{4+b^2}}{2}t}\right) + \frac{\sqrt{4+b^2}}{2}e^{-\frac{2+b}{2}t}\left(e^{-\frac{\sqrt{4+b^2}}{2}t} + e^{\frac{\sqrt{4+b^2}}{2}t}\right)\right) + \kappa^*\\
&= \frac{n}{n+d+1}\frac{2}{\sqrt{4+b^2}}e^{-\frac{2+b}{2}t}\left(-\frac{b}{2}\sinh\left(\frac{\sqrt{4+b^2}}{2}t\right) + \frac{\sqrt{4+b^2}}{2}\cosh\left(\frac{\sqrt{4+b^2}}{2}t\right)\right) + \kappa^*\\
&= \frac{n}{n+d+1}\frac{2}{\sqrt{4+b^2}}e^{-\frac{2+b}{2}t}\left(-\sinh(\tau_b)\sinh\left(\frac{\sqrt{4+b^2}}{2}t\right) + \cosh(\tau_b)\cosh\left(\frac{\sqrt{4+b^2}}{2}t\right)\right) + \kappa^*\\
&= \frac{n}{n+d+1}\frac{2}{\sqrt{4+b^2}}e^{-\frac{2+b}{2}t}\cosh\left(\frac{\sqrt{4+b^2}}{2}t - \tau_b\right) + \kappa^*.
\end{aligned}$$

□

Proof of Theorem 4.3. If b is small, then we can ignore higher order terms and obtain

$$(4+b^2)^{1/2} = 2(1+(b/2)^2)^{1/2} = 2+\Theta(b^2)$$

and $\frac{b\pm\sqrt{4+b^2}}{2} = \pm 1 + \Theta(b)$. For large b, we have

$$(4+b^2)^{1/2} = b(4b^{-2}+1)^{1/2} = b+\Theta(b^{-1})$$

and thus, $b+\sqrt{4+b^2} = \Theta(b)$ and $b-\sqrt{4+b^2} = \Theta(b^{-1})$. To summarize, we have the eigenvectors

$$\begin{aligned}
\boldsymbol{v}_+ &= \begin{bmatrix}1\\ \frac{b+\sqrt{4+b^2}}{2}\end{bmatrix} = \begin{cases}[1,\ 1+\Theta(b)]^\top & \text{if } b \ll 1\\ [1,\ \Theta(b)]^\top & \text{if } b \gg 1\end{cases},\\
\boldsymbol{v}_- &= \begin{bmatrix}1\\ \frac{b-\sqrt{4+b^2}}{2}\end{bmatrix} = \begin{cases}[1,\ -1+\Theta(b)]^\top & \text{if } b \ll 1\\ [1,\ \Theta(1/b)]^\top & \text{if } b \gg 1\end{cases}.
\end{aligned}$$

Similarly, for the eigenvalues, we have $\lambda_\pm = \frac{2+b\pm\sqrt{4+b^2}}{2} = 1+\frac{b\pm\sqrt{4+b^2}}{2}$, and thus

$$\begin{aligned}
\lambda_+ &= \begin{cases}2+\Theta(b) & \text{if } b \ll 1\\ \Theta(b) & \text{if } b \gg 1\end{cases},\\
\lambda_- &= \begin{cases}\Theta(b) & \text{if } b \ll 1\\ 1+\Theta(1/b) & \text{if } b \gg 1\end{cases}.
\end{aligned}$$

□

E Extra Figures

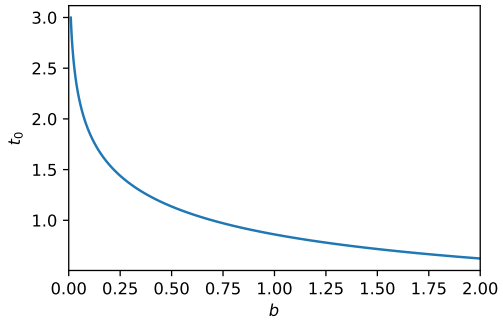

Figure 5: The critical point $t_0 = \frac{1}{\gamma_b}\mathrm{arctanh}(\gamma_b/\beta_b) = \frac{2}{\sqrt{4+b^2}}\mathrm{arctanh}\left(\frac{\sqrt{4+b^2}}{2+b}\right)$ that $\alpha'(t)$ changes its sign from negative to positive becomes earlier as we increase the value of b.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Our theoretical analysis is limited to the 2-parameter model, but empirical results demonstrate that the full-parameter model also exhibits similar behavior. See Fig 3.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Assumptions 3.1 and 3.2, and Appendix D.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See the captions of each Figure (e.g., Fig 3).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We modify the code from `https://github.com/chengxiang/LinearTransformer`. See the supplemental material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See the captions of each Figure (e.g., Fig 3).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: There is no experiment that requires such error bars.

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Our random linear regression experiments do not require a lot of resources. We used a single A40 GPU, but much smaller one would suffice.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This is a theoretical paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This is a theoretical paper.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: See the supplementary material for the license.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: See the supplementary material.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This is a theoretical paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This is a theoretical paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: No LLM is used.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.