On the Robustness of Transformers against Context Hijacking for Linear Classification

Tianle Li[1]*, **Chenyang Zhang**[2]*, **Xingwu Chen**[2], **Yuan Cao**[2], **Difan Zou**[1,2]†
[1]Institute of Data Science, The University of Hong Kong
[2]School of Computing & Data Science, The University of Hong Kong
{tianleli, chyzhang, xingwu}@connect.hku.hk
{yuancao, dzou}@hku.hk

Abstract

Transformer-based Large Language Models (LLMs) have demonstrated powerful in-context learning capabilities. However, their predictions can be disrupted by factually correct context, a phenomenon known as context hijacking, revealing a significant robustness issue. To understand this phenomenon theoretically, we explore an in-context linear classification problem based on recent advances in linear transformers. In our setup, context tokens are designed as factually correct query-answer pairs, where the queries are similar to the final query but have opposite labels. Then, we develop a general theoretical analysis on the robustness of the linear transformers, which is formulated as a function of the model depth, training context lengths, and number of hijacking context tokens. A key finding is that a well-trained deeper transformer can achieve higher robustness, which aligns with empirical observations. We show that this improvement arises because deeper layers enable more fine-grained optimization steps, effectively mitigating interference from context hijacking. This is also well supported by our numerical and real-world experiments. Our findings provide theoretical insights into the benefits of deeper architectures and contribute to enhancing the understanding of transformer architectures.

1 Introduction

Transformers [67] have demonstrated remarkable capabilities in various fields of deep learning, such as natural language processing [60, 1, 68, 66, 52, 26]. A common view of the superior performance of transformers lies in its remarkable in-context learning ability [14, 18, 44], that is, transformers can flexibly adjust predictions based on additional data given in context contained in the input sequence itself, without updating parameters. This impressive ability has triggered a series of theoretical studies attempting to understand the in-context learning mechanism of transformers [51, 30, 77, 31, 76]. These studies suggest that transformers can behave as meta learners [18], implementing certain meta algorithms (such as gradient descent [69, 2, 85]) based on context examples, and then applying these algorithms to the queried input.

Despite the benefits of in-context learning abilities in transformers, this feature can also lead to certain negative impacts. Specifically, while well-designed in-context prompts can help generate desired responses, they can also mislead the transformer into producing incorrect or even harmful outputs, raising significant concerns about the robustness of transformers [21, 45, 86]. For instance, a significant body of work focuses on jailbreaking attacks [15, 50, 62, 25, 81], which aim to design

*Equal contribution.
†Corresponding author.

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

Model	Context Input	Output
All models	Rafael Nadal's best sport is	tennis
GPT2-SMALL (12 Layers)	2×{Rafael Nadal is not good at playing basketball.} Rafael Nadal's best sport is	basketball
GPT2-MEDIUM (24 Layers)	8×{Rafael Nadal is not good at playing basketball.} Rafael Nadal's best sport is	basketball
GPT2-LARGE (36 Layers)	11×{Rafael Nadal is not good at playing basketball.} Rafael Nadal's best sport is	basketball

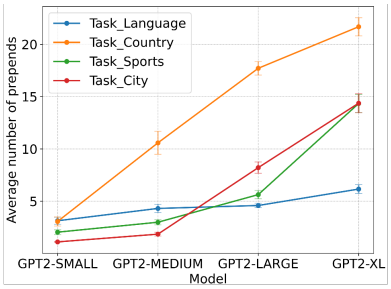

Figure 1: **Context hijacking phenomenon in LLMs of different depths.** *Left*: If there are no or only a few factually correct prepends, LLMs of different depths can correctly predict the next token. When the number of prepends increases, the outputs of models are disrupted. *Right*: Four different types of tasks are introduced, each with a fixed template, and tested on LLMs of different depths. The horizontal axis is the model with depth from small to large, and the vertical axis is the average number of prepends required to successfully interfere with the model output. Experiments show that deeper models perform more robustly. (Experimental setup is given in Appendix H.1)

specific context prompts that can bypass the defense mechanisms of large language models (LLMs) to produce answers to dangerous or harmful questions (e.g., "`how to build a bomb?`"). It has been demonstrated that, as long as the context prompt is sufficiently long and flexible to be adjusted, almost all LLMs can be successfully attacked [6]. These studies can be categorized under adversarial robustness, where an attacker is allowed to perturb the contextual inputs arbitrarily to induce the transformer model to generate targeted erroneous outputs [63, 53, 22, 80].

However, in addition to the adversarial attack that may use harmful or incorrect context examples, it has been shown that the predictions of LLMs can also be disrupted by harmless and factually correct context. Such a phenomenon is referred to as *context hijacking* [37, 36], which is primarily discovered on fact retrieval tasks, i.e. the output of the LLMs can be simply manipulated by modifying the context with additional factual information. For example, as shown in Figure 1, the GPT2 model can correctly answer the question "`Rafael Nadal's best sport is`" with "`tennis`" when giving context examples. However, if factually correct context examples such as "`Rafael Nadal is not good at playing basketball`" are provided before the question, the GPT-2 model may incorrectly respond with "`basketball`". Then, it is interesting to investigate whether such a phenomenon depends on different tasks and transformer architectures. To this end, we developed a class of context hijacking tasks and counted the number of context examples that led to incorrect outputs (see Figure 1). Our findings indicate that increasing the number of prepended context examples amplifies the effect on the transformer's prediction, making it more likely to generate incorrect outputs. Additionally, we observed that deeper transformer models exhibit higher robustness to context hijacking, requiring more prepended context examples to alter the model's output. Therefore, conducting a precise robustness analysis regarding context hijacking could provide valuable insights in understanding the architecture of the transformer model.

In this paper, we aim to develop a comprehensive theoretical analysis on the robustness of transformer against context hijacking. In particular, we follow the general design of previous theoretical works [51, 2, 27] on the in-context learning of transformers, by considering the multi-layer linear transformer models for linear classification tasks, where the hijacking examples are designed as the data on the boundary but with an opposite label to the queried input. Starting from the view that the L-layer transformer models can implement L-step gradient descent on the context examples, with an arbitrary initialization, we formulate the transformer training as finding the optimal multi-step gradient descent methods with respect to the learning rates and initialization. Then, we prove the optimal multi-step gradient strategy, and formulate the optimal learning rate and initialization as the function of the iteration number (i.e., model depth) and the context length. Furthermore, we deliver the theoretical analysis on the robustness based on the proved optimal gradient descent strategy, which shows that as the transformer become deeper, the corresponding more fine-grained optimization steps can be less affected by the hijacking examples, thus leading to higher robustness. This is well aligned with the empirical findings and validated by our numerical and real-world experiments. We summarize the main contributions of this paper as follows:

- We develop the first theoretical framework to study the robustness of multi-layer transformer model against context hijacking, where the hijacked context example is designed as the data with the factually correct label but close to the prediction boundary. This is different from a very recent related work on the robustness of transformer [7] that allows the context data to be arbitrarily perturbed, which could be factually incorrect.
- Based on the developed theoretical framework, we formulate the test robust accuracy of the transformer as a function with respect to the training context length, number of hijacked context examples, and the depth of the transformer model. The key of our analysis is that we model the in-context learning mechanism of a well-trained multi-layer transformer as an optimized multi-step gradient descent, where the corresponding optimal initialization and learning rates can be precisely characterized. This could of independent interest to other problems that involve the gradient descent methods on linear problems.
- Based on the developed theoretical results, we demonstrate that deeper transformers are more robust because they are able to perform more fine-grained optimization steps on the context samples, which can potentially explain the practical observations of LLMs in the real world (see Figure 1). The theoretical results are well supported by synthetic numerical experiments and real-world LLMs experiments in various settings.

Related Works. Our work is related to recent works on *in-context learning via transformers*, *mechanism interpretability of transformers*, and *robustness of transformers*. Due to space limit, we defer them to Appendix A.

2 Preliminaries

2.1 Data model

To understand the mechanism of context hijacking phenomenon, we model it as a binary classification task, where the query-answer pair is modeled as the input-response pair ($(\mathbf{x}, y) \in \mathbb{R}^d \times \{\pm 1\}$). In particular, we present the definition of the data model as follows:

Definition 2.1 (Data distribution)**.** Let $\mathbf{w}^* \in \mathbb{R}^d$ be a vector drawn from a prior distribution on the d dimensional unit sphere $\mathbb{S}^{d-1}$, denoted by $p_{\boldsymbol{\beta}^*}(\cdot)$, where $\boldsymbol{\beta}^* \in \mathbb{S}^{d-1}$ denotes the expected direction of $\mathbf{w}^*$. Then given the generated $\mathbf{w}^*$, the data pair $(\mathbf{x}, y)$ is generated as follows: the feature vector is $\mathbf{x} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$ and the corresponding label is $y = \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x} \rangle)$.

Modeling natural language problems as linear problems is a common setting in theoretical research, because linear problems have sufficient representation power, supported by many previous works [69, 2, 85, 84, 33, 47]. The innovation of our construction is that we precisely exploit the characteristics of the context hijacking phenomenon and model it as a semantic binary classification problem. Compared with previous works that directly study linear problems, such as linear regression, our modeling is more practical. Moreover, to extend our theoretical results to more complex situations, we also consider nonlinear models and nonlinear tasks, which is given in Appendix I.3.

Based on the data distribution of each instance, we then introduce the detailed setup of the in-context learning task in our work. In particular, we consider the setting that the transformer is trained on the data with clean context examples and evaluated on the data with hijacked context.

Training phase. During the training phase, we are given n clean context examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ and a query $\mathbf{x}_{\mathrm{query}}$ with its label y_{query}. In particular, here we mean the clean examples as the $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ are drawn from the same data distribution $\mathcal{D}_{\mathbf{w}^*}$ as $(\mathbf{x}_{\mathrm{query}}, y_{\mathrm{query}})$. Then, the input data matrix for in-context learning is designed as follows:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{x}_1 & \ldots & \mathbf{x}_n & \mathbf{x}_{\mathrm{query}} \\ y_1 & \ldots & y_n & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(n+1)}. \tag{2.1}$$

Here, to ensure that the dimension of $\mathbf{x}_{\mathrm{query}}$ aligns with those of other input pair $(\mathbf{x}_i, y_i)$, we concatenate it with 0 as a placeholder for the unknown label y_{query}. Ideally, we anticipate that given the input $\mathbf{Z}$, the output of the transformer model, denoted by $\widehat{y}_{\mathrm{query}}$ can match the ground truth one. Moreover, we also emphasize that within each data matrix $\mathbf{Z}$, the context examples and the queried data should be generated based on the same ground truth vector $\mathbf{w}^*$, while for different input matrices, e.g., $\mathbf{Z}$ and $\mathbf{Z}'$, we allow their corresponding ground truth vectors could be different, which are i.i.d. drawn from the prior $p_{\boldsymbol{\beta}^*}(\cdot)$.

The training data distribution simulates the pre-training data of the large language model. Unlike existing works [2, 51] where the prior of $\mathbf{w}^*$ is assumed to have a zero mean, we consider a setting where $\mathbf{w}^*$ has a non-zero mean (i.e., $\boldsymbol{\beta}^*$). This approach is inspired by empirical observations (see Figure 1) that transformer models can perform accurate zero-shot predictions. Consequently, our model can encapsulate both memorization and in-context learning, where the former corresponds to recovering the mean of the prior distribution, i.e., $\boldsymbol{\beta}^*$, and the latter aims to manipulate the $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ effectively. In contrast, existing works primarily focus on the latter, thereby failing to fully explain the interplay between memorization and in-context learning.

Test phase. During the test phase, context examples are designed based on the query input $\mathbf{x}_{\text{query}}$ to effectively execute the attack. Inspired by empirical observations (Figure 1) and prior experience with jailbreaking attacks [6], we choose to use repeated hijacking context examples during the test phase. Specifically, since the hijacked context should be factually correct, we consider data similar to the queried input but with a correct and opposite label of low confidence. Mathematically, this involves projecting $\mathbf{x}_{\text{query}}$ onto the classification boundary. To this end, given the target query data $(\mathbf{x}_{\text{query}}, y_{\text{query}})$, we formalize the design of the hijacked context example as follows.

Definition 2.2 (Hijacked context data)**.** Let $(\mathbf{x}, y)$ be a input pair and $\mathbf{w}^*$ be the corresponding ground truth vector. Additionally, denote $\mathbf{x}_\perp$ as the projection of $\mathbf{x}$ on the boundary of classifier, i.e. $\mathbf{x}_\perp = (\mathbf{I}_d - \mathbf{w}^*(\mathbf{w}^*)^\top) \cdot \mathbf{x}$. Then, the query pair $(\mathbf{x}_{\text{query}}, y_{\text{query}})$ is generated as $\mathbf{x}_{\text{query}} = \mathbf{x}_\perp + \sigma \mathbf{w}^*$ and $y_{\text{query}} = \text{sign}\left(\langle \mathbf{w}^*, \mathbf{x}_{\text{query}} \rangle\right) = \text{sign}(\sigma)$ with σ being a random variable, and the hijacked context example is designed as $\mathbf{x}_{\text{hc}} = \mathbf{x}_\perp$ and $y_{\text{hc}} = -y_{\text{query}}$.

Note that we pick $\langle \mathbf{x}_{\text{hc}}, \mathbf{w}^* \rangle = 0$ to enforce hijacked context lies on the boundary of the classifier. A more rigorous design is to set $\mathbf{x}_{\text{hc}} = \mathbf{x}_\perp - \eta \cdot y_{\text{query}} \cdot \mathbf{w}^*$ for some positive quantity η, where it can be clearly shown that $y_{\text{hc}} = \text{sign}(\langle \mathbf{x}_{\text{hc}}, \mathbf{w}^* \rangle) = -y_{\text{query}}$. Definition 2.2 concerns the limiting regime by enforcing $\eta \to 0^+$.

Then, based on the above design, the input data matrix in the test phase is constructed as follows:

$$\mathbf{Z}^{\text{hc}} = \begin{bmatrix} \mathbf{x}_{\text{hc}} & \cdots & \mathbf{x}_{\text{hc}} & \mathbf{x}_{\text{query}} \\ y_{\text{hc}} & \cdots & y_{\text{hc}} & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(N+1)}. \tag{2.2}$$

Here we use N to denote the number of hijacked context examples. The example $(\mathbf{x}_{\text{hc}}, y_{\text{hc}})$ can also be interpreted to the closest data to $\mathbf{x}_{\text{query}}$ but with a different label $-y_{\text{query}}$, which principally has the ability to perturb the prediction of $\mathbf{x}_{\text{query}}$. An innovation compared with previous work is that our data model design is semantic. For example, in Figure 1, we can assume that $\mathbf{x}_{\text{hc}}$ = "Rafael Nadal is not good at playing", y_{hc} = "basketball", $\mathbf{x}_{\text{query}}$ = "Rafael Nadal's best sport is", and y_{query} = "tennis". Additionally, because the prediction is highly likely to be correct in the zero-shot regime (i.e., $N = 0$), the prediction in the test phase can be viewed as a competition between model memorization and adversarial in-context learning. This dynamic is primarily influenced by the number of hijacked context examples.

2.2 Transformer model

Following the extensive prior theoretical works for transformer [84, 85, 16, 27, 2], we consider linear attention-only transformers, a prevalent simplified structure to investigate the behavior of transformer models. In particular, We define an L-layer linear transformer $\texttt{TF}$ as a stack of L single-head linear attention-only layers. For the input matrix $\mathbf{Z}_{i-1} \in \mathbb{R}^{(d+1)\times(n+1)}$, the i-th single-head linear attention-only layer $\texttt{TF}_i$ updates the input as follows:

$$\mathbf{Z}_i = \texttt{TF}_i(\mathbf{Z}_{i-1}) = \mathbf{Z}_{i-1} + \mathbf{P}_i \mathbf{Z}_i \mathbf{M} (\mathbf{Z}_{i-1}^\top \mathbf{Q}_i \mathbf{Z}_{i-1}), \tag{2.3}$$

where $\mathbf{M} := \begin{pmatrix} \mathbf{I}_n & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$ is the mask matrix. We design this architecture to constrain the model's focus to the first n in-context examples. Moreover, the matrix $\mathbf{P} := \mathbf{W}_v \in \mathbb{R}^{(d+1)\times(d+1)}$ serves as the value matrix in the standard self-attention layer, while the matrix $\mathbf{Q} := \mathbf{W}_k^\top \mathbf{W}_q \in \mathbb{R}^{(d+1)\times(d+1)}$ consolidates the key matrix and query matrix. This mild re-parameterization has been widely considered in numerous recent theoretical works [34, 72, 65, 35]. To adapt the transformer for solving the linear classification problem, we introduce an additional linear embedding layer $\mathbf{W}_E \in \mathbb{R}^{(d+1)\times(d+1)}$. Then the output of the transformer $\texttt{TF}$ is defined as

$$\widehat{y}_{\text{query}} = \texttt{TF}(\mathbf{Z}_0; \mathbf{W}_E, \{\mathbf{P}_\ell, \mathbf{Q}_\ell\}_{\ell=1}^L)$$

$$= -[\mathtt{TF}_L \circ \cdots \circ \mathtt{TF}_1 \circ \mathbf{W}_E(\mathbf{Z}_0)]_{(d+1),(n+1)}$$
$$= -[\mathbf{Z}_L]_{(d+1),(n+1)}, \tag{2.4}$$

i.e. the negative of the $(d+1, n+1)$-th entry of $\mathbf{Z}_L$, and this position is replaced by 0 in the input $\mathbf{Z}_0$. The reason for taking the minus sign here is to align with previous work [69, 62], which will be explained in Proposition 3.1.

