ADVERSARIALLY PRETRAINED TRANSFORMERS MAY BE UNIVERSALLY ROBUST IN-CONTEXT LEARNERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adversarial training is one of the most effective adversarial defenses, but it incurs a high computational cost. In this study, we present the first theoretical analysis suggesting that adversarially pretrained transformers can serve as *universally robust foundation models*—models that can robustly adapt to diverse downstream tasks with only lightweight tuning. Specifically, we demonstrate that single-layer linear transformers, after adversarial pretraining across a variety of classification tasks, can robustly generalize to unseen classification tasks through in-context learning from clean demonstrations (i.e., without requiring additional adversarial training or examples). This universal robustness stems from the model's ability to adaptively focus on robust features within given tasks. We also show the two open challenges for attaining robustness: accuracy–robustness trade-off and sample-hungry training. This study initiates the discussion on the utility of universally robust foundation models. While their training is expensive, the investment would prove worthwhile as downstream tasks can enjoy free adversarial robustness.

1 INTRODUCTION

Adversarial examples—subtle and often imperceptible perturbations to inputs that lead machine learning models to make incorrect predictions—reveal a fundamental vulnerability in modern deep learning systems (Szegedy et al., 2014). Adversarial training is one of the most effective defenses against such attacks (Goodfellow et al., 2015; Madry et al., 2018), where classification loss is minimized over worst-case (i.e., adversarial) perturbations. This min–max optimization significantly increases the computational cost compared to standard training. Despite extensive efforts to develop alternative defenses, most of them have subsequently been shown to offer only spurious robustness (Athalye et al., 2018; Croce & Hein, 2020; Tramer et al., 2020). Consequently, adversarial training remains the de facto standard, and practitioners must incur this cost to obtain adversarially robust models.

Recently, it has become common to leverage foundation models for target tasks. Thanks to large-scale pretraining, these models can be adapted to diverse downstream tasks with only lightweight tuning. This naturally raises the question: *Can adversarially trained foundation models enable efficient and robust adaptation to a wide range of downstream tasks?* Although training such models is expensive, the investment would be worthwhile if numerous downstream tasks can inherit adversarial robustness for free, without requiring costly adversarial training themselves. While this is a promising research direction, the utility of such *universally robust foundation models* remains largely unknown, as their training is computationally and financially prohibitive to empirically evaluate across multiple runs.

In this study, we present the first theoretical analysis suggesting that adversarially pretrained transformers can serve as universally robust foundation models. Specifically, we show that single-layer linear transformers, after adversarial pretraining across a variety of classification tasks, can robustly generalize to previously unseen classification tasks through in-context learning (Brown et al., 2020) from clean demonstrations. Namely, these transformers can adapt robustly without requiring any adversarial examples or additional training. In-context learning is a recently uncovered capability of transformers that allows them to efficiently adapt to new tasks from a few input–output demonstrations in the prompt, without any parameter updates.

Our analysis builds upon the conceptual framework of robust features (class-discriminative and human-interpretable) and non-robust features (human-imperceptible yet predictive) (Ilyas et al., 2019; Tsipras et al., 2019). Based on this framework, we show that adversarially pretrained single-layer linear transformers can adaptively focus on robust features within given downstream tasks, rather than non-robust or non-predictive features, thereby achieving universal robustness. This framework also reveals that the universal robustness holds under mild conditions, except in an unrealistic scenario where non-robust features overwhelmingly outnumber robust ones.

We also show that two open challenges in robust machine learning (Schmidt et al., 2018; Tsipras et al., 2019) still remain in our scenario. First, when adversarially pretrained, single-layer linear transformers exhibit lower clean accuracy than their standard counterparts. Second, to achieve clean accuracy comparable to standard models, these transformers require more in-context demonstrations.

Our contributions are summarized as follows:

- We provide the first theoretical support for universally robust foundation models: under mild conditions, adversarially pretrained transformers with a single linear self-attention layer can robustly adapt to unseen classification tasks through in-context learning.
- Based on the framework of robust and non-robust features, we derive the condition for successful robust adaptation. Moreover, we show that the universal robustness arises from the model's adaptive focus on robust features within given (unseen) tasks.
- As open problems for these transformers, we identify the accuracy–robust trade-off and sample-hungry in-context learning.

This study explores the potential of universally robust foundation models, which can endow diverse downstream tasks with adversarial robustness without adversarial training. A key challenge is the cost of adversarial pretraining. We assume that, as with standard foundation models, such efforts would be undertaken by large companies, which could offset development costs through licensing or API fees. The growing demand for safe and reliable AI strengthens this incentive. Encouragingly, advances in acceleration techniques for adversarial training, such as fast adversarial training (Wong et al., 2020) and adversarial finetuning (Jeddi et al., 2020), suggest that the cost of adversarial training could approach that of standard training. We regard our theoretical analysis as an important first step toward fostering the practical development of universally robust foundation models.

2 Related Work

Additional related work can be found in Appendix A.

Adversarial Training. Adversarial training (Goodfellow et al., 2015; Madry et al., 2018), which augments training data with adversarial examples (Szegedy et al., 2014), is one of the most effective adversarial defenses. Its major limitation is the high computational cost. To address this, several methods have focused on the efficient generation of adversarial examples (Andriushchenko & Flammarion, 2020; Kim et al., 2021; Park & Lee, 2021; Shafahi et al., 2019; Wong et al., 2020; Zhang et al., 2019a) and adversarial finetuning (Jeddi et al., 2020; Mao et al., 2023; Suzuki et al., 2023; Wang et al., 2024a). However, these methods still require task-specific adversarial training. In this study, we introduce the concept of universally robust foundation models, which can adapt to a wide range of downstream tasks without requiring any adversarial training or examples.

Robust and Non-Robust Features. It is often suggested that adversarial vulnerability arises from the reliance of models on non-robust features (Ilyas et al., 2019; Tsipras et al., 2019). While robust features are class-discriminative, human-interpretable, and semantically meaningful, non-robust features are subtle, often imperceptible to humans, yet statistically correlated with labels and therefore predictive. Humans can rely only on robust features, whereas models can leverage both features to maximize accuracy. Tsipras et al. (2019) showed that standard classifiers depend heavily on non-robust features, making them vulnerable to adversarial perturbations that can manipulate these subtle features. They also showed that adversarial training forces models to rely solely on robust features, thereby enhancing robustness, but often reduces clean accuracy, known as the accuracy–robustness trade-off (Dobriban et al., 2023; Mehrabi et al., 2021; Raghunathan et al., 2019; 2020; Su et al., 2018; Tsipras et al., 2019; Yang et al., 2020; Zhang et al., 2019b). Subsequent studies have confirmed that adversarially trained neural networks place greater emphasis on robust features (Augustin et al., 2020; Chalasani et al., 2020; Engstrom et al., 2019; Etmann et al., 2019; Kaur et al., 2019; Santurkar et al., 2019; Srinivas et al., 2023; Tsipras et al., 2019; Zhang & Zhu, 2019). In this study, building on this perspective, we employ datasets consisting of robust and non-robust features. Based on this framework, we find that adversarially pretrained single-layer linear transformers prioritize robust features rather than non-robust features, and exhibit the accuracy–robustness trade-off.

3 Theoretical Results

Notation. For $n \in \mathbb{N}$, let $[n] := \{1, \dots, n\}$. Denote the i-th element of a vector $\boldsymbol{a}$ by a_i, and the element in the i-th row and j-th column of a matrix $\boldsymbol{A}$ by $A_{i,j}$. Let $U(\mathcal{S})$ be the uniform distribution over a set $\mathcal{S} \subset \mathbb{R}$. The sign function is denoted as $\operatorname{sgn}(\cdot)$. For $d_1, d_2 \in \mathbb{N}$, let $\mathbf{1}_{d_1}$ and $\mathbf{1}_{d_1,d_2}$ be the d_1-dimensional all-ones vector and $d_1 \times d_2$ all-ones matrix, respectively. The $d_1 \times d_1$ identity matrix is denoted as $\boldsymbol{I}_{d_1}$. Similarly, we write the all-zeros vector and matrix as $\mathbf{0}_{d_1}$ and $\mathbf{0}_{d_1,d_2}$, respectively. We use $\gtrsim$, $\lesssim$, and $\approx$ only to hide constant factors in informal statements.

3.1 Problem Setup

Overview. We adversarially train a single-layer linear transformer on $d \in \mathbb{N}$ distinct datasets. The c-th training data distribution is denoted by $\mathcal{D}_c^{\mathrm{tr}}$ for $c \in [d]$. The c-th dataset consists of $N+1$ samples, $\{(\boldsymbol{x}_n^{(c)}, y_n^{(c)})\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_c^{\mathrm{tr}}$. The transformer is encouraged to adaptively learn data structures from N clean in-context demonstrations $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$ and generalize to the $(N+1)$-th perturbed sample $\boldsymbol{x}_{N+1} + \boldsymbol{\Delta}$, where $\boldsymbol{\Delta}$ represents an adversarial perturbation. We then evaluate the adversarial robustness of the trained transformer on a test dataset $\{(\boldsymbol{x}_n^{\mathrm{te}}, y_n^{\mathrm{te}})\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\mathrm{te}}$, which may exhibit different structures from all training distributions.

Transformer. We first define the input sequence for a transformer as

$$\boldsymbol{Z}_{\boldsymbol{\Delta}} := \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N & \boldsymbol{x}_{N+1} + \boldsymbol{\Delta} \\ y_1 & y_2 & \cdots & y_N & 0 \end{bmatrix} \in \mathbb{R}^{(d+1)\times(N+1)}, \tag{1}$$

where $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \in \mathbb{R}^d$ are training data, $y_1, \dots, y_N \in \{\pm 1\}$ are their binary labels, $\boldsymbol{x}_{N+1} \in \mathbb{R}^d$ is a test (query) sample, and $\boldsymbol{\Delta} \in \mathbb{R}^d$ is an adversarial perturbation (see later). A transformer is expected to adaptively learn data structures from N demonstrations $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$ and to predict the label of $\boldsymbol{x}_{N+1}$. The $(d+1, N+1)$-th element of $\boldsymbol{Z}_{\boldsymbol{\Delta}}$ serves as a placeholder for the prediction of $\boldsymbol{x}_{N+1} + \boldsymbol{\Delta}$. We define a single-layer linear transformer $\boldsymbol{f} : \mathbb{R}^{(d+1)\times(N+1)} \to \mathbb{R}^{(d+1)\times(N+1)}$, which is commonly employed in theoretical studies of in-context learning (Ahn et al., 2023; Cheng et al., 2024; Gatmiry et al., 2024; Mahankali et al., 2024; Zhang et al., 2024b), as follows:

$$\boldsymbol{f}(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}, \boldsymbol{Q}) := \frac{1}{N} \boldsymbol{P} \boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{Q} \boldsymbol{Z}_{\boldsymbol{\Delta}}, \qquad \boldsymbol{M} := \begin{bmatrix} \boldsymbol{I}_n & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{(N+1)\times(N+1)}, \tag{2}$$

where $\boldsymbol{P} \in \mathbb{R}^{(d+1)\times(d+1)}$ serves as the value weight matrix and $\boldsymbol{Q} \in \mathbb{R}^{(d+1)\times(d+1)}$ serves as the product of the key and query weight matrices. The mask matrix M is adopted from recent literature on in-context learning to prevent tokens from attending to the query token (Ahn et al., 2023; Cheng et al., 2024; Gatmiry et al., 2024; Li et al., 2025).

Training Data Distribution. The transformer is pretrained on d distinct datasets. Inspired by Tsipras et al. (2019), we consider the following data structure that explicitly separates robust and non-robust features (cf. Section 2) according to their dimensional indices:

Assumption 3.1 (Individual training data distribution)**.** Let $c \in [d]$ be the index of the training data distribution and $\mathcal{D}_c^{\mathrm{tr}}$ be the c-th distribution. A sample $(\boldsymbol{x}, y) \sim \mathcal{D}_c^{\mathrm{tr}}$ satisfies the following:

$$y \sim U(\{\pm 1\}), \qquad x_c = y, \qquad \forall i \in [d], i \neq c: \; x_i \sim \begin{cases} U([0, y\lambda]) & (y = 1) \\ U([y\lambda, 0]) & (y = -1) \end{cases}, \tag{3}$$

where $0 < \lambda < 1$. For any $i \neq j$, x_i and x_j are independent, given y.

In this distribution, a sample has a feature strongly correlated with its label (i.e., robust feature) at the c-th dimension and has features weakly correlated (i.e., non-robust features) at other dimensions. The correlation between non-robust features and the label is bounded by λ. The robust feature mimics human-interpretable, semantically meaningful attributes in natural objects (e.g., shape). The non-robust features mimic human-imperceptible yet predictive attributes (e.g., texture).

Test Data Distribution. The test data distribution may exhibit more diverse structures than the training data, and may include non-predictive features in addition to robust and non-robust features.

Assumption 3.2 (Test data distribution)**.** Let the index sets of robust, non-robust, and irrelevant features be $\mathcal{S}_{\mathrm{rob}}, \mathcal{S}_{\mathrm{vul}}, \mathcal{S}_{\mathrm{irr}} \subset [d]$, respectively. Suppose that these sets are disjoint, i.e., $\mathcal{S}_{\mathrm{rob}} \cap \mathcal{S}_{\mathrm{vul}} = \mathcal{S}_{\mathrm{vul}} \cap \mathcal{S}_{\mathrm{irr}} = \mathcal{S}_{\mathrm{irr}} \cap \mathcal{S}_{\mathrm{rob}} = \emptyset$ and that $\mathcal{S}_{\mathrm{rob}} \cup \mathcal{S}_{\mathrm{vul}} \cup \mathcal{S}_{\mathrm{irr}} = [d]$. Let the number of robust, non-robust, and irrelevant features be $d_{\mathrm{rob}} := |\mathcal{S}_{\mathrm{rob}}|$, $d_{\mathrm{vul}} := |\mathcal{S}_{\mathrm{vul}}|$, and $d_{\mathrm{irr}} := |\mathcal{S}_{\mathrm{irr}}|$, respectively. Let the scales of the robust, non-robust, and irrelevant features be $\alpha > 0$, $\beta > 0$, and $\gamma \geq 0$, respectively. Let $\mathcal{D}^{\mathrm{te}}$ be the test data distribution. A sample $(\boldsymbol{x}, y) \sim \mathcal{D}^{\mathrm{te}}$ satisfies the following:

(1. Label) The label y follows the uniform distribution $U(\{\pm 1\})$.

(2. Expectation and Moments) For every $i \in \mathcal{S}_{\mathrm{irr}}$, $\mathbb{E}[x_i] = 0$. For every $i \in [d]$ and $n \in \{2, 3, 4\}$, there exist constants $C_i > 0$ and $C_{i,n} \geq 0$ such that

$$\mathbb{E}[yx_i] = \begin{cases} C_i \alpha & (i \in \mathcal{S}_{\mathrm{rob}}) \\ C_i \beta & (i \in \mathcal{S}_{\mathrm{vul}}) \\ 0 & (i \in \mathcal{S}_{\mathrm{irr}}) \end{cases}, \qquad |\mathbb{E}[(yx_i - \mathbb{E}[yx_i])^n]| \leq \begin{cases} C_{i,n}\alpha^n & (i \in \mathcal{S}_{\mathrm{rob}}) \\ C_{i,n}\beta^n & (i \in \mathcal{S}_{\mathrm{vul}}) \\ C_{i,n}\gamma^n & (i \in \mathcal{S}_{\mathrm{irr}}) \end{cases}. \tag{4}$$

(3. Covariance) There exist constants $0 \leq q_{\mathrm{rob}}, q_{\mathrm{vul}} < 1$ such that

$$\left|\left\{ \quad i \in \mathcal{S}_{\mathrm{rob}} \quad \middle| \quad \textstyle\sum_{j \in \mathcal{S}_{\mathrm{rob}} \cup \mathcal{S}_{\mathrm{vul}}} \mathbb{E}[(x_i - \mathbb{E}[x_i])(x_j - \mathbb{E}[x_j])] < 0 \quad \right\}\right| \leq q_{\mathrm{rob}} d_{\mathrm{rob}}, \tag{5}$$

$$\left|\left\{ \quad i \in \mathcal{S}_{\mathrm{vul}} \quad \middle| \quad \textstyle\sum_{j \in \mathcal{S}_{\mathrm{rob}} \cup \mathcal{S}_{\mathrm{vul}}} \mathbb{E}[(x_i - \mathbb{E}[x_i])(x_j - \mathbb{E}[x_j])] < 0 \quad \right\}\right| \leq q_{\mathrm{vul}} d_{\mathrm{vul}}. \tag{6}$$

(4. Independence) For every $i \in \mathcal{S}_{\mathrm{irr}}$, x_i is independent of y and all x_j for $j \neq i$.

In contrast to the training distribution, the test distribution may contain d_{rob} robust features and d_{irr} irrelevant features. The latter simulates natural noise or redundant dimensions commonly found in real-world data. For example, in MNIST (Deng, 2012), the top-left pixel is always zero and thus not predictive. Assumption 4 requires each irrelevant feature to be independent of both the label and all the other features. Robust and non-robust features are not assumed to be mutually independent.

Assumption 2 (Expectation) ensures that robust and non-robust features exhibit positive correlation with the label. Given sufficient data, it is always possible to preprocess features to positively align with the label. For example, with a large N, this can be achieved by multiplying each feature x_i by $\mathrm{sgn}(\mathbb{E}[yx_i]) \approx \mathrm{sgn}(\sum_{n=1}^{N} y_n x_{n,i})$, ensuring $\mathbb{E}[y(\mathrm{sgn}(\mathbb{E}[yx_i])x_i)] = |\mathbb{E}[yx_i]| \geq 0$.

Assumption 2 (Moments) bounds the n-th central moment of each feature by a constant multiple of the n-th power of its expectation. This property, commonly referred to as Taylor's law (Taylor, 1961), is observed in a wide range of natural datasets and distributions.

Assumption 3 bounds the number of features whose total covariance with other informative features (i.e., robust and non-robust features) is negative. As stated in Theorem 3.6, we typically assume that q_{rob} and q_{vul} are small (but not necessarily infinitesimal). This assumption prevents unrealistic cases where useful features are overly anti-correlated with others, which could hinder learning. When all predictive features are independent conditioned on the label, $q_{\mathrm{rob}} = 0$ and $q_{\mathrm{vul}} = 0$ satisfy this assumption. We can observe that q_{rob} and q_{vul} are small in real-world datasets (cf. Fig. A2).

These conditions encompass a wide class of realistic data distributions.

- **Example 1: Training data distribution.** The training distribution $\mathcal{D}_c^{\mathrm{tr}}$ is a special case of the test distribution $\mathcal{D}^{\mathrm{te}}$. In this case, the number of robust features is $d_{\mathrm{rob}} = 1$ with scale $\alpha \approx 1$. Similarly, $d_{\mathrm{vul}} = d - 1$ and $\beta \approx \lambda$. There are no irrelevant features, i.e., $d_{\mathrm{irr}} = 0$. By construction, and due to the properties of the uniform distribution, this distribution satisfies all the assumptions.
- **Example 2: Basic distributions.** The test distribution class includes basic distributions, such as uniform, normal, exponential, beta, gamma, Bernoulli, binomial distributions, etc. For example, consider normal distribution. Assumptions 3 and 4 are satisfied if all features are mutually independent. The expectation and second-moment constraints can be satisfied by setting appropriate mean and covariance. The third- and fourth-moment constraints are inherently satisfied.

- **Example 3: MNIST/Fashion-MNIST/CIFAR-10.** Empirical evidence suggests that preprocessed MNIST (Deng, 2012), Fashion-MNIST (Xiao et al., 2017), and CIFAR-10 (Krizhevsky, 2009) approximately satisfy our assumptions. Consider MNIST. Let $\{\boldsymbol{x}_n^{(0)}\}_{n=1}^N, \{\boldsymbol{x}_n^{(1)}\}_{n=1}^N \in [0,1]^{784}$ denote the samples of digits zero and one, respectively. We assign $y=1$ to digit zero and $y=-1$ to digit one. Center the data via $\boldsymbol{x}' \leftarrow \boldsymbol{x} - \bar{\boldsymbol{x}}$ with $\bar{\boldsymbol{x}} := (1/2N)\sum_{n=1}^N(\boldsymbol{x}_n^{(0)} + \boldsymbol{x}_n^{(1)})$ and align features with the label using $\boldsymbol{x}'' \leftarrow \operatorname{sgn}(\sum_{n=1}^N(\boldsymbol{x}_n^{(0)} - \boldsymbol{x}_n^{(1)})) \odot \boldsymbol{x}'$. In this representation, common background features yield near-zero expectations (i.e., $\gamma \approx 0$), while discriminative features—such as the left and right arcs of zero or the vertical stroke of one—correlate strongly with the label (i.e., $\alpha \approx 0.2$) (cf. Fig. A2). Additionally, some outlier-dependent pixels (e.g., corners occasionally activated by slanted digits) exhibit weak correlation with the label (i.e., $\beta \approx 0.01$), reflecting non-robust but predictive attributes. Empirical analysis reveals that most dimensions exhibit positive total covariance with others, consistent with Assumption 3 (cf. Fig. A2). The main departure from our test distribution lies in the fact that real-world datasets exhibit a gradual transition in feature importance rather than a binary separation between robust and non-robust features.
- **Example 4: Linear combination of orthonormal bases.** Under mild conditions, any distribution comprising robust and non-robust directions forming an orthonormal basis can be transformed into our setting via principal component analysis (cf. Appendix B).

Adversarial Attack. We assume that the test query $\boldsymbol{x}_{N+1}$ is subject to an adversarial perturbation $\boldsymbol{\Delta}$ constrained in the ℓ_∞ norm, i.e., $\|\boldsymbol{\Delta}\|_\infty \le \epsilon$, where $\epsilon \ge 0$ denotes the perturbation budget. In practice, ϵ is chosen to match the scale of non-robust features (e.g., $\epsilon \approx \lambda$ for the training and $\epsilon \approx \beta$ for the test distribution). This ensures that perturbations effectively manipulate non-robust features while leaving robust features intact and remaining imperceptible to humans.

Pretraining with In-Context Loss. For pretraining, we consider the following problem based on the in-context loss (Ahn et al., 2023; Bai et al., 2023; Mahankali et al., 2024; Zhang et al., 2024b):

$$\min_{\boldsymbol{P},\boldsymbol{Q}\in[0,1]^{(d+1)\times(d+1)}} \mathop{\mathbb{E}}_{c\sim U([d]),\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_c^{\text{tr}}} \left[\max_{\|\boldsymbol{\Delta}\|_\infty\le\epsilon} -y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}};\boldsymbol{P},\boldsymbol{Q})]_{d+1,N+1}\right]. \quad (7)$$

This formulation encourages the transformer to extract robust, generalizable representations from N clean in-context demonstrations and accurately classify an adversarially perturbed query sample.

3.2 Warm-Up: Linear Classifiers and Oracle

Standard Linear Classifiers Extract All Features and Thus are Vulnerable. As a warm-up, consider standard training of a linear classifier parameterized by $\boldsymbol{w} \in \mathbb{R}^d$ on the c-th training distribution $\mathcal{D}_c^{\text{tr}}$. Standard training results in $\boldsymbol{w}^{\text{std}} := \arg\min_{\boldsymbol{w}\in[0,1]^d} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_c^{\text{tr}}}[-y\boldsymbol{w}^\top\boldsymbol{x}] = \mathbf{1}_d$. This classifier utilizes all features, including the robust feature at the c-th dimension and other non-robust features. Although $\boldsymbol{w}^{\text{std}}$ achieves correct predictions on clean samples, $\mathbb{E}[y{\boldsymbol{w}^{\text{std}}}^\top\boldsymbol{x}] > 0$, it is vulnerable to adversarial perturbations, $\mathbb{E}[\min_{\|\boldsymbol{\Delta}\|_\infty\le\epsilon} y{\boldsymbol{w}^{\text{std}}}^\top(\boldsymbol{x}+\boldsymbol{\Delta})] \le 0$ for $\epsilon \ge \frac{1+(d-1)(\lambda/2)}{d}$.[1] This implies that, for a small d, the perturbation must be of the order $\epsilon \gtrsim 1$, which affects the robust feature and is human-perceptible. However, as d increases, the threshold decreases to $\epsilon \gtrsim \lambda$, which is at the scale of non-robust features and imperceptible yet can break the classifier predictions.

Linear Classifiers can be Specific Robust, but not Universally Robust. Consider adversarial training $\min_{\boldsymbol{w}\in[0,1]^d} \mathbb{E}[\max_{\|\boldsymbol{\Delta}\|_\infty\le\epsilon} -y\boldsymbol{w}^\top(\boldsymbol{x}+\boldsymbol{\Delta})]$. For $\epsilon \ge \frac{\lambda}{2}$, the optimal solution $\boldsymbol{w}^{\text{adv}}$ has one at the c-th dimension and zero otherwise. The classifier relies solely on the robust feature at the c-th dimension and ignores all non-robust features. Unlike $\boldsymbol{w}^{\text{std}}$, this classifier can correctly classify both clean and adversarial samples for $0 \le \epsilon < 1$; linear classifiers can be robust for a specific training distribution. However, $\boldsymbol{w}^{\text{adv}}$ tailored to $\mathcal{D}_c^{\text{tr}}$ is vulnerable on other distributions $\mathcal{D}_{c'}^{\text{tr}}$ indexed by $c' \ne c$; linear classifiers cannot be universally robust.

Universally Robust Classifiers Exist. Although linear classifiers cannot exhibit universal robustness across all c, universally robust classifiers do exist. For example, the classifier $h(\boldsymbol{x}) := \operatorname{sgn}(x_i)$ with $i := \arg\max_{i'\in[d]} |x_{i'}|$ always produces correct predictions for clean data $\boldsymbol{x} \sim \mathcal{D}_c^{\text{tr}}$ for any c and perturbed data $\boldsymbol{x}+\boldsymbol{\Delta}$ with $\|\boldsymbol{\Delta}\|_\infty \le \frac{1}{2}$.

[1] $\mathbb{E}[\min_{\|\boldsymbol{\Delta}\|_\infty\le\epsilon} y{\boldsymbol{w}^{\text{std}}}^\top(\boldsymbol{x}+\boldsymbol{\Delta})] = {\boldsymbol{w}^{\text{std}}}^\top(\mathbb{E}[y\boldsymbol{x}] - \epsilon\mathbf{1}_d) = \{1+(d-1)(\lambda/2)\} - d\epsilon \le 0$.

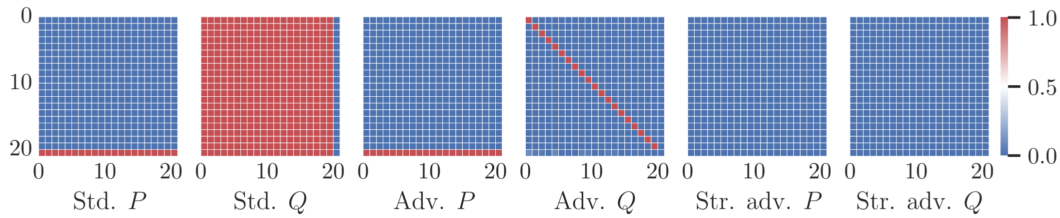

Figure 1: Parameter heatmaps induced by adversarial training (7) with $d = 20$ and $\lambda = 0.1$. For the standard, adversarial, and strong adversarial regimes, we used $\epsilon = 0$, $\frac{1+(d-1)(\lambda/2)}{d} = 0.098$, and $\frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3} = 0.95$, respectively. We optimized (7) by stochastic gradient descent. Detailed experimental settings can be found in Appendix C.

3.3 Adversarial Pretraining

In this section, we consider the global solution for the minimization problem (7).

Optimization Challenges. Although the training distributions are relatively simple, the minimization problem (7) remains nontrivial due to the non-linearity and non-convexity in the trainable parameters $\boldsymbol{P}$ and $\boldsymbol{Q}$. The high non-linearity of self-attention and inner-maximization are also obstacles. Indeed, the minimization problem (7) is rearranged as the following non-linear maximization problem:

Lemma 3.3 (Transformation of original optimization problem)**.** The minimization problem (7) can be transformed into the maximization problem $\max_{\boldsymbol{b}\in\{0,1\}^{d+1}} \sum_{i=1}^{d(d+1)} \max(0, \sum_{j=1}^{d+1} b_j h_{i,j})$, where $h_{i,j} \in \mathbb{R}$ is an (i,j)-dependent constant, and there exists a mapping from $\boldsymbol{b}$ to $\boldsymbol{P}$ and $\boldsymbol{Q}$.

The proof can be found in Appendix D. This lemma highlights the inherent difficulty of optimizing (7), which requires selecting a binary vector $\boldsymbol{b}$ that balances $d(d+1)$ interdependent non-linear terms.

Global Solution. Considering the symmetric property of $\boldsymbol{b}$ and further transformation of the problem in Lemma 3.3, we identify the global solution of (7) for some perturbation cases.

Theorem 3.4 (Parameters induced by adversarial pretraining)**.** The global minimizer of (7) is

$$\Big(\text{1. Standard; } \epsilon = 0\Big) \quad \boldsymbol{P} = \boldsymbol{P}^{\text{std}} := \begin{bmatrix} \boldsymbol{0}_{d,d+1} \\ \boldsymbol{1}_{d+1}^\top \end{bmatrix} \text{ and } \boldsymbol{Q} = \boldsymbol{Q}^{\text{std}} := [\boldsymbol{1}_{d+1,d} \quad \boldsymbol{0}_{d+1}].$$

$$\Big(\text{2. Adversarial; } \epsilon = \tfrac{1+(d-1)(\lambda/2)}{d}\Big) \quad \boldsymbol{P} = \boldsymbol{P}^{\text{adv}} := \begin{bmatrix} \boldsymbol{0}_{d,d+1} \\ \boldsymbol{1}_{d+1}^\top \end{bmatrix} \text{ and } \boldsymbol{Q} = \boldsymbol{Q}^{\text{adv}} := \begin{bmatrix} \boldsymbol{I}_d & \boldsymbol{0}_d \\ \boldsymbol{0}_d^\top & 0 \end{bmatrix}.$$

$$\Big(\text{3. Strongly adversarial; } \epsilon \geq \tfrac{\lambda}{2} + \tfrac{3}{2}\tfrac{2-\lambda}{(d-1)\lambda^2+3}\Big) \quad \boldsymbol{P} = \boldsymbol{0}_{d+1,d+1} \text{ and } \boldsymbol{Q} = \boldsymbol{0}_{d+1,d+1}.$$

The proof and optimal parameters for different ϵ can be found in Appendix D. Importantly, the optimal $\boldsymbol{P}$ and $\boldsymbol{Q}$ are independent of any specific training distribution (i.e., index c), reflecting that the transformer obtains learnability from demonstrations rather than memorizing individual tasks. The experimental results via gradient descent completely align with our theoretical predictions (Fig. 1).

Failure Case. In the strong adversarial regime, the global optimum becomes $\boldsymbol{P} = \boldsymbol{Q} = \boldsymbol{0}$, causing the transformer to always output zero regardless of the input. Namely, no universally robust single-layer linear transformers exist, despite the existence of universally robust classifiers (cf. Section 3.2). The perturbation scale $\epsilon \geq \frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3}$ decreases in d: it transitions from $\epsilon = 1$ when $d = 1$ to $\epsilon \to \frac{\lambda}{2}$ as $d \to \infty$. In moderate dimensions ($d \approx \frac{1}{\lambda}$), adversarial perturbations must be $\epsilon \gtrsim 1$ to break robustness. They are comparable to the scale of robust features and thus perceptible to humans, contradicting the concept of adversarial perturbations. However, in extremely high dimensions ($d \gtrsim \frac{1}{\lambda^2}$), it suffices to perturb by only $\epsilon \gtrsim \lambda$, which is on the same scale as non-robust features and typically imperceptible, keeping the perturbation concept. This can be rephrased as: under our training distributions, single-layer linear transformers cannot achieve universal robustness when the non-robust dimensions (i.e., $d - 1$) substantially outnumber the robust dimension (i.e., 1).

3.4 Universal Robustness

In this section, we show that universal robustness (over seen and unseen distributions) can be attained by adversarial pretraining and clean in-context demonstrations.

Standard Pretraining Leads to Vulnerability. We begin by showing that the standard model fails to classify adversarially perturbed inputs.

Theorem 3.5 (Standard pretraining case)**.** There exist a constant $C > 0$ and a strictly positive function $g(d_{\text{rob}}, d_{\text{vul}}, d_{\text{irr}}, \alpha, \beta, \gamma)$ such that

$$\begin{aligned}&\mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\text{te}}}\left[\min_{\|\boldsymbol{\Delta}\|_\infty \le \epsilon} y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}^{\text{std}}, \boldsymbol{Q}^{\text{std}})]_{d+1,N+1}\right] \\ \le \quad & g(d_{\text{rob}}, d_{\text{vul}}, d_{\text{irr}}, \alpha, \beta, \gamma) \quad \Big\{ \quad \underbrace{C(d_{\text{rob}}\alpha + d_{\text{vul}}\beta)}_{\text{Prediction for original data}} \quad - \quad \underbrace{(d_{\text{rob}} + d_{\text{vul}} + d_{\text{irr}})\epsilon}_{\text{Adversarial effect}} \quad \Big\}. \end{aligned} \tag{8}$$

The proof can be found in Appendix E. This result analyzes the expectation of the product between the true label and model prediction for the query. A positive value indicates correct classification and a nonpositive value indicates failure. Since $g(d_{\text{rob}}, d_{\text{vul}}, d_{\text{irr}}, \alpha, \beta, \gamma)$ is always positive, a nonpositive $C(d_{\text{rob}}\alpha + d_{\text{vul}}\beta) - (d_{\text{rob}} + d_{\text{vul}} + d_{\text{irr}})\epsilon$ implies incorrect classification.

The standard model extracts both features and thus is vulnerable. Assume $d_{\text{irr}} = 0$. Like standard linear classifiers, the standard model leverages both robust features $d_{\text{rob}}\alpha$ and non-robust features $d_{\text{vul}}\beta$. This also makes vulnerability to adversarial perturbations contributing to the term $(d_{\text{rob}} + d_{\text{vul}})\epsilon$. The prediction becomes incorrect, $C(d_{\text{rob}}\alpha + d_{\text{vul}}\beta) - (d_{\text{rob}} + d_{\text{vul}})\epsilon \le 0$, when $\epsilon \gtrsim \frac{d_{\text{rob}}\alpha + d_{\text{vul}}\beta}{d_{\text{rob}} + d_{\text{vul}}}$. The perturbation size ϵ is at the same scale as non-robust features, $\epsilon \approx \beta$, when $d_{\text{vul}} \gtrsim d_{\text{rob}} \frac{\alpha - \beta}{\beta}$. In typical cases where the scale of robust features is much larger than that of non-robust ones, $\alpha \gg \beta$, we can informally conclude:

For $\epsilon \approx \beta$, if $d_{\text{vul}} \gtrsim \frac{\alpha}{\beta} d_{\text{rob}}$, then the normally pretrained single-layer linear transformer is vulnerable.

Non-predictive features accelerate vulnerability. Redundant dimensions d_{irr} do not contribute to the first term, i.e., accuracy, but they increase the second term, i.e., vulnerability. Therefore, they degrade robustness without providing any benefit to prediction. In addition, d_{irr} amplifies the adversarial effect at a rate of $d_{\text{irr}}\epsilon$, which is comparable to the effect from the useful dimensions, $d_{\text{rob}}\epsilon$ and $d_{\text{vul}}\epsilon$.

Adversarial Pretraining Leads to Universal Robustness. We now establish the universal robustness of the adversarially pretrained model.

Theorem 3.6 (Adversarial pretraining case)**.** Suppose that q_{rob} and q_{vul} defined in Assumption 3.2 are sufficiently small. There exist constants $C_1, C_2 > 0$ such that

$$\begin{aligned}&\mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\text{te}}}\left[\min_{\|\boldsymbol{\Delta}\|_\infty \le \epsilon} y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}^{\text{adv}}, \boldsymbol{Q}^{\text{adv}})]_{d+1,N+1}\right] \\ &\ge \underbrace{C_1(d_{\text{rob}}\alpha + d_{\text{vul}}\beta + 1)(d_{\text{rob}}\alpha^2 + d_{\text{vul}}\beta^2)}_{\text{Prediction for original data}} \\ &\quad \underbrace{- C_2\left\{(d_{\text{rob}}\alpha + d_{\text{vul}}\beta + 1)\left(d_{\text{rob}}\alpha + d_{\text{vul}}\beta + \frac{d_{\text{irr}}\gamma}{\sqrt{N}}\right) + d_{\text{irr}}\left(\sqrt{\frac{d_{\text{irr}}}{N}} + 1\right)\gamma^2\right\}\epsilon}_{\text{Adversarial effect}}. \end{aligned} \tag{9}$$

The proof and generalized theorem can be found in Appendix E and Theorem E.1. For notational simplicity, we assume small q_{rob} and q_{vul}. However, we do not require infinitesimal q_{rob} and q_{vul}. See Theorem E.1 and Appendix B. In contrast to Theorem 3.5, this theorem provides the lower bound. A positive right-hand side implies correct classification under adversarial perturbations.

The adversarially trained model prioritizes robust features. Assume $d_{\text{irr}} = 0$. Up to constant factors, the lower bound reduces to $(d_{\text{rob}}\alpha + d_{\text{vul}}\beta + 1)\{d_{\text{rob}}\alpha^2 + d_{\text{vul}}\beta^2 - (d_{\text{rob}}\alpha + d_{\text{vul}}\beta)\epsilon\}$. The important factor is $d_{\mathrm{rob}}\alpha^2 + d_{\mathrm{vul}}\beta^2 - (d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta)\epsilon$, which determines the sign. As shown in Theorem 3.5, the standard models extract features at scales $d_{\mathrm{rob}}\alpha$ and $d_{\mathrm{vul}}\beta$. In contrast, the adversarially trained models extract them at quadratic scales $d_{\mathrm{rob}}\alpha^2$ and $d_{\mathrm{vul}}\beta^2$. Since robust features typically have larger magnitude ($\alpha^2 \gg \beta^2$), the adversarially trained model places greater emphasis on robust features and mitigate the influence of non-robust features, comparing the standard counterpart.

It is universally robust. As shown in the above discussion, to flip the prediction of the adversarially trained model, the perturbation must satisfy $\epsilon \gtrsim \frac{d_{\mathrm{rob}}\alpha^2 + d_{\mathrm{vul}}\beta^2}{d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta}$. To maintain $\epsilon \approx \beta$, d_{vul} needs to be $d_{\mathrm{vul}} \gtrsim \frac{d_{\mathrm{rob}}\alpha(\alpha-\beta)}{\beta^2}$. In typical cases where $\alpha \gg \beta$, we can informally conclude:

For $\epsilon \approx \beta$, if $d_{\mathrm{vul}} \lesssim (\frac{\alpha}{\beta})^2 d_{\mathrm{rob}}$, then the adversarially pretrained single-layer linear transformer is universally robust.

This threshold represents a substantial improvement over the standard model's robustness condition of $d_{\mathrm{vul}} \lesssim \frac{\alpha}{\beta} d_{\mathrm{rob}}$. For example, when $\alpha = 160/255$ and $\beta = 8/255$, the standard model becomes vulnerable at $d_{\mathrm{vul}} \gtrsim 20 d_{\mathrm{rob}}$, whereas the adversarially pretrained model remains robust up to $d_{\mathrm{vul}} \lesssim 400 d_{\mathrm{rob}}$. This result also suggests that they become vulnerable when non-robust dimensions significantly outnumber robust ones, consistent with the failure case in Section 3.3.

It is more robust to attacks that exploit non-predictive features. Theorem 3.6 shows that even though the adversary may exploit redundant dimensions, their effect is significantly attenuated. Assume $N \to \infty$ for simplicity. The adversarial effect from irrelevant features then scales as $d_{\mathrm{irr}}\gamma^2\epsilon$, which is linear in d_{irr}. In contrast, the clean prediction scales as $d_{\mathrm{rob}}^2\alpha^3$ and $d_{\mathrm{vul}}^2\beta^3$, i.e., quadratically in the number of informative features. Thus, as long as useful features dominate in magnitude and number, the influence of redundant features on the model's robustness remains limited.

3.5 Open Challenges

In this section, we show that two open challenges in robust classification (Schmidt et al., 2018; Tsipras et al., 2019) persist in our setting.

Accuracy–Robustness Trade-Off. Inspired by Tsipras et al. (2019), we consider a situation where robust features correlate with the label with some probability, yet non-robust features always correlate.

Theorem 3.7 (Accuracy–robustness trade-off)**.** Assume $|\mathcal{S}_{\mathrm{rob}}| = 1$, $|\mathcal{S}_{\mathrm{vul}}| = d-1$, and $|\mathcal{S}_{\mathrm{irr}}| = 0$. In addition to Assumption 3.2, for $(\boldsymbol{x}, y) \sim \mathcal{D}^{\mathrm{te}}$, suppose that yx_i takes α with probability $p > 0.5$ and $-\alpha$ with probability $1-p$ for $i \in \mathcal{S}_{\mathrm{rob}}$. Moreover, yx_i takes β with probability one for $i \in \mathcal{S}_{\mathrm{vul}}$. Let $\tilde{f}(\boldsymbol{P}, \boldsymbol{Q}) := \mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\mathrm{te}}}[y_{N+1}[f(\boldsymbol{Z_0}; \boldsymbol{P}, \boldsymbol{Q})]_{d+1,N+1}]$. Then, there exist strictly positive functions $g_1(d, \alpha, \beta)$ and $g_2(d, \alpha, \beta)$ such that

$$\tilde{f}(\boldsymbol{P}^{\mathrm{std}}, \boldsymbol{Q}^{\mathrm{std}}) = \begin{cases} g_1(d, \alpha, \beta)(\alpha + (d-1)\beta) & (\text{w.p.} \quad p) \\ g_1(d, \alpha, \beta)(-\alpha + (d-1)\beta) & (\text{w.p.} \quad 1-p) \end{cases}, \tag{10}$$

$$\tilde{f}(\boldsymbol{P}^{\mathrm{adv}}, \boldsymbol{Q}^{\mathrm{adv}}) \le g_2(d, \alpha, \beta)\{-(2p-1)\alpha^2 + (d-1)\beta^2\} \quad (\text{w.p.} \quad 1-p). \tag{11}$$

The proof can be found in Appendix F. Different from Theorems 3.5 and 3.6, this theorem considers the expectation over $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, instead of $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}$. The query $(\boldsymbol{x}_{N+1}, y_{N+1})$ behaves probabilistically. If $d \gtrsim \frac{\alpha}{\beta}$, the standard model can always produce correct predictions. However, if $d \lesssim (2p-1)(\frac{\alpha}{\beta})^2$, the adversarially trained model produces incorrect predictions with probability $1-p$. This discrepancy arises because the robust model discards non-robust but predictive features.

Need for Larger In-Context Sample Sizes. Building on the assumptions of Theorem 3.7, we informally summarize Theorem G.1 as follows (omitting constant factors for clarity):

Consider $\mathbb{E}_{\boldsymbol{x}_{N+1}, y_{N+1}}[y_{N+1}[f(\boldsymbol{Z_0}; \boldsymbol{P}, \boldsymbol{Q})]_{d+1,N+1}]$. Assume $d \lesssim \frac{\alpha}{\beta}$, $p \to 0.5$, and a small N regime. With probability at least $1 - \exp(-N)$, the standard transformer outputs correct answers. With probability at most $1 - \frac{1}{\sqrt{N}}$, the adversarially trained transformer outputs correct answers.

Table 1: Accuracy (%) of normally and adversarially pretrained single-layer linear transformers. Left values represent clean accuracy; right values represent robust accuracy. For $\mathcal{D}^{\text{tr}}$ (cf. Assumption 3.1), we used $d = 100$ and $\lambda = 0.1$. For $\mathcal{D}^{\text{te}}$ (cf. Assumption 3.2), we constructed a test distribution from multivariate normal distributions with $d_{\text{rob}} = 10$, $d_{\text{vul}} = 90$, $d_{\text{irr}} = 0$, $\alpha = 1.0$, and $\beta = 0.1$. For the real-world datasets, values were averaged across all 45 binary classification pairs from the 10 classes. The perturbation budgets were set as follows: $\epsilon = 0.15$ for $\mathcal{D}^{\text{tr}}$, 0.2 for $\mathcal{D}^{\text{te}}$, 0.1 for MNIST and CIFAR-10, and 0.15 for Fashion-MNIST. See Appendix C for details.

	$\mathcal{D}^{\text{tr}}$	$\mathcal{D}^{\text{te}}$	MNIST	FMNIST	CIFAR10
Normally pretrained model	**100** / 0	**100** / 0	**94** / 4	**91** / 20	**68** / 21
Adversarially pretrained model	**100** / **100**	99 / **95**	93 / **72**	89 / **62**	64 / **34**

This result indicates that the adversarially pretrained model requires substantially more in-context demonstrations to match the clean accuracy of the standard model. In low-sample regimes, the standard model rapidly approaches high accuracy, while the robust model converges more slowly due to its reliance on robust features, which are underrepresented in small-sample regimes.

4 Experimental Results

Additional results and detailed experimental settings are provided in Appendix C.

Verification of Theorem 3.4. We trained single-layer linear transformers (2) using stochastic gradient descent over $[0, 1]^d$ with in-context loss (7). The training distribution was configured with $d = 20$ and $\lambda = 0.1$. We used $\epsilon = 0$, $\frac{1+(d-1)(\lambda/2)}{d} = 0.098$, and $\frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3} = 0.95$ for the standard, adversarial, and strong adversarial regimes, respectively. The heatmaps of the learned parameters are shown in Fig. 1. These results completely align with the theoretical predictions of Theorem 3.4.

Verification of Theorems 3.5 to 3.7. We evaluated normally and adversarially pretrained single-layer linear transformers on $\mathcal{D}^{\text{tr}}$, $\mathcal{D}^{\text{te}}$, MNIST (Deng, 2012), Fashion-MNIST (Xiao et al., 2017), and CIFAR-10 (Krizhevsky, 2009). These results are provided in Tab. 1. They suggest that the standard models achieve high clean accuracy but suffer severe degradation under adversarial attacks, consistent with Theorem 3.5. In contrast, the adversarially pretrained models maintain high robustness, supporting Theorem 3.6, while their clean accuracy is lower, aligning with the accuracy–robustness trade-off described in Theorem 3.7.