2.3 Evaluation metrics

Based on the illustration regarding the transformer architecture, we first define the in-context learning risk of a L-layer model TF in the training phase. In particular, let $\mathcal{D}_{\mathrm{tr}}$ be the distribution of the input data matrix $\mathbf{Z}$ in (2.1) and the target y_{query}, which covers the randomness of both $(\mathbf{x}, y)$ and $\mathbf{w}^*$, then the risk function in the training phase is defined as:

$$\mathcal{R}\left(\mathtt{TF}\right) := \mathbb{E}_{\mathbf{Z}, y_{\mathrm{query}} \sim \mathcal{D}_{\mathrm{tr}}} \left[\left(\mathtt{TF}(\mathbf{Z}; \boldsymbol{\theta}) - y_{\mathrm{query}}\right)^2 \right]. \tag{2.5}$$

where $\boldsymbol{\theta} = \{\mathbf{W}_E, \{\mathbf{P}_\ell, \mathbf{Q}_\ell\}_{\ell=1}^L\}$ denotes the collection of all trainable parameters of TF. This risk function will be leveraged for training the transformer models (where we use the stochastic gradient in the experiments).

Additionally, in the test phase, let $\mathcal{D}_{\mathrm{te}}$ be the distribution of the input data matrix $\mathbf{Z}^{\mathrm{hc}}$ in (2.2) and the target y_{query}, we consider the following population prediction error:

$$\mathcal{E}\left(\mathtt{TF}\right) := \mathbb{P}_{\mathbf{Z}^{\mathrm{hc}}, y_{\mathrm{query}} \sim \mathcal{D}_{\mathrm{te}}} \left[\mathtt{TF}(\mathbf{Z}^{\mathrm{hc}}; \boldsymbol{\theta}) \cdot y_{\mathrm{query}} < 0 \right]. \tag{2.6}$$

3 Main theory

In this section, we present how we establish our theoretical analysis framework regarding the robustness of transformers against context hijacking. In summary, we can briefly sketch our framework into the following several steps:

- Step 1. We establish the equivalence between the L-layer transformers and L steps gradient descent, converting the original problem of identifying well-trained transformers to the problem of finding the optimal parameters of gradient descent (i.e., initialization and learning rates).
- Step 2. We derive the optimal learning rates and initialization of gradient descent, revealing its relationship with the number of layers L and training context length n.
- Step 3. By formulating the classification error of a linear model obtained by L steps gradient descent with optimal parameters on hijacking distribution $\mathcal{D}_{\mathrm{te}}$, we characterize how the number of layers L, the training context length n and test context length N affect the robustness.

3.1 Optimizing over in-context examples

Inspired by a line of recent works [85, 9, 16, 2, 51] which connects the in-context learning of transformer with the gradient descent algorithm, we follow a similar approach by showing that, in the following proposition, multi-layer transformer can implement multi-step gradient descent, starting from any initialization, on the context examples.

Proposition 3.1. For any L-layer single-head linear transformer, let $\widehat{y}_{\mathrm{query}}^{(l)}$ be the output of the l-th layer of the transformer, i.e. the $(d+1, n+1)$-th entry of $\mathbf{Z}_l$. Then, there exists a single-head linear transformer with L layers such that $\widehat{y}_{\mathrm{query}}^{(l)} = -\langle \mathbf{w}_{\mathrm{gd}}^{(l)}, \mathbf{x}_{\mathrm{query}} \rangle$. Here, $\mathbf{w}_{\mathrm{gd}}^{(l)}$'s are the parameter vectors obtained by the following gradient descent iterative rule and the initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$ can be arbitrary:

$$\mathbf{w}_{\mathrm{gd}}^{(l+1)} = \mathbf{w}_{\mathrm{gd}}^{(l)} - \boldsymbol{\Gamma}_l \nabla \widetilde{L}(\mathbf{w}_{\mathrm{gd}}^{(l)}),$$
$$\text{where} \quad \widetilde{L}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{n} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2. \tag{3.1}$$

Here $\boldsymbol{\Gamma}_l$ can be any $d \times d$ matrix.

As $\mathbf{\Gamma}_l$ could be any $d \times d$ matrix, Proposition 3.1 demonstrates that the output of the L-layer transformers is equivalent to that of a linear model trained via L-steps of full-batch preconditioned gradient descent on the context examples, with $\{\mathbf{\Gamma}_l\}_{l=0}^{L-1}$ being the learning rates. This suggests that each L-layer transformer defined in (2.4), with different parameters, can be viewed as an optimization process of a linear model characterized by a distinct set of initialization and learning rates $\{\mathbf{w}_{\mathrm{gd}}^{(0)}, \mathbf{\Gamma}_0, \ldots, \mathbf{\Gamma}_{L-1}\}$. Therefore, it suffice to directly find the optimal parameters of the gradient descent process, without needing to infer the specific parameters of the well-trained transformers.

Among all related works presenting similar conclusions that transformers can implement gradient descent, our result is general as we prove that transformers can implement multi-step gradient descent from any initialization. In comparison, for example, [85] shows that a single-layer transformer with MLP can implement one-step gradient descent from non-zero initialization. [2] demonstrate that linear transformers can implement gradient descent, but only from $\mathbf{0}$ initialization.

3.2 Optimal multi-step gradient descent

Based on the discussion in the previous section, Proposition 3.1 successfully transforms the original problem of identifying the parameters of well-trained transformers into the task of finding the optimal learning rates and initialization for the gradient descent process (3.1). In this section, we present our conclusions regarding these optimal parameters. As we consider optimizing over the general training distribution $\mathcal{D}_{\mathrm{tr}}$, where the tokens $\mathbf{x}_i$'s follow the isotropic distribution, it follows that the updating step size should be equal in each direction from the perspective of expectation. Therefore we consider the case $\mathbf{\Gamma}_l = \alpha_l \mathbf{I}_d$ to simplify the problem, with α_l being a scalar for all $l \in \{0, \ldots, L-1\}$. In the following, we focus on the optimal set of parameters $\{\mathbf{w}_{\mathrm{gd}}^{(0)}, \alpha_0, \ldots, \alpha_{L-1}\}$. Specifically, we consider the population loss for $\mathbf{w}_{\mathrm{gd}}^{(L)}$ as

$$\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) := \mathbb{E}_{T, \mathbf{w}^* \sim \mathcal{D}_{\mathrm{tr}}}\big[\big(\langle \mathbf{w}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}} \rangle - y_{\mathrm{query}}\big)^2\big],$$

where $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n), (\mathbf{x}_{\mathrm{query}}, y_{\mathrm{query}})\}$ is the set of all classification pairs [3]. This definition resembles $\mathcal{R}(\mathtt{TF})$ defined in (2.5). We attempt to find the $\mathbf{w}_{\mathrm{gd}}^{(L)}$ that minimizes this population loss, along with the corresponding learning rates $\{\alpha_l\}_{l=0}^{L}$ and initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$, which can generate this $\mathbf{w}$ via gradient descent. We first present the following proposition demonstrating that there exists commutative invariance among the learning rates $\{\alpha_l\}_{l=0}^{L}$ for producing $\mathbf{w}_{\mathrm{gd}}^{(L)}$.

Proposition 3.2. Let $\{\alpha_0, \alpha_1, \ldots, \alpha_{L-1}\}$ be a set of learning rates, and $\{\alpha_0', \alpha_1', \ldots, \alpha_{L-1}'\}$ be another set of learning rates that is a permutation of $\{\alpha_0, \alpha_1, \ldots, \alpha_{L-1}\}$, meaning both sets contain the same elements, with the only difference being the order of these elements. With $\mathbf{w}_{\mathrm{gd}}^{(L)} \in \mathbb{R}^d$ denoting as the parameters achieved by learning rates $\{\alpha_0, \alpha_1, \ldots, \alpha_{L-1}\}$ and $\mathbf{w}_{\mathrm{gd}'}^{(L)} \in \mathbb{R}^d$ as the parameters achieved by learning rates $\{\alpha_0', \alpha_1', \ldots, \alpha_{L-1}'\}$ from the same initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$, it holds that $\mathbf{w}_{\mathrm{gd}}^{(L)} = \mathbf{w}_{\mathrm{gd}'}^{(L)}$.

Proposition 3.2 implies that the learning rates at different steps contribute equally to the overall optimization process. Consequently, we will consider a consistent learning rate α through the entire gradient descent procedure, which significantly reduces the difficulty of analysis and does not incur any loss of generality. Additionally, given the correspondence between each step of gradient descent and each layer of transformers, Proposition 3.2 implies the possibility of analogous behavior among layers of deep transformers during training. Such a result provides a promising approach to the optimization of deep transformers. Now we are ready to present our main results regarding the derivation of the optimal parameters α and $\mathbf{w}_{\mathrm{gd}}^{(0)}$.

Theorem 3.3. For training distribution $\mathcal{D}_{\mathrm{tr}}$ in Definition 2.1, suppose that the training context length n is sufficiently large such that $n \geq \widetilde{\Omega}(\max\{d^2, dL\})$. Additionally, suppose that the perturbation of $\mathbf{w}^*$ around its expectation $\boldsymbol{\beta}^*$ is smaller than $\frac{\pi}{2}$, i.e. $\langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle > 0$. Based on these assumptions,

[3] Here we slightly abuse the notation of $\mathcal{D}_{\mathrm{tr}}$ to denote both the distribution of $\mathbf{Z}, \mathbf{w}_{\mathrm{query}}$ and $T, \mathbf{w}^*$.

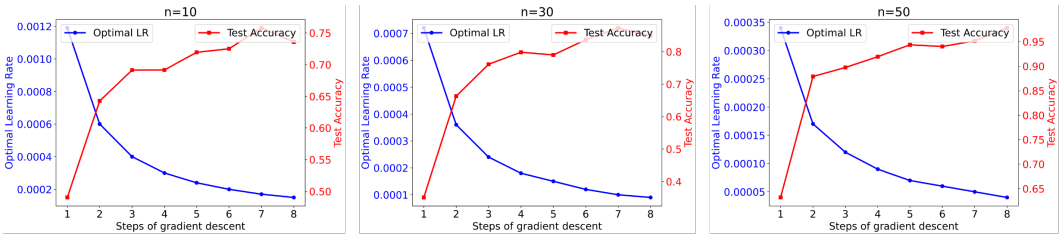

Figure 2: Gradient descent experiments using a single-layer neural network. We use grid search to obtain the optimal learning rate for different training context lengths n and different steps of gradient descent L. Then we use the corresponding optimal learning rate to perform multi-step gradient descent optimization on the test dataset. The results show that longer training context lengths and more gradient descent steps lead to smaller optimal learning rate and better optimization.

the optimal learning rate α and initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$, i.e. $\alpha, \mathbf{w}_{\mathrm{gd}}^{(0)} = \arg\min_{\alpha, \mathbf{w}_{\mathrm{gd}}^{(0)}} \mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)})$, take the value as follows:

$$\alpha = \widetilde{\Theta}\bigg(\frac{1}{nL}\bigg); \qquad \mathbf{w}_{\mathrm{gd}}^{(0)} = c\boldsymbol{\beta}^*,$$

where c is an absolute constant.

Theorem 3.3 clearly identifies the optimal learning rate α and initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$. Specifically, it shows that the optimal initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$ aligns the direction of the expectation $\boldsymbol{\beta}^*$, with its length independent of the number of steps L, and the context length n. Such a conclusion complies with our intuitions as the initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$ represents the memory of large language models, which is not dependent on the task-specific context examples. In contrast, the optimal learning rate α is inversely related to both n and L. This suggests that in both cases: (i) with more in-context examples; and (ii) with more layers, the output of pre-trained transformers will equal to that of a more fine-grained gradient descent process using a smaller learning rate. Generally, a small-step strategy ensures the convergence of the objective, highlighting the potential benefits of deeper architectures and training inputs with longer context.

3.3 Robustness against context hijacking

The previous two subsections illustrate that for any input with context examples, we can obtain the corresponding prediction for that input from the well-trained transformers by applying gradient descent with the optimal parameters we derived in Theorem 3.3. As we model $\mathcal{D}_{\mathrm{te}}$ the distribution of hijacking examples, to examine the robustness of L-layer transformers against hijacking, we only need to check whether the linear model achieved by L-step gradient on $(\mathbf{x}_{\mathrm{hc}}, y_{\mathrm{hc}})$ can still conduct successful classification on $\mathbf{x}_{\mathrm{query}}$. Specifically, we consider the classification error of the parameter vector $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}$ as,

$$\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) := \mathbb{P}_{T, \mathbf{w}^* \sim \mathcal{D}_{\mathrm{te}}}\big(y_{\mathrm{query}} \cdot \langle \widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}} \rangle < 0\big),$$

where $T = \{(\mathbf{x}_{\mathrm{hc}}, y_{\mathrm{hc}}), (\mathbf{x}_{\mathrm{query}}, y_{\mathrm{query}})\}$, and $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}$ is obtained by implementing gradient descent on $(\mathbf{x}_{\mathrm{hc}}, y_{\mathrm{hc}})$ with L steps and the optimal α and $\mathbf{w}_{\mathrm{gd}}^{(0)}$. Similar to the previous result, $\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)})$ is identical to $\mathcal{E}(\mathtt{TF})$ defined in (2.6). Based on these preliminaries, we are ready to present our results regarding the robustness against context hijacking. We first introduce the following lemma illustrating that when the context length of hijacking examples is small, we hardly observe the label flipping phenomenons of the prediction from well-trained transformers.

Lemma 3.4. Assume that all assumptions in Theorem 3.3 still hold. Additionally, assume that the length of hijacking examples N is small such that $N \leq \widetilde{O}\big(\frac{n}{d^{3/2}}\big)$ and σ follows any continuous distribution. Based on these assumptions, it holds that

$$\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) \leq \mathcal{E}(\mathbf{w}_{\mathrm{gd}}^{(0)}) + o(1).$$

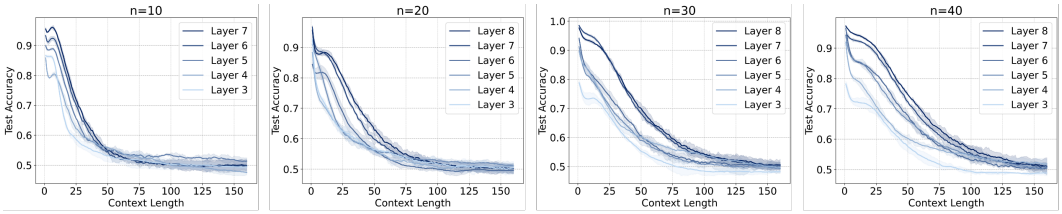

Figure 3: Linear transformers experiments with different depths and different training context lengths. By testing the trained linear transformers on the test set, we can find that as the number of interference samples increases, the model prediction accuracy becomes worse. However, deeper models have higher accuracy, indicating stronger robustness. As the training context length increases, the model robustness will also increase because the accuracy converges significantly more slowly.