5 Conclusion and Limitations

We theoretically demonstrated that single-layer linear transformers, after adversarial pretraining across classification tasks, can robustly adapt to previously unseen classification tasks through in-context learning, without any additional training. These results pave the way for universally robust foundation models. We also showed that these transformers can adaptively focus on robust features, exhibit an accuracy–robustness trade-off, and require a larger number of in-context demonstrations.

Our limitations include the assumptions on the data distributions and architectures. While we assume that the data distributions consist of clearly separated robust and non-robust features, real-world datasets typically exhibit a more gradual transition (cf. Section 3.1, especially Example 3). Single-layer linear transformers lack the practical characteristics of multi-layer models and softmax attention. Although such theoretical assumptions are standard and comparable in strength to those in prior work (cf. studies on in-context learning in Appendix A), they limit the applicability of our results.

The cost of adversarial pretraining is the limitation of the concept of universally robust foundation models. We expect that such efforts would be undertaken by large companies, which could offset development costs through API fees. In addition, acceleration techniques for adversarial training, which have been extensively studied in the existing literature, can reduce this cost to a level comparable to standard training. Our theoretical analysis is an important first step toward fostering the practical development of universally robust foundation models. See also the last paragraph in Section 1.

Reproducibility statement

All experimental procedures are described in Section 4 and Appendix C. The supplementary material includes the source code to reproduce our experimental results. Proofs of the theorems are provided in Appendices D to G.

The Use of Large Language Models (LLMs)

We used LLMs to improve our writing. No essential contributions were made by the LLMs.

References

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. In *NeurIPS*, volume 36, pp. 45614–45650, 2023.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *ICLR*, 2023.

Ahmed Aldahdooh, Wassim Hamidouche, and Olivier Deforges. Reveal of vision transformers robustness against adversarial attacks. *arXiv:2106.03734*, 2021.

Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. In *NeurIPS*, volume 33, pp. 16048–16059, 2020.

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

Robert B Ash. *Information Theory*. Courier Corporation, 1990.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, pp. 274–283, 2018.

Maximilian Augustin, Alexander Meinke, and Matthias Hein. Adversarial robustness on in-and out-distribution improves explainability. In *ECCV*, pp. 228–245, 2020.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. In *NeurIPS*, volume 36, pp. 57125–57211, 2023.

Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? In *NeurIPS*, volume 34, pp. 26831–26843, 2021.

Philipp Benz, Soomin Ham, Chaoning Zhang, Adil Karjauv, and In So Kweon. Adversarial robustness comparison of vision transformer and MLP-mixer to CNNs. In *BMVC*, 2021.

Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. In *ICCV*, pp. 10231–10241, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, volume 33, pp. 1877–1901, 2020.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *USENIX Security*, pp. 2633–2650, 2021.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *ICLR*, 2022.

Prasad Chalasani, Jiefeng Chen, Amrita Roy Chowdhury, Xi Wu, and Somesh Jha. Concise explanations of neural networks using adversarial training. In *ICML*, pp. 1383–1391, 2020.

Xiang Cheng, Yuxin Chen, and Suvrit Sra. Transformers implement functional gradient descent to learn non-linear functions in context. In *ICML*, 2024.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, pp. 2206–2216, 2020.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can gpt learn in-context? language models implicitly perform gradient descent as meta-optimizers. In *ACL*, 2023.

Edoardo Debenedetti, Vikash Sehwag, and Prateek Mittal. A light recipe to train robust vision transformers. In *SaTML*, pp. 225–253, 2023.

Li Deng. The MNIST database of handwritten digit images for machine learning research. *Signal Processing Magazine*, 29(6):141–142, 2012.

Edgar Dobriban, Hamed Hassani, David Hong, and Alexander Robey. Provable tradeoffs in adversarially robust classification. *IEEE Transactions on Information Theory*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations. *arXiv:1906.00945*, 2019.

Christian Etmann, Sebastian Lunz, Peter Maass, and Carola-Bibiane Schönlieb. On the connection between adversarial robustness and saliency map interpretability. In *ICML*, 2019.

Hao Fan, Zhaoyang Ma, Yong Li, Rui Tian, Yunli Chen, and Chenlong Gao. Mixprompt: Enhancing generalizability and adversarial robustness for vision-language models via prompt fusion. In *ICIC*, pp. 328–339, 2024.

Spencer Frei and Gal Vardi. Trained transformer classifiers generalize and exhibit benign overfitting in-context. In *ICLR*, 2025.

Deqing Fu, Tian-qi Chen, Robin Jia, and Vatsal Sharan. Transformers learn to achieve second-order convergence rates for in-context linear regression. In *NeurIPS*, volume 37, pp. 98675–98716, 2024.

Shaopeng Fu, Liang Ding, and Di Wang. ”short-length” adversarial training helps llms defend ”long-length” jailbreak attacks: Theoretical and empirical evidence. *arXiv:2502.04204*, 2025.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In *NeurIPS*, volume 35, pp. 30583–30598, 2022.

Siddhant Garg and Goutham Ramakrishnan. BAE: BERT-based adversarial examples for text classification. In *EMNLP*, pp. 6174–6181, 2020.

Khashayar Gatmiry, Nikunj Saunshi, Sashank J Reddi, Stefanie Jegelka, and Sanjiv Kumar. Can looped transformers learn to implement multi-step gradient descent for in-context learning? In *ICML*, 2024.

Angeliki Giannou, Liu Yang, Tianhao Wang, Dimitris Papailiopoulos, and Jason D Lee. How well can transformers emulate in-context newton's method? *arXiv:2403.03183*, 2024.

Micah Goldblum, Liam Fowl, and Tom Goldstein. Adversarially robust few-shot learning: A meta-learning approach. In *NeurIPS*, volume 33, pp. 17886–17895, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Yufan Hou, Lixin Zou, and Weidong Liu. Task-based focal loss for adversarially robust meta-learning. In *ICPR*, pp. 2824–2829, 2021.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *NeurIPS*, volume 32, pp. 125–136, 2019.

Ahmadreza Jeddi, Mohammad Javad Shafiee, and Alexander Wong. A simple fine-tuning is all you need: Towards robust deep learning via adversarial fine-tuning. *arXiv:2012.13628*, 2020.

Xiaojun Jia, Sensen Gao, Simeng Qin, Ke Ma, Xinfeng Li, Yihao Huang, Wei Dong, Yang Liu, and Xiaochun Cao. Evolution-based region adversarial prompt learning for robustness enhancement in vision-language models. *arXiv:2503.12874*, 2025.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is BERT really robust? a strong baseline for natural language attack on text classification and entailment. In *AAAI*, volume 34, pp. 8018–8025, 2020.

Simran Kaur, Jeremy Cohen, and Zachary C Lipton. Are perceptually-aligned gradients a general property of robust classifiers? In *NeurIPS WS*, 2019.

Hoki Kim, Woojin Lee, and Jaewook Lee. Understanding catastrophic overfitting in single-step adversarial training. In *AAAI*, volume 35, pp. 8119–8127, 2021.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. In *NeurIPS*, volume 36, pp. 43057–43083, 2023.

Lin Li, Haoyan Guan, Jianing Qiu, and Michael Spratling. One prompt word is enough to boost adversarial robustness for pre-trained vision-language models. In *CVPR*, pp. 24408–24419, 2024.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. BERT-ATTACK: Adversarial attack against BERT using BERT. In *EMNLP*, pp. 6193–6202, 2020.

Tianle Li, Chenyang Zhang, Xingwu Chen, Yuan Cao, and Difan Zou. On the robustness of transformers against context hijacking for linear classification. *arXiv:2502.15609*, 2025.

Licong Lin, Yu Bai, and Song Mei. Transformers as decision makers: Provable in-context reinforcement learning via supervised pretraining. In *ICLR*, 2024.

Chang Liu, Yinpeng Dong, Wenzhao Xiang, Xiao Yang, Hang Su, Jun Zhu, Yuefeng Chen, Yuan He, Hui Xue, and Shibao Zheng. A comprehensive study on robustness of image classification models: Benchmarking and rethinking. *IJCV*, 133(2):567–589, 2025.

Fan Liu, Shuyu Zhao, Xuelong Dai, and Bin Xiao. Long-term cross adversarial training: A robust meta-learning method for few-shot classification tasks. In *ICML WS*, 2021.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. In *ICLR*, 2024.

Lin Luo, Xin Wang, Bojia Zi, Shihao Zhao, and Xingjun Ma. Adversarial prompt distillation for vision-language models. *arXiv:2411.15244*, 2024.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Arvind Mahankali, Tatsunori B Hashimoto, and Tengyu Ma. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. In *ICLR*, 2024.

Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. In *ICCV*, pp. 7838–7847, 2021.

Chengzhi Mao, Scott Geng, Junfeng Yang, Xin Wang, and Carl Vondrick. Understanding zero-shot adversarial robustness for large-scale models. In *ICLR*, 2023.

Mohammad Mehrabi, Adel Javanmard, Ryan A Rossi, Anup Rao, and Tung Mai. Fundamental tradeoffs in distributionally adversarial training. In *ICML*, pp. 7544–7554, 2021.

Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. In *NeurIPS*, volume 34, pp. 23296–23308, 2021.

Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A Feder Cooper, Daphne Ippolito, Christopher A Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from (production) language models. *arXiv:2311.17035*, 2023.

Geon Yeong Park and Sang Wan Lee. Reliably fast adversarial training via latent adversarial perturbation. In *ICCV*, pp. 7758–7767, 2021.

Sayak Paul and Pin-Yu Chen. Vision transformers are robust learners. In *AAAI*, volume 36, pp. 2071–2081, 2022.

Fábio Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. In *NeurIPS WS*, 2022.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang. Adversarial training can hurt generalization. In *ICML WS*, 2019.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. In *ICML*, 2020.

Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. In *NeurIPS*, volume 32, 2019.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *NeurIPS*, volume 31, 2018.

Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, 2019.

Rulin Shao, Zhouxing Shi, Jinfeng Yi, Pin-Yu Chen, and Cho-Jui Hsieh. On the adversarial robustness of vision transformers. *TMLR*, 2022.

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 *ACM CCS*, pp. 1671–1685, 2024.

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 *ICML*, pp. 31210–31227, 2023.

Zhenmei Shi, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. Why larger language models do in-context learning differently? In *ICML*, 2024.

Suraj Srinivas, Sebastian Bordt, and Himabindu Lakkaraju. Which models have perceptually-aligned gradients? an explanation via off-manifold robustness. In *NeurIPS*, volume 36, 2023.

Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *ECCV*, pp. 631–648, 2018.

Satoshi Suzuki, Shin'ya Yamaguchi, Shoichiro Takeda, Sekitoshi Kanai, Naoki Makishima, Atsushi Ando, and Ryo Masumura. Adversarial finetuning with latent representation constraint to mitigate accuracy-robustness tradeoff. In *ICCV*, pp. 4367–4378, 2023.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

Shiyu Tang, Ruihao Gong, Yan Wang, Aishan Liu, Jiakai Wang, Xinyun Chen, Fengwei Yu, Xianglong Liu, Dawn Song, Alan Yuille, et al. RobustART: Benchmarking robustness on architecture design and training techniques. *arXiv:2109.05211*, 2021.

L. R. Taylor. Aggregation, variance and the mean. *Nature*, 189:732–735, 1961.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, volume 33, pp. 1633–1645, 2020.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *ICLR*, 2019.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *ICML*, pp. 35151–35174, 2023.

Johannes Von Oswald, Maximilian Schlegel, Alexander Meulemans, Seijin Kobayashi, Eyvind Niklasson, Nicolas Zucchet, Nino Scherrer, Nolan Miller, Mark Sandler, Max Vladymyrov, et al. Uncovering mesa-optimization algorithms in transformers. In *ICLR WS*, 2024.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing NLP. In *EMNLP-IJCNLP*, 2019.

Ren Wang, Kaidi Xu, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Chuang Gan, and Meng Wang. On fast adversarial robustness adaptation in model-agnostic meta-learning. In *ICLR*, 2021.

Sibo Wang, Jie Zhang, Zheng Yuan, and Shiguang Shan. Pre-trained model guided fine-tuning for zero-shot adversarial robustness. In *CVPR*, pp. 24502–24511, 2024a.

Xin Wang, Kai Chen, Xingjun Ma, Zhineng Chen, Jingjing Chen, and Yu-Gang Jiang. Advqdet: Detecting query-based adversarial attacks with adversarial contrastive prompt tuning. In *ACM MM*, pp. 6212–6221, 2024b.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? In *NeurIPS*, volume 36, pp. 80079–80110, 2023a.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv:2303.03846*, 2023b.

Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning. In *NeurIPS*, volume 36, pp. 36637–36651, 2023.

Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

Boxi Wu, Jindong Gu, Zhifeng Li, Deng Cai, Xiaofei He, and Wei Liu. Towards efficient adversarial training on vision transformers. In *ECCV*, pp. 307–325, 2022.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747*, 2017.

Fan Yang, Mingxuan Xia, Sangzhou Xia, Chicheng Ma, and Hui Hui. Revisiting the robust generalization of adversarial prompt tuning. *arXiv:2405.11154*, 2024.

Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Russ R Salakhutdinov, and Kamalika Chaudhuri. A closer look at accuracy vs. robustness. In *NeurIPS*, volume 33, pp. 8588–8601, 2020.

Chengxiang Yin, Jian Tang, Zhiyuan Xu, and Yanzhi Wang. Adversarial meta-learning. *arXiv:1806.03316*, 2018.

Yuan Zang, Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Meng Zhang, Qun Liu, and Maosong Sun. Word-level textual adversarial attacking as combinatorial optimization. In *ACL*, pp. 6066–6080, 2020.

Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. In *NeurIPS*, 2019a.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *ICML*, pp. 7472–7482, 2019b.

Jiaming Zhang, Xingjun Ma, Xin Wang, Lingyu Qiu, Jiaqi Wang, Yu-Gang Jiang, and Jitao Sang. Adversarial prompt tuning for vision-language models. In *ECCV*, pp. 56–72, 2024a.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *JMLR*, 25(49):1–55, 2024b.

Tianyuan Zhang and Zhanxing Zhu. Interpreting adversarially trained convolutional neural networks. In *ICML*, pp. 7502–7511, 2019.

Yufeng Zhang, Fengzhuo Zhang, Zhuoran Yang, and Zhaoran Wang. What and how does in-context learning learn? bayesian model averaging, parameterization, and generalization. *arXiv:2305.19420*, 2023.

Yiwei Zhou, Xiaobo Xia, Zhiwei Lin, Bo Han, and Tongliang Liu. Few-shot adversarial prompt learning on vision-language models. In *NeurIPS*, volume 37, pp. 3122–3156, 2024.

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

A Additional Related Work

In-Context Learning. In-context learning has emerged as a remarkable property of large language models, enabling them to adapt to a new task from a few input–output demonstrations without any parameter updates (Brown et al., 2020). Recent work has shown that in-context learning can implement various algorithms (Bai et al., 2023; Garg et al., 2022). One research direction has linked in-context learning with preconditioned gradient descent through empirical (Akyürek et al., 2023; Dai et al., 2023; Garg et al., 2022; Von Oswald et al., 2023; 2024) and theoretical analyses (Ahn et al., 2023; Bai et al., 2023; Cheng et al., 2024; Gatmiry et al., 2024; Mahankali et al., 2024; Zhang et al., 2024b). Additional results have indicated that in-context learning can implement ridge regression (Akyürek et al., 2023; Bai et al., 2023), second-order optimization (Fu et al., 2024; Giannou et al., 2024), reinforcement learning (Lee et al., 2023; Lin et al., 2024), and Bayesian model averaging (Zhang et al., 2023). In terms of robustness, some studies have shown that in-context learning can act as a nearly optimal predictor under noisy linear data (Bai et al., 2023) and noisy labels (Frei & Vardi, 2025). Moreover, it has been demonstrated that in-context learning is robust to shifts in the query distribution (Wies et al., 2023; Zhang et al., 2024b), but not necessarily to shifts in the context (Shi et al., 2023; 2024; Wei et al., 2023b; Zhang et al., 2024b). In this study, we focus on the adversarial robustness of in-context learning, rather than the underlying algorithms or its robustness to random noise and distribution shifts. Specifically, we examine whether a single adversarially pretrained transformer can robustly adapt to a broad range of tasks through in-context learning.

Norm- and Token-Bounded Adversarial Examples. Adversarial examples were originally introduced as subtle perturbations to natural data, designed to induce misclassifications in models (Croce & Hein, 2020; Goodfellow et al., 2015; Madry et al., 2018; Szegedy et al., 2014). These perturbations are typically constrained by a norm-based distance from the original inputs. The robustness of transformers to such norm-bounded adversarial examples has been studied primarily in vision transformers (Dosovitskiy et al., 2021). Several studies have shown that standard vision transformers are as vulnerable to these attacks as conventional vision models (Bai et al., 2021; Mahmood et al., 2021), though some have reported marginal differences (Aldahdooh et al., 2021; Benz et al., 2021; Bhojanapalli et al., 2021; Naseer et al., 2021; Paul & Chen, 2022; Shao et al., 2022; Tang et al., 2021). In contrast, adversarial attacks on language models are often neither norm-constrained nor imperceptible to humans. They involve substantial token modifications (Garg & Ramakrishnan, 2020; Jin et al., 2020; Li et al., 2020; Zang et al., 2020), the insertion of adversarial tokens (Liu et al., 2024; Shen et al., 2024; Wallace et al., 2019; Wei et al., 2023a; Zou et al., 2023), and the construction of entirely new adversarial prompts (Carlini et al., 2021; 2022; Nasr et al., 2023; Perez & Ribeiro, 2022; Wei et al., 2023a). These attacks aim not only to induce misclassification (Garg & Ramakrishnan, 2020; Jin et al., 2020; Li et al., 2020; Wallace et al., 2019; Zang et al., 2020), but also to provoke objectionable outputs (Liu et al., 2024; Perez & Ribeiro, 2022; Shen et al., 2024; Wei et al., 2023a; Zou et al., 2023) or to extract private information from training data (Carlini et al., 2021; 2022; Nasr et al., 2023). They are generally bounded by token-level metrics (e.g., the number of modified tokens). In this study, we focus exclusively on norm-bounded adversarial examples. Token-bounded ones are out of scope.

Adversarial Training. Adversarial training, which augments training data with adversarial examples, is one of the most effective adversarial defenses (Goodfellow et al., 2015; Madry et al., 2018). Although originally developed for conventional neural architectures, adversarial training has also proven effective for transformers (Debenedetti et al., 2023; Liu et al., 2025; Shao et al., 2022; Tang et al., 2021; Wu et al., 2022). A major limitation of adversarial training is its high computational cost. To address this, several methods have focused on more efficient generation of adversarial examples (Andriushchenko & Flammarion, 2020; Kim et al., 2021; Park & Lee, 2021; Shafahi et al., 2019; Wong et al., 2020; Zhang et al., 2019a) and adversarial finetuning of standard pretrained models (Jeddi et al., 2020; Mao et al., 2023; Suzuki et al., 2023; Wang et al., 2024a). More recently, researchers have introduced adversarial prompt tuning, which trains visual (Mao et al., 2023; Wang et al., 2024b), textual (Fan et al., 2024; Li et al., 2024; Zhang et al., 2024a), or bimodal prompts (Jia et al., 2025; Luo et al., 2024; Yang et al., 2024; Zhou et al., 2024) in an adversarial manner. However, these methods require retraining for each task. In this study, we explore the potential of adversarially pretrained transformers for robust task adaptation via in-context learning, thereby eliminating the task-specific retraining and associated computational overhead.

Adversarial Meta-Learning. Adversarial meta-learning seeks to develop a universally robust meta-learner that can swiftly and reliably adapt to new tasks under adversarial conditions. Existing approaches adversarially train a neural network on multiple tasks, and then finetune it on a target task using clean (Goldblum et al., 2020; Hou et al., 2021; Liu et al., 2021; Wang et al., 2021; Yin et al., 2018) or adversarial samples (Yin et al., 2018). In this study, we similarly aim to train such a meta-learner. However, rather than relying on neural networks and finetuning, we employ a transformer as the meta-learner and leverage its in-context learning ability for task adaptation.

Related but Distinct Work. We here review theoretical work on the adversarial robustness of in-context learning. Assuming token-bounded adversarial examples, prior studies have shown that even a single token modification in the context can significantly alter the output of a normally trained model on a clean query (Anwar et al., 2024), and deeper layers can mitigate this (Li et al., 2025). Assuming norm- and token-bounded examples, Fu et al. have shown that adversarial training with short adversarial contexts can provide robustness against longer ones (Fu et al., 2025). They considered a clean query and adversarial tokens appended to the original context. In this study, we explore how adversarially trained models handle norm-bounded perturbations to a query in a clean context. As a result, we reveal their universal robustness that can be generalized to a new task from a few demonstrations.

B Additional Theoretical Support and Insights

B.1 Linear Combination of Orthonormal Bases can be Transformed into Our Test Distribution.

Our test data distribution, Assumption 3.2, can implicitly represent data distributions comprising robust and non-robust directions forming an orthonormal basis. Consider d orthonormal bases,

$\{\boldsymbol{e}_i\}_{i=1}^d$. We set $d_{\text{irr}} = 0$, namely $d = d_{\text{rob}} + d_{\text{vul}}$. Each data point is represented as $\boldsymbol{x} = c_1\boldsymbol{e}_1 + c_2\boldsymbol{e}_2 + \cdots + c_d\boldsymbol{e}_d$, where coefficients c_i are sampled probabilistically. These coefficients satisfy $\mathbb{E}[yc_i] = C_i\alpha$ for $i \in \mathcal{S}_{\text{rob}}$ and β for $i \in \mathcal{S}_{\text{vul}}$. In addition, $|\mathbb{E}[(yc_i - \mathbb{E}[yc_i])^n]| \leq C_{i,n}\alpha^n$ for $i \in \mathcal{S}_{\text{rob}}$ and $C_{i,n}\beta^n$ for $i \in \mathcal{S}_{\text{vul}}$. Given a dataset of N i.i.d. samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, if $c_{n,i}$ is independent of $c_{n,j}$ for $i \neq j$ conditional on y, and N is sufficiently large, then the covariance of $y\boldsymbol{x}$ can be approximated as:

$$\begin{aligned}
&\frac{1}{N}\sum_{n=1}^N \left(y_n\boldsymbol{x}_n - \sum_{k=1}^N y_k\boldsymbol{x}_k\right)\left(y_n\boldsymbol{x}_n - \sum_{k=1}^N y_k\boldsymbol{x}_k\right)^\top \\
&\approx \mathbb{E}[(y\boldsymbol{x} - \mathbb{E}[y\boldsymbol{x}])(y\boldsymbol{x} - \mathbb{E}[y\boldsymbol{x}])^\top] &\text{(A12)}\\
&= \mathbb{E}\left[\left(\sum_{i=1}^d (y_ic_i - \mathbb{E}[yc_i])\boldsymbol{e}_i\right)\left(\sum_{i=1}^d (y_ic_i - \mathbb{E}[yc_i])\boldsymbol{e}_i\right)^\top\right] &\text{(A13)}\\
&= \sum_{i,j=1}^d \mathbb{E}[(yc_i - \mathbb{E}[yc_i])(yc_j - \mathbb{E}[yc_j])]\boldsymbol{e}_i\boldsymbol{e}_j^\top &\text{(A14)}\\
&= \sum_{i\in\mathcal{S}_{\text{rob}}}^d C_{i,2}\alpha^2\boldsymbol{e}_i\boldsymbol{e}_i^\top + \sum_{i\in\mathcal{S}_{\text{vul}}}^d C_{i,2}\beta^2\boldsymbol{e}_i\boldsymbol{e}_i^\top. &\text{(A15)}
\end{aligned}$$

This implies that through principal component analysis for $y_n\boldsymbol{x}_n$, we can obtain d orthonormal bases, $\{\boldsymbol{e}_i\}_{i=1}^d$. By projecting a sample $\boldsymbol{x}_n$ onto these bases, we obtain a transformed sample $\boldsymbol{x}'_n := \{c_{n,1}, c_{n,2}, \ldots, c_{n,d}\}$. This demonstrates that when data is sampled from a distribution comprising robust and non-robust directions forming an orthonormal basis, if the coefficients are mutually independent and the sample size is sufficiently large, we can preprocess the data to satisfy Assumption 3.2. Importantly, this preprocessing relies solely on statistics derivable from training samples.

B.2 Sufficient Number of Datasets to Provide Universal Robustness

What determines the sufficient number of datasets needed to provide universal robustness to transformers? We conjecture that this may be determined by the number of robust bases. In this paper, we trained transformers using d datasets. This stems from training with datasets where only one dimension is robust (in other words, datasets with a single robust basis), the number of dimensions d, and the assumption that all dimensions might contain robust features. If we assume that robust features never appear in the latter d' dimensions, following the procedure in Appendix D, we can train robust transformers using only $d - d'$ datasets that describe the first $d - d'$ robust features. From this observation, we conjecture that the sufficient number of datasets required to provide universal robustness to transformers depends on the number of robust bases in the assumed data structure.

B.3 Effects of q_{rob} and q_{vul}

We here analyze how q_{rob} and q_{vul} affect the robustness of adversarially trained transformer. As defined in Assumption 3.2, these parameters control the proportion of features whose total covariance with other features is negative. Theorem E.1 suggests that the transformer prediction for unperturbed data can be expressed as

$$C(d_{\text{rob}}\alpha + d_{\text{vul}}\beta)\left\{(1 - cq_{\text{rob}})d_{\text{rob}}\alpha^2 + (1 - cq_{\text{vul}})d_{\text{vul}}\beta^2\right\} + C'(d_{\text{rob}}\alpha^2 + d_{\text{vul}}\beta^2), \tag{A16}$$

where

$$c := \frac{(\max_{i\in\mathcal{S}_{\text{rob}}\cup\mathcal{S}_{\text{vul}}} C_i)(\max_{i\in\mathcal{S}_{\text{rob}}\cup\mathcal{S}_{\text{vul}}} C_{i,2})}{\min_{i\in\mathcal{S}_{\text{rob}}\cup\mathcal{S}_{\text{vul}}} C_i^3}. \tag{A17}$$

Examining the term $(1 - cq_{\text{rob}})d_{\text{rob}}\alpha^2 + (1 - cq_{\text{vul}})d_{\text{vul}}\beta^2$, we observe that larger values of q_{rob} and q_{vul} generally diminish the magnitude of transformer predictions. This indicates that negative correlations between features degrade the robustness of adversarially trained transformers. Additionally, the coefficient c is characterized by $\max_{i\in\mathcal{S}_{\text{rob}}\cup\mathcal{S}_{\text{vul}}} C_{i,2}$, which represents a variance coefficient. This suggests that smaller feature variances enhance the robustness of adversarially trained transformers. For example, if each feature variance $C_{i,2}$ is sufficiently small, even $q_{\mathrm{rob}} = 1$ and $q_{\mathrm{vul}} = 1$ may be tolerated without significantly compromising robustness.

B.4 Disadvantage of Standard Finetuning: Parameter Selection Perspective

In this study, we investigate task adaptation through in-context learning. As an alternative lightweight approach, standard finetuning—where all or part of the model parameters are updated—can also be employed. However, a key drawback of standard finetuning is that it requires parameter updates, whereas in-context learning does not. Moreover, finetuning necessitates careful selection of which parameters to update. Our analysis shows that improper parameter selection during finetuning can compromise the robustness initially established by adversarial pretraining. Consider adversarially pretrained parameters, $\boldsymbol{P}^{\mathrm{adv}}$ and $\boldsymbol{Q}^{\mathrm{adv}}$, and $\mathcal{D}_c^{\mathrm{tr}}$ as a downstream data distribution.

First, we examine the scenario where only $\boldsymbol{P}$ is updated while keeping $\boldsymbol{Q}^{\mathrm{adv}}$ fixed, formulated as:

$$\min_{\boldsymbol{P}\in[0,1]^{(d+1)\times(d+1)}} \mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_c^{\mathrm{tr}}} \left[-y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{0}};\boldsymbol{P},\boldsymbol{Q}^{\mathrm{adv}})]_{d+1,N+1}\right]. \tag{A18}$$

In this case, as shown in the proof in Appendix D, $\boldsymbol{P} = \boldsymbol{P}^{\mathrm{std}} (= \boldsymbol{P}^{\mathrm{adv}})$ is the global solution. Consequently, as demonstrated in Theorem 3.6, the model's robustness is preserved.

Conversely, consider training $\boldsymbol{Q}$ while keeping $\boldsymbol{P}^{\mathrm{adv}}$ fixed, formulated as:

$$\min_{\boldsymbol{Q}\in[0,1]^{(d+1)\times(d+1)}} \mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_c^{\mathrm{tr}}} \left[-y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{0}};\boldsymbol{P}^{\mathrm{adv}},\boldsymbol{Q})]_{d+1,N+1}\right]. \tag{A19}$$

In this scenario, $\boldsymbol{Q} = \boldsymbol{Q}^{\mathrm{std}}$ is the global solution. As established in Theorems 3.5, 3.7 and G.1, while this configuration enables the transformer to perform well on unperturbed queries, it fails to maintain robustness against perturbed inputs.

These findings highlight a critical insight: achieving robust task adaptation through standard finetuning requires careful parameter selection; otherwise, the pretrained model's adversarial robustness may be compromised. This parameter sensitivity represents a disadvantage compared to in-context learning, which preserves robustness without requiring parameter updates.

B.5 Naive Adversarial Context may not Improve Robustness

One approach to enhancing the robustness of a normally trained transformer is to incorporate adversarial examples into the context. In this section, we show that this is not the case in our setting. Consider the following transformer input:

$$\boldsymbol{Z}' := \begin{bmatrix} \boldsymbol{x}_1 + \boldsymbol{\Delta}_1 & \boldsymbol{x}_2 + \boldsymbol{\Delta}_2 & \cdots & \boldsymbol{x}_N + \boldsymbol{\Delta}_N & \boldsymbol{x}_{N+1} + \boldsymbol{\Delta}_{N+1} \\ y_1 & y_2 & \cdots & y_N & 0 \end{bmatrix}. \tag{A20}$$

The adversarial perturbations for the context, $\boldsymbol{\Delta}_1, \dots, \boldsymbol{\Delta}_N$, are defined as $\boldsymbol{\Delta}_n := -\epsilon y_n \boldsymbol{1}_d$. In this setting, for $\epsilon \geq \frac{1+(d-1)(\lambda/2)}{d}$, the standard transformer prediction is given by:

$$\mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_c^{\mathrm{tr}}} \left[\min_{\|\boldsymbol{\Delta}_{N+1}\|_\infty \leq \epsilon} y_{N+1}[f(\boldsymbol{Z}';\boldsymbol{P}^{\mathrm{std}},\boldsymbol{Q}^{\mathrm{std}})]_{d+1,N+1}\right] \leq 0. \tag{A21}$$

This result suggests that, in our setting, naive adversarial demonstrations do not improve the performance of the standard transformer. Intuitively, because adversarial training generates new adversarial examples at each step of gradient descent, fixed adversarial demonstrations may fail to counter newly generated adversarial perturbations to the query.

C Additional Experimental Results

All experiments were conducted on Ubuntu 20.04.6 LTS, Intel Xeon Gold 6226R CPUs, and NVIDIA RTX 6000 Ada GPUs.

C.1 Support for Assumption 3.2.

The statistics of preprocessed MNIST, Fashion-MNIST, and CIFAR-10 are provided in Fig. A2. Preprocessing was conducted as follows: (i) selection of two different classes from the ten available classes and assignment of binary labels to every sample from the training dataset, creating $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$; (ii) centering the data via $\boldsymbol{x}' \leftarrow \boldsymbol{x} - \bar{\boldsymbol{x}}$ with $\bar{\boldsymbol{x}} := (1/N) \sum_{n=1}^N \boldsymbol{x}_n$; and (iii) aligning features with the label using $\boldsymbol{x}'' \leftarrow \operatorname{sgn}(\sum_{n=1}^N y_n \boldsymbol{x}_n) \odot \boldsymbol{x}'$. These preprocessed datasets exhibit that each dimension has a positive correlation with the label and that few dimensions have negative total covariance. The main distinction from Assumption 3.2 is that their features are not clearly separated as robust or non-robust. Instead, they gradually transition from robust to non-robust characteristics.

C.2 Verification of Theorem 3.4.

We trained a single-layer transformer (2) with the in-context loss (7). The training distribution was configured with $d = 20$ and $\lambda = 0.1$ in Fig. 1 and with $d = 100$ and $\lambda = 0.1$ in Fig. A3. For standard, adversarial, and strong adversarial regimes, we used $\epsilon = 0$, $\frac{1+(d-1)(\lambda/2)}{d} = 0.098$, and $\frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3} = 0.95$ in Fig. 1 and $\epsilon = 0$, $\frac{1+(d-1)(\lambda/2)}{d} = 0.06$, and $\frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3} = 0.77$ in Fig. A3. Optimization was conducted using stochastic gradient descent with momentum 0.9. Learning rates were set to 0.1 for all regimes in Fig. 1, and to 1.0 for standard and strong adversarial regimes and 0.2 for the adversarial regime in Fig. A3. Training ran for 100 epochs with a learning rate scheduler that multiplied the rate by 0.1 when the loss did not improve within 10 epochs. In each iteration of stochastic gradient descent, we sampled 1,000 datasets $\{(\boldsymbol{x}_n^{(c)}, y_n^{(c)})\}_{n=1}^{N+1}$ with $N = 1,000$. The distribution index c was randomly sampled from $U([d])$, meaning that in each iteration, each of the 1,000 datasets may have different c values. After each parameter update, we projected the parameters to $[0,1]^d$. Adversarial perturbation was calculated as $\boldsymbol{\Delta} := -\epsilon y_n \operatorname{sgn}(\boldsymbol{P}_{d+1,\cdot} \boldsymbol{Z_0} \boldsymbol{M} \boldsymbol{Z_0}^\top \boldsymbol{Q}_{\cdot,:d})$, which represents the optimal attack. The heatmaps of the learned parameters in Figs. 1 and A3 completely align with the theoretical predictions of Theorem 3.4.

C.3 Verification of Theorems 3.5 to 3.7 and G.1

We evaluated normally and adversarially pretrained single-layer transformers on $\mathcal{D}^{\text{tr}}$, $\mathcal{D}^{\text{te}}$, the preprocessed MNIST, Fashion-MNIST, and CIFAR-10 datasets. For network parameters, we used the theoretically predicted $\boldsymbol{P}^{\text{std}}$ and $\boldsymbol{Q}^{\text{std}}$ as standard model parameters and $\boldsymbol{P}^{\text{adv}}$ and $\boldsymbol{Q}^{\text{adv}}$ as adversarially trained model parameters. This approach allowed us to circumvent the computationally expensive adversarial pretraining for every distinct d setting. As described previously, our empirical results completely align with the theoretically predicted parameter configurations.

Configuration in Figs. A4 and A5. In Fig. A4, the basic settings were $d = 100$, $\lambda = 0.1$, $N = 1,000$, and $\epsilon = 0.15$. In Fig. A5, they were $d_{\text{rob}} = 10$, $d_{\text{vul}} = 90$, $d_{\text{irr}} = 0$, $\alpha = 1.0$, $\beta = 0.1$, $\gamma = 0.1$, and $\epsilon = 0.2$. The basic perturbation budget was set to 0.1. We considered 1,000 batches where each batch contained 1,000 in-context demonstrations (i.e., $N = 1000$), and 1,000 queries. The test distribution $\mathcal{D}^{\text{te}}$ was constructed based on normal distribution. During sampling, yx_i was sampled from $\mathcal{N}(\alpha, \alpha^2)$ for $i \in \mathcal{S}_{\text{rob}}$, $\mathcal{N}(\beta, \beta^2)$ for $i \in \mathcal{S}_{\text{vul}}$, and $\mathcal{N}(0, \gamma^2)$ for $i \in \mathcal{S}_{\text{irr}}$. Each dimension is independent, given y.

Configuration in Fig. A6. The preprocessing procedure is described in Appendix C.1. As batches, we considered 45 binary class pairs from ten classes. The basic perturbation budget was set to 0.1. In the first row of Fig. A6, we used all training samples in the training dataset. As queries, we used all test samples in the test dataset.

Analysis. In Figs. A4 to A6, standard transformers consistently demonstrate vulnerability to adversarial attacks, whereas adversarially trained transformers maintain a certain level of robustness, validating Theorems 3.5 and 3.6. However, adversarially pretrained transformers exhibit lower clean accuracy, supporting Theorem 3.7.

In Figs. A4 and A5, we observe that a larger number of vulnerable dimensions increases model vulnerability. Conversely, Fig. A5 shows that a larger number of robust dimensions enhances model

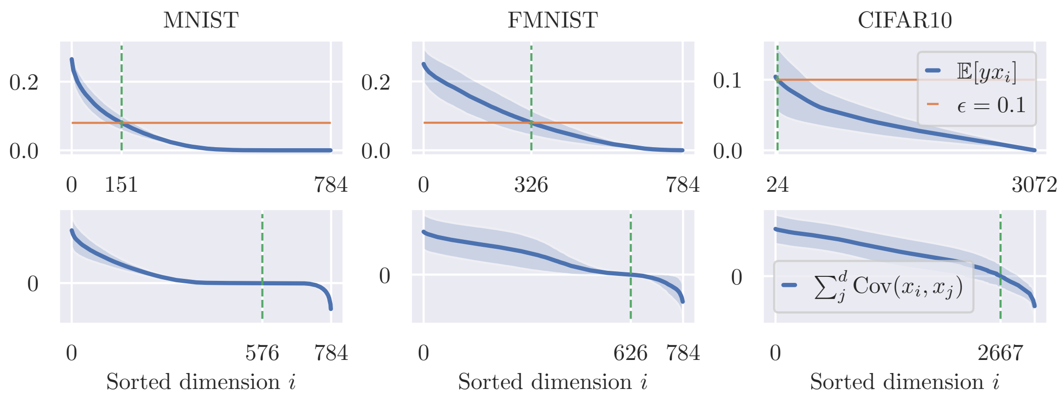

Figure A2: Statistical properties of preprocessed MNIST, Fashion-MNIST, and CIFAR-10 datasets. **First row:** Blue lines represent the mean of $(1/N)\sum_{n=1}^{N} y_n \boldsymbol{x}_n$ across 45 binary class pairs and shaded regions represent the sample standard deviation. Orange lines represent typical perturbation magnitude. Green dashed lines represent the (pseudo) threshold between robust and non-robust dimensions. **Second row:** Blue lines represent the total covariance of each dimension with other dimensions and shaded regions represent sample standard deviation across the 45 binary class pairs. Green dashed lines represent the boundary between positive and negative total covariance.

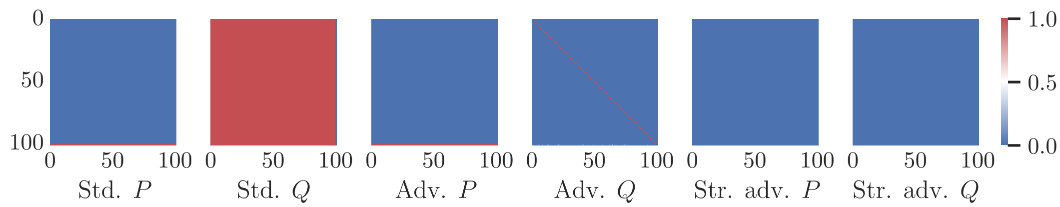

Figure A3: Parameter heatmaps induced by adversarial training (7) with $d = 100$ and $\lambda = 0.1$. For the standard, adversarial, and strong adversarial regimes, we used $\epsilon = 0$, $\frac{1+(d-1)(\lambda/2)}{d} = 0.06$, and $\frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3} = 0.77$, respectively. We optimized (7) by stochastic gradient descent.

robustness. Robust models are less susceptible to increasing vulnerable dimensions and benefit more from increasing robust dimensions.

Additionally, as predicted in Theorems 3.5 and 3.6, standard training exhibits vulnerability to increasing redundant dimensions, which is more detrimental than the harmful effect from increasing vulnerable dimensions, since redundant dimensions do not benefit predictions and are only harmful for robustness. In contrast, adversarially trained transformers exhibit significant resistance to increases in these dimensions.

The second row of Fig. A6 indicates that standard transformers still achieve high classification accuracy in small demonstration regimes, whereas adversarially trained transformers show degraded performance. These results align with our theoretical predictions, Theorem G.1.

D Proof of Lemma 3.3 and Theorem 3.4 (Pretraining)

Lemma 3.3 (Transformation of original optimization problem)**.** The minimization problem (7) can be transformed into the maximization problem $\max_{\boldsymbol{b}\in\{0,1\}^{d+1}} \sum_{i=1}^{d(d+1)} \max(0, \sum_{j=1}^{d+1} b_j h_{i,j})$, where $h_{i,j} \in \mathbb{R}$ is an (i,j)-dependent constant, and there exists a mapping from $\boldsymbol{b}$ to $\boldsymbol{P}$ and $\boldsymbol{Q}$.

Proof. See "Overview" below. □

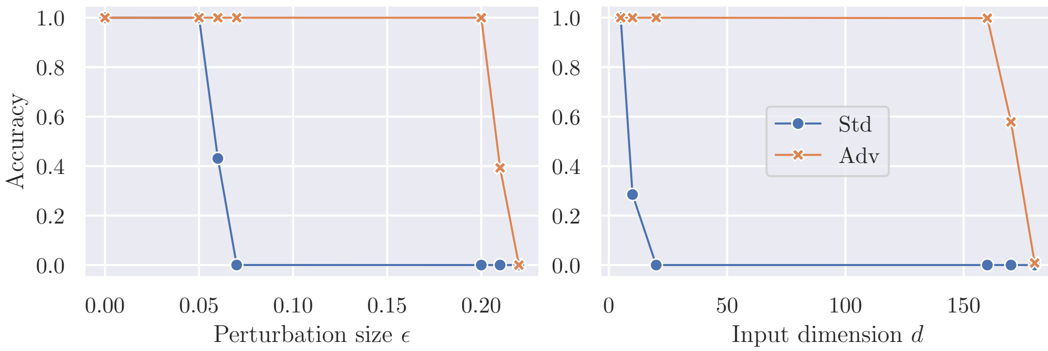

Figure A4: Accuracy (%) of normally and adversarially pretrained single-layer transformers. Lines represent mean accuracy across batches and shaded regions represent unbiased standard deviation (notably small in magnitude). We used 1,000 batches, each containing 1,000 in-context demonstrations ($N = 1000$) and 1,000 query examples. Base configuration parameters were $d = 100$, $\lambda = 0.1$, and $\epsilon = 0.15$.

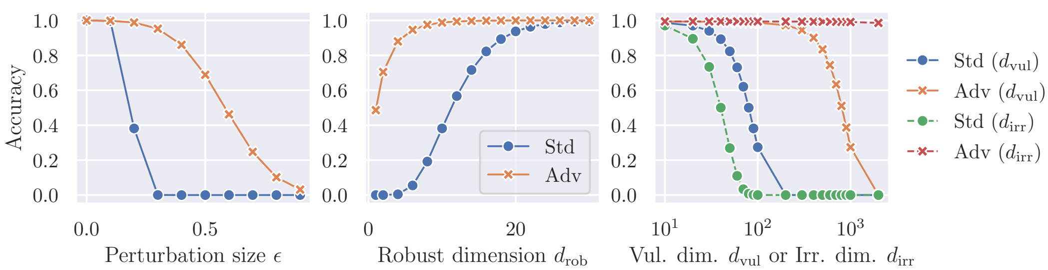

Figure A5: Accuracy (%) of normally and adversarially pretrained single-layer transformers. Lines represent mean accuracy across batches and shaded regions represent unbiased standard deviation. We used 1,000 batches, each containing 1,000 in-context demonstrations ($N = 1000$) and 1,000 query examples. Base configuration parameters were $d_{\mathrm{rob}} = 10$, $d_{\mathrm{vul}} = 90$, $d_{\mathrm{irr}} = 0$, $\alpha = 1.0$, $\beta = 0.1$, $\gamma = 0.1$, and $\epsilon = 0.2$.

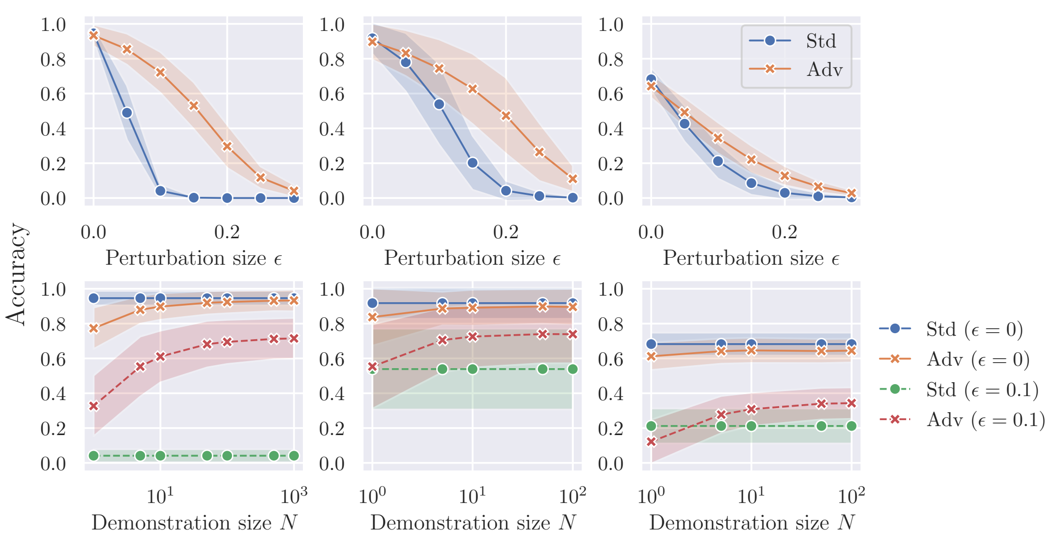

Figure A6: Accuracy (%) of normally and adversarially pretrained single-layer transformers. Lines represent mean accuracy across 45 binary classification tasks (derived from all possible pairs of the ten classes) and shaded regions represent the unbiased standard deviation. The perturbation size was basically $\epsilon = 0.1$.