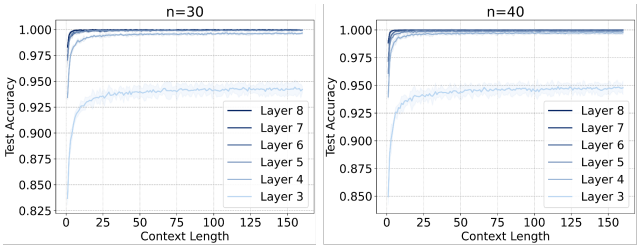

Figure 4: Linear transformers experiments on training dataset. By testing trained linear transformers on the training set, the initial accuracy of the model is high and can be improved with the increase of context length, indicating that the model can use in-context learning to fine-tune $\boldsymbol{\beta}^\star$ to $\mathbf{w}^\star$. And deeper models have stronger optimization capabilities.

Lemma 3.4 demonstrates that when the context length of hijacking examples is small, the classification error of the linear model obtained through gradient descent on these hijacking examples is very close to that of the optimal initialization. The reasoning is straightforward: When N is relatively smaller compared to training context length n, and since the optimal learning rate α is on the order of the reciprocal of n, the contributions from the hijacking examples become almost negligible in gradient descent iterations, allowing the model to remain close to its initialization. Consequently, we consider the case that N is comparable with n in the following theorem.

Theorem 3.5. Assume that all assumptions in Theorem 3.3 still hold. Additionally, assume that $N \geq \widetilde{\Omega}\big(\frac{n}{d^{3/2}}\big)$, $n \geq \widetilde{\Omega}\big(Nd\big)$, and σ follows some uniform distribution. Based on these assumptions, it holds that

$$\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) \leq c_1 - c_2\left(1 - \widetilde{\Theta}\left(\frac{Nd}{nL}\right)\right)^L,$$

where c_1, c_2 are two positive scalar solely depending on the distribution of σ and $\mathbf{w}^*$.

Based on a general assumption that σ follows the uniform distribution, Theorem 3.5 formulates the upper bound of the classification error as a function of the training context length n, the number of hijacking examples N, and the number of layers L. Specifically, this upper bound contains a term proportional to $-\big(1 - \widetilde{\Theta}\left(\frac{Nd}{nL}\right)\big)^L$. As $\big(1 - \widetilde{\Theta}\left(\frac{Nd}{nL}\right)\big)^L$ is a monotonically increasing function for N and a monotonically decreasing function for n and L, Theorem 3.5 successfully demonstrates two facts: (i) well-trained transformers with deeper architectures, or those pre-trained on longer context examples, will exhibit more robustness against context hijacking; and (ii) for a given well-trained transformer, the context hijacking phenomenon is easier to observe when provided with more hijacking examples. These conclusions align well with our experimental observations (Figure 3).

4 Experiments

4.1 Optimal gradient descent with different steps

In our theoretical framework, the optimal gradient descent with more steps (L) or longer training context length (n) will have a smaller learning rate per step (Theorem 3.3), and this combination

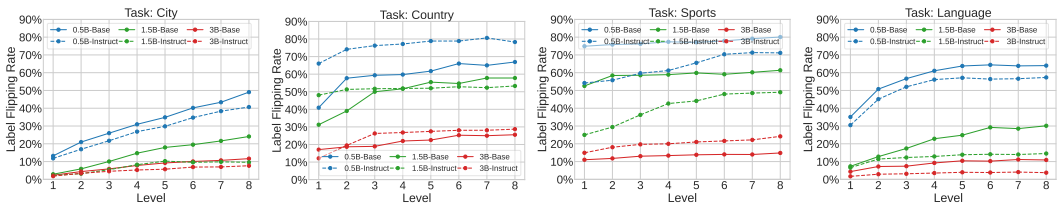

Figure 5: Context hijacking in real-world LLMs. We investigate the label flipping rates of Qwen2.5-Base and Qwen2.5-Instruct models of varying sizes, facing varying levels of context hijacking. Hijacking levels ranging from 1 to 8 represent context hijacking prefixes ranging in length from 10 to 80 sentences. Label flipping occurs when the model predicts the next token to be the topic described by the context prefix, which differs from the correct answer to the question. Our experimental results on real-world LLMs are still completely consistent with our theoretical conclusions: deeper models are more robust to context hijacking.

of more steps with small learning rates will perform better on the optimization process over context samples (Theorem 3.5). Our theory shows that a trained transformer will learn the optimal multi-step gradient descent, which will make it more robust during testing. Therefore, we directly verified the consistency between practice and theory in the multi-step gradient descent experiment.

We construct a single-layer neural network to conduct optimal multi-step gradient descent experiments. Each training sample $(\mathbf{x}_i, y_i)$ is drawn i.i.d. from the distribution $\mathcal{D}_{\mathrm{tr}}$ defined in Section 2.1. We consider the learning rate that minimizes the loss of the test sample when the single-layer neural network is trained using 1 to 8 steps of gradient descent, that is, the optimal learning rate α_L corresponding to L-step gradient descent, which can be obtained by grid search. Figure 2 shows that α_L decreases as L and n increases, which is aligned with our theoretical results (Theorem 3.3).

Next, we discuss the second part of the theoretical framework, i.e., gradient descent with more steps and small step size performs a more fine-grained optimization (Theorem 3.5). We apply the optimal learning rate searched in the training phase to the test phase, and perform gradient descent optimization on the test samples drawn from $\mathcal{D}_{\mathrm{te}}$ with the optimal learning rate and its corresponding number of steps. We can find that with the increase in the number of gradient descent steps and the decrease in the learning rate, the performance of the model will be significantly improved.

4.2 Robustness of linear transformers with different number of layers

Applying our theoretical framework to the context hijacking task on transformers can explain it well, indicating that our theory has practical significance. We train linear transformers with different depths and context lengths on the training dataset ($\mathcal{D}_{\mathrm{tr}}$). We mainly investigate the impact of training context length n, and model depth L and the testing context length N on model classification accuracy.

We first test the trained transformers on the training dataset to verify that the model can fine-tune the memorized $\boldsymbol{\beta}^*$ to $\mathbf{w}^\star$. According to the Figure 4, we can find that the model has a high classification accuracy when there are very few samples at the beginning. This means that the model successfully memorizes the shared signal $\boldsymbol{\beta}^*$. As the context length increases, the accuracy of the model gradually increases and converges, meaning that the model can fine-tune the pre-trained $\boldsymbol{\beta}^*$ by using the context samples. In addition, deeper models can converge to larger values faster, corresponding to the theoretical view that deeper models can perform more sophisticated optimization.

Then we conduct experiments on the test set. Observing the experiment results (Figure 3), we can see that as the context length increases, the accuracy of the model decreases significantly and converges to 50%, showing that the model is randomly classifying the final query $\mathbf{x}_{\mathrm{query}}$. This is consistent with the context hijacking phenomenon that the model's robustness will deteriorate as the number of interference prompt words increases. When the number of layers increases, the models with different depths show the same trend as the context length increases, but the accuracy of the model will increase significantly, which is consistent with the phenomenon that deeper models show stronger robustness in practical applications. In addition, the model becomes significantly more robust as the training context length increases, because the accuracy converges more slowly as the length increases.

4.3 Context hijacking in real-world LLMs

To further validate our theoretical results in more realistic scenarios, we investigate the label flipping rates of Qwen2.5-Base and Qwen2.5-Instruct models of varying sizes, facing different levels of hijacking. The 0.5B, 1.5B, and 3B models have 24, 28, and 36 layers, respectively. We first construct four different datasets. Each question in the dataset consists of two parts: a factually correct prefix of a certain length and a fact retrieval question. Each sentence of the context prefix will describe a topic from a different perspective and with different words. Therefore, rather than using repeated sentences, we use diverse sentences to describe a fact, which is more realistic. We divide the context hijacking into eight different levels according to the length of the context prefix, from level 1 to level 8, which means the context has 10 to 80 sentences. We then observe whether the model predicts the correct next token. If the next token changes to the topic described by the context prefix (different from the answer to the fact retrieval question), we call this phenomenon "label flipping". For more detailed experimental details and examples, please refer to Appendix H.3.

Figure 5 shows how the label flipping rate of different models changes with the hijacking level. Although models exhibit varying resistance to context hijacking across different tasks, we can find that in practical LLMs, longer hijacking context will significantly increase the label flipping rate (leading to lower accuracy), while increasing the model depth can well alleviate this problem. The experiment results are consistent with our theoretical conclusions, indicating that our theoretical results can be generalized to deeper and larger LLMs in practice. Additionally, we find that instruction fine-tuning can improve the model's robustness to context hijacking in most cases, but the effect is not significant, which provides new insights for future work, such as adversarial optimization.

5 Conclusion and limitations

In this paper, we explore the robustness of transformers from the perspective of context hijacking [37]. We build a solid theoretical framework by modeling context hijacking phenomenon as a linear classification problem. We first demonstrate the context hijacking phenomenon by conducting experiments on LLMs with different depths, i.e., the output of the LLMs can be simply manipulated by modifying the context with factually correct information. This reflects an intuition: deeper models may be more robust. Then we develop a comprehensive theoretical analysis of the robustness of transformer, showing that the well-trained transformers can achieve the optimal gradient descent strategy. More specifically, we show that as the number of model layers or the length of training context increase, the model will be able to perform more fine-grained optimization steps over context samples, which can be less affected by the hijacking examples, leading to stronger robustness. Specifically considering the context hijacking task, our theory can fully explain the various phenomena, supported by a series of numerical experiments. We also conduct nonlinear experiments to extend our theoretical results.

The limitation of our work is that our model is specifically designed for context hijacking, a recent phenomenon from the real world. We only focus on the hijacking task, so new constructions may be required for other problems, and different results may be derived based on our technical framework. Besides, our current analysis follows from the recent works on modeling transformer model as a gradient descent optimizer, it is also possible that the corresponding meta optimization algorithms are more complicated ones. This will also lead to different theoretical results based on our techniques.

Our work provides a new perspective for the robustness explanation of transformers and the understanding of in-context learning ability, which offer new insights to understand the benefit of deeper architecture. Besides, our analysis on the optimal multi-step gradient descent may also be leveraged to other problems that involve the numerical optimization for linear problems.

Acknowledgments and Disclosure of Funding

We would like to thank the anonymous reviewers and area chairs for their helpful comments. This work is supported by NSFC 62306252, Hong Kong ECS award 27309624, Guangdong NSF 2024A1515012444, and the central fund from HKU IDS.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36:45614–45650, 2023.

[3] Yaroslav Aksenov, Nikita Balagansky, Sofia Lo Cicero Vaina, Boris Shaposhnikov, Alexey Gorbatovski, and Daniil Gavrilov. Linear transformers with learnable kernel functions are better in-context models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9584–9597, 2024.

[4] 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*, 2022.

[5] 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*, 2022.

[6] Cem Anil, Esin Durmus, Nina Rimsky, Mrinank Sharma, Joe Benton, Sandipan Kundu, Joshua Batson, Meg Tong, Jesse Mu, Daniel J Ford, et al. Many-shot jailbreaking. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[7] Usman Anwar, Johannes Von Oswald, Louis Kirsch, David Krueger, and Spencer Frei. Adversarial robustness of in-context learning in transformers for linear regression. *arXiv preprint arXiv:2411.05189*, 2024.

[8] Giovanni Apruzzese, Hyrum S Anderson, Savino Dambra, David Freeman, Fabio Pierazzi, and Kevin Roundy. "real attackers don't compute gradients": bridging the gap between adversarial ml research and practice. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 339–364. IEEE, 2023.

[9] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Advances in neural information processing systems*, 36, 2024.

[10] Luke Bailey, Euan Ong, Stuart Russell, and Scott Emmons. Image hijacks: Adversarial images can control generative models at runtime. In *Forty-first International Conference on Machine Learning*, 2023.

[11] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

[12] 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*, 2023.

[13] Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. *URL https://openaipublic. blob. core. windows. net/neuron-explainer/paper/index. html.(Date accessed: 14.05. 2023)*, 2, 2023.

[14] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[15] Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

[16] Xingwu Chen, Lei Zhao, and Difan Zou. How transformers utilize multi-head attention in in-context learning? a case study on sparse linear regression. *arXiv preprint arXiv:2408.04532*, 2024.

[17] Xingwu Chen and Difan Zou. What can transformer learn with varying depth? case studies on sequence learning tasks. *arXiv preprint arXiv:2404.01601*, 2024.

[18] Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. Meta-learning via language model in-context tuning. In *60th Annual Meeting of the Association for Computational Linguistics, ACL 2022*, pages 719–730. Association for Computational Linguistics (ACL), 2022.

[19] Xiang Cheng, Yuxin Chen, and Suvrit Sra. Transformers implement functional gradient descent to learn non-linear functions in context. In *Forty-first International Conference on Machine Learning*, 2023.

[20] Yixin Cheng, Markos Georgopoulos, Volkan Cevher, and Grigorios G Chrysos. Leveraging the context through multi-round interactions for jailbreaking attacks. *arXiv preprint arXiv:2402.09177*, 2024.

[21] Arijit Ghosh Chowdhury, Md Mofijul Islam, Vaibhav Kumar, Faysal Hossain Shezan, Vinija Jain, and Aman Chadha. Breaking down the defenses: A comparative survey of attacks on large language models. *arXiv preprint arXiv:2403.04786*, 2024.

[22] Antonia Creswell, Murray Shanahan, and Irina Higgins. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*, 2022.

[23] Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can gpt learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, 2023.

[24] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2019.

[25] Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. Jailbreaker: Automated jailbreak across multiple large language model chatbots. *arXiv preprint arXiv:2307.08715*, 2023.

[26] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[27] Spencer Frei and Gal Vardi. Trained transformer classifiers generalize and exhibit benign overfitting in-context. *arXiv preprint arXiv:2410.01774*, 2024.

[28] Dan Friedman, Alexander Wettig, and Danqi Chen. Learning transformer programs. *Advances in Neural Information Processing Systems*, 36, 2024.

[29] Jingwen Fu, Tao Yang, Yuwang Wang, Yan Lu, and Nanning Zheng. How does representation impact in-context learning: A exploration on a synthetic task. *arXiv preprint arXiv:2309.06054*, 2023.

[30] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

[31] Tianyu Guo, Wei Hu, Song Mei, Huan Wang, Caiming Xiong, Silvio Savarese, and Yu Bai. How do transformers learn in-context beyond simple functions? a case study on learning with representations. In *The Twelfth International Conference on Learning Representations*, 2023.

[32] Pengfei He, Han Xu, Yue Xing, Hui Liu, Makoto Yamada, and Jiliang Tang. Data poisoning for in-context learning. *arXiv preprint arXiv:2402.02160*, 2024.

[33] Jianhao Huang, Zixuan Wang, and Jason D Lee. Transformers learn to implement multi-step gradient descent with chain of thought. In *The Thirteenth International Conference on Learning Representations*, 2025.

[34] Yu Huang, Yuan Cheng, and Yingbin Liang. In-context convergence of transformers. In *Forty-first International Conference on Machine Learning*, 2023.

[35] Samy Jelassi, Michael Sander, and Yuanzhi Li. Vision transformers provably learn spatial structure. *Advances in Neural Information Processing Systems*, 35:37822–37836, 2022.

[36] Joonhyun Jeong. Hijacking context in large multi-modal models. *arXiv preprint arXiv:2312.07553*, 2023.

[37] Yibo Jiang, Goutham Rajendran, Pradeep Kumar Ravikumar, and Bryon Aragam. Do llms dream of elephants (when told not to)? latent concept association and associative memory in transformers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[38] Aounon Kumar, Chirag Agarwal, Suraj Srinivas, Aaron Jiaxun Li, Soheil Feizi, and Himabindu Lakkaraju. Certifying llm safety against adversarial prompting. *arXiv preprint arXiv:2309.02705*, 2023.

[39] Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[40] Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, pages 19565–19594. PMLR, 2023a.

[41] Yingcong Li, Ankit Singh Rawat, and Samet Oymak. Fine-grained analysis of in-context linear estimation: Data, architecture, and beyond. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[42] Licong Lin, Yu Bai, and Song Mei. Transformers as decision makers: Provable in-context reinforcement learning via supervised pretraining. In *The Twelfth International Conference on Learning Representations*, 2023.

[43] David Lindner, János Kramár, Sebastian Farquhar, Matthew Rahtz, Tom McGrath, and Vladimir Mikulik. Tracr: Compiled transformers as a laboratory for interpretability. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[45] Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. Trustworthy llms: A survey and guideline for evaluating large language models' alignment. *arXiv preprint arXiv:2308.05374*, 2023.

[46] Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.

[47] Arvind V Mahankali, Tatsunori Hashimoto, and Tengyu 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*, 2023.

[48] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

[49] Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. Metaicl: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, 2022.

[50] Zhenxing Niu, Haodong Ren, Xinbo Gao, Gang Hua, and Rong Jin. Jailbreaking attack against multimodal large language model. *arXiv preprint arXiv:2402.02309*, 2024.

[51] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

[52] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[53] Lalchand Pandia and Allyson Ettinger. Sorting through the noise: Testing robustness of information processing in pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1583–1596, 2021.

[54] Onkar Pandit and Yufang Hou. Probing for bridging inference in transformer language models. In *NAACL 2021-Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2021.

[55] Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. In *The Twelfth International Conference on Learning Representations*, 2023.

[56] Reese Pathak, Rajat Sen, Weihao Kong, and Abhimanyu Das. Transformers can optimally learn regression mixture models. In *The Twelfth International Conference on Learning Representations*, 2023.

[57] Fábio Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. In *NeurIPS ML Safety Workshop*, 2022.

[58] Jorge Pérez, Pablo Barceló, and Javier Marinkovic. Attention is turing-complete. *Journal of Machine Learning Research*, 22(75):1–35, 2021.

[59] Yao Qiang, Xiangyu Zhou, and Dongxiao Zhu. Hijacking large language models via adversarial in-context learning. *arXiv preprint arXiv:2311.09948*, 2023.

[60] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[61] Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. *Advances in Neural Information Processing Systems*, 36, 2024.

[62] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1671–1685, 2024.

[63] Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR, 2023.

[64] Chen Siyu, Sheen Heejune, Wang Tianhao, and Yang Zhuoran. Training dynamics of multi-head softmax attention for in-context learning: Emergence, convergence, and optimality. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 4573–4573. PMLR, 2024.

[65] Yuandong Tian, Yiping Wang, Beidi Chen, and Simon S Du. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. *Advances in Neural Information Processing Systems*, 36:71911–71947, 2023.

[66] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[68] Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, 2019.

[69] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

[70] Jindong Wang, HU Xixu, Wenxin Hou, Hao Chen, Runkai Zheng, Yidong Wang, Linyi Yang, Wei Ye, Haojun Huang, Xiubo Geng, et al. On the robustness of chatgpt: An adversarial and out-of-distribution perspective. In *ICLR 2023 Workshop on Trustworthy and Reliable Large-Scale Machine Learning Models*, 2023.

[71] Jiongxiao Wang, Zichen Liu, Keun Hee Park, Zhuojun Jiang, Zhaoheng Zheng, Zhuofeng Wu, Muhao Chen, and Chaowei Xiao. Adversarial demonstration attacks on large language models. *arXiv preprint arXiv:2305.14950*, 2023.

[72] Zixuan Wang, Stanley Wei, Daniel Hsu, and Jason D Lee. Transformers provably learn sparse token selection while fully-connected nets cannot. In *Forty-first International Conference on Machine Learning*, 2024.

[73] Colin Wei, Yining Chen, and Tengyu Ma. Statistically meaningful approximation: a case study on approximating turing machines with transformers. *Advances in Neural Information Processing Systems*, 35:12071–12083, 2022.

[74] Zeming Wei, Yifei Wang, Ang Li, Yichuan Mo, and Yisen Wang. Jailbreak and guard aligned language models with only few in-context demonstrations. *arXiv preprint arXiv:2310.06387*, 2023.

[75] Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. In *International Conference on Machine Learning*, pages 11080–11090. PMLR, 2021.

[76] 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*, 2023.

[77] 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*, 2021.

[78] Xilie Xu, Keyi Kong, Ning Liu, Lizhen Cui, Di Wang, Jingfeng Zhang, and Mohan Kankanhalli. An llm can fool itself: A prompt-based adversarial attack. In *The Twelfth International Conference on Learning Representations*, 2023.

[79] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, page 100211, 2024.

[80] Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations*, 2023.

[81] Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. Gptfuzzer: Red teaming large language models with auto-generated jailbreak prompts. *arXiv preprint arXiv:2309.10253*, 2023.

[82] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020.

[83] Chenyang Zhang, Xuran Meng, and Yuan Cao. Transformer learns optimal variable selection in group-sparse classification. In *The Thirteenth International Conference on Learning Representations*, 2025.

[84] Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

[85] Ruiqi Zhang, Jingfeng Wu, and Peter L Bartlett. In-context learning of a linear transformer block: benefits of the mlp component and one-step gd initialization. *arXiv preprint arXiv:2402.14951*, 2024.

[86] Yunqing Zhao, Tianyu Pang, Chao Du, Xiao Yang, Chongxuan Li, Ngai-Man Man Cheung, and Min Lin. On evaluating adversarial robustness of large vision-language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[87] Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua M Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. In *The Twelfth International Conference on Learning Representations*, 2023.

[88] Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Zhenqiang Gong, et al. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *arXiv e-prints*, pages arXiv–2306, 2023.

[89] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: Main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   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: The paper discuss the limitations of the work.

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: The paper provides the full set of assumptions and a complete (and correct) proof for each theoretical result in appendix.

   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: We provide experimental settings and training details in appendix.

   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: [No]

   Justification: The code is being organized and we will provide it later.

   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: We provide experimental settings and training details in appendix.

   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: [Yes]

   Justification: We provide error bars and some other appropriate information about the statistical significance for most experiments.

   Guidelines:

   - 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: [Yes]

   Justification: We provide experimental settings and training details in appendix.

   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: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   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: There is no societal impact of the work performed.

    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: The paper poses no such risks.

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

    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: [NA]

    Justification: The paper does not release new assets.

    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: The paper does not involve crowdsourcing nor research with human subjects.

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: The paper does not involve crowdsourcing nor research with human subjects.

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: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

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.

A Related works

In-context learning via transformers. The powerful performance of transformers is generally believed to come from its in-context learning ability [14, 18, 49, 44, 77]. A line of recent works study the phenomenon of in-context learning from both theoretical [9, 31, 42, 16, 27, 34, 64, 41] and empirical [30, 4, 40, 61, 56, 55, 12, 29, 39, 3] perspectives on diverse settings. [14] first showed that GPT-3 can perform in-context learning. [16] studied the role of different heads within transformers in performing in-context learning focusing on the sparse linear regression setting. [27] studied the ability of one-layer linear transformers to perform in-context learning for linear classification tasks. [3] demonstrated linear transformers perform less well on in-context tasks, but the linear transformer here refers to a model that uses a kernel function to approximate the standard attention computation for computational efficiency, albeit at the expense of some in-context learning performance. In contrast, the linear transformer in our paper is a model that removes the activation function from the standard transformer. As discussed in the main paper 2.2, this is a very common setting in transformer theory research. In addition, while not considering the ICL setting, [83] studied the training of one-layer transformers on the group-sparse linear classification setting.

Mechanism interpretability of transformers. Among the various theoretical interpretations of transformers [28, 82, 24, 43, 54, 58, 13, 73, 75, 87, 17, 83], one of the most widely studied theories is the ability of transformers to implement optimization algorithms such as gradient descent [69, 2, 85, 9, 76, 19, 5, 23, 84]. [69] theoretically and empirically proved that transformers can learn in-context by implementing a single step of gradient descent per layer. [2] theoretically analyzed that transformers can learn to implement preconditioned gradient descent for in-context learning. [85] considered ICL in the setting of linear regression with a non-zero mean Gaussian prior, a more general and common scenario where different tasks share a signal, which is highly relevant to our work.

Robustness of transformers. The security issues of large language models have always attracted a great deal of attention [79, 46, 57, 89, 8]. However, most of the research focuses on jail-breaking black-box models [21], such as context-based adversarial attacks [38, 74, 78, 71, 88, 20, 70]. There is very little white-box interpretation work of attacks on the transformer, the foundation model of LLMs [59, 10, 32, 7, 37]. [59] first considered attacking large language models during in-context learning, but they did not study the role of transformers in robustness. [37] proposed the phenomenon of context hijacking, which became the key motivation of our work. They analyzed this problem from the perspective of associative memory models instead of the in-context learning ability of transformers.

B Notations

Given two sequences $\{x_n\}$ and $\{y_n\}$, we denote $x_n = O(y_n)$ if there exist some absolute constant $C_1 > 0$ and $N > 0$ such that $|x_n| \leq C_1|y_n|$ for all $n \geq N$. Similarly, we denote $x_n = \Omega(y_n)$ if there exist $C_2 > 0$ and $N > 0$ such that $|x_n| \geq C_2|y_n|$ for all $n > N$. We say $x_n = \Theta(y_n)$ if $x_n = O(y_n)$ and $x_n = \Omega(y_n)$ both holds. Additionally, we denote $x_n = o(y_n)$ if, for any $\epsilon > 0$, there exists some $N(\epsilon) > 0$ such that $|x_n| \leq \epsilon|y_n|$ for all $n \geq N(\epsilon)$, and we denote $x_n = \omega(y_n)$ if $y_n = o(x_n)$. We use $\widetilde{O}(\cdot)$, $\widetilde{\Omega}(\cdot)$, and $\widetilde{\Theta}(\cdot)$ to hide logarithmic factors in these notations respectively. Finally, for any $n \in \mathbb{N}_+$, we use $[n]$ to denote the set $\{1, 2, \cdots, n\}$.

C Proof of Proposition 3.1

In this section we provide a proof for Proposition 3.1.

Proof of Proposition 3.1. Our proof is inspired by Lemma 1 in [2], while we consider a non-zero initialization. We first provide the parameters $\mathbf{W}_E, \mathbf{P}_\ell, \mathbf{Q}_\ell \in \mathbb{R}^{(d+1)\times(d+1)}$ of a L-layers transformer.

$$\mathbf{W}_E = \begin{bmatrix} \mathbf{I}_n & 0 \\ -\mathbf{w}_{\mathrm{gd}}^{(0)} & 1 \end{bmatrix}, \quad \mathbf{P}_\ell = \begin{bmatrix} \mathbf{0}_{d\times d} & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{Q}_\ell = \begin{bmatrix} -\mathbf{\Gamma}_\ell & 0 \\ 0 & 0 \end{bmatrix} \quad \text{where } \mathbf{\Gamma}_\ell \in \mathbb{R}^{d\times d}.$$

For the linear classification problem, the input sample $\mathbf{Z}_0 \in \mathbb{R}^{(d+1)\times(n+1)}$ consists of $\{(\mathbf{x}_i, y_i)\}_i = 1^n$ and $(\mathbf{x}_{\mathrm{query}}, y_{\mathrm{query}})$ in (2.1) , which will first be embedded by $\mathbf{W}_E$. Let $\mathbf{X}^{(0)} \in \mathbb{R}^{d\times(n+1)}$ denote

the first d rows of $\mathbf{W}_E(\mathbf{Z}_0)$ and let $\mathbf{Y}^{(0)} \in \mathbb{R}^{1\times(n+1)}$ denote the $(d+1)$-th row of $\mathbf{W}_E(\mathbf{Z}_0)$. In subsequent iterative updates in (2.3), the values at the same position will be denoted as $\mathbf{X}^{(l)}$ and $\mathbf{Y}^{(l)}$, for $l = 1, \ldots, L$. Similarly, define $\bar{\mathbf{X}}^{(l)} \in \mathbb{R}^{d\times n}$ and $\bar{\mathbf{Y}}^{(l)} \in \mathbb{R}^{1\times n}$ as matrices that exclude the last query sample $(\mathbf{x}_{\text{query}}^{(l)}, y_{\text{query}}^{(l)})$. That is, they only contain the first n columns of the output of the l-th layer. Let $\mathbf{x}_i^{(l)}$ and $y_i^{(l)}$ be the i-th pair of samples output by the l-th layer. Define a function $g(\mathbf{x}, y, l) : \mathbb{R}^d \times \mathbb{R} \times \mathbb{Z} \to \mathbb{R}$: let $\mathbf{x}_{\text{query}}^{(0)} = \mathbf{x}$ and $y_{\text{query}}^{(0)} = y - \langle \mathbf{w}_{\text{gd}}^{(0)}, \mathbf{x}\rangle$, then $g(\mathbf{x}, y, l) := y_{\text{query}}^{(k)}$. Next, based on the update formula (2.3) and the parameters constructed above, we have:

$$\mathbf{X}^{(l+1)} = \mathbf{X}^{(l)} = \cdots = \mathbf{X}^{(0)}, \quad \mathbf{Y}^{(l+1)} = \mathbf{Y}^{(l)} - \mathbf{Y}^{(l)}\mathbf{M}(\mathbf{X}^{(0)})^\top \mathbf{\Gamma}_l \mathbf{X}^{(0)}.$$

Then for all $i \in \{1, \ldots, n\}$,

$$y_i^{(l+1)} = y_i^{(l)} - \sum_{j=1}^{n} \mathbf{x}_i{}^\top \mathbf{\Gamma}_l \mathbf{x}_j y_j^{(l)}.$$

So $y_i^{(l+1)}$ does not depend on $y_{\text{query}}^{(l+1)}$. For query position,

$$y_{\text{query}}^{(l+1)} = y_{\text{query}}^{(l)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_l \mathbf{x}_j y_j^{(l)}.$$

Then we obtain $g(\mathbf{x}, y, l)$ and $g(\mathbf{x}, 0, l)$:

$$\begin{aligned}
g(\mathbf{x}, y, l) &= y_{\text{query}}^{(l-1)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)} \\
&= y_{\text{query}}^{(l-2)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-2} \mathbf{x}_j y_j^{(l-2)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)} \\
&\vdots \\
&= y_{\text{query}}^{(0)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_0 \mathbf{x}_j y_j^{(0)} - \cdots - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)} \\
&= y - \langle \mathbf{w}_{\text{gd}}^{(0)}, \mathbf{x}_{\text{query}}\rangle - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_0 \mathbf{x}_j y_j^{(0)} - \cdots - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)}; \\
g(\mathbf{x}, 0, k) &= y_{\text{query}}^{(l-1)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)} \\
&= y_{\text{query}}^{(l-2)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-2} \mathbf{x}_j y_j^{(l-2)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)} \\
&\vdots \\
&= y_{\text{query}}^{(0)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_0 \mathbf{x}_j y_j^{(0)} - \cdots - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)} \\
&= -\langle \mathbf{w}_{\text{gd}}^{(0)}, \mathbf{x}_{\text{query}}\rangle - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_0 \mathbf{x}_j y_j^{(0)} - \cdots - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \mathbf{\Gamma}_{l-1} \mathbf{x}_j y_j^{(l-1)};.
\end{aligned}$$

So we have $g(\mathbf{x}, y, l) = g(\mathbf{x}, 0, l) + y$. Observing $g(\mathbf{x}, 0, l)$, we can find that it is linear in $\mathbf{x}$ for the reason that every term of $g(\mathbf{x}, 0, l)$ is linear in $\mathbf{x}_{\text{query}}$, which means we can rewrite it. We verify that there exists a $\boldsymbol{\theta}_l \in \mathbb{R}^d$ for each $l \in [L]$, such that for all $\mathbf{x}, y$,

$$g(\mathbf{x}, y, l) = g(\mathbf{x}, 0, k) + y = \langle \boldsymbol{\theta}_l, \mathbf{x}\rangle + y.$$

Let $l = 0$, we have $\langle \boldsymbol{\theta}_0, \mathbf{x}\rangle = g(\mathbf{x}, y, 0) - y = y_{\text{query}}^{(0)} - y = -\langle \mathbf{w}_{\text{gd}}^{(0)}, \mathbf{x}_{\text{query}}\rangle$, so $\boldsymbol{\theta}_0 = -\mathbf{w}_{\text{gd}}^{(0)}$. Next, we will show that for all $(\mathbf{x}_i, y_i) \in \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_n, y_n), (\mathbf{x}_{\text{query}}, y_{\text{query}})\}$,

$$g(\mathbf{x}_i, y_i, l) = y_i^{(l)} = \langle \boldsymbol{\theta}_l, \mathbf{x}_i\rangle + y_i.$$

Observing the update formulas for $y_i^{(l+1)}$ and $y_{\text{query}}^{(l+1)}$, if we let $\mathbf{x}_{\text{query}} := \mathbf{x}_i$ for some i, we can get that $y_i^{(l+1)} = y_{\text{query}}^{(l+1)}$ because $y_i^{(0)} = y_{\text{query}}^{(0)}$ by definition. This indicates that

$$\bar{\mathbf{Y}}^{(l)} = \bar{\mathbf{Y}}^{(0)} + \boldsymbol{\theta}_l^T \bar{\mathbf{X}}.$$