Theorem 3.4 (Parameters induced by adversarial pretraining)**.** The global minimizer of (7) is

$$\left(1.\ \text{Standard}; \epsilon = 0\right) \quad \boldsymbol{P} = \boldsymbol{P}^{\mathrm{std}} := \begin{bmatrix} \mathbf{0}_{d,d+1} \\ \mathbf{1}_{d+1}^\top \end{bmatrix} \text{ and } \boldsymbol{Q} = \boldsymbol{Q}^{\mathrm{std}} := [\mathbf{1}_{d+1,d} \quad \mathbf{0}_{d+1}].$$

$$\left(2.\ \text{Adversarial}; \epsilon = \tfrac{1+(d-1)(\lambda/2)}{d}\right) \quad \boldsymbol{P} = \boldsymbol{P}^{\mathrm{adv}} := \begin{bmatrix} \mathbf{0}_{d,d+1} \\ \mathbf{1}_{d+1}^\top \end{bmatrix} \text{ and } \boldsymbol{Q} = \boldsymbol{Q}^{\mathrm{adv}} := \begin{bmatrix} \boldsymbol{I}_d & \mathbf{0}_d \\ \mathbf{0}_d^\top & 0 \end{bmatrix}.$$

$$\left(3.\ \text{Strongly adversarial}; \epsilon \geq \tfrac{\lambda}{2} + \tfrac{3}{2}\tfrac{2-\lambda}{(d-1)\lambda^2+3}\right) \quad \boldsymbol{P} = \mathbf{0}_{d+1,d+1} \text{ and } \boldsymbol{Q} = \mathbf{0}_{d+1,d+1}.$$

Proof. This is the special case of the following theorem. □

Theorem D.1 (General case of Theorem 3.4)**.** The global minimizer of (7) is as follows:

- If
$$0 \leq \epsilon \leq \frac{\lambda(\lambda(d-2)+4)}{2(\lambda(d-1)+2)}, \tag{A22}$$
then $\boldsymbol{P} = \begin{bmatrix} \mathbf{0}_{d,d+1} \\ \mathbf{1}_{d+1}^\top \end{bmatrix}$ and $\boldsymbol{Q} = [\mathbf{1}_{d+1,d} \quad \mathbf{0}_{d+1}]$.
- If
$$\epsilon = \frac{1+(d-1)(\lambda/2)}{d}, \tag{A23}$$
then $\boldsymbol{P} = \begin{bmatrix} \mathbf{0}_{d,d+1} \\ \mathbf{1}_{d+1}^\top \end{bmatrix}$ and $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{I}_d & \mathbf{0}_d \\ \mathbf{0}_d^\top & 0 \end{bmatrix}$.
- If
$$\epsilon \geq \frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{(d-1)\lambda^2+3}, \tag{A24}$$
then $\boldsymbol{P} = \mathbf{0}_{d+1,d+1}$ and $\boldsymbol{Q} = \mathbf{0}_{d+1,d+1}$.

Proof.

Overview. The loss function $\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q})$ is determined only by the last row of $\boldsymbol{P}$ and the first d columns of $\boldsymbol{Q}$. Let

$$\boldsymbol{P} := \begin{bmatrix} \mathbf{0}_{d,d+1} \\ \boldsymbol{b}^\top \end{bmatrix}, \qquad \boldsymbol{Q} := [\boldsymbol{A} \quad \mathbf{0}_{d+1}], \tag{A25}$$

where $\boldsymbol{b} \in \mathbb{R}^{d+1}$ and $\boldsymbol{A} := [\boldsymbol{a}_1 \cdots \boldsymbol{a}_d] \in \mathbb{R}^{(d+1)\times d}$. With $\boldsymbol{b}$, $\boldsymbol{A}$, and $\boldsymbol{G} := \boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top / N$, the loss function $\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q})$ can be represented as:

$$\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q}) := \mathbb{E}_{c,\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1}}\left[\max_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} -y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}, \boldsymbol{Q})]_{d+1,N+1}\right] \tag{A26}$$

$$= \mathbb{E}_{c,\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1}}\left[\max_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} -y_{N+1}\left[\boldsymbol{Z}_{\boldsymbol{\Delta}} + \frac{1}{N}\boldsymbol{P}\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{Q}\boldsymbol{Z}_{\boldsymbol{\Delta}}\right]_{d+1,N+1}\right] \tag{A27}$$

$$= \mathbb{E}_{c,\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1}}\left[\max_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} -y_{N+1}\boldsymbol{b}^\top \boldsymbol{G}\boldsymbol{A}(\boldsymbol{x}_{N+1} + \boldsymbol{\Delta})\right]. \tag{A28}$$

Using $\boldsymbol{b}$ and $\boldsymbol{A}$, we redefine the loss function as $\mathcal{L}(\boldsymbol{b}, \boldsymbol{A}) := \mathcal{L}(\boldsymbol{P}, \boldsymbol{Q})$. Since $\boldsymbol{G}$ does not include $\boldsymbol{\Delta}$ and $\max_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} \boldsymbol{w}^\top \boldsymbol{\Delta} = \epsilon\|\boldsymbol{w}\|_1$ for $\boldsymbol{w} \in \mathbb{R}^d$, the inner maximization can be solved as:

$$\mathcal{L}(\boldsymbol{b}, \boldsymbol{A}) = \mathbb{E}_{c,\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1}}\left[-y_{N+1}\boldsymbol{b}^\top \boldsymbol{G}\boldsymbol{A}\boldsymbol{x}_{N+1} + \epsilon\|\boldsymbol{b}^\top \boldsymbol{G}\boldsymbol{A}\|_1\right]. \tag{A29}$$

When $0 \leq \boldsymbol{b} \leq 1$ and $0 \leq \boldsymbol{A} \leq 1$, then $\|\boldsymbol{b}^\top \boldsymbol{G}\boldsymbol{A}\|_1 = \boldsymbol{b}^\top \boldsymbol{G}\boldsymbol{A}\mathbf{1}$ since all the elements of $\boldsymbol{G}$ are nonnegative. Thus,

$$\min_{0 \leq \boldsymbol{b} \leq 1, 0 \leq \boldsymbol{A} \leq 1} \mathcal{L}(\boldsymbol{b}, \boldsymbol{A})$$

$$= \min_{0 \le \boldsymbol{b} \le 1, 0 \le \boldsymbol{A} \le 1} \mathbb{E}_{c, \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}} \left[-y_{N+1} \boldsymbol{b}^\top \boldsymbol{G} \boldsymbol{A} \boldsymbol{x}_{N+1} + \epsilon \boldsymbol{b}^\top \boldsymbol{G} \boldsymbol{A} \boldsymbol{1} \right]. \tag{A30}$$

Let the i-th row of $\boldsymbol{G}$ be $\boldsymbol{g}_i^\top$. Rearranging the argument of the expectation as:

$$-y_{N+1} \boldsymbol{b}^\top \boldsymbol{G} \boldsymbol{A} \boldsymbol{x}_{N+1} + \epsilon \boldsymbol{b}^\top \boldsymbol{G} \boldsymbol{A} \boldsymbol{1} = -\sum_{j=1}^{d+1} \sum_{k=1}^{d} A_{j,k} \left(\sum_{i=1}^{d+1} b_i g_{i,j} (y_{N+1} x_{N+1,k} - \epsilon) \right). \tag{A31}$$

Thus, the objective function can be represented as:

$$\max_{0 \le \boldsymbol{b} \le 1, 0 \le \boldsymbol{A} \le 1} \sum_{j=1}^{d+1} \sum_{k=1}^{d} A_{j,k} \left(\sum_{i=1}^{d+1} b_i \mathbb{E}_{c, \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}} [g_{i,j} (y_{N+1} x_{N+1,k} - \epsilon)] \right). \tag{A32}$$

Since the objective function is linear with respect to $\boldsymbol{b}$ and $\boldsymbol{A}$, respectively, the optimal solution exists on the boundary:

$$\max_{\boldsymbol{b} \in \{0,1\}^{d+1}, \boldsymbol{A} \in \{0,1\}^{(d+1) \times d}} \sum_{j=1}^{d+1} \sum_{k=1}^{d} A_{j,k} \left(\sum_{i=1}^{d+1} b_i \mathbb{E}_{c, \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}} [g_{i,j} (y_{N+1} x_{N+1,k} - \epsilon)] \right). \tag{A33}$$

This is maximized by $A_{j,k} = 1$ if $\sum_{i=1}^{d+1} b_i \mathbb{E}_{c, \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}} [g_{i,j} (y_{N+1} x_{N+1,k} - \epsilon)]) \ge 0$ and 0 otherwise. Now,

$$\max_{\boldsymbol{b} \in \{0,1\}^{d+1}} \sum_{j=1}^{d+1} \sum_{k=1}^{d} \phi \left(\sum_{i=1}^{d+1} b_i \mathbb{E}_{c, \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}} [g_{i,j} (y_{N+1} x_{N+1,k} - \epsilon)] \right), \tag{A34}$$

where $\phi(x) := \max(0, x)$. Calculating the expectation and optimizing $\boldsymbol{b}$, we obtain the solution.

Calculation of the expectation. First, we consider the expectation given c. Since $y_n x_{n,i} = 1$ if $i = c$ and $y_n x_{n,i} \sim U(0, \lambda)$ otherwise, the expectation of $y_n \boldsymbol{x}_n$ can be calculated as:

$$\mathbb{E}[y_n x_{n,i} \mid c] = \begin{cases} 1 & (i = c) \\ \frac{\lambda}{2} & (i \ne c) \end{cases}, \qquad \mathbb{E}[y_n \boldsymbol{x}_n^\top \mid c] = \begin{bmatrix} \frac{\lambda}{2} & \cdots & \frac{\lambda}{2} & \underbrace{1}_{c\text{-th}} & \frac{\lambda}{2} & \cdots & \frac{\lambda}{2} \end{bmatrix}. \tag{A35}$$

The expectation of $\boldsymbol{G}$ can be calculated as:

$$\begin{aligned} \mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}} [\boldsymbol{G} \mid c] &= \frac{1}{N} \mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}} [\boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \mid c] && \text{(A36)} \\ &= \frac{1}{N} \begin{bmatrix} \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{x}_n} [\boldsymbol{x}_n \boldsymbol{x}_n^\top \mid c] & \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{x}_n, y_n} [y_n \boldsymbol{x}_n \mid c] \\ \sum_{n=1}^{N} \mathbb{E}_{\boldsymbol{x}_n, y_n} [y_n \boldsymbol{x}_n^\top \mid c] & N \end{bmatrix} && \text{(A37)} \\ &= \begin{bmatrix} \mathbb{E}_{\boldsymbol{x}_n} [\boldsymbol{x}_n \boldsymbol{x}_n^\top \mid c] & \mathbb{E}_{\boldsymbol{x}_n, y_n} [y_n \boldsymbol{x}_n \mid c] \\ \mathbb{E}_{\boldsymbol{x}_n, y_n} [y_n \boldsymbol{x}_n^\top \mid c] & 1 \end{bmatrix}. && \text{(A38)} \end{aligned}$$

For $y_n = 1$ and $i, j \ne c$, $\mathbb{E}[x_{n,i}^2 \mid c] = \int_0^\lambda x^2 / \lambda \, \mathrm{d}x = \lambda^2 / 3$ and $\mathbb{E}[x_{n,i} x_{n,j} \mid c] = \mathbb{E}[x_{n,i} \mid c] \mathbb{E}[x_{n,j} \mid c] = \lambda^2 / 4$. Thus,

$$\mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}} [g_{i,j} \mid c] = \begin{cases} 1 & (i = c) \wedge (j = i, d+1) \\ \frac{\lambda}{2} & (i = c) \wedge (j \ne i, d+1) \\ \frac{\lambda^2}{3} & (i \in [d], i \ne c) \wedge (j = i) \\ \frac{\lambda}{2} & (i \in [d], i \ne c) \wedge (j = c, d+1) \\ \frac{\lambda^2}{4} & (i \in [d], i \ne c) \wedge (j \ne i, c, d+1) \\ 1 & (i = d+1) \wedge (j = c, d+1) \\ \frac{\lambda}{2} & (i = d+1) \wedge (j \ne c, d+1) \end{cases}. \tag{A39}$$

Note that

$$\mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}} [\boldsymbol{G} \mid c]$$

$$= \begin{bmatrix} \lambda^2/3 & \lambda^2/4 & \lambda^2/4 & \cdots & \lambda^2/4 & \overbrace{\lambda/2}^{c\text{-th}} & \lambda^2/4 & \cdots & \lambda^2/4 & \lambda/2 \\ \lambda^2/4 & \lambda^2/3 & \lambda^2/4 & \cdots & \lambda^2/4 & \lambda/2 & \lambda^2/4 & \cdots & \lambda^2/4 & \lambda/2 \\ \vdots \\ \lambda^2/4 & \lambda^2/4 & \lambda^2/4 & \cdots & \lambda^2/3 & \lambda/2 & \lambda^2/4 & \cdots & \lambda^2/4 & \lambda/2 \\ \lambda/2 & \lambda/2 & \lambda/2 & \cdots & \lambda/2 & 1 & \lambda/2 & \cdots & \lambda/2 & 1 \\ \lambda^2/4 & \lambda^2/4 & \lambda^2/4 & \cdots & \lambda^2/4 & \lambda/2 & \lambda^2/3 & \cdots & \lambda^2/4 & \lambda/2 \\ \vdots \\ \lambda^2/4 & \lambda^2/4 & \lambda^2/4 & \cdots & \lambda^2/4 & \lambda/2 & \lambda^2/4 & \cdots & \lambda^2/3 & \lambda/2 \\ \lambda/2 & \lambda/2 & \lambda/2 & \cdots & \lambda/2 & 1 & \lambda/2 & \cdots & \lambda/2 & 1 \end{bmatrix} \}c\text{-th}. \tag{A40}$$

Let

$$h_i(j;k;c) := \mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}}[g_{i,j}(y_{N+1}x_{N+1,k} - \epsilon) \mid c]. \tag{A41}$$

Let $\epsilon_+ := 1 - \epsilon$ and $\epsilon_- := \lambda/2 - \epsilon$. By Eqs. (A35) and (A39),

$$h_i(j;k;c) = \begin{cases} \epsilon_+ & (i \in [d]) \wedge (j = i, d+1) \wedge (k = i) \wedge (c = i) \\ \epsilon_- & (i \in [d]) \wedge (j = i, d+1) \wedge (k \neq i) \wedge (c = i) \\ \frac{\lambda}{2}\epsilon_+ & (i \in [d]) \wedge (j \neq i, d+1) \wedge (k = i) \wedge (c = i) \\ \frac{\lambda}{2}\epsilon_- & (i \in [d]) \wedge (j \neq i, d+1) \wedge (k \neq i) \wedge (c = i) \\ \frac{\lambda^2}{3}\epsilon_- & (i \in [d]) \wedge (j = i) \wedge (k = i) \wedge (c \neq i) \\ \frac{\lambda}{2}\epsilon_- & (i \in [d]) \wedge (j = c, d+1) \wedge (k = i) \wedge (c \neq i) \\ \frac{\lambda^2}{4}\epsilon_- & (i \in [d]) \wedge (j \neq i, c, d+1) \wedge (k = i) \wedge (c \neq i) \\ \frac{\lambda^2}{3}\epsilon_+ & (i \in [d]) \wedge (j = i) \wedge (k = c) \wedge (c \neq i) \\ \frac{\lambda}{2}\epsilon_+ & (i \in [d]) \wedge (j = c, d+1) \wedge (k = c) \wedge (c \neq i) \\ \frac{\lambda^2}{4}\epsilon_+ & (i \in [d]) \wedge (j \neq i, c, d+1) \wedge (k = c) \wedge (c \neq i) \\ \frac{\lambda^2}{3}\epsilon_- & (i \in [d]) \wedge (j = i) \wedge (k \neq i, c) \wedge (c \neq i) \\ \frac{\lambda}{2}\epsilon_- & (i \in [d]) \wedge (j = c, d+1) \wedge (k \neq i, c) \wedge (c \neq i) \\ \frac{\lambda^2}{4}\epsilon_- & (i \in [d]) \wedge (j \neq i, c, d+1) \wedge (k \neq i, c) \wedge (c \neq i) \\ \epsilon_+ & (i = d+1) \wedge (j = c, d+1) \wedge (k = c) \\ \epsilon_- & (i = d+1) \wedge (j = c, d+1) \wedge (k \neq c) \\ \frac{\lambda}{2}\epsilon_+ & (i = d+1) \wedge (j \neq c, d+1) \wedge (k = c) \\ \frac{\lambda}{2}\epsilon_- & (i = d+1) \wedge (j \neq c, d+1) \wedge (k \neq c) \end{cases}. \tag{A42}$$

Then, we compute the expectation along c. Note that

$$\mathbb{E}_{c,\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}}[g_{i,j}(y_{N+1}x_{N+1,k} - \epsilon)] = \frac{1}{d}\sum_{c=1}^{d} h_i(j;k;c). \tag{A43}$$

Let $H_{i,j,k} := \sum_{c=1}^{d} h_i(j;k;c)$. The summation of h_i along c can be calculated as:

For $(i \in [d]) \wedge (j = i) \wedge (k = i)$,

$$H_{i,j,k} = h_i(j = i; k = i; c = i) + \sum_{c \neq i}^{d} h_i(j = i; k = i; c \neq i) = \epsilon_+ + \frac{\lambda^2}{3}(d-1)\epsilon_- \tag{A44}$$

$$=: r_1. \tag{A45}$$

For $(i \in [d]) \wedge (j = i) \wedge (k \neq i)$,

$$H_{i,j,k} = h_i(j = i; k \neq i; c = i) + h_i(j = i; k = c; c \neq i) + \sum_{c \neq i,k}^{d} h(j = i; k \neq i, c; c \neq i) \tag{A46}$$

$$= \epsilon_- + \frac{\lambda^2}{3}\epsilon_+ + \frac{\lambda^2}{3}(d-2)\epsilon_- \tag{A47}$$

$$=: r_2. \tag{A48}$$

For $(i \in [d]) \wedge (j = d+1) \wedge (k = i)$,

$$H_{i,j,k} = h_i(j = d+1; k = i; c = i) + \sum_{c \neq i}^{d} h_i(j = d+1; k = i; c \neq i) \tag{A49}$$

$$= \epsilon_+ + \frac{\lambda}{2}(d-1)\epsilon_- \tag{A50}$$

$$=: r_3. \tag{A51}$$

For $(i \in [d]) \wedge (j = d+1) \wedge (k \neq i)$,

$$H_{i,j,k} = h_i(j = d+1; k \neq i; c = i) + h_i(j = d+1; k = c; c \neq i) + \sum_{c \neq i,k}^{d} h_i(j = d+1; k \neq i, c; c \neq i) \tag{A52}$$

$$= \epsilon_- + \frac{\lambda}{2}\epsilon_+ + \frac{\lambda}{2}(d-2)\epsilon_- \tag{A53}$$

$$=: r_4. \tag{A54}$$

For $(i \in [d]) \wedge (j \neq i, d+1) \wedge (k = i)$,

$$H_{i,j,k} = h_i(j \neq i, d+1; k = i; c = i) + h_i(j = c; k = i; c \neq i) + \sum_{c \neq i,j}^{d} h_i(j \neq i, c, d+1; k = i; c \neq i) \tag{A55}$$

$$= \frac{\lambda}{2}\epsilon_+ + \frac{\lambda}{2}\epsilon_- + \frac{\lambda^2}{4}(d-2)\epsilon_- \tag{A56}$$

$$=: r_5. \tag{A57}$$

For $(i \in [d]) \wedge (j \neq i, d+1) \wedge (k \neq i) \wedge (j = k)$,

$$H_{i,j,k} = h_i(j \neq i, d+1; k \neq i; c = i) + h_i(j = c; k = c; c \neq i) + \sum_{c \neq i,j,k}^{d} h_i(j \neq i, c, d+1; k \neq i, c; c \neq i) \tag{A58}$$

$$= \frac{\lambda}{2}\epsilon_- + \frac{\lambda}{2}\epsilon_+ + \frac{\lambda^2}{4}(d-2)\epsilon_- \tag{A59}$$

$$=: r_5. \tag{A60}$$

For $(i \in [d]) \wedge (j \neq i, d+1) \wedge (k \neq i) \wedge (j \neq k)$,

$$H_{i,j,k} = h_i(j \neq i, d+1; k \neq i; c = i) + h_i(j = c; k \neq i, c; c \neq i) + h_i(j \neq i, c, d+1; k = c; c \neq i) + \sum_{c \neq i,j,k}^{d} h_i(j \neq i, c, d+1; k \neq i, c; c \neq i) \tag{A61}$$

$$= \frac{\lambda}{2}\epsilon_- + \frac{\lambda}{2}\epsilon_- + \frac{\lambda^2}{4}\epsilon_+ + \frac{\lambda^2}{4}(d-3)\epsilon_- \tag{A62}$$

$$=: r_6. \tag{A63}$$

For $(i = d+1) \wedge (j = d+1)$,

$$H_{i,j,k} = h_i(j = d+1; k = c; c = k) + \sum_{c \neq k}^{d} h_i(j = d+1; k \neq c; c \neq k) \tag{A64}$$

$$= \epsilon_+ + (d-1)\epsilon_- \tag{A65}$$

$$=: r_7. \tag{A66}$$

For $(i = d+1) \wedge (j \neq d+1) \wedge (j = k)$,

$$H_{i,j,k} = h_i(j = c; k = c; c = k) + \sum_{c \neq k}^{d} h_i(j \neq d+1; k \neq c; c \neq k) \tag{A67}$$

$$= \epsilon_+ + \frac{\lambda}{2}(d-1)\epsilon_- \tag{A68}$$

$$=: r_3. \tag{A69}$$

For $(i = d+1) \wedge (j \neq d+1) \wedge (j \neq k)$,

$$H_{i,j,k} = h_i(j = c; k \neq c; c \neq k) + h_i(j \neq c; k = c; c = k) + \sum_{c \neq j,k}^{d} h_i(j \neq c, d+1; k \neq c; c \neq k) \tag{A70}$$

$$= \epsilon_- + \frac{\lambda}{2}\epsilon_+ + \frac{\lambda}{2}(d-2)\epsilon_- \tag{A71}$$

$$=: r_4. \tag{A72}$$

Optimization of A and b. From Eq. (A34), we redefine the objective function as:

$$d \max_{\boldsymbol{b} \in \{0,1\}^{d+1}} \sum_{j=1}^{d+1} \sum_{k=1}^{d} \phi\left(\sum_{i=1}^{d+1} b_i \mathbb{E}_{c, \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1}} [g_{i,j}(y_{N+1} x_{N+1,k} - \epsilon)] \right)$$

$$= \max_{\boldsymbol{b} \in \{0,1\}^{d+1}} \sum_{j=1}^{d+1} \sum_{k=1}^{d} \phi\left(\sum_{i=1}^{d+1} b_i H_{i,j,k} \right). \tag{A73}$$