Finally, we can rewrite the update formula for $y_k^{(n+1)}$

$$\begin{aligned} y_{\text{query}}^{(l+1)} &= y_{\text{query}}^{(l)} - \sum_{j=1}^{n} \mathbf{x}_{\text{query}}^\top \boldsymbol{\Gamma}_l \mathbf{x}_j y_j^{(l)}. \\ &= y_{\text{query}}^{(l)} - \langle \boldsymbol{\Gamma}_l \bar{\mathbf{X}} (\bar{\mathbf{Y}}^{(l)})^\top, \mathbf{x}_{\text{query}} \rangle \\ \Rightarrow \quad \left\langle \boldsymbol{\theta}_{l+1}, \mathbf{x}_{\text{query}} \right\rangle &= \langle \boldsymbol{\theta}_l, \mathbf{x}_{\text{query}} \rangle - \langle \boldsymbol{\Gamma}_l \bar{\mathbf{X}} \big(\bar{\mathbf{X}}^\top \boldsymbol{\theta}_l + (\bar{\mathbf{Y}}^{(0)})^\top \big), \mathbf{x}_{\text{query}} \Big\rangle \end{aligned}$$

Since $\mathbf{x}_{\text{query}}$ is an arbitrary variable, we get the more general update formula for $\boldsymbol{\theta}_l$:

$$\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}_l - \langle \boldsymbol{\Gamma}_l \bar{\mathbf{X}} \left(\bar{\mathbf{X}}^\top \boldsymbol{\theta}_l + (\bar{\mathbf{Y}}^{(0)})^\top \right) \rangle.$$

Notice that we use the mean squared error, we have

$$\begin{aligned} \widetilde{L}(\mathbf{w}) &= \frac{1}{2} \sum_{i=1}^{n} (\langle \mathbf{w}, \mathbf{x}_i \rangle - y_i)^2 \\ &= \frac{1}{2} \| \bar{\mathbf{X}}^\top \mathbf{w} - (\bar{\mathbf{Y}}^{(0)})^\top \|_2. \end{aligned}$$

Then we get its gradient $\nabla \widetilde{L}(\mathbf{w}) = \bar{\mathbf{X}} \left(\bar{\mathbf{X}}^\top \mathbf{w} - (\bar{\mathbf{Y}}^{(0)})^\top \right)$. Let $\mathbf{w}_{\text{gd}}^{(l)} := -\boldsymbol{\theta}_l$, we have

$$\begin{aligned} \boldsymbol{\theta}_{l+1} &= \boldsymbol{\theta}_l - \langle \boldsymbol{\Gamma}_l \bar{\mathbf{X}} \left(\bar{\mathbf{X}}^\top \boldsymbol{\theta}_l + \bar{\mathbf{Y}}_0^\top \right) \rangle \\ \Rightarrow \quad \mathbf{w}_{\text{gd}}^{(l+1)} &= \mathbf{w}_{\text{gd}}^{(l)} - \langle \boldsymbol{\Gamma}_k \bar{\mathbf{X}} \left(\bar{\mathbf{X}}^\top \mathbf{w}_{\text{gd}}^{(l)} - (\bar{\mathbf{Y}}^{(0)})^\top \right) \rangle \\ &= \mathbf{w}_{\text{gd}}^{(l)} - \boldsymbol{\Gamma}_l \nabla \widetilde{L}(\mathbf{w}_{\text{gd}}^{(l)}). \end{aligned}$$

And the output of the l-th layer $y_{\text{query}}^{(l)}$ is

$$g\left(\mathbf{x}_{\text{query}}, y_{\text{query}}, l\right) = y_{\text{query}} + \langle \boldsymbol{\theta}_l, \mathbf{x}_{\text{query}} \rangle = y_{\text{query}} - \langle \mathbf{w}_{\text{gd}}^{(l)}, \mathbf{x}_{\text{query}} \rangle.$$

In our settings, we have $y_k^{n+1} = -\langle \mathbf{w}_{\text{gd}}^{(l)}, \mathbf{x}_{\text{query}} \rangle$ because the input query label is 0. □

D Gradient descent updates of parameters

In this section, we provide further details regarding the updating of parameters $\mathbf{w}_{\text{gd}}^{(l)}$, which will be utilized in subsequent proof. Besides, it can directly imply Proposition 3.2. Before demonstrating the mathematical, we first introduce several utility notations, which will be used in subsequent technical derivations and proofs. We denote $\mathcal{S}_{l,k}$ as the set of all k-dimensional tuples whose entries are drawn from $\{0, 1, \ldots, l-1\}$ without replacement, i.e.

$$\mathcal{S}_{t,k} = \big\{ (j_1, j_2, \ldots, j_k) | j_1, j_2, \ldots, j_k \in \{0, 1, \ldots, l-1\}; j_1 \neq j_2 \neq \ldots \neq j_k \big\}.$$

Then given the set of all historical learning rates before or at l-th iteration, i.e. $\{\alpha_0, \alpha_1, \ldots, \alpha_{l-1}\}$, and $\mathcal{S}_{l,k}$ defined above, we define $A_{l,k}$ as

$$A_{l,k} := \sum_{(j_1, j_2, \ldots, j_k) \in \mathcal{S}_{l,k}} \prod_{\kappa=1}^{k} \alpha_{j_\kappa}.$$

Then we can observe that the permutation of elements of $\{\alpha_0, \alpha_1, \ldots, \alpha_{l-1}\}$ would not change the value of $A_{l,k}$. Then based on these notations, we present mathematical derivation in the following.

By some basic gradient calculations, we can re-write the iterative rule of gradient descent (3.1) as

$$\mathbf{w}_{\text{gd}}^{(l+1)} = \mathbf{w}_{\text{gd}}^{(l)} - \alpha_l \nabla L(\mathbf{w}_{\text{gd}}^{(l)})$$

$$= \mathbf{w}_{\mathrm{gd}}^{(l)} - \alpha_l \sum_{i=1}^{n} \big(\langle \mathbf{w}_{\mathrm{gd}}^{(l)}, \mathbf{x}_i\rangle - \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\big) \cdot \mathbf{x}_i$$
$$= \bigg(\mathbf{I}_d - \alpha_l \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)\bigg) \cdot \mathbf{w}_{\mathrm{gd}}^{(l)} + \alpha_l \bigg(\sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle) \cdot \mathbf{x}_i\bigg). \tag{D.1}$$

Based on this detailed iterative formula, and the definition of $\mathcal{S}_{l,k}$ and $A_{l,k}$ above, we present and prove the following lemma, which characterizes the closed-form expression for $\mathbf{w}^{(l)}$.

Lemma D.1. For the iterates of gradient descent, i.e. $\mathbf{w}_{\mathrm{gd}}^{(l)}$'s with $l \in \{0, 1, \ldots, L-1\}$, it holds that

$$\mathbf{w}_{\mathrm{gd}}^{(l)} = \bigg(\mathbf{I}_d + \sum_{k=1}^{l} A_{l,k} \Big(-\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)^k\bigg) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + \bigg(\sum_{k=1}^{l} A_{l,k} \Big(-\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)^{k-1}\bigg)$$
$$\cdot \bigg(\sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle) \cdot \mathbf{x}_i\bigg). \tag{D.2}$$

Proof of Lemma D.1. Before we demonstrate our proof, we first present some conclusions regarding $\mathcal{S}_{l,k}$ and $A_{l,k}$. By directly applying the Binomial theorem and the definition of $A_{l,k}$, we can obtain that

$$\prod_{k=0}^{l-1} \bigg(\mathbf{I}_d + \alpha_k \Big(-\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)\bigg) = \mathbf{I}_d + \sum_{k=1}^{l} A_{l,k} \Big(-\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)^k. \tag{D.3}$$

Additionally, by utilizing the definition of $\mathcal{S}_{l,k}$, we can easily derive that

$$\begin{aligned}
\mathcal{S}_{l+1,k} =& \big\{(j_1, j_2, \ldots, j_k) | j_1, j_2, \ldots, j_k \in \{0, 1, \ldots, l\}; j_1 \neq j_2 \neq \ldots \neq j_k\big\} \\
=& \big\{(j_1, j_2, \ldots, j_k) | j_1, j_2, \ldots, j_k \in \{0, 1, \ldots, l-1\}; j_1 \neq j_2 \neq \ldots \neq j_k\big\} \\
& \cup \big\{(j_1, j_2, \ldots, j_{k-1}, l) | j_1, j_2, \ldots, j_{k-1} \in \{0, 1, \ldots, l-1\}; j_1 \neq j_2 \neq \ldots \neq j_{k-1}\big\} \\
=& \mathcal{S}_{l,k} \cup \big\{(j_1, j_2, \ldots, j_{k-1}, l) | (j_1, j_2, \ldots, j_{k-1}) \in \mathcal{S}_{l,k-1}\big\},
\end{aligned}$$

holds when $k \leq l$. This result can further imply that

$$\begin{aligned}
A_{l+1,k} =& \sum_{(j_1, j_2, \ldots, j_k) \in \mathcal{S}_{l+1,k}} \prod_{\kappa=1}^{k} \alpha_{j_\kappa} = \sum_{(j_1, j_2, \ldots, j_k) \in \mathcal{S}_{l,k}} \prod_{\kappa=1}^{k} \alpha_{j_\kappa} + \sum_{(j_1, j_2, \ldots, j_{k-1}) \in \mathcal{S}_{l,k-1}} \prod_{\kappa=1}^{k-1} \alpha_{j_\kappa} \alpha_l \\
=& A_{l,k} + \alpha_l A_{l,k-1}.
\end{aligned} \tag{D.4}$$

holds when $k \leq l$. Additionally, it is straightforward that

$$A_{l+1,l+1} = \alpha_l A_{l,l}; \quad A_{l+1,1} = A_{l,1} + \alpha_l. \tag{D.5}$$

With these conclusions in hands, we will begin proving this lemma by induction. When $l = 1$, by the iterative rule (D.1), we can obtain that

$$\mathbf{w}_{\mathrm{gd}}^{(1)} = \bigg(\mathbf{I}_d - \alpha_0 \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)\bigg) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + \alpha_0 \bigg(\sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle) \cdot \mathbf{x}_i\bigg),$$

which follows the conclusion of (D.2) due to (D.3) and the definition of $A_{1,1}$. By induction, we assume that (D.2) holds at l-th iteration. Then at $(l+1)$-th iteration, we can obtain that,

$$\begin{aligned}
\mathbf{w}_{\mathrm{gd}}^{(l+1)} =& \bigg(\mathbf{I}_d - \alpha_l \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)\bigg) \cdot \mathbf{w}_{\mathrm{gd}}^{(l)} + \alpha_l \bigg(\sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle) \cdot \mathbf{x}_i\bigg) \\
=& \alpha_l \bigg(\sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle) \cdot \mathbf{x}_i\bigg) + \bigg(\mathbf{I}_d - \alpha_l \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)\bigg) \\
& \cdot \bigg\{\prod_{\tau=0}^{l-1} \bigg(\mathbf{I}_d - \alpha_\tau \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)\bigg) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + \bigg(\sum_{k=1}^{t} A_{l,k} \Big(-\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top\Big)^{k-1}\bigg)
\end{aligned}$$

$$
\begin{aligned}
&\cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \bigg\} \\
=&\alpha_l \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) + \prod_{k=0}^{l} \Big(\mathbf{I}_d - \alpha_k \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big) \Big) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} \\
&+ \bigg(\sum_{k=1}^{l} A_{l,k} \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k-1} + \sum_{k=1}^{l} \alpha_l A_{l,k} \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k} \bigg) \\
&\cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
=&\prod_{k=0}^{l} \Big(\mathbf{I}_d - \alpha_k \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big) \Big) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + (A_{l,1} + \alpha_l) \cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
&+ \bigg(\sum_{k=2}^{l} (A_{l,k} + \alpha_l A_{l,k-1}) \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k-1} \bigg) \cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
&+ \alpha_l A_{l,l} \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{l} \cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
=&\prod_{k=0}^{l} \Big(\mathbf{I}_d - \alpha_k \Big(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big) \Big) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + A_{l+1,1} \cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
&+ \bigg(\sum_{k=2}^{t} A_{l+1,k} \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k-1} \bigg) \cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
&+ A_{l+1,l+1} \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k} \cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg) \\
=&\bigg(\mathbf{I}_d + \sum_{k=1}^{l+1} A_{l+1,k} \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k} \bigg) \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + \bigg(\sum_{k=1}^{l+1} A_{l+1,k} \Big(- \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \Big)^{k-1} \bigg) \\
&\cdot \bigg(\sum_{i=1}^{n} \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \cdot \mathbf{x}_i \bigg).
\end{aligned}
$$

The second equality holds by substituting $\mathbf{w}_{\mathrm{gd}}^{(l)}$ with its expansion from (D.2), assuming it is valid at the l-th iteration by induction. The third and fourth equalities are established by rearranging the terms. The penultimate equality is derived by applying the conclusions regarding $A_{l,k}$ from (D.4) and (D.5). The final equality is obtained by applying (D.3). This demonstrates that (D.2) still holds at $l+1$-th iteration given it holds at l-th iteration, which finishes the proof of induction. □

Lemma D.1 demonstrate that the learning rates α_l's will only influence $\mathbf{w}_{\mathrm{gd}}^{(L)}$ by determining the value of $A_{L,k}$'s. While as we have discussed above, the values of $A_{L,k}$'s depend solely on the elements in the $\{\alpha_0, \ldots, \alpha_{L-1}\}$, and remain unchanged when the order of these learning rates is rearranged. Consequently, the permutation of $\{\alpha_0, \ldots, \alpha_{L-1}\}$ will also not affect the value of $\mathbf{w}_{\mathrm{gd}}^{(L)}$, thereby confirming that Proposition 3.2 holds.

E Proof of Theorem 3.3

In this section, we provide a detailed proof for Theorem 3.3. We begin by introducing and proving a lemma that demonstrates how $\mathbf{w}_{\mathrm{gd}}^{(0)}$ must align with the direction of $\boldsymbol{\beta}^*$. This alignment constrains the choice of $\mathbf{w}_{\mathrm{gd}}^{(0)}$ to a scalar multiple of $\boldsymbol{\beta}^*$, specifically in the form of $c_0 \cdot \boldsymbol{\beta}^*$. Additionally, in the subsequent sections, we will use the notation $\widehat{\boldsymbol{\Sigma}} = \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top$.

Lemma E.1. Under the same conditions with Theorem 3.3, to minimize the loss $\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)})$, $\mathbf{w}_{\mathrm{gd}}^{(0)}$ is always in the form of $c_0 \cdot \boldsymbol{\beta}^*$.

Proof of Lemma E.1. Utilizing the independence among the examples in $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n), (\mathbf{x}_{\mathrm{query}}, y_{\mathrm{query}})\}$, and $\mathbf{w}^*$, we can expand $\mathcal{R}_{\mathbf{w}_{\mathrm{gd}}^{(L)}}$ by law of total expectation as

$$\begin{aligned}
\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) =& \mathbb{E}_{T,\mathbf{w}^*}\big[\big(\langle \mathbf{w}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}}\rangle - \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_{\mathrm{query}}\rangle)\big)^2\big] \\
=& \mathbb{E}_{T,\mathbf{w}^*}\big[\langle \mathbf{w}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}}\rangle^2 - 2\,\mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_{\mathrm{query}}\rangle)\langle \mathbf{w}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}}\rangle\big] + 1 \\
=& \mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\Big[\mathbb{E}_{(\mathbf{x}_{\mathrm{query}}, y_{\mathrm{query}})}\big[\langle \mathbf{w}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}}\rangle^2 \\
& - 2\,\mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_{\mathrm{query}}\rangle)\langle \mathbf{w}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}}\rangle \big| \{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*\big]\Big] + 1 \\
=& \mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\left[\left\|\mathbf{w}_{\mathrm{gd}}^{(L)} - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\right\|_2^2\right] + 1 - \frac{2}{\pi},
\end{aligned}$$

where the last equality holds since $\mathbb{E}_{\mathbf{x}_{\mathrm{query}}}[\langle \mathbf{w}, \mathbf{x}_{\mathrm{query}}\rangle^2] = \mathbf{w}^\top \mathbb{E}_{\mathbf{x}_{\mathrm{query}}}[\mathbf{x}_{\mathrm{query}}\mathbf{x}_{\mathrm{query}}^\top]\mathbf{w} = \|\mathbf{w}\|_2^2$ when $\mathbf{w}$ is independent with $\mathbf{x}_{\mathrm{query}}$, and $\mathbb{E}_{\mathbf{x}_{\mathrm{query}}}[\langle \mathbf{w}_1, \mathbf{x}_{\mathrm{query}}\rangle \,\mathrm{sign}(\langle \mathbf{w}_2, \mathbf{x}_{\mathrm{query}}\rangle)] = \sqrt{\frac{2}{\pi}}\langle \mathbf{w}^*, \mathbf{w}_{\mathrm{gd}}^{(L)}\rangle$ implied by Lemma G.1. Therefore in the next we attempt to optimize the first term $\mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\left[\left\|\mathbf{w}_{\mathrm{gd}}^{(L)} - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\right\|_2^2\right]$. By applying the closed form of $\mathbf{w}_{\mathrm{gd}}^{(L)}$ in Lemma D.1 with all $\alpha_l = \alpha$, we have

$$\mathbf{w}_{\mathrm{gd}}^{(L)} = \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^L \cdot \mathbf{w}_{\mathrm{gd}}^{(0)} + \alpha \sum_{l=0}^{L-1} \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^l \cdot \Big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\Big).$$

Based on this, we can further derive that

$$\begin{aligned}
\mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\left[\left\|\mathbf{w}_{\mathrm{gd}}^{(L)} - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\right\|_2^2\right] =& (\mathbf{w}_{\mathrm{gd}}^{(0)})^\top \mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\big[\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^{2L}\big]\mathbf{w}_{\mathrm{gd}}^{(0)} \\
& - 2\alpha(\mathbf{w}_{\mathrm{gd}}^{(0)})^\top \mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\Big[\sum_{l=0}^{L-1}\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^{l+L} \\
& \cdot \Big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\Big)\Big] + C \\
=& c_1\big\|\mathbf{w}_{\mathrm{gd}}^{(0)}\big\|_2^2 - 2c_2\langle \mathbf{w}_{\mathrm{gd}}^{(0)}, \boldsymbol{\beta}^*\rangle + C \\
=& c_1\Big\|\mathbf{w}_{\mathrm{gd}}^{(0)} - \frac{c_2}{c_1}\boldsymbol{\beta}^*\Big\|_2^2 + C - \frac{c_2^2}{c_1},
\end{aligned} \tag{E.1}$$

where c_1, c_2, C are some scalar independent of $\mathbf{w}_{\mathrm{gd}}^{(0)}$. The second inequality holds since $\mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\big[\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^{2L}\big] = c_1\mathbf{I}_d$ for some scalar c_1, guaranteed by Lemma G.2, and $\mathbb{E}_{\{(\mathbf{x}_i, y_i)\}_{i=1}^n, \mathbf{w}^*}\big[\sum_{l=0}^{L-1}\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^{l+L} \cdot \big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\big)\big] = c_2\boldsymbol{\beta}^*$ for some scalar c_2, guaranteed by Lemma G.3. As the result of (D.3) is a quartic function of $\mathbf{w}_{\mathrm{gd}}^{(0)}$, we can easily conclude that it achieves the minimum value when $\mathbf{w}_{\mathrm{gd}}^{(0)} = c_0\boldsymbol{\beta}^*$ for some scalar c_0, which completes the proof. □

Based on Lemma E.1, in the following proof, we will directly replace $\mathbf{w}_{\mathrm{gd}}^{(0)}$ with $c_0\boldsymbol{\beta}^*$ and attempt to find the optimal c_0. Now we are ready to prove the following theorem, a representation of Theorem 3.3.

Theorem E.2 (Restate of Theorem 3.3)**.** For training distribution $\mathcal{D}_{\mathrm{tr}}$ in Definition 2.1, suppose that the training context length n is sufficiently large such that $n \geq \widetilde{\Omega}(\max\{d^2, dL\})$. Additionally,

suppose that the perturbation of $\mathbf{w}^*$ around its expectation $\boldsymbol{\beta}^*$ is smaller than $\frac{\pi}{2}$, i.e. $\langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle > 0$. then for any learning rate α and initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$, it holds that

$$\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) \leq \Theta\big((1-\alpha n)^L \|\mathbf{w}_{\mathrm{gd}}^{(0)} - c_1 \mathbf{w}^*\|_2^2\big) + \widetilde{\Theta}(\alpha d L) + C,$$

where both c_1, C are absolute constants. Additionally, by taking $\mathbf{w}_{\mathrm{gd}}^{(0)} = c_1 \boldsymbol{\beta}^*$ and $\alpha = \widetilde{\Theta}(\frac{1}{nL})$, the upper bound above achieve its optimal rates as

$$\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) \leq \widetilde{\Theta}\Big(\frac{d}{n}\Big) + C.$$

Proof of Theorem E.2. Utilizing the fact that $\mathbf{I}_d - \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^L = \alpha \sum_{l=0}^{L-1} \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^l \widehat{\boldsymbol{\Sigma}}$ and $\mathbf{w}_{\mathrm{gd}}^{(0)} = c_0 \boldsymbol{\beta}^*$, we can re-write the close form of $\mathbf{w}_{\mathrm{gd}}^{(L)}$ as

$$\begin{aligned}
\mathbf{w}_{\mathrm{gd}}^{(L)} =& \Big(\mathbf{I}_d - \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^L\Big) \cdot \sqrt{\frac{2}{\pi}}\mathbf{w}^* + \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^L \cdot c_0\boldsymbol{\beta}^* \\
& - \alpha \sum_{l=0}^{L-1} \big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^l \cdot \bigg(\sqrt{\frac{2}{\pi}}\widehat{\boldsymbol{\Sigma}}\mathbf{w}^* - \Big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\Big)\bigg)
\end{aligned}$$

Then by the similar calculation to Lemma E.1, we have

$$\begin{aligned}
\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) =& \mathbb{E}\bigg[\Big\|\mathbf{w}_{\mathrm{gd}}^{(L)} - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\Big\|_2^2\bigg] + C \\
=& \mathbb{E}\bigg[\bigg\|\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^L \cdot \Big(c_0\boldsymbol{\beta}^* - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\Big) - \alpha\sum_{l=0}^{L-1}\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^l \\
& \cdot \bigg(\sqrt{\frac{2}{\pi}}\widehat{\boldsymbol{\Sigma}}\mathbf{w}^* - \Big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\Big)\bigg)\bigg\|_2^2\bigg] + C \\
\leq& 2\,\underbrace{\mathbb{E}\bigg[\Big\|\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^L \cdot \Big(c_0\boldsymbol{\beta}^* - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\Big)\Big\|_2^2\bigg]}_{I} \\
& + 2\,\underbrace{\mathbb{E}\bigg[\alpha^2\Big\|\sum_{l=0}^{L-1}\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)^l \cdot \bigg(\sqrt{\frac{2}{\pi}}\widehat{\boldsymbol{\Sigma}}\mathbf{w}^* - \Big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\Big)\bigg)\Big\|_2^2\bigg]}_{II} + C
\end{aligned}$$

where the last inequality hols by $(a+b)^2 \leq 2a^2 + 2b^2$, and C is an absolute constant. Therefore, in the following, we discuss the upper-bounds for I and II respectively. For I, we have

$$I \leq \mathbb{E}\Big[\Big\|\big(\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big)\Big\|_2^{2L}\Big] \cdot \mathbb{E}\bigg[\Big\|\Big(c_0\boldsymbol{\beta}^* - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\Big)\Big\|_2^2\bigg] \leq O\big((1-\alpha n)^{2L}\big) \cdot \mathbb{E}\bigg[\Big\|\Big(c_0\boldsymbol{\beta}^* - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\Big)\Big\|_2^2\bigg],$$

where the first inequality is derived by the independence among $\mathbf{x}_i$ and $\mathbf{w}^*$ and the submultiplicativity of ℓ_2 norm, and the second inequality holds by the concentration results regarding $\|\widehat{\boldsymbol{\Sigma}}\|_2$ provided in Lemma G.4. For II, we can derive that

$$II \leq \underbrace{\mathbb{E}\Big[\alpha^2 \sum_{l_1,l_2=0}^{L-1} \big\|\mathbf{I}_d - \alpha\widehat{\boldsymbol{\Sigma}}\big\|_2^{l_1+l_2}\Big]}_{II.1} \underbrace{\mathbb{E}\bigg[\Big\|\sqrt{\frac{2}{\pi}}\widehat{\boldsymbol{\Sigma}}\mathbf{w}^* - \Big(\sum_{i=1}^n \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i\Big)\Big\|_2^2\bigg]}_{II.2},$$

where the inequality is guaranteed by the submultiplicativity of ℓ_2 norm. Then we discuss $II.1$ and $II.2$ respectively. For $II.1$, we have

$$II.1 \leq \frac{\alpha}{\|\widehat{\boldsymbol{\Sigma}}\|_2} \sum_{l_1,l_2=0}^{L-1} \frac{1}{l_1+l_2+1} \leq \frac{\alpha L}{\|\widehat{\boldsymbol{\Sigma}}\|_2} \sum_{l_1=0}^{L-1} \frac{1}{l_1+1} \leq O\Big(\frac{\alpha L \log L}{n}\Big).$$

The first inequality holds by the fact that $x(1-x)^k \leq \frac{1}{k+1}$ for $x \in [0,1]$. The second inequality holds by replace $\frac{1}{l_1+l_2+1}$ with its upper bound $\frac{1}{l_1+1}$. The third inequality holds by $\sum_{l_1=0}^{L-1} \frac{1}{l_1+1} \leq \log L$ and $\|\widehat{\boldsymbol{\Sigma}}\|_2 = \Theta(n)$ demonstrated in Lemma G.4. For $II.2$, we have

$$
\begin{aligned}
II.2 &= \mathbb{E}\bigg[\bigg\|\sqrt{\frac{2}{\pi}}(\widehat{\boldsymbol{\Sigma}} - n\mathbf{I}_d)\mathbf{w}^* - \Big(\sum_{i=1}^n \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i - n\sqrt{\frac{2}{\pi}}\mathbf{w}^*\Big)\bigg\|_2^2\bigg] \\
&\leq \frac{4}{\pi}\mathbb{E}\|\widehat{\boldsymbol{\Sigma}} - n\mathbf{I}_d\|_2^2 + 2\mathbb{E}\bigg[\bigg\|\sum_{i=1}^n \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i - n\sqrt{\frac{2}{\pi}}\mathbf{w}^*\bigg\|_2^2\bigg] \leq \widetilde{O}(nd).
\end{aligned}
$$

The first equality adds and minuses the same term. The first inequality holds by the submultiplicativity of ℓ_2 norm, and the fact $(a+b)^2 \leq 2a^2+2b^2$. The second inequality holds as $\|\widehat{\boldsymbol{\Sigma}} - n\mathbf{I}_d\|_2 \leq \widetilde{O}(\sqrt{nd})$, proved in Lemma G.4 and $\big\|\sum_{i=1}^n \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle)\mathbf{x}_i - n\sqrt{\frac{2}{\pi}}\mathbf{w}^*\big\|_2 \leq \widetilde{O}(\sqrt{nd})$, proved in Lemma G.5. Combining all the preceding results, we can obtain that

$$
\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) \leq O\big((1-\alpha n)^{2L}\big) \cdot \mathbb{E}\bigg[\bigg(c_0\boldsymbol{\beta}^* - \sqrt{\frac{2}{\pi}}\mathbf{w}^*\bigg)\bigg\|_2^2\bigg] + \widetilde{O}(\alpha d L) + C.
$$

It is straightforward that when taking $c_0 = \sqrt{\frac{2}{\pi}}$, the expectation term will achieve its minimum, which is the variance of $\mathbf{w}^*$ multiplying by a factor $\sqrt{\frac{2}{\pi}}$. This finishes the proof that the optimal initialization takes the value as $\mathbf{w}_{\mathrm{gd}}^{(0)} = \sqrt{\frac{2}{\pi}}\boldsymbol{\beta}^*$. We re-plug this result into the upper-bound above and utilize the fact that the variance is at the constant order. Then to find the optimal learning rate α is actually to optimize the summation of $(1-\alpha n)^{2L}$ and $\alpha d L$. We can note that the first term will decrease as α increases, while the second term will increase as α increases. Therefore, minimizing the summation of these two terms is essentially equivalent to finding an optimal α such that both terms are of the same order. Then we can notice that when consider $\alpha = \frac{\log(n/d)}{2nL}$, the first term can be bounded as

$$
(1-\alpha n)^{2L} = \bigg(1 - \frac{\log(n/d)}{2L}\bigg)^{2L} \leq \frac{d}{n}.
$$

Additionally, it is straightforward that $\alpha d L = \frac{d\log(n/d)}{n}$. When omitting the factors of $\log$, we conclude that these two terms are at the same order. Therefore, the optimal choice of learning rate is $\alpha = \widetilde{\Theta}(\frac{1}{nL})$, which can optimize the excess risk as

$$
\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) - C \leq \widetilde{O}\bigg(\frac{d}{n}\bigg).
$$

This completes the proof. □

Here we provide further discussions regarding the upper bound for the population loss achieved when choosing the optimal learning rate and initialization. The constant C represents an irreducible term arising from the variance of the model. Such an irreducible term always exists when considering least-squares loss, similar to the noise variance in classic linear regression problems. Therefore, when considering the problems with least-square loss function, it is common to define $\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) - C$ as the excess risk and attempt to minimize this term. Consequently, Theorem E.2 reveals that when using the optimal parameters, the excess risk $\mathcal{R}(\mathbf{w}_{\mathrm{gd}}^{(L)}) - C$ will converge to 0 as the context length n goes to infinity.

F Proof of Lemma 3.4 and Theorem 3.5

In this section, we provide the proof for both Lemma 3.4 and Theorem 3.5. W.L.O.G, we assume that $\sigma > 0$ in the subsequent proof. This implies that $y_{\mathrm{hc}} = -1$, $y_{\mathrm{query}} = 1$ and $\mathcal{E}(\mathbf{w}) = \mathbb{P}(\langle \mathbf{w}, \mathbf{x}_{\mathrm{query}}\rangle < 0)$ for any $\mathbf{w}$. Then we first introduce a lemma providing a closed form for $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(l)}$, which is the parameter vector of the linear model trained by gradient descent with the optimal parameters derived in Theorem 3.3 and data $(\mathbf{x}_{\mathrm{hc}}, y_{\mathrm{hc}})$.

Lemma F.1. For the gradient descent iterates $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(l)}$, it holds that

$$\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(l)} = c\boldsymbol{\beta}^* + a(l) \cdot \mathbf{x}_\perp \tag{F.1}$$

for all $l \in \{0, 1, \ldots, L\}$. c is the coefficient of $\boldsymbol{\beta}^*$ of initialization $\mathbf{w}_{\mathrm{gd}}^{(0)}$ and $a(l)$ follows that

$$a(l) = -\Big(1 - \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^l\Big) \frac{1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle}{\|\mathbf{x}_\perp\|_2^2}$$

Proof of Lemma F.1. We prove this lemma by induction. It is straightforward that $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(0)} = c\boldsymbol{\beta}^*$ and $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(1)} = c\boldsymbol{\beta}^* - \alpha N \mathbf{x}_\perp$, complying with the formula (F.1). By induction, we assume (F.1) still holds for l-th iteration. Then at the $l+1$-th iteration, we have

$$\begin{aligned}
\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(l+1)} &= \big(\mathbf{I}_d - \alpha N \mathbf{x}_\perp \mathbf{x}_\perp^\top\big) \cdot \widetilde{\mathbf{w}}_{\mathrm{gd}}^{(l)} - \alpha N \mathbf{x}_\perp \\
&= \big(\mathbf{I}_d - \alpha N \mathbf{x}_\perp \mathbf{x}_\perp^\top\big) \cdot \big(c\boldsymbol{\beta}^* + a(l)\mathbf{x}_\perp\big) - \alpha N \mathbf{x}_\perp \\
&= c\boldsymbol{\beta}^* + \big(a(l)(1 - \alpha N \|\mathbf{x}_\perp\|_2^2) - \alpha N (1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle)\big) = c\boldsymbol{\beta}^* + a(l+1)\mathbf{x}_\perp.
\end{aligned}$$

Additionally, by the fact $a(l+1) = a(l)(1 - \alpha N \|\mathbf{x}_\perp\|_2^2) - \alpha N (1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle)$, we can derive that

$$\begin{aligned}
\bigg(a(l+1) + \frac{1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle}{\|\mathbf{x}_\perp\|_2^2}\bigg) &= \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big) \bigg(a(l) + \frac{1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle}{\|\mathbf{x}_\perp\|_2^2}\bigg) \\
&= \cdots \\
&= -\big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^l \frac{1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle}{\|\mathbf{x}_\perp\|_2^2}.
\end{aligned}$$

This implies that

$$a(l) = -\Big(1 - \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^l\Big) \frac{1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle}{\|\mathbf{x}_\perp\|_2^2},$$

which completes the proof. □

Based on the closed form of $\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}$ obtained by Lemma F.1, we are ready to prove Lemma 3.4 and Theorem 3.5.