Recall that we set $A_{j,k} = 1$ if $\sum_{i=1}^{d+1} b_i H_{i,j,k} \geq 0$ and 0 otherwise. Let $[d]' := \{i \in [d] \mid b_i = 1\}$ and $d' := |[d]'|$. Now,

$$\sum_{j=1}^{d+1} \sum_{k=1}^{d} \phi\left(\sum_{i=1}^{d+1} b_i H_{i,j,k} \right) = \sum_{k=1}^{d} \phi\left(b_{d+1} H_{d+1,d+1,k} + \mathbb{1}[k \in [d]'] H_{k,d+1,k} + \sum_{i \in [d]', i \neq k} H_{i,d+1,k} \right)$$
$$+ \sum_{j=1}^{d} \phi\left(b_{d+1} H_{d+1,j,j} + \mathbb{1}[j \in [d]'] H_{j,j,j} + \sum_{i \in [d]', i \neq j} H_{i,j,j} \right)$$
$$+ \sum_{j=1}^{d} \sum_{k \neq j}^{d} \phi\left(b_{d+1} H_{d+1,j,k} + \mathbb{1}[j \in [d]'] H_{i,i,k} + \mathbb{1}[k \in [d]'] H_{i,j,i} + \sum_{i \in [d]', i \neq j,k} H_{i,j,k} \right). \tag{A74}$$

By Eqs. (A51), (A54) and (A66),

$$\sum_{k=1}^{d} \phi\left(b_{d+1} H_{d+1,d+1,k} + \mathbb{1}[k \in [d]'] H_{k,d+1,k} + \sum_{i \in [d]', i \neq k} H_{i,d+1,k} \right)$$

$$= \sum_{k=1}^{d} \phi\left(b_{d+1} r_7 + \mathbb{1}[k \in [d]'] r_3 + \sum_{i \in [d]', i \neq k} r_4 \right) \tag{A75}$$

$$= d' \phi(\underbrace{b_{d+1} r_7 + r_3 + (d'-1) r_4}_{=: s_1(d', b_{d+1})}) + (d - d') \phi(\underbrace{b_{d+1} r_7 + d' r_4}_{=: s_2(d', b_{d+1})}). \tag{A76}$$

By Eqs. (A45), (A60) and (A69),

$$\sum_{j=1}^{d} \phi\left(b_{d+1} H_{d+1,j,j} + \mathbb{1}[j \in [d]'] H_{j,j,j} + \sum_{i \in [d]', i \neq j} H_{i,j,j} \right)$$

$$= \sum_{j=1}^{d} \phi\left(b_{d+1}r_3 + \mathbb{1}[j \in [d]']r_1 + \sum_{i\in[d]', i\neq j} r_5\right) \tag{A77}$$
$$= d'\phi(\underbrace{b_{d+1}r_3 + r_1 + (d'-1)r_5}_{=:s_3(d',b_{d+1})}) + (d-d')\phi(\underbrace{b_{d+1}r_3 + d'r_5}_{=:s_4(d',b_{d+1})}). \tag{A78}$$

By Eqs. (A48), (A57), (A63) and (A72),

$$\sum_{j=1}^{d}\sum_{k\neq j}^{d} \phi\left(b_{d+1}H_{d+1,j,k} + \mathbb{1}[j\in[d]']H_{i,i,k} + \mathbb{1}[k\in[d]']H_{i,j,i} + \sum_{i\in[d]', i\neq j,k} H_{i,j,k}\right)$$
$$= \sum_{j=1}^{d}\sum_{k\neq j}^{d} \phi\left(b_{d+1}r_4 + \mathbb{1}[j\in[d]']r_2 + \mathbb{1}[k\in[d]']r_5 + \sum_{i\in[d]', i\neq j,k} r_6\right) \tag{A79}$$
$$= d'(d'-1)\phi(\underbrace{b_{d+1}r_4 + r_2 + r_5 + (d'-2)r_6}_{=:s_5(d',b_{d+1})}) + d'(d-d')\phi(\underbrace{b_{d+1}r_4 + r_2 + (d'-1)r_6}_{=:s_6(d',b_{d+1})})$$
$$+ d'(d-d')\phi(\underbrace{b_{d+1}r_4 + r_5 + (d'-1)r_6}_{=:s_7(d',b_{d+1})}) + (d-d')(d-d'-1)\phi(\underbrace{b_{d+1}r_4 + d'r_6}_{=:s_8(d',b_{d+1})}). \tag{A80}$$

Now,

$$\sum_{j=1}^{d+1}\sum_{k=1}^{d}\phi\left(\sum_{i=1}^{d+1} b_i H_{i,j,k}\right) = d'\phi(s_1(d',b_{d+1})) + (d-d')\phi(s_2(d',b_{d+1})) + d'\phi(s_3(d',b_{d+1}))$$
$$+ (d-d')\phi(s_4(d',b_{d+1})) + d'(d'-1)\phi(s_5(d',b_{d+1}))$$
$$+ d'(d-d')\phi(s_6(d',b_{d+1})) + d'(d-d')\phi(s_7(d',b_{d+1}))$$
$$+ (d-d')(d-d'-1)\phi(s_8(d',b_{d+1})) \tag{A81}$$
$$=: \mathrm{score}(d',b_{d+1}). \tag{A82}$$

We shall now summarize the discussion to Lemma D.2. The rest of the proof is left to Lemma D.3.

□

Optimization of transformed problem.

Lemma D.2. Let $\phi(x) := \max(0,x)$, $d \in \mathbb{N}$, $0 < \lambda < 1$, $0 \le \epsilon < 1$, $\epsilon_+ := 1-\epsilon$, and $\epsilon_- := \lambda/2-\epsilon$. In addition, for $d' \in \{0,\dots,d\}$ and $b_{d+1} \in \{0,1\}$,

$$r_1 := \epsilon_+ + \frac{\lambda^2}{3}(d-1)\epsilon_-, \tag{A83}$$
$$r_2 := \epsilon_- + \frac{\lambda^2}{3}\epsilon_+ + \frac{\lambda^2}{3}(d-2)\epsilon_-, \tag{A84}$$
$$r_3 := \epsilon_+ + \frac{\lambda}{2}(d-1)\epsilon_-, \tag{A85}$$
$$r_4 := \epsilon_- + \frac{\lambda}{2}\epsilon_+ + \frac{\lambda}{2}(d-2)\epsilon_-, \tag{A86}$$
$$r_5 := \frac{\lambda}{2}\epsilon_+ + \frac{\lambda}{2}\epsilon_- + \frac{\lambda^2}{4}(d-2)\epsilon_-, \tag{A87}$$
$$r_6 := \frac{\lambda}{2}\epsilon_- + \frac{\lambda}{2}\epsilon_- + \frac{\lambda^2}{4}\epsilon_+ + \frac{\lambda^2}{4}(d-3)\epsilon_-, \tag{A88}$$
$$r_7 := \epsilon_+ + (d-1)\epsilon_-, \tag{A89}$$

$$s_1(d',b_{d+1}) := b_{d+1}r_7 + r_3 + (d'-1)r_4, \tag{A90}$$
$$s_2(d',b_{d+1}) := b_{d+1}r_7 + d'r_4, \tag{A91}$$
$$s_3(d',b_{d+1}) := b_{d+1}r_3 + r_1 + (d'-1)r_5, \tag{A92}$$

$$s_4(d', b_{d+1}) := b_{d+1} r_3 + d' r_5, \tag{A93}$$
$$s_5(d', b_{d+1}) := b_{d+1} r_4 + r_2 + r_5 + (d'-2) r_6, \tag{A94}$$
$$s_6(d', b_{d+1}) := b_{d+1} r_4 + r_2 + (d'-1) r_6, \tag{A95}$$
$$s_7(d', b_{d+1}) := b_{d+1} r_4 + r_5 + (d'-1) r_6, \tag{A96}$$
$$s_8(d', b_{d+1}) := b_{d+1} r_4 + d' r_6, \tag{A97}$$

$$\begin{aligned}\mathrm{score}(d', b_{d+1}) := {} & d'\phi(s_1(d', b_{d+1})) + (d-d')\phi(s_2(d', b_{d+1})) + d'\phi(s_3(d', b_{d+1})) \\ & + (d-d')\phi(s_4(d', b_{d+1})) + d'(d'-1)\phi(s_5(d', b_{d+1})) \\ & + d'(d-d')\phi(s_6(d', b_{d+1})) + d'(d-d')\phi(s_7(d', b_{d+1})) \\ & + (d-d')(d-d'-1)\phi(s_8(d', b_{d+1})).\end{aligned} \tag{A98}$$

Considering the following optimization problem:

$$\max_{d' \in \{0,\ldots,d\}, b_{d+1} \in \{0,1\}} \mathrm{score}(d', b_{d+1}). \tag{A99}$$

Then, setting $\boldsymbol{P}, \boldsymbol{Q} \in \mathbb{R}^{(d+1)\times(d+1)}$ to

$$\boldsymbol{P} = \begin{bmatrix} \mathbf{0}_{d,d+1} \\ \boldsymbol{b}^\top \end{bmatrix}, \quad \boldsymbol{Q} = [\boldsymbol{A} \quad \mathbf{0}_{d+1}], \quad \boldsymbol{b}^\top = [\underbrace{1 \quad 1 \quad \cdots \quad 1}_{d'} \quad \underbrace{0 \quad 0 \quad \cdots \quad 0}_{d-d'} \quad b_{d+1}], \tag{A100}$$

$$A_{jk} = \begin{cases} \mathbb{1}[b_{d+1} r_7 + \mathbb{1}[k \le d'] r_3 + (d' - \mathbb{1}[k \le d']) r_4 \ge 0] \\ \qquad (j = d+1) \\ \mathbb{1}[b_{d+1} r_3 + \mathbb{1}[j \le d'] r_1 + (d' - \mathbb{1}[j \le d']) r_5 \ge 0] \\ \qquad (j \ne d+1) \wedge (j = k) \\ \mathbb{1}[b_{d+1} r_4 + \mathbb{1}[j \le d'] r_2 + \mathbb{1}[k \le d'] r_5 + (d' - \mathbb{1}[j \le d'] - \mathbb{1}[k \le d']) r_6 \ge 0] \\ \qquad (j \ne d+1) \wedge (j \ne k) \end{cases}, \tag{A101}$$

the global maximizer of (A99) is the global minimizer of (7).

Proof. See the above discussion. ☐

Lemma D.3. The global maximizer of (A99) is as follows:

(a) If

$$0 \le \epsilon \le \frac{\lambda(\lambda(d-2)+4)}{2(\lambda(d-1)+2)}, \tag{A102}$$

then $d' = d$ and $b_{d+1} = 1$. This corresponds to $\boldsymbol{b} = \mathbf{1}_{d+1}$ and $\boldsymbol{A} = \mathbf{1}_{d+1,d}$.

(b) If

$$\epsilon = \frac{\lambda(d-1)+2}{2d}, \tag{A103}$$

then $d' = d$ and $b_{d+1} = 1$. This corresponds to $\boldsymbol{b} = \mathbf{1}_{d+1}$ and $\boldsymbol{A} = [\boldsymbol{I}_d \ \ \mathbf{0}_d]^\top$.

(c) If

$$\epsilon \ge \frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{\lambda^2(d-1)+3}, \tag{A104}$$

then $d' = 0$ and $b_{d+1} = 0$. This corresponds to $\boldsymbol{b} = \mathbf{1}_{d+1}$ and $\boldsymbol{A} = \mathbf{0}_{d+1,d}$.

Proof. For notational simplicity, we abbreviate terms including variables such as $x_1, x_2, \dots$ (e.g., $x_1^2 + 3x_2 + \cdots$) using the notation $\Theta(x_1, x_2, \dots)$. In particular, when the expression is strictly nonnegative (e.g., $x_1^2 + x_2^2$) or nonpositive, we use $\Theta_+(x_1, x_2, \dots)$ or $\Theta_-(x_1, x_2, \dots)$, respectively. These terms are not essential to the analysis and too long. They can be derived by simple basic arithmetic operations. These concrete values can be showed by our python codes.

We define $\epsilon_1, \dots, \epsilon_7$ as

$$r_1 = 0 \iff \epsilon = \frac{\lambda}{2} + \frac{3}{2}\frac{2-\lambda}{\lambda^2(d-1)+3} =: \epsilon_1, \tag{A105}$$

$$r_2 = 0 \iff \epsilon = \frac{\lambda(\lambda^2(d-2)+2\lambda+3)}{2(\lambda^2(d-1)+3)} =: \epsilon_2, \tag{A106}$$

$$r_3 = 0 \iff \epsilon = \frac{\lambda^2(d-1)+4}{2(\lambda(d-1)+2)} =: \epsilon_3, \tag{A107}$$

$$r_4 = 0 \iff \epsilon = \frac{\lambda(\lambda(d-2)+4)}{2(\lambda(d-1)+2)} =: \epsilon_4, \tag{A108}$$

$$r_5 = 0 \iff \epsilon = \frac{\lambda^2(d-2)+2\lambda+4}{2(\lambda(d-2)+4)} =: \epsilon_5, \tag{A109}$$

$$r_6 = 0 \iff \epsilon = \frac{\lambda(\lambda(d-3)+6)}{2(\lambda(d-2)+4)} =: \epsilon_6, \tag{A110}$$

$$r_7 = 0 \iff \epsilon = \frac{\lambda(d-1)+2}{2d} =: \epsilon_7, \tag{A111}$$

$$s_5(d,1) = 0 \iff \epsilon = \frac{\lambda}{2}\frac{3d^2\lambda^2 - 8d\lambda^2 + 24d\lambda + 4\lambda^2 - 34\lambda + 48}{3d^2\lambda^2 - 5d\lambda^2 + 18d\lambda + 2\lambda^2 - 18\lambda + 24} =: \epsilon_{s_5}. \tag{A112}$$

Since

$$\epsilon_1 - \epsilon_3 = \frac{\lambda(d-1)(2-\lambda)(3-2\lambda)}{2(\lambda(d-1)+2)(\lambda^2(d-1)+3)} \geq 0, \tag{A113}$$

$$\epsilon_3 - \epsilon_5 = \frac{(2-\lambda)^2}{(\lambda(d-2)+4)(\lambda(d-1)+2)} \geq 0, \tag{A114}$$

$$\epsilon_5 - \epsilon_7 = \frac{(d-2)(2-\lambda)^2}{2d(\lambda(d-2)+4)} \geq 0, \tag{A115}$$

$$\epsilon_7 - \epsilon_{s_5} = \frac{(2-\lambda)(-3d\lambda^2 + 6d\lambda + 2\lambda^2 - 18\lambda + 24)}{2d(3d^2\lambda^2 - 5d\lambda^2 + 18d\lambda + 2\lambda^2 - 18\lambda + 24)} \geq 0, \tag{A116}$$

$$\epsilon_{s_5} - \epsilon_4 = \frac{\lambda^2(2-\lambda)}{(\lambda(d-1)+2)(3d^2\lambda^2 - 5d\lambda^2 + 18d\lambda + 2\lambda^2 - 18\lambda + 24)} \geq 0, \tag{A117}$$

$$\epsilon_4 - \epsilon_6 = \frac{\lambda(2-\lambda)^2}{2(\lambda(d-2)+4)(\lambda(d-1)+2)} \geq 0, \tag{A118}$$

$$\epsilon_6 - \epsilon_2 = \frac{\lambda(3-\lambda)(2-\lambda)(1-\lambda)}{2(\lambda(d-2)+4)(\lambda^2(d-1)+3)} \geq 0, \tag{A119}$$

for $d \geq 2$, they are ordered as

$$\epsilon_2 \leq \epsilon_6 \leq \epsilon_4 \leq \epsilon_{s_5} \leq \epsilon_7 \leq \epsilon_5 \leq \epsilon_3 \leq \epsilon_1. \tag{A120}$$

In score, b_{d+1} appears as $b_{d+1}r_3$, $b_{d+1}r_4$, or $b_{d+1}r_7$, each with a positive coefficient in d and d'. Thus, if $r_3, r_4, r_7 \leq 0$, then b_{d+1} should be zero. If $r_3, r_4, r_7 \geq 0$, then b_{d+1} should be one. Considering Ineq. (A120), for $d \geq 2$, the optimal b_{d+1} is one if $\epsilon \leq \epsilon_4$ and zero if $\epsilon \geq \epsilon_3$.

One-Dimensional Case. If $d = 1$,

$$\begin{aligned} &\text{score}(d', b_{d+1}) \\ &= \mathbb{1}[d'=0](\phi(b_{d+1}r_7) + \phi(b_{d+1}r_3)) + \mathbb{1}[d'=1](\phi(b_{d+1}r_7 + r_3) + \phi(b_{d+1}r_3 + r_1)) \qquad \text{(A121)} \\ &= \mathbb{1}[d'=0](\phi(b_{d+1}\epsilon_+) + \phi(b_{d+1}\epsilon_+)) \end{aligned}$$

$$+ \mathbb{1}[d' = 1](\phi(b_{d+1}\epsilon_+ + \epsilon_+) + \phi(b_{d+1}\epsilon_+ + \epsilon_+)). \tag{A122}$$

As ϵ_+ is always positive for $0 \le \epsilon < 1$, $d' = d = 1$ and $b_{d+1} = 1$ are the optimal. This aligns with the following case analysis.

Weak Adversarial (Case 1). Assume $d \ge 2$ and $0 \le \epsilon \le \epsilon_6$. As $\epsilon \le \epsilon_6 \le \epsilon_4$, $b_{d+1} = 1$ is the optimal. By Ineq. (A120), $r_1, r_3, r_4, r_5, r_6, r_7 \ge 0$. The sign of r_2 depends on ϵ. Thus, $s_1(d',1), s_2(d',1), s_3(d',1), s_4(d',1), s_7(d',1), s_8(d',1) \ge 0$ for $0 \le d' \le d$. In addition, for $d' \ge 2$,

$$\begin{aligned}
s_5(d',1) &\ge r_4 + r_2 &\text{(A123)}\\
&= \frac{\lambda^3}{6}(d-2) + \frac{\lambda^2}{12}(3d-2) + \frac{3\lambda}{2} - \frac{\epsilon}{6}(2\lambda^2(d-1) + 3\lambda(d-1) + 12) &\text{(A124)}\\
&\ge \frac{\lambda^2(2-\lambda)(5-2\lambda)}{12(\lambda(d-2)+4)} \qquad (\because \epsilon \le \epsilon_6) &\text{(A125)}\\
&\ge 0. &\text{(A126)}
\end{aligned}$$

Thus, $d'(d'-1)s_5(d',1)$ is nonnegative for $0 \le d' \le d$. Similarly, by $s_6(d',1) \ge r_4 + r_2 \ge 0$ for $d' \ge 1$, $d'(d'-1)s_6(d',1)$ is nonnegative for $0 \le d' \le d$. Thus,

$$\begin{aligned}
\mathrm{score}(d',1) := \;& d's_1(d',1) + (d-d')s_2(d',1) + d's_3(d',1) + (d-d')s_4(d',1)\\
&+ d'(d'-1)s_5(d',1) + d'(d-d')s_6(d',1) + d'(d-d')s_7(d',1)\\
&+ (d-d')(d-d'-1)s_8(d',1) &\text{(A127)}\\
=\;& dr_7 + d'r_3 + d'(d-1)r_4 + dr_3 + d'r_1 + d'(d-1)r_5\\
&+ dr_4 + d'r_2 + d'r_5 + d'(d-1)(d-2)r_6. &\text{(A128)}
\end{aligned}$$

This monotonically increases in d'. Therefore, $d' = d$ is the optimal. By Lemma D.2, $\boldsymbol{b} = \mathbf{1}_{d+1}$. In addition, from $s_1(d,1), s_3(d,1), s_5(d,1) \ge 0$, $\boldsymbol{A} = \mathbf{1}_{d+1,d}$.

Weak Adversarial (Case 2). Assume $d \ge 2$ and $\epsilon_6 \le \epsilon \le \epsilon_4$. As $\epsilon \le \epsilon_4$, $b_{d+1} = 1$ is the optimal. By Ineq. (A120), $r_1, r_3, r_4, r_5, r_7 \ge 0$ and $r_2, r_6 \le 0$. Thus, $s_1(d',1), s_2(d',1), s_3(d',1), s_4(d',1) \ge 0$. In addition,

$$s_5(d',1) \ge s_5(d,1) \ge \frac{\lambda^2(2-\lambda)}{12(\lambda(d-1)+2)} \ge 0 \qquad (\because \epsilon \le \epsilon_4), \tag{A129}$$

$$s_7(d',1) \ge s_7(d,1) \ge \frac{\lambda(2-\lambda)^3}{8(\lambda(d-1)+2)} \ge 0 \qquad (\because \epsilon \le \epsilon_4). \tag{A130}$$

Due to the following inequality, $s_8(d',1)$ is always larger than $s_6(d',1)$:

$$\begin{aligned}
s_8(d',1) - s_6(d',1) &= -\frac{\lambda^3}{24}(d+1) + \frac{5\lambda^2}{12} - \frac{\lambda}{2} + \frac{\epsilon}{12}(\lambda^2(d+2) + 12(1-\lambda)) &\text{(A131)}\\
&\ge \frac{\lambda(3-\lambda)(2-\lambda)(1-\lambda)}{6(\lambda(d-2)+4)} \qquad (\because \epsilon \ge \epsilon_6) &\text{(A132)}\\
&\ge 0. &\text{(A133)}
\end{aligned}$$

If $s_6(d',1), s_8(d',1) \ge 0$,

$$\frac{\mathrm{d}\,\mathrm{score}(d',1)}{\mathrm{d}d'} = \frac{(2 + \lambda(d-1) - 2d\epsilon)(\lambda^2(3d^2 - 5d + 2) + 18\lambda(d-1) + 24)}{24} \ge 0. \tag{A134}$$

We used

$$2 + \lambda(d-1) - 2d\epsilon \ge \frac{(2-\lambda)^2}{\lambda(d-1)+2} \ge 0 \qquad (\because \epsilon \le \epsilon_4). \tag{A135}$$

If $s_6(d',1) \le 0, s_8(d',1) \ge 0$,

$$\frac{\mathrm{d}\,\mathrm{score}(d',1)}{\mathrm{d}d'} = \Theta(d,d',\lambda) - \frac{\epsilon}{12}\{3d\lambda^2((d-d')^2 + 2d'^2) + 6\lambda(2-\lambda)\left\{\left(d - \frac{1}{2}d'\right)^2 + \frac{11}{4}d'^2\right\}$$

$$+ 8dd'\lambda^2 + d'(4\lambda^2 - 36\lambda + 48)\} \quad \text{(A136)}$$

$$\geq \Theta(d,\lambda) - \frac{\lambda(2-\lambda)}{24(\lambda(d-1)+2)} d'(9d'\lambda(2-\lambda) + 6\lambda^2(d+1) - 4\lambda(3d+7) + 24)$$

$$(\because \epsilon \leq \epsilon_4) \quad \text{(A137)}$$

$$\geq \frac{(2-\lambda)(d\lambda^3 + d\lambda(12-7\lambda) - \lambda^3 + 11\lambda^2 - 30\lambda + 24)}{12(\lambda(d-1)+2)} \quad \text{(A138)}$$

$$\geq 0. \quad \text{(A139)}$$

We used for $0 \leq d' \leq d$,

$$d'(9d'\lambda(2-\lambda) + 6\lambda^2(d+1) - 4\lambda(3d+7) + 24)$$
$$\leq d\lambda(3d\lambda(2-\lambda) + 6\lambda^2 - 28\lambda + 24). \quad \text{(A140)}$$

If $s_6(d', 1) \leq 0, s_8(d', 1) \leq 0$,

$$\frac{\mathrm{d}\,\mathrm{score}(d',1)}{\mathrm{d}d'}$$
$$= \Theta(d,d',\lambda) - \frac{\epsilon}{12}\{3d^2\lambda(\lambda+4) + 6d(-\lambda^2 - \lambda + 2) + 6\lambda + 12(d-1)$$
$$+ 2d'(3d^2\lambda^2 + 8d\lambda(-\lambda+1) + 4(2\lambda^2 + (d-6)\lambda + 3)\} \quad \text{(A141)}$$
$$\geq \Theta(d,\lambda) - \frac{\lambda(2-\lambda)}{12(\lambda(d-1)+2)} d'(-3d\lambda^2 + 6d\lambda + 6\lambda^2 - 20\lambda + 12) \quad (\because \epsilon \leq \epsilon_4) \quad \text{(A142)}$$
$$\geq \frac{(2-\lambda)(-d\lambda^3 - 8d\lambda^2 + 24d\lambda - 2\lambda^3 + 22\lambda^2 - 60\lambda + 48)}{24(\lambda(d-1)+2)} \quad (\because d' \leq d) \quad \text{(A143)}$$
$$\geq 0. \quad \text{(A144)}$$

From the above discussion, for any case, ($s_6, s_8 \geq 0$), ($s_6 \leq 0$ and $s_8 \geq 0$), or ($s_6, s_8 \leq 0$), the derivative of $\mathrm{score}(d', 1)$ with respect to d' is nonnegative. Thus, $d' = d$ is the optimal. By Lemma D.2, $\boldsymbol{b} = \mathbf{1}_{d+1}$. In addition, from $s_1(d,1), s_3(d,1), s_5(d,1) \geq 0$, $\boldsymbol{A} = \mathbf{1}_{d+1,d}$.

Adversarial. Assume $d \geq 2$ and $\epsilon = \epsilon_7$. By Ineq. (A120), $r_1, r_3, r_5 \geq 0$, $r_7 = 0$, and $r_2, r_4, r_6 \leq 0$. Thus, $s_3(d', b_{d+1}), s_4(d', b_{d+1}) \geq 0$ and $s_2(d', b_{d+1}), s_6(d', b_{d+1}), s_8(d', b_{d+1}) \leq 0$. Now,

$$s_1(d',1) = s_1(d',0) \geq \frac{(d-d')(2-\lambda)^2}{4d} \geq 0 \quad (\because \epsilon = \epsilon_7). \quad \text{(A145)}$$

Thus,

$$\mathrm{score}(d', b_{d+1}) = d's_1(d',0) + d's_3(d',b_{d+1}) + (d-d')s_4(d',b_{d+1})$$
$$+ d'(d'-1)\phi(s_5(d',b_{d+1})) + d'(d-d')\phi(s_7(d',b_{d+1})) \quad \text{(A146)}$$
$$= d's_1(d',0) + d'r_1 + (d-1)d'r_5 + db_{d+1}r_3$$
$$+ d'(d'-1)\phi(b_{d+1}r_4 + r_2 + r_5 + (d'-2)r_6)$$
$$+ d'(d-d')\phi(b_{d+1}r_4 + r_5 + (d'-1)r_6). \quad \text{(A147)}$$

Since r_4 is nonpositive, this indicates that score changes by $dr_3 + d'(d-1)r_4$ at least by switching b_{d+1} to one from zero. Moreover,

$$dr_3 + d'(d-1)r_4 \geq \frac{(d-1)(d-d')(2-\lambda)^2}{4d} \geq 0 \quad (\because \epsilon = \epsilon_7). \quad \text{(A148)}$$

Therefore, $b_{d+1} = 1$ is the optimal. From Ineq. (A120) and $\epsilon = \epsilon_7$, $s_7(d', b_{d+1}) - s_5(d', b_{d+1}) \geq 0$. If $s_5(d',1), s_7(d',1) \geq 0$,

$$\frac{\mathrm{d}\,\mathrm{score}(d',1)}{\mathrm{d}d'} = \Theta(d,d',\lambda) - \Theta_+(d,d',\lambda)\epsilon \quad \text{(A149)}$$
$$= \Theta(d,\lambda) - \Theta_+(d,\lambda)d' \quad (\because \epsilon = \epsilon_7) \quad \text{(A150)}$$
$$\geq 0 \quad (\because d' \leq d_{s_5}), \quad \text{(A151)}$$

where

$$s_5(d',1) \geq 0 \iff d' \leq \frac{3d\lambda^2 - 6d\lambda + 2\lambda^2 - 18\lambda + 24}{6\lambda(\lambda-2)} =: d_{s_5}. \tag{A152}$$

When $s_5(d',1) \leq 0, s_7(d',1) \geq 0$, then $\frac{\mathrm{d}\,\mathrm{score}(d',1)}{\mathrm{d}d'} \geq 0$ similarly holds. If $s_5(d',1), s_7(d',1) \leq 0$, $\frac{\mathrm{d}\,\mathrm{score}(d',1)}{\mathrm{d}d'} \geq 0$ for $d' \leq d-1$. Comparing $\mathrm{score}(d',1)$ with $d' = d-1$ and $d' = d$, we obtain $\mathrm{score}(d,1) \geq \mathrm{score}(d-1,1)$. In summary, $d' = d$ is the optimal. By Lemma D.2, $\boldsymbol{b} = \mathbf{1}_{d+1}$. In addition, from $s_3(d,1) \geq 0$, $s_1(d,1) = 0$, and $s_5(d,1) < 0$, $\boldsymbol{A} = [\boldsymbol{I}_d \ \ \mathbf{0}_d]^\top$.

Strong Adversarial. Assume $d \geq 2$ and $\epsilon \geq \epsilon_1$. By Ineq. (A120), $r_1, \dots, r_7$ are nonpositive. Thus, $s_1(d', b_{d+1}), \dots, s_8(d', b_{d+1})$ are nonpositive. Therefore, $d' = 0$ and $b_{d+1} = 0$ are the optimal. By Lemma D.2, $\boldsymbol{b} = \mathbf{0}_{d+1}$ and $\boldsymbol{A} = \mathbf{0}_{d+1,d}$. ☐

E Proof of Theorems 3.5 and 3.6 (Robustness)

For notational convenience, we occasionally describe representations and equations under the assumption that $\mathcal{S}_{\mathrm{rob}} := \{1, \dots, d_{\mathrm{rob}}\}$, $\mathcal{S}_{\mathrm{vul}} := \{d_{\mathrm{rob}}+1, \dots, d_{\mathrm{rob}}+d_{\mathrm{vul}}\}$, and $\mathcal{S}_{\mathrm{irr}} := \{d_{\mathrm{rob}}+d_{\mathrm{vul}}+1, \dots, d_{\mathrm{rob}}+d_{\mathrm{vul}}+d_{\mathrm{irr}}\}$. This assumption is made without loss of generality.

We use *uniform* big-O and -Theta notation. Denote $f(x) = \mathcal{O}(g(x))$ if there exists a positive constant $C > 0$ such that $|f(x)| \leq C|g(x)|$ for *every* x in the domain. Denote $f(x) = \Theta(g(x))$ if there exist $C_1, C_2 > 0$ such that $C_1|g(x)| \leq |f(x)| \leq C_2|g(x)|$ for *every* x in the domain.

For notational simplicity, we abbreviate the following matrix:

$$\begin{bmatrix} C_1\alpha \\ C_2\alpha \\ \vdots \\ C_{d_{\mathrm{rob}}}\alpha \\ C_{d_{\mathrm{rob}}+1}\beta \\ \vdots \\ C_{d_{\mathrm{rob}}+d_{\mathrm{vul}}}\beta \\ C_{d_{\mathrm{rob}}+d_{\mathrm{vul}}+1}\gamma \\ \vdots \\ C_{d_{\mathrm{rob}}+d_{\mathrm{vul}}+d_{\mathrm{irr}}}\gamma \end{bmatrix} \quad \text{as} \quad \begin{bmatrix} C_i\alpha \\ C_i\beta \\ C_i\gamma \end{bmatrix}. \tag{A153}$$

Theorem 3.5 (Standard pretraining case)**.** There exist a constant $C > 0$ and a strictly positive function $g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma)$ such that

$$\begin{aligned} &\mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^{N+1} \overset{\mathrm{i.i.d.}}{\sim} \mathcal{D}^{\mathrm{te}}}\left[\min_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}^{\mathrm{std}}, \boldsymbol{Q}^{\mathrm{std}})]_{d+1,N+1}\right] \\ \leq \quad & g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma) \left\{ \underbrace{C(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta)}_{\text{Prediction for original data}} - \underbrace{(d_{\mathrm{rob}} + d_{\mathrm{vul}} + d_{\mathrm{irr}})\epsilon}_{\text{Adversarial effect}} \right\}. \end{aligned} \tag{8}$$

Proof. Since $\boldsymbol{b} = \mathbf{1}_{d+1}$, $\boldsymbol{A} = \mathbf{1}_{d+1,d}$, and $\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top$ is positive semidefinite, every entry in $\boldsymbol{b}^\top\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top\boldsymbol{A}$ is nonnegative. Thus, we can solve the inner minimization as

$$\min_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} y_{N+1}[\boldsymbol{f}(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}, \boldsymbol{Q})]_{d+1,N+1} = \min_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} \frac{1}{N}\boldsymbol{b}^\top\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top\boldsymbol{A}y_{N+1}(\boldsymbol{x}_{N+1} + \boldsymbol{\Delta}) \tag{A154}$$

$$= \frac{1}{N}\boldsymbol{b}^\top\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top\boldsymbol{A}(y_{N+1}\boldsymbol{x}_{N+1} - \epsilon\mathbf{1}_d). \tag{A155}$$

Using $(\boldsymbol{x}, y) \sim \mathcal{D}^{\mathrm{te}}$,

$$\mathbb{E}\left[\frac{1}{N}\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top\right]$$

$$= \begin{bmatrix} \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top] & \mathbb{E}[y\boldsymbol{x}] \\ \mathbb{E}[y\boldsymbol{x}^\top] & 1 \end{bmatrix} \tag{A156}$$

$$= \begin{bmatrix} \mathbb{E}[y\boldsymbol{x}]\mathbb{E}[y\boldsymbol{x}^\top] & \mathbb{E}[y\boldsymbol{x}] \\ \mathbb{E}[y\boldsymbol{x}^\top] & 1 \end{bmatrix} + \begin{bmatrix} \mathbb{E}[(y\boldsymbol{x} - \mathbb{E}[y\boldsymbol{x}])(y\boldsymbol{x} - \mathbb{E}[y\boldsymbol{x}])^\top] & \mathbf{0}_d \\ \mathbf{0}_d^\top & 0 \end{bmatrix}. \tag{A157}$$

Since the second term is positive semidefinite,

$$\mathbb{E}\left[\frac{1}{N}\mathbf{1}_{d+1}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \mathbf{1}_{d+1}\right]$$

$$= \mathbf{1}_{d+1}^\top \left(\begin{bmatrix} \mathbb{E}[y\boldsymbol{x}]\mathbb{E}[y\boldsymbol{x}^\top] & \mathbb{E}[y\boldsymbol{x}] \\ \mathbb{E}[y\boldsymbol{x}^\top] & 1 \end{bmatrix} + \begin{bmatrix} \mathbb{E}[(y\boldsymbol{x} - \mathbb{E}[y\boldsymbol{x}])(y\boldsymbol{x} - \mathbb{E}[y\boldsymbol{x}])^\top] & \mathbf{0}_d \\ \mathbf{0}_d^\top & 0 \end{bmatrix} \right) \mathbf{1}_{d+1} \tag{A158}$$

$$\geq \mathbf{1}_{d+1}^\top \begin{bmatrix} \mathbb{E}[y\boldsymbol{x}^\top]\mathbb{E}[y\boldsymbol{x}] & \mathbb{E}[y\boldsymbol{x}] \\ \mathbb{E}[y\boldsymbol{x}^\top] & 1 \end{bmatrix} \mathbf{1}_{d+1}. \tag{A159}$$

Since every entry of $\mathbb{E}[y\boldsymbol{x}^\top]\mathbb{E}[y\boldsymbol{x}]$ and $\mathbb{E}[y\boldsymbol{x}]$ is nonnegative,

$$\mathbb{E}\left[\frac{1}{N}\mathbf{1}_{d+1}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \mathbf{1}_{d+1}\right] \geq \mathbf{1}_{d+1}^\top \begin{bmatrix} \mathbb{E}[y\boldsymbol{x}^\top]\mathbb{E}[y\boldsymbol{x}] & \mathbb{E}[y\boldsymbol{x}] \\ \mathbb{E}[y\boldsymbol{x}^\top] & 1 \end{bmatrix} \mathbf{1}_{d+1} \geq 1. \tag{A160}$$

Representing $\mathbb{E}[\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A}/N] = [g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma) \ \cdots \ g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma)]$ using some positive function $g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma) > 0$, there exists a positive constant $C > 0$ such that

$$\mathbb{E}\left[\frac{1}{N}\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}} \boldsymbol{M} \boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A}(y_{N+1}\boldsymbol{x}_{N+1} - \epsilon \mathbf{1}_d)\right]$$

$$= \begin{bmatrix} g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma) \\ \vdots \\ g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma) \end{bmatrix}^\top (\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}] - \epsilon \mathbf{1}_d) \tag{A161}$$

$$= g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma)(\Theta(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta) - d\epsilon) \tag{A162}$$

$$\leq g(d_{\mathrm{rob}}, d_{\mathrm{vul}}, d_{\mathrm{irr}}, \alpha, \beta, \gamma)(C(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta) - (d_{\mathrm{rob}} + d_{\mathrm{vul}} + d_{\mathrm{irr}})\epsilon). \tag{A163}$$

□

Theorem 3.6 (Adversarial pretraining case)**.** Suppose that q_{rob} and q_{vul} defined in Assumption 3.2 are sufficiently small. There exist constants $C_1, C_2 > 0$ such that

$$\mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\mathrm{te}}}\left[\min_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}^{\mathrm{adv}}, \boldsymbol{Q}^{\mathrm{adv}})]_{d+1,N+1}\right]$$
$$\geq \underbrace{C_1(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)(d_{\mathrm{rob}}\alpha^2 + d_{\mathrm{vul}}\beta^2)}_{\text{Prediction for original data}}$$
$$\underbrace{- C_2\left\{(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)\left(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + \frac{d_{\mathrm{irr}}\gamma}{\sqrt{N}}\right) + d_{\mathrm{irr}}\left(\sqrt{\frac{d_{\mathrm{irr}}}{N}} + 1\right)\gamma^2\right\}\epsilon}_{\text{Adversarial effect}}. \tag{9}$$

Proof. This is the special case of the following theorem. □

Theorem E.1 (General case of Theorem 3.6)**.** There exist constants $C, C', C'' > 0$ such that

$$\mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N+1} \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\mathrm{te}}}\left[\min_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{\Delta}}; \boldsymbol{P}^{\mathrm{adv}}, \boldsymbol{Q}^{\mathrm{adv}})]_{d+1,N+1}\right]$$
$$\geq C(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta)\{(1 - cq_{\mathrm{rob}})d_{\mathrm{rob}}\alpha^2 + (1 - cq_{\mathrm{vul}})d_{\mathrm{vul}}\beta^2\} + C'(d_{\mathrm{rob}}\alpha^2 + d_{\mathrm{vul}}\beta^2)$$
$$- C''\left\{(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)\left(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + \frac{d_{\mathrm{irr}}\gamma}{\sqrt{N}}\right) + d_{\mathrm{irr}}\left(\sqrt{\frac{d_{\mathrm{irr}}}{N}} + 1\right)\gamma^2\right\}\epsilon, \tag{A164}$$

where

$$c := \frac{(\max_{i\in\mathcal{S}_{\rm rob}\cup\mathcal{S}_{\rm vul}} C_i)(\max_{i\in\mathcal{S}_{\rm rob}\cup\mathcal{S}_{\rm vul}} C_{i,2})}{\min_{i\in\mathcal{S}_{\rm rob}\cup\mathcal{S}_{\rm vul}} C_i^3}. \tag{A165}$$

In particular, if there exists a constant $C''' > 0$ such that $1 - cq_{\rm rob} \geq C'''$ and $1 - cq_{\rm vul} \geq C'''$, then there exist constants $C_1, C_2 > 0$ such that Ineq. (9) holds.

Proof. Similarly to Eq. (A29), we can solve the minimization as

$$\begin{aligned}
&\min_{\|\boldsymbol{\Delta}\|_\infty\leq\epsilon} y_{N+1}[\boldsymbol{f}(\boldsymbol{Z}_{\boldsymbol{\Delta}};\boldsymbol{P},\boldsymbol{Q})]_{d+1,N+1} \\
&= \min_{\|\boldsymbol{\Delta}\|_\infty\leq\epsilon} \frac{1}{N}\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A} y_{N+1}(\boldsymbol{x}_{N+1}+\boldsymbol{\Delta}) &\text{(A166)}\\
&= \frac{1}{N}\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A} y_{N+1}\boldsymbol{x}_{N+1} - \epsilon\left\|\frac{1}{N}\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A}\right\|_1. &\text{(A167)}
\end{aligned}$$

By Eq. (A157), we can rearrange the first term as

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{N}\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A} y_{N+1}\boldsymbol{x}_{N+1}\right] \\
&= \boldsymbol{1}_{d+1}^\top \begin{bmatrix}\mathbb{E}[\boldsymbol{x}]\mathbb{E}[\boldsymbol{x}^\top] \\ \mathbb{E}[y\boldsymbol{x}^\top]\end{bmatrix}\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}] + \boldsymbol{1}_d^\top\mathbb{E}[(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])^\top]\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}]. &\text{(A168)}
\end{aligned}$$

The first term of Eq. (A168) can be rearranged as

$$\begin{aligned}
&\boldsymbol{1}_{d+1}^\top \begin{bmatrix}\mathbb{E}[\boldsymbol{x}]\mathbb{E}[\boldsymbol{x}^\top] \\ \mathbb{E}[y\boldsymbol{x}^\top]\end{bmatrix}\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}] \\
&= \boldsymbol{1}_{d+1}^\top \begin{bmatrix} C_iC_j\alpha^2 & C_iC_j\alpha\beta & \boldsymbol{0} \\ C_iC_j\alpha\beta & C_iC_j\beta^2 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & C_i^2\gamma^2\boldsymbol{I} \\ C_i\alpha & C_i\beta & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}C_i\alpha \\ C_i\beta \\ \boldsymbol{0}\end{bmatrix} &\text{(A169)}\\
&= \left(\sum_{i\in\mathcal{S}_{\rm rob}} C_i\alpha + \sum_{i\in\mathcal{S}_{\rm vul}} C_i\beta + 1\right)\left(\sum_{i\in\mathcal{S}_{\rm rob}} C_i^2\alpha^2 + \sum_{i\in\mathcal{S}_{\rm vul}} C_i^2\beta^2\right) &\text{(A170)}\\
&= \left(\min_{i\in\mathcal{S}_{\rm rob}\cup\mathcal{S}_{\rm vul}} C_i^3\right)(d_{\rm rob}\alpha + d_{\rm vul}\beta)(d_{\rm rob}\alpha^2 + d_{\rm vul}\beta^2) + \sum_{i\in\mathcal{S}_{\rm rob}} C_i^2\alpha^2 + \sum_{i\in\mathcal{S}_{\rm vul}} C_i^2\beta^2. &\text{(A171)}
\end{aligned}$$

Consider the second term of Eq. (A168). Now,

$$\begin{aligned}
&|\mathbb{E}[(x_i - \mathbb{E}[x_i])(x_j - \mathbb{E}[x_j])]| \\
&\leq \begin{cases}\sqrt{C_{i,2}}\sqrt{C_{j,2}}\alpha^2 & (i,j\in\mathcal{S}_{\rm rob}) \\ \sqrt{C_{i,2}}\sqrt{C_{j,2}}\beta^2 & (i,j\in\mathcal{S}_{\rm vul}) \\ \sqrt{C_{i,2}}\sqrt{C_{j,2}}\alpha\beta & (i\in\mathcal{S}_{\rm rob}\wedge j\in\mathcal{S}_{\rm vul})\vee(i\in\mathcal{S}_{\rm vul}\wedge j\in\mathcal{S}_{\rm rob})\end{cases}. &\text{(A172)}
\end{aligned}$$

Let

$$\mathcal{S} := \left\{ i\in\mathcal{S}_{\rm rob}\cup\mathcal{S}_{\rm vul} \mid \sum_{j\in\mathcal{S}_{\rm rob}\cup\mathcal{S}_{\rm vul}} \mathbb{E}[(x_i - \mathbb{E}[x_i])(x_j - \mathbb{E}[x_j])] < 0 \right\}. \tag{A173}$$

The second term of Eq. (A168) can be computed as

$$\boldsymbol{1}_d^\top\mathbb{E}[(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])^\top]\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}]$$

$$\geq - \begin{bmatrix} \left.\begin{matrix} \sqrt{C_{i,2}}\alpha\left(\sum_{j\in\mathcal{S}_{\mathrm{rob}}}\sqrt{C_{j,2}}\alpha+\sum_{j\in\mathcal{S}_{\mathrm{vul}}}\sqrt{C_{j,2}}\beta\right) \\ \vdots \\ \sqrt{C_{i,2}}\alpha\left(\sum_{j\in\mathcal{S}_{\mathrm{rob}}}\sqrt{C_{j,2}}\alpha+\sum_{j\in\mathcal{S}_{\mathrm{vul}}}\sqrt{C_{j,2}}\beta\right) \end{matrix}\right\} \leq q_{\mathrm{rob}}d_{\mathrm{rob}} \\ \mathbf{0} \\ \left.\begin{matrix} \sqrt{C_{i,2}}\beta\left(\sum_{j\in\mathcal{S}_{\mathrm{rob}}}\sqrt{C_{j,2}}\alpha+\sum_{j\in\mathcal{S}_{\mathrm{vul}}}\sqrt{C_{j,2}}\beta\right) \\ \vdots \\ \sqrt{C_{i,2}}\beta\left(\sum_{j\in\mathcal{S}_{\mathrm{rob}}}\sqrt{C_{j,2}}\alpha+\sum_{j\in\mathcal{S}_{\mathrm{vul}}}\sqrt{C_{j,2}}\beta\right) \end{matrix}\right\} \leq q_{\mathrm{vul}}d_{\mathrm{vul}} \\ \mathbf{0} \end{bmatrix}^{\top} \begin{bmatrix} C_i\alpha \\ C_i\beta \\ \mathbf{0} \end{bmatrix} \tag{A174}$$

$$= -\left(\sum_{i\in\mathcal{S}_{\mathrm{rob}}}\sqrt{C_{i,2}}\alpha+\sum_{i\in\mathcal{S}_{\mathrm{vul}}}\sqrt{C_{i,2}}\beta\right)$$
$$\times\left(\sum_{i\in\mathcal{S}_{\mathrm{rob}}\cap\mathcal{S}}C_i\sqrt{C_{i,2}}\alpha^2+\sum_{i\in\mathcal{S}_{\mathrm{vul}}\cap\mathcal{S}}C_i\sqrt{C_{i,2}}\beta^2\right) \tag{A175}$$

$$\geq -\left(\max_{i\in\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}}}\sqrt{C_{i,2}}\right)\left(\max_{i\in(\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}})\cap\mathcal{S}}C_i\sqrt{C_{i,2}}\right)$$
$$\times(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta)(q_{\mathrm{rob}}d_{\mathrm{rob}}\alpha^2+q_{\mathrm{vul}}d_{\mathrm{vul}}\beta^2) \tag{A176}$$

$$\geq -\left(\max_{i\in\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}}}C_i\right)\left(\max_{i\in\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}}}C_{i,2}\right)(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta)(q_{\mathrm{rob}}d_{\mathrm{rob}}\alpha^2+q_{\mathrm{vul}}d_{\mathrm{vul}}\beta^2). \tag{A177}$$

By Lemma E.2, we can compute the second term as

$$\mathbb{E}\left[\left\|\frac{1}{N}\boldsymbol{b}^{\top}\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^{\top}\boldsymbol{A}\right\|_1\right]$$
$$=\mathcal{O}\left((d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta+1)\left(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta+\frac{d_{\mathrm{irr}}\gamma}{\sqrt{N}}\right)+d_{\mathrm{irr}}\left(\sqrt{\frac{d_{\mathrm{irr}}}{N}}+1\right)\gamma^2\right). \tag{A178}$$

Finally,

$$\mathbb{E}\left[\frac{1}{N}\boldsymbol{b}^{\top}\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^{\top}\boldsymbol{A}y_{N+1}\boldsymbol{x}_{N+1}\right]-\epsilon\mathbb{E}\left[\left\|\frac{1}{N}\boldsymbol{b}^{\top}\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^{\top}\boldsymbol{A}\right\|_1\right]$$
$$\geq\left(\min_{i\in\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}}}C_i^3\right)(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta)(d_{\mathrm{rob}}\alpha^2+d_{\mathrm{vul}}\beta^2)+\sum_{i\in\mathcal{S}_{\mathrm{rob}}}C_i^2\alpha^2+\sum_{i\in\mathcal{S}_{\mathrm{vul}}}C_i^2\beta^2$$
$$-\left(\max_{i\in\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}}}C_i\right)\left(\max_{i\in\mathcal{S}_{\mathrm{rob}}\cup\mathcal{S}_{\mathrm{vul}}}C_{i,2}\right)(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta)(q_{\mathrm{rob}}d_{\mathrm{rob}}\alpha^2+q_{\mathrm{vul}}d_{\mathrm{vul}}\beta^2)$$
$$+\mathcal{O}\left((d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta+1)\left(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta+\frac{d_{\mathrm{irr}}\gamma}{\sqrt{N}}\right)+d_{\mathrm{irr}}\left(\sqrt{\frac{d_{\mathrm{irr}}}{N}}+1\right)\gamma^2\right). \tag{A179}$$

□

Lemma E.2. If $(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)$ are i.i.d. and follow $\mathcal{D}^{\mathrm{te}}$, then

$$\mathbb{E}\left[\left\|\frac{1}{N}\boldsymbol{b}^{\top}\boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^{\top}\boldsymbol{A}\right\|_1\right]$$
$$=\mathcal{O}\left((d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta+1)\left(d_{\mathrm{rob}}\alpha+d_{\mathrm{vul}}\beta+\frac{d_{\mathrm{irr}}\gamma}{\sqrt{N}}\right)+d_{\mathrm{irr}}\left(\sqrt{\frac{d_{\mathrm{irr}}}{N}}+1\right)\gamma^2\right), \tag{A180}$$

where $\boldsymbol{b} = \mathbf{1}_{d+1}$ and $\boldsymbol{A}^{\top} := [\boldsymbol{I}_d \quad \mathbf{0}_d]$.

Proof. We can rearrange the given expectation as

$$\mathbb{E}\left[\left\|\frac{1}{N}\boldsymbol{b}^\top \boldsymbol{Z}_{\boldsymbol{\Delta}}\boldsymbol{M}\boldsymbol{Z}_{\boldsymbol{\Delta}}^\top \boldsymbol{A}\right\|_1\right] = \mathbb{E}\left[\left\|\frac{1}{N}\boldsymbol{1}_{d+1}^\top \begin{bmatrix}\sum_{n=1}^N \boldsymbol{x}_n\boldsymbol{x}_n^\top & \sum_{n=1}^N y_n\boldsymbol{x}_n \\ \sum_{n=1}^N y_n\boldsymbol{x}_n^\top & N\end{bmatrix}\begin{bmatrix}\boldsymbol{I}_d \\ \boldsymbol{0}_d^\top\end{bmatrix}\right\|_1\right] \tag{A181}$$

$$= \mathbb{E}\left[\left\|\frac{1}{N}\boldsymbol{1}_{d+1}^\top \begin{bmatrix}\sum_{n=1}^N \boldsymbol{x}_n\boldsymbol{x}_n^\top \\ \sum_{n=1}^N y_n\boldsymbol{x}_n^\top\end{bmatrix}\right\|_1\right] \tag{A182}$$

$$= \sum_{i=1}^d \mathbb{E}\left[\left|\frac{1}{N}\sum_{n=1}^N\left(y_n + \sum_{j=1}^d x_{n,j}\right)x_{n,i}\right|\right]. \tag{A183}$$

By the Lyapunov inequality, for $N+1$ i.i.d. random variables $X, X_1, \dots, X_N$,

$$\mathbb{E}\left[\left|\frac{1}{N}\sum_{n=1}^N X_n\right|\right] \le \sqrt{\mathbb{E}\left[\left(\frac{1}{N}\sum_{n=1}^N X_n\right)^2\right]} = \sqrt{\frac{1}{N}\mathbb{E}[X^2] + \frac{N-1}{N}\mathbb{E}[X]^2}. \tag{A184}$$

Thus, using $(\boldsymbol{x}, y) \sim \mathcal{D}^{\mathrm{te}}$,

$$\sum_{i=1}^d \mathbb{E}\left[\left|\frac{1}{N}\sum_{n=1}^N\left(y_n + \sum_{j=1}^d x_{n,j}\right)x_{n,i}\right|\right]$$
$$\le \sum_{i=1}^d \sqrt{\frac{1}{N}\mathbb{E}\left[\left(y + \sum_{j=1}^d x_j\right)^2 x_i^2\right] + \frac{N-1}{N}\mathbb{E}\left[\left(y + \sum_{j=1}^d x_j\right)x_i\right]^2}. \tag{A185}$$

From Lemma E.3, we can compute the second term of using

$$\mathbb{E}\left[\left(y + \sum_{j=1}^d x_j\right)x_i\right] = \mathbb{E}[yx_i] + \sum_{j=1}^d \mathbb{E}[x_jx_i] \tag{A186}$$

$$= \begin{cases}\mathcal{O}(\alpha(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)) & (i \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)) & (i \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}})\end{cases}. \tag{A187}$$

From Lemma E.3, we can compute the first term of using

$$\mathbb{E}\left[\left(y + \sum_{j=1}^d x_j\right)^2 x_i^2\right] = \mathbb{E}[x_i^2] + 2\sum_{j=1}^d \mathbb{E}[yx_jx_i^2] + \sum_{j,k=1}^d \mathbb{E}[x_jx_kx_i^2] \tag{A188}$$

$$= \begin{cases}\mathcal{O}(\alpha^2\{(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)^2 + d_{\mathrm{irr}}\gamma^2\}) & (i \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^2\{(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)^2 + d_{\mathrm{irr}}\gamma^2\}) & (i \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\gamma^2\{(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)^2 + d_{\mathrm{irr}}\gamma^2\}) & (i \in \mathcal{S}_{\mathrm{irr}})\end{cases}. \tag{A189}$$

Thus,

$$\sum_{i=1}^d \sqrt{\frac{1}{N}\mathbb{E}\left[\left(y + \sum_{j=1}^d x_j\right)^2 x_i^2\right] + \frac{N-1}{N}\mathbb{E}\left[\left(y + \sum_{j=1}^d x_j\right)x_i\right]^2}$$
$$= \mathcal{O}\left(d_{\mathrm{rob}}\left(\alpha(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1) + \sqrt{\frac{d_{\mathrm{irr}}}{N}}\alpha\gamma\right)\right.$$
$$\left. + d_{\mathrm{vul}}\left(\beta(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1) + \sqrt{\frac{d_{\mathrm{irr}}}{N}}\beta\gamma\right)\right.$$

$$+ d_{\mathrm{irr}}\left(\gamma^2 + \frac{\gamma}{\sqrt{N}}\Big((d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1) + \sqrt{d_{\mathrm{irr}}}\gamma\Big)\right)\Bigg) \tag{A190}$$

$$= \mathcal{O}\left((d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + 1)\left(d_{\mathrm{rob}}\alpha + d_{\mathrm{vul}}\beta + \frac{d_{\mathrm{irr}}\gamma}{\sqrt{N}}\right) + d_{\mathrm{irr}}\left(\sqrt{\frac{d_{\mathrm{irr}}}{N}} + 1\right)\gamma^2\right). \tag{A191}$$

□

Lemma E.3. If $(\boldsymbol{x}, y) \sim \mathcal{D}^{\mathrm{te}}$, then

(a)

$$\mathbb{E}[x_j x_i] = \begin{cases} \mathcal{O}(\alpha^2) & (i, j \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^2) & (i, j \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\gamma^2) & (i = j) \wedge (i, j \in \mathcal{S}_{\mathrm{irr}}) \\ \mathcal{O}(\alpha\beta) & (i \in \mathcal{S}_{\mathrm{rob}} \wedge j \in \mathcal{S}_{\mathrm{vul}}) \vee (i \in \mathcal{S}_{\mathrm{vul}} \wedge j \in \mathcal{S}_{\mathrm{rob}}) \\ 0 & (i \neq j) \wedge (i \in \mathcal{S}_{\mathrm{irr}} \vee j \in \mathcal{S}_{\mathrm{irr}}) \end{cases}. \tag{A192}$$

(b)

$$\mathbb{E}[y x_j x_i^2] = \begin{cases} \mathcal{O}(\alpha^3) & (i, j \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^3) & (i, j \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\alpha^2\beta) & (i \in \mathcal{S}_{\mathrm{rob}} \wedge j \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\alpha\beta^2) & (i \in \mathcal{S}_{\mathrm{vul}} \wedge j \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\alpha\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}} \wedge j \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}} \wedge j \in \mathcal{S}_{\mathrm{vul}}) \\ 0 & (j \in \mathcal{S}_{\mathrm{irr}}) \end{cases}. \tag{A193}$$

(c)

$$\mathbb{E}[x_j x_k x_i^2] = \begin{cases} \mathcal{O}(\alpha^4) & (i, j, k \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^4) & (i, j, k \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\gamma^4) & (j = k) \wedge (i, j, k \in \mathcal{S}_{\mathrm{irr}}) \\ \mathcal{O}(\alpha^3\beta) & (i \in \mathcal{S}_{\mathrm{rob}}) \wedge \{(j \in \mathcal{S}_{\mathrm{rob}} \wedge k \in \mathcal{S}_{\mathrm{vul}}) \vee (j \in \mathcal{S}_{\mathrm{vul}} \wedge k \in \mathcal{S}_{\mathrm{rob}})\} \\ \mathcal{O}(\alpha\beta^3) & (i \in \mathcal{S}_{\mathrm{vul}}) \wedge \{(j \in \mathcal{S}_{\mathrm{rob}} \wedge k \in \mathcal{S}_{\mathrm{vul}}) \vee (j \in \mathcal{S}_{\mathrm{vul}} \wedge k \in \mathcal{S}_{\mathrm{rob}})\} \\ \mathcal{O}(\alpha^2\beta^2) & (i \in \mathcal{S}_{\mathrm{rob}} \wedge j, k \in \mathcal{S}_{\mathrm{vul}}) \vee (i \in \mathcal{S}_{\mathrm{vul}} \wedge j, k \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\alpha^2\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}} \wedge j, k \in \mathcal{S}_{\mathrm{rob}}) \vee (j = k \wedge j, k \in d_{\mathrm{irr}} \wedge i \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^2\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}} \wedge j, k \in \mathcal{S}_{\mathrm{vul}}) \vee (j = k \wedge j, k \in d_{\mathrm{irr}} \wedge i \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\alpha\beta\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}}) \wedge \{(j \in \mathcal{S}_{\mathrm{rob}} \wedge k \in \mathcal{S}_{\mathrm{vul}}) \vee (j \in \mathcal{S}_{\mathrm{vul}} \wedge k \in \mathcal{S}_{\mathrm{rob}})\} \\ 0 & (j \neq k) \wedge (j \in \mathcal{S}_{\mathrm{irr}} \vee k \in \mathcal{S}_{\mathrm{irr}}) \end{cases}. \tag{A194}$$

Proof. We first note that

$$\mathbb{E}[x_i^2] = \mathbb{E}[(y x_i)^2] = \mathbb{E}[(y x_i - \mathbb{E}[y x_i])^2] + \mathbb{E}[y x_i]^2 = \begin{cases} \mathcal{O}(\alpha^2) & (i \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^2) & (i \in \mathcal{S}_{\mathrm{vul}}) \\ \mathcal{O}(\gamma^2) & (i \in \mathcal{S}_{\mathrm{irr}}) \end{cases}, \tag{A195}$$

$$\mathbb{E}[y x_i^3] = \mathbb{E}[(y x_i)^3] \tag{A196}$$

$$= \mathbb{E}[(y x_i - \mathbb{E}[y x_i])^3] + 3\mathbb{E}[(y x_i)^2]\mathbb{E}[y x_i] - 2\mathbb{E}[y x_i]^3 \tag{A197}$$

$$= \begin{cases} \mathcal{O}(\alpha^3) & (i \in \mathcal{S}_{\mathrm{rob}}) \\ \mathcal{O}(\beta^3) & (i \in \mathcal{S}_{\mathrm{vul}}) \\ 0 & (i \in \mathcal{S}_{\mathrm{irr}}) \end{cases}, \tag{A198}$$

$$\mathbb{E}[x_i^4] = \mathbb{E}[(y x_i - \mathbb{E}[y x_i])^4] + 4\mathbb{E}[y x_i^3]\mathbb{E}[y x_i] - 6\mathbb{E}[x_i^2]\mathbb{E}[y x_i]^2 + 3\mathbb{E}[y x_i]^4 \tag{A199}$$

$$= \begin{cases} \mathcal{O}(\alpha^4) & (i \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta^4) & (i \in \mathcal{S}_{\text{vul}}) \\ \mathcal{O}(\gamma^4) & (i \in \mathcal{S}_{\text{irr}}) \end{cases}. \tag{A200}$$

(a) For $(i \neq j) \wedge (i \in \mathcal{S}_{\text{irr}} \vee j \in \mathcal{S}_{\text{irr}})$, $\mathbb{E}[x_j x_i] = \mathbb{E}[x_j]\mathbb{E}[x_i] = 0$. Using the Cauthy-Schwarz inequality,

$$\mathbb{E}[x_j x_i] \leq \sqrt{\mathbb{E}[x_j^2]}\sqrt{\mathbb{E}[x_i^2]} \tag{A201}$$

$$= \begin{cases} \mathcal{O}(\alpha^2) & (i, j \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta^2) & (i, j \in \mathcal{S}_{\text{vul}}) \\ \mathcal{O}(\gamma^2) & (i, j \in \mathcal{S}_{\text{irr}}) \wedge (i = j) \\ \mathcal{O}(\alpha\beta) & (i \in \mathcal{S}_{\text{rob}} \wedge j \in \mathcal{S}_{\text{vul}}) \vee (i \in \mathcal{S}_{\text{vul}} \wedge j \in \mathcal{S}_{\text{rob}}) \end{cases}. \tag{A202}$$

(b) For $j \in \mathcal{S}_{\text{irr}}, j = i$, $\mathbb{E}[y x_j x_i^2] = \mathbb{E}[y]\mathbb{E}[x_i^3] = 0$. For $j \in \mathcal{S}_{\text{irr}}, j \neq i$, $\mathbb{E}[y x_j x_i^2] = \mathbb{E}[x_j]\mathbb{E}[y x_i^2] = 0$. Using the Cauthy-Schwarz inequality,

$$\mathbb{E}[y x_j x_i^2] \leq \sqrt{\mathbb{E}[x_j^2]}\sqrt{\mathbb{E}[x_i^4]} = \begin{cases} \mathcal{O}(\alpha^3) & (i, j \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta^3) & (i, j \in \mathcal{S}_{\text{vul}}) \\ \mathcal{O}(\alpha^2\beta) & (i \in \mathcal{S}_{\text{rob}} \wedge j \in \mathcal{S}_{\text{vul}}) \\ \mathcal{O}(\alpha\beta^2) & (i \in \mathcal{S}_{\text{vul}} \wedge j \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\alpha\gamma^2) & (i \in \mathcal{S}_{\text{irr}} \wedge j \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta\gamma^2) & (i \in \mathcal{S}_{\text{irr}} \wedge j \in \mathcal{S}_{\text{vul}}) \end{cases}. \tag{A203}$$

(c) For $(j \neq k) \wedge (j \in \mathcal{S}_{\text{irr}} \vee k \in \mathcal{S}_{\text{irr}})$, $\mathbb{E}[x_j x_k x_i^2] = 0$. For $j = k$, using the Cauthy-Schwarz inequality,

$$\mathbb{E}[x_j x_k x_i^2] \leq \sqrt{\mathbb{E}[x_j^4]}\sqrt{\mathbb{E}[x_i^4]} = \begin{cases} \mathcal{O}(\gamma^4) & (j = k) \wedge (i, j, k \in \mathcal{S}_{\text{irr}}) \\ \mathcal{O}(\alpha^2\gamma^2) & (j = k) \wedge (j, k \in d_{\text{irr}} \wedge i \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta^2\gamma^2) & (j = k) \wedge (j, k \in d_{\text{irr}} \wedge i \in \mathcal{S}_{\text{vul}}) \end{cases}. \tag{A204}$$

Using the Cauthy-Schwarz inequality,

$$\mathbb{E}[x_j x_k x_i^2]$$

$$\leq \sqrt{\mathbb{E}[x_j^2]}\sqrt{\mathbb{E}[x_k^2]}\sqrt{\mathbb{E}[x_i^4]} \tag{A205}$$

$$= \begin{cases} \mathcal{O}(\alpha^4) & (i, j, k \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta^4) & (i, j, k \in \mathcal{S}_{\text{vul}}) \\ \mathcal{O}(\alpha^3\beta) & (i \in \mathcal{S}_{\text{rob}}) \wedge \{(j \in \mathcal{S}_{\text{rob}} \wedge k \in \mathcal{S}_{\text{vul}}) \vee (j \in \mathcal{S}_{\text{vul}} \wedge k \in \mathcal{S}_{\text{rob}})\} \\ \mathcal{O}(\alpha\beta^3) & (i \in \mathcal{S}_{\text{vul}}) \wedge \{(j \in \mathcal{S}_{\text{rob}} \wedge k \in \mathcal{S}_{\text{vul}}) \vee (j \in \mathcal{S}_{\text{vul}} \wedge k \in \mathcal{S}_{\text{rob}})\} \\ \mathcal{O}(\alpha^2\beta^2) & (i \in \mathcal{S}_{\text{rob}} \wedge j, k \in \mathcal{S}_{\text{vul}}) \vee (i \in \mathcal{S}_{\text{vul}} \wedge j, k \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\alpha^2\gamma^2) & (i \in \mathcal{S}_{\text{irr}} \wedge j, k \in \mathcal{S}_{\text{rob}}) \\ \mathcal{O}(\beta^2\gamma^2) & (i \in \mathcal{S}_{\text{irr}} \wedge j, k \in \mathcal{S}_{\text{vul}}) \\ \mathcal{O}(\alpha\beta\gamma^2) & (i \in \mathcal{S}_{\text{irr}}) \wedge \{(j \in \mathcal{S}_{\text{rob}} \wedge k \in \mathcal{S}_{\text{vul}}) \vee (j \in \mathcal{S}_{\text{vul}} \wedge k \in \mathcal{S}_{\text{rob}})\} \end{cases}. \tag{A206}$$

□

F PROOF OF THEOREM 3.7 (TRADE-OFF)

Theorem 3.7 (Accuracy–robustness trade-off)**.** Assume $|\mathcal{S}_{\text{rob}}| = 1$, $|\mathcal{S}_{\text{vul}}| = d-1$, and $|\mathcal{S}_{\text{irr}}| = 0$. In addition to Assumption 3.2, for $(\boldsymbol{x}, y) \sim \mathcal{D}^{\text{te}}$, suppose that $y x_i$ takes α with probability $p > 0.5$ and $-\alpha$ with probability $1-p$ for $i \in \mathcal{S}_{\text{rob}}$. Moreover, $y x_i$ takes β with probability one for $i \in \mathcal{S}_{\text{vul}}$. Let $\tilde{f}(\boldsymbol{P}, \boldsymbol{Q}) := \mathbb{E}_{\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N \overset{\text{i.i.d.}}{\sim} \mathcal{D}^{\text{te}}}[y_{N+1}[f(\boldsymbol{Z}_{\boldsymbol{0}}; \boldsymbol{P}, \boldsymbol{Q})]_{d+1,N+1}]$. Then, there exist strictly positive functions $g_1(d,\alpha,\beta)$ and $g_2(d,\alpha,\beta)$ such that

$$\tilde{f}(\boldsymbol{P}^{\mathrm{std}},\boldsymbol{Q}^{\mathrm{std}}) = \begin{cases} g_1(d,\alpha,\beta)(\alpha+(d-1)\beta) & (\text{w.p.} \quad p) \\ g_1(d,\alpha,\beta)(-\alpha+(d-1)\beta) & (\text{w.p.} \quad 1-p) \end{cases}, \tag{10}$$

$$\tilde{f}(\boldsymbol{P}^{\mathrm{adv}},\boldsymbol{Q}^{\mathrm{adv}}) \le g_2(d,\alpha,\beta)\{-(2p-1)\alpha^2+(d-1)\beta^2\} \quad (\text{w.p.} \quad 1-p). \tag{11}$$

Proof. Using $\boldsymbol{b}$ and $\boldsymbol{A}$ defined in Appendix D, we can rearrange $\tilde{f}(\boldsymbol{P},\boldsymbol{Q})$ as

$$\tilde{f}(\boldsymbol{P},\boldsymbol{Q}) := \mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^N}[y_{N+1}[f(\boldsymbol{Z_0};\boldsymbol{P},\boldsymbol{Q})]_{d+1,N+1}] \tag{A207}$$

$$= \frac{1}{N}\boldsymbol{b}^\top \mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^N}[\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top]\boldsymbol{A}y_{N+1}\boldsymbol{x}_{N+1}. \tag{A208}$$

Standard Transformer. Similarly to the proof of Theorem 3.5, using some positive function $g(d,\alpha,\beta)>0$, we can represent $\mathbb{E}[\boldsymbol{b}^\top\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\boldsymbol{A}/N] = [g(d,\alpha,\beta) \ \cdots \ g(d,\alpha,\beta)]$. Thus,

$$\frac{1}{N}\boldsymbol{b}\mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^N}[\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top]\boldsymbol{A}y_{N+1}\boldsymbol{x}_{N+1} = \begin{bmatrix} g(d,\alpha,\beta) \\ \vdots \\ g(d,\alpha,\beta)\end{bmatrix}^\top y_{N+1}\boldsymbol{x}_{N+1} \tag{A209}$$

$$= g