Proof of Lemma 3.4. By Lemma F.1, the output of the linear model trained via gradient descent on $\mathbf{x}_{\mathrm{query}}$ can be expanded as

$$\begin{aligned}
\langle \widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}, \mathbf{x}_{\mathrm{query}} \rangle &= \langle c\boldsymbol{\beta}^* + a(L)\mathbf{x}_\perp, \mathbf{x}_\perp + \sigma \mathbf{w}^* \rangle \\
&= c\langle \boldsymbol{\beta}^*, \mathbf{x}_\perp \rangle + a(L)\|\mathbf{x}_\perp\|_2^2 + c\sigma \langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle \\
&= c\langle \boldsymbol{\beta}^*, \mathbf{x}_\perp \rangle - \Big(1 - \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^L\Big)\big(1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle\big) + c\sigma \langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle \\
&= c\sigma \langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle - 1 + \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^L \big(1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle\big).
\end{aligned} \tag{F.2}$$

By utilizing the independence among $\mathbf{w}^*$, σ, and $\mathbf{x}_\perp$ and law of total expectation, we can derive that

$$\begin{aligned}
\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) &= \mathbb{P}\Big(c\sigma \langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle - 1 + \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^L \big(1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle\big) \le 0\Big) \\
&= \mathbb{E}\Big[\mathbb{P}\Big(c\sigma \langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle - 1 + \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^L \big(1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle\big) \le 0 \Big| \mathbf{w}^*, \mathbf{x}_\perp\Big)\Big] \\
&= \mathbb{E}\bigg[F_\sigma\bigg(\frac{1 - \big(1 - \alpha N \|\mathbf{x}_\perp\|_2^2\big)^L \big(1 + c\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle\big)}{c\langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle}\bigg)\bigg],
\end{aligned} \tag{F.3}$$

where $F_\sigma(\cdot)$ is the cumulative distribution function of σ. Similarly, we also have

$$\mathcal{E}(\mathbf{w}_{\mathrm{gd}}^{(0)}) = \mathbb{E}\bigg[F_\sigma\Big(-\frac{\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle}{\langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle}\Big)\bigg].$$

Therefore, by Taylor's first order expansion, we have

$$\begin{aligned}\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) - \mathcal{E}(\mathbf{w}_{\mathrm{gd}}^{(0)}) &= \mathbb{E}\bigg[F_\sigma'(\boldsymbol{\xi})\frac{\big(1-\big(1-\alpha N\|\mathbf{x}_\perp\|_2^2\big)^L\big)\big(1+c\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle\big)}{1+c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg] \\ &\leq \bigg(1-\bigg(1-\Theta\Big(\frac{Nd}{nL}\Big)\bigg)^L\bigg)\widetilde{O}(\sqrt{d}) \leq \widetilde{O}\Big(\frac{Nd^{3/2}}{n}\Big) \leq \widetilde{o}(1),\end{aligned}$$

where the first inequality utilizing the concentration results that $\|\mathbf{x}_\perp\|_2^2\Theta(d)$ and $\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle = \widetilde{O}(\sqrt{d})$. The second inequality holds by the fact $\Big(1-\Theta\big(\frac{Nd}{nL}\big)\Big)^L = 1-\Theta\big(\frac{Nd}{n}\big)$ by our condition $n \leq o(d^{3/2}/n)$, which also implies the last inequality holds. Therefore, we finish the proof. □

In the next, we prove Theorem 3.5

Proof of Theorem 3.5. Similar to the proof of Lemma 3.4, we have that

$$\begin{aligned}\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) &= \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\alpha N\|\mathbf{x}_\perp\|_2^2\big)^L\big(1+c\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle\big)}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg)\bigg] \\ &= \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\alpha N\|\mathbf{x}_\perp\|_2^2\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg) - F_\sigma'(\boldsymbol{\xi})\frac{\big(1-\alpha N\|\mathbf{x}_\perp\|_2^2\big)^L|\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle|\operatorname{sign}(\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle)}{\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg] \\ &= \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\alpha N\|\mathbf{x}_\perp\|_2^2\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg)\bigg] = \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg)\bigg],\end{aligned}$$

where the third inequality holds as $\operatorname{sign}(\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle)$ is independent with $|\langle\mathbf{x}_\perp,\boldsymbol{\beta}^*\rangle|$ and $\|\mathbf{x}_\perp\|_2^2$, and $F'(\boldsymbol{\xi})$ is a constant. Additionally, let σ follows the uniform distribution from a to b, then we can expand the expectation above as

$$\begin{aligned}\mathcal{E}(\widetilde{\mathbf{w}}_{\mathrm{gd}}^{(L)}) =& \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg)\mathbb{1}\bigg\{\frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle} \leq a\bigg\}\bigg] \\ &+ \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg)\mathbb{1}\bigg\{a < \frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle} \leq b\bigg\}\bigg] \\ &+ \mathbb{E}\bigg[F_\sigma\bigg(\frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\bigg)\mathbb{1}\bigg\{\frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle} > b\bigg\}\bigg] \\ \leq& \bigg(1-\bigg(1-\widetilde{\Theta}\Big(\frac{Nd}{nL}\Big)\bigg)^L\bigg)\mathbb{E}\bigg[\frac{1}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle}\mathbb{1}\bigg\{a < \frac{1-\big(1-\widetilde{\Theta}(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle} \leq b\bigg\}\bigg] \\ &+ \mathbb{E}\bigg[\mathbb{1}\bigg\{\frac{1-\big(1-\Theta(\frac{Nd}{nL})\big)^L}{c\langle\mathbf{w}^*,\boldsymbol{\beta}^*\rangle} > b\bigg\}\bigg] \\ \leq& c_1 - c_2\bigg(1-\widetilde{\Theta}\Big(\frac{Nd}{nL}\Big)\bigg)^L\end{aligned}$$

where c_1, c_2 are two positive scalars solely depending on a, b and the distribution of $\mathbf{w}^*$. This completes the proof. □

G Technical lemmas

In this section, we introduce and prove some technical lemmas utilized in the previous proof.

Lemma G.1. Let $\mathbf{x} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$, and $\mathbf{w}_1, \mathbf{w} \in \mathbb{R}^d$ be two vectors independent of $\mathbf{x}$, with $\|\mathbf{w}_1\|_2 = 1$, then it holds that

$$\mathbb{E}_{\mathbf{x}}[\langle\mathbf{w},\mathbf{x}\rangle\operatorname{sign}(\langle\mathbf{w}_1,\mathbf{x}\rangle)] = \sqrt{\frac{2}{\pi}}\langle\mathbf{w},\mathbf{w}_1\rangle.$$

Proof of Lemma G.1. Since $\|\mathbf{w}_1\|_2 = 1$, let $\mathbf{\Gamma} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d] \in \mathbb{R}^d$ be the orthogonal matrix with $\mathbf{w}_1$ being its first column. Then we have

$$\begin{aligned}\mathbb{E}_{\mathbf{x}}[\langle \mathbf{w}, \mathbf{x}\rangle \operatorname{sign}(\langle \mathbf{w}_1, \mathbf{x}\rangle)] &= \mathbb{E}_{\mathbf{x}}[\mathbf{w}^\top \mathbf{\Gamma}\mathbf{\Gamma}^\top \mathbf{x} \operatorname{sign}(\langle \mathbf{w}_1, \mathbf{x}\rangle)] \\ &= \sum_{k=1}^{d} \langle \mathbf{w}, \mathbf{w}_k\rangle \mathbb{E}_{\mathbf{x}}[\langle \mathbf{w}_k, \mathbf{x}\rangle \operatorname{sign}(\langle \mathbf{w}_1, \mathbf{x}\rangle)] = \sqrt{\frac{2}{\pi}}\langle \mathbf{w}, \mathbf{w}_1\rangle,\end{aligned}$$

where the last equality holds since $\langle \mathbf{w}_k, \mathbf{x}\rangle \sim \mathcal{N}(0, 1)$ for all $k \in [d]$, $\langle \mathbf{w}_{k_1}, \mathbf{x}\rangle$ and $\langle \mathbf{w}_{k_2}, \mathbf{x}\rangle$ are independent when $k_1 \neq k_2$, and $\mathbb{E}[\langle \mathbf{w}_1, \mathbf{x}\rangle \operatorname{sign}(\langle \mathbf{w}_1, \mathbf{x}\rangle)] = \mathbb{E}[|\langle \mathbf{w}_1, \mathbf{x}\rangle|] = \sqrt{\frac{2}{\pi}}$. This completes the proof. □

For the next lemmas, we follow the notation we used in previous section that $\widehat{\mathbf{\Sigma}} = \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top$.

Lemma G.2. For any $k \in \mathbb{N}$, it holds that $\mathbb{E}[\widehat{\mathbf{\Sigma}}^k] = c\mathbf{I}_d$, where c is a scalar.

Proof of Lemma G.2. Let $\mathbf{\Gamma}$ be any orthogonal matrix, then we have $\mathbf{\Gamma}\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$. This implies that $\sum_{i=1}^n (\mathbf{\Gamma}\mathbf{x}_i)(\mathbf{\Gamma}\mathbf{x}_i)^\top$ has the same distribution with $\widehat{\mathbf{\Sigma}}$. Therefore, we can derive that

$$\mathbf{\Gamma}\mathbb{E}[\widehat{\mathbf{\Sigma}}^k]\mathbf{\Gamma} = \mathbb{E}\Big[\Big(\sum_{i=1}^n (\mathbf{\Gamma}\mathbf{x}_i)(\mathbf{\Gamma}\mathbf{x}_i)^\top\Big)^k\Big] = \mathbb{E}[\widehat{\mathbf{\Sigma}}^k]$$

holds for any orthogonal matrix $\mathbf{\Gamma}$, which implies that $\mathbb{E}[\widehat{\mathbf{\Sigma}}^k]$ must be at the form $c\mathbf{I}_d$. This completes the proof. □

Lemma G.2 implies that $\mathbb{E}[(\mathbf{I}_d - \widehat{\mathbf{\Sigma}})^k] = c\mathbf{I}_d$ for some scalar c as by binomial formula it can be expanded as a summation of polynomials of $\widehat{\mathbf{\Sigma}}$, which all have the expectations with the form $c\mathbf{I}_d$.

Lemma G.3. For any $k \in \mathbb{N}$, it holds that

$$\mathbb{E}\Big[\widehat{\mathbf{\Sigma}}^k\Big(\sum_{i=1}^n \mathbf{x}_i y_i\Big)\Big] = c\boldsymbol{\beta}^*,$$

where c is some scalar.

Proof of Lemma G.3. By binomial theorem, we have

$$\mathbb{E}\Big[\widehat{\mathbf{\Sigma}}^k\Big(\sum_{i=1}^n \mathbf{x}_i y_i\Big)\Big] = \sum_{i=1}^n \sum_{k_1=0}^{k} \binom{k}{k_1} \mathbb{E}\Big[\Big(\sum_{i'\neq i} \mathbf{x}_{i'}\mathbf{x}_{i'}^\top\Big)^{k-k_1}\Big] \mathbb{E}[(\mathbf{x}_i\mathbf{x}_i^\top)^{k_1}\mathbf{x}_i y_i].$$

By Lemma G.2, we already obtain that $\mathbb{E}\Big[\Big(\sum_{i'\neq i} \mathbf{x}_{i'}\mathbf{x}_{i'}^\top\Big)^{k-k_1}\Big] = c\mathbf{I}_d$ for some scalar c. In the next, it suffices to show that $\mathbb{E}[(\mathbf{x}_i\mathbf{x}_i^\top)^{k_1}\mathbf{x}_i y_i] = c\boldsymbol{\beta}^*$ for some scalar c. Since $\|\mathbf{w}^*\|_2 = 1$, let $\mathbf{\Gamma} = [\mathbf{w}^*, \mathbf{w}_2, \ldots, \mathbf{w}_d] \in \mathbb{R}^d$ be the orthogonal matrix with $\mathbf{w}^*$ being its first column, and let $\mathbf{x}_i' = \mathbf{\Gamma}^\top\mathbf{x}_i \sim \mathcal{N}(0, \mathbf{I}_d)$. This implies that $y_i = \operatorname{sign}(\langle \mathbf{w}^*, \mathbf{x}_i\rangle) = \operatorname{sign}(\mathbf{x}_{i,1}')$, which is the first coordinate of $\mathbf{x}_i'$. Based on this, for any fixed $\mathbf{w}^*$, we can further derive that

$$\mathbb{E}[(\mathbf{x}_i\mathbf{x}_i^\top)^{k_1}\mathbf{x}_i y_i|\mathbf{w}^*] = \mathbf{\Gamma}\mathbb{E}[(\mathbf{x}_i'\mathbf{x}_i'^\top)^{k_1}\mathbf{x}_i' \operatorname{sign}(\mathbf{x}_{i,1}')] = \mathbf{\Gamma}\mathbb{E}[\|\mathbf{x}_i'\|_2^{2k_1}\mathbf{x}_i' \operatorname{sign}(\mathbf{x}_{i,1}')] = c\mathbf{w}^*.$$

The last equality holds as $\|\mathbf{x}_i'\|_2^{2k_1}$ is a even function for each coordinate of $\mathbf{x}_i'$, which implies that $\mathbb{E}[\|\mathbf{x}_i'\|_2^{2k_1}\mathbf{x}_{i,j}' \operatorname{sign}(\mathbf{x}_{i,1}')] = 0$ for any $j \in [d]$ and $j \neq 1$. Therefore, we can finally obtain that

$$\mathbb{E}[(\mathbf{x}_i\mathbf{x}_i^\top)^{k_1}\mathbf{x}_i y_i] = \mathbb{E}\big[\mathbb{E}[(\mathbf{x}_i\mathbf{x}_i^\top)^{k_1}\mathbf{x}_i y_i|\mathbf{w}^*]\big] = c\mathbb{E}[\mathbf{w}^*] = c\boldsymbol{\beta}^*,$$

which completes the proof. □

Lemma G.4 (Theorem 9 in [11])**.** For any $\delta > 0$, with probability at least $1 - \delta$, it holds that,

$$\Big\|\frac{1}{n}\widehat{\mathbf{\Sigma}} - \mathbf{I}_d\Big\|_2 \leq O\bigg(\max\bigg\{\frac{d}{n}, \sqrt{\frac{d}{n}}, \frac{\log(1/\delta)}{n}, \sqrt{\frac{\log(1/\delta)}{n}}\bigg\}\bigg).$$

Lemma G.5. For any $\delta > 0$, with probability at least $1 - \delta$, it holds that,

$$\Big\| \sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \mathbf{x}_i - n\sqrt{\frac{2}{\pi}} \mathbf{w}^* \Big\|_2 \leq O\big(\sqrt{nd \log(d/\delta)}\big).$$

Proof of Lemma G.5. Similar to the previous proof technique, let $\mathbf{\Gamma} = [\mathbf{w}^*, \mathbf{w}_2, \ldots, \mathbf{w}_d] \in \mathbb{R}^d$ be the orthogonal matrix with $\mathbf{w}^*$ being its first column, and let $\mathbf{x}_i' = \mathbf{\Gamma}^\top \mathbf{x}_i \sim \mathcal{N}(0, \mathbf{I}_d)$. Then we can derive that

$$\sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \mathbf{x}_i - n\sqrt{\frac{2}{\pi}} \mathbf{w}^* = \bigg[\sum_{i=1}^{n} \Big(|\mathbf{x}_{i,1}'| - \sqrt{\frac{2}{\pi}} \Big) \bigg] \cdot \mathbf{w}^* + \sum_{j=2}^{d} \bigg[\sum_{i=1}^{n} \mathrm{sign}(\mathbf{x}_{i,1}') \mathbf{x}_{i,j}' \bigg] \cdot \mathbf{w}_j.$$

Since $|\mathbf{x}_{i,1}'|$ is a subgaussian random variable with expectation $\sqrt{\frac{2}{\pi}}$, by Hoeffding's inequality we can derive that with probability at least $1 - \delta/d$,

$$\sum_{i=1}^{n} \Big(|\mathbf{x}_{i,1}'| - \sqrt{\frac{2}{\pi}} \Big) \leq O\big(\sqrt{n \log(d/\delta)}\big).$$

Additionally, when $j \neq 1$, $\mathrm{sign}(\mathbf{x}_{i,1}') \mathbf{x}_{i,j}'$ still follows a standard normal distribution (A standard normal random variable times an independent Rademacher random variable is still a standard normal random). Therefore, we can also derive that

$$\sum_{i=1}^{n} \mathrm{sign}(\mathbf{x}_{i,1}') \mathbf{x}_{i,j}' \leq O\big(\sqrt{n \log(d/\delta)}\big)$$

holds with probability at least $1 - \frac{\delta}{d}$. Then by taking an union bound, we can finally obtain that

$$\begin{aligned} \Big\| \sum_{i=1}^{n} \mathrm{sign}(\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \mathbf{x}_i - n\sqrt{\frac{2}{\pi}} \mathbf{w}^* \Big\|_2^2 &= \bigg[\sum_{i=1}^{n} \Big(|\mathbf{x}_{i,1}'| - \sqrt{\frac{2}{\pi}} \Big) \bigg]^2 + \sum_{j=2}^{d} \bigg[\sum_{i=1}^{n} \mathrm{sign}(\mathbf{x}_{i,1}') \mathbf{x}_{i,j}' \bigg]^2 \\ &\leq O\big(nd \log(d/\delta)\big). \end{aligned}$$

The first equality holds by the orthogonality among $\mathbf{w}^*, \mathbf{w}_2, \ldots, \mathbf{w}^*$. This completes the proof. □

Lemma G.6. For any $\delta > 0$, with probability at least $1 - \delta$, it holds that,

$$\begin{aligned} &\big| \|\mathbf{x}_\perp\|_2^2 - (d-1) \big| \leq O\big(\sqrt{d \log(1/\delta)}\big); \\ &\big| \langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle \big| \leq O\big(\sqrt{d \log(1/\delta)}\big). \end{aligned}$$

Proof of Lemma G.6. By the fact that $\|\mathbf{x}_\perp\|_2^2 \sim \chi_{d-1}^2$, we have $\mathbb{E}[\|\mathbf{x}_\perp\|_2^2] = d - 1$. Then by the Bernstein's inequality, we can obtain that

$$\big| \|\mathbf{x}_\perp\|_2^2 - (d-1) \big| \leq O\big(\sqrt{d \log(1/\delta)}\big)$$

holds with probability at least $1 - \delta/2$. Besides, since $\langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle \sim \mathcal{N}(0, 1 - \langle \mathbf{w}^*, \boldsymbol{\beta}^* \rangle)$, by applying the tail bounds of Gaussian distribution, we can obtain that

$$\big| \langle \mathbf{x}_\perp, \boldsymbol{\beta}^* \rangle \big| \leq O\big(\sqrt{d \log(1/\delta)}\big)$$

holds with probability at least $1 - \delta/2$. By applying a union bound, we obtain the final result. □

H Experimental setup

H.1 Context hijacking in GPT-2

This section will describe our experimental setup for context hijacking on LLMs of different depths. We first construct four datasets for different tasks, including language, country, sports, and city. The samples in each dataset consist of four parts: prepend,

error result, query, and correct result. Each task has a fixed template for the sample. For the language, the template is "`{People} did not speak {error result}. The native language of {People} is {correct result}`". For the country, the template is "`{People} does not live in {error result}. {People} has a citizenship from {correct result}`". For the sports, the template is "`{People} is not good at playing {error result}. {People}'s best sport is {correct result}`". For the city, the template is "`{Landmarks} is not in {error result}. {Landmarks} is in the city of {correct result}`". We allow samples to have certain deviations from the templates, but they must generally conform to the semantics of the templates. Instance always match the reality, and the main source of instances is the *CounterFact* dataset [48]. In our dataset, each task contains three hundred to seven hundred specific instances. We conduct experiments on GPT2 [60] of different sizes. Specifically, we consider GPT2, GPT2-MEDIUM, GPT2-LARGE, and GPT2-XL. They have 12 layers, 24 layers, 36 layers, and 48 layers, respectively. We construct a pipeline that test each model on each task, recording the number of prepends for which the context just succeeded in perturbing the output. For those samples that fail to perturb within a certain number of prepends (which is determined by the maximum length of the pre-trained model), we exclude them from the statistics. Finally, we verify the relationship between model depth and robustness by averaging the number of prepends required to successfully perturb the output.

H.2 Numerical experiments

We use extensive numerical experiments to verify our theoretical results, including gradient descent and linear transformers.

Gradient descent: We use a single-layer neural network as the gradient descent model, which contains only one linear hidden layer. Its input dimension is the dimension d of feature $\mathbf{x}$, and we mainly experiment on $d = \{15, 20, 25\}$. Its output dimension is 1, because we only need to judge the classification result by its sign. We use the mean square error as the loss function and *SGD* as the optimizer. All data comes from the defined training distribution $\mathcal{D}_{\text{tr}}$. The hyperparameters we set include training context length $N = 50$, mean of the Gaussian distribution $\boldsymbol{\beta}^\star = 1$, variance of the Gaussian distribution $\boldsymbol{\Sigma} = 0.1$ (then normalized). We initialize the neural network to $c\boldsymbol{\beta}$, and then perform gradient descents with steps $Steps = \{1, 2, ..., 8\}$ and learning rate lr. We use grid search to search for the optimal c and lr for the loss function. This is equivalent to the trained transformers of layers 1 to 8 learning to obtain the shared signal $c\boldsymbol{\beta}$ and the optimal learning rate lr for the corresponding number of layers. Then they can use in-context data to fine-tune $c\boldsymbol{\beta}$ to a specific $\mathbf{w}^\star$.

After obtaining the optimal initialization and learning rate, we test it on the dataset from $\mathcal{D}_{\text{te}}$. Again, we set exactly the same hyperparameters as above. In addition, we set the σ in the test distribution to 0.1.

Linear Transformer: We train on multi-layer single-head linear transformers and use *Adam* as the optimizer. The training settings for models with different numbers of layers are exactly the same. We use the initial learning rate $lr \in \{0.0001, 0.0002\}$, and the training steps are 600,000. We use a learning rate decay mechanism, where the learning rate is decayed by half every 50,000 training steps. For training and testing data, we set the data dimension $d = 20$ and the training context length $N = \{10, 20, 30, 40\}$. We use a batchsize of 5,000 and apply gradient clipping with a threshold of 1. Each experiment takes about four hours on average on a single NVIDIA GeForce RTX 4090 GPU.

H.3 Context hijacking in real-world LLMs

Similar to Appendix H.1, we construct four different topics of datasets, including city, country, sport, and language. However, this dataset is more diverse, which helps simulate more realistic situations. Below is a detailed description of the dataset construction process, using an example from the dataset.

First, we will design a fact retrieval problem. It is a direct question, such as "Of all the sports, Maria Sharapova is most professional in which one? The answer is". We want the model to predict the next token is "tennis".

Next, we will choose a topic that is factually correct. For the example above, we can choose the topic that "Maria Sharapova is not a professional in rugby".

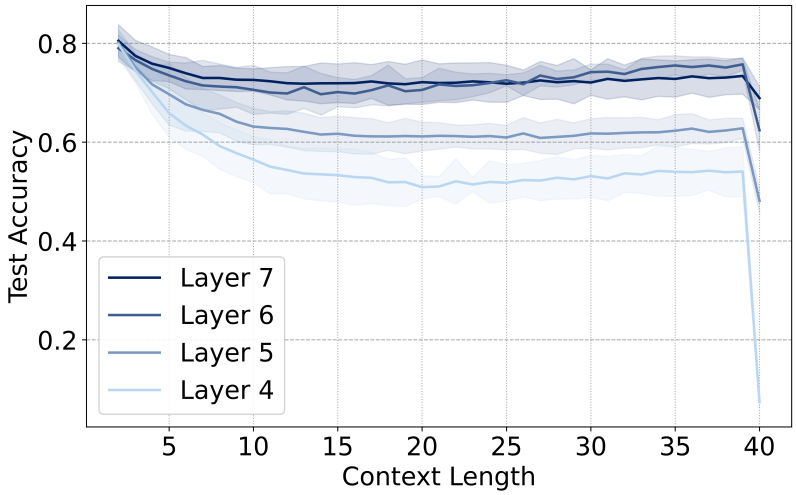

Figure 6: Standard transformers experiments with different depths. Testing the trained standard transformers (GPT-2 architecture [60]) on the test set, as the number of interference samples increases, the model classification accuracy decreases and gradually converges. The results also show that deeper models are more robust.

Finally, we will add factually correct context prefixes of varying lengths before the question. Each sentence of this context prefix will describe the topic that has been determined from a different perspective and with different words. That is, paraphrase the hijacking context instead of repeating them. In our example, these sentences could be "Rugby is not a sport that Maria Sharapova is adept at playing", "Maria Sharapova's tennis skills do not translate well to rugby", "The physical demands of rugby are not ones with which Maria Sharapova is familiar", etc. The model is then asked the same question. If the model predicts "tennis", then it is correct. If the model predicts "rugby", we call this "label flipping".

The number of samples in each dataset ranges from hundreds to thousands. We filter out questions that are too difficult based on the model's own capabilities and the difficulty of the questions, which means that the model could always correctly answer direct questions without hijacking context. We conduct experiments on Qwen2.5 base models of different sizes (depths) and corresponding instruction fine-tuned versions.

I Additional experiments

I.1 Robustness of standard transformers with different number of layers

To generalize the results to more realistic settings, we transfer the experiments from linear transformers to larger and standard transformers, such as GPT-2 [60]. We train and test GPT-2 with different numbers of layers based on exactly the same settings as the linear transformers experiments. The results once again verify our theory (Figure 6). As the context length increases, the model's accuracy decreases, but increasing the number of layers of the model significantly improves the robustness, indicating that our theory has more practical significance. Then we describe the setup of standard transformers experiments briefly.

Setup: We use the standard transformers of the GPT-2 architecture for the experiments, and the main settings are similar to [30]. We set the embedding size to 256, the number of heads to 8, and the batch size to 64. We use a learning rate decay mechanism similar to linear transformers experiments, with an initial learning rate of 0.0002, and then reduced by half every 200,000 steps, for a total of 600,000 steps. We use *Adam* as the optimizer.

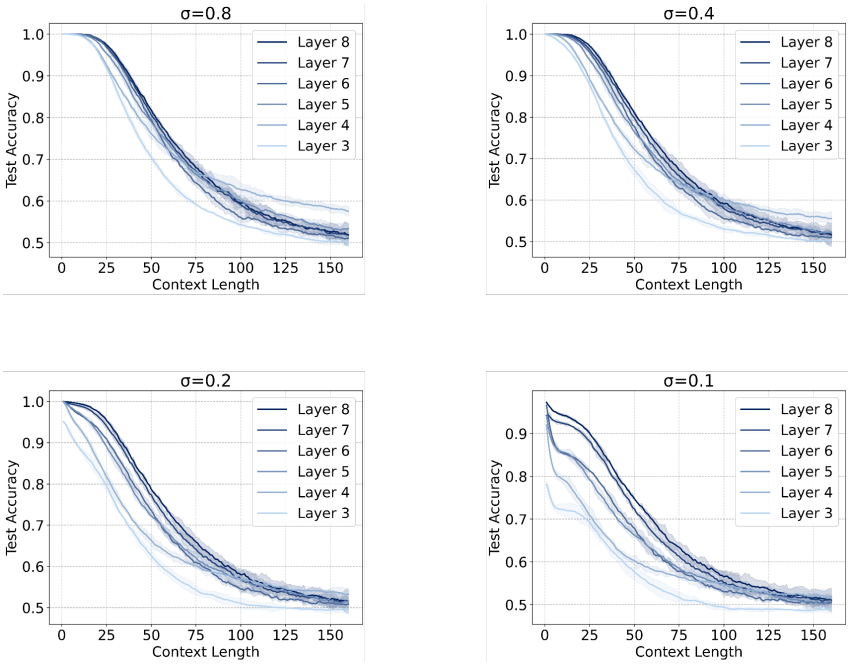

Figure 7: Linear transformers experiments with different depths and different σ. In real-world semantics, smaller σ means stronger interference. Comparing the test performance of the model under different σ, we can find that as σ decreases, the robustness of the model decreases significantly, which verifies the rationality of our modeling.

I.2 Linear transformers facing different interference intensity

In this section, we mainly discuss how the robustness of the model changes with the interference intensity. In our modeling, the interference intensity is determined only by the distance between the query sample and the similar interference samples defined in the test set, that is, by the variable σ in $\mathcal{D}_{\text{te}}$. In real-world observations, according to the idea of the induction head [51], the more similar the context prepend used for interference is to the query, the more likely the model is to use in-context learning to output incorrect results. Therefore, we examine different σ to determine whether the model conforms to the actual real-world interference situation, that is, to verify the rationality of our modeling.

Observing the experiment results in Figure 7, when σ gradually decreases from 0.8 to 0.1, that is, the interference intensity of the data gradually increases, the classification accuracy of the model decreases significantly. When σ is larger and the interference context is less, the model can always classify accurately, indicating that weak interference does not affect the performance of the model, which is consistent with real observations. Various experimental phenomena show that our modeling of the context hijacking task by the distance between the interference sample and the query sample is consistent with the real semantics.

I.3 Robustness of nonlinear transformers for nonlinear classification

To extend our theoretical results to more complex situations, we consider nonlinear models and nonlinear tasks. We conduct preliminary experiments on nonlinear classification (Figure 8). More specifically, we change $\langle \mathbf{w}, \mathbf{x} \rangle$ to $\langle \mathbf{w}, \mathbf{x} \rangle^2 - C$, where C is a constant. We conduct the experiments on the multi-layer *ReLU* attention transformers. The results show that even in the nonlinear case, the model still tends to be more robust as it gets deeper, which is consistent with our theoretical results. This strengthens our conclusions and shows that our theory is not limited to the linear case, but is also valid on more complex and practical nonlinear tasks.

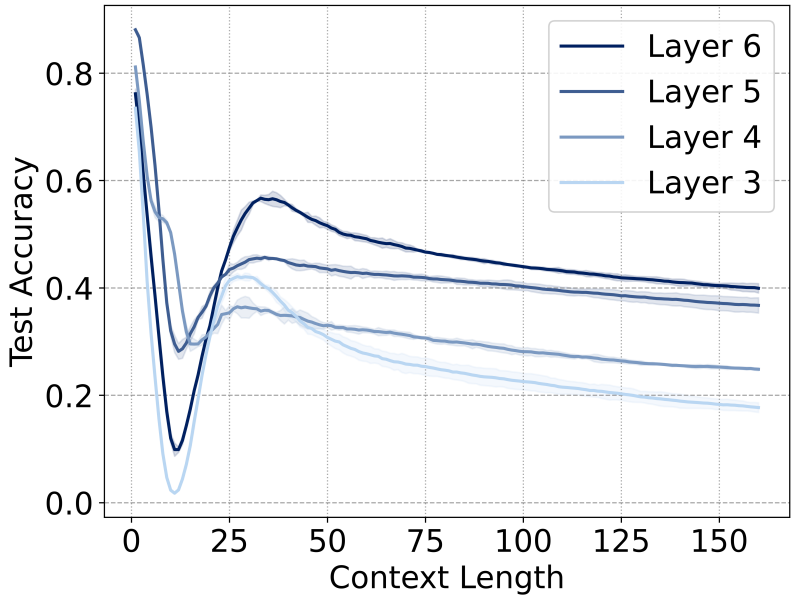

Figure 8: We introduce a nonlinear problem by changing $\langle \mathbf{w}, \mathbf{x} \rangle$ to $\langle \mathbf{w}, \mathbf{x} \rangle^2 - C$, where C is a constant. We conduct the experiments on the multi-layer *ReLU* attention transformers. The results show that the model still tends to be more robust as it gets deeper, which is consistent with our theoretical results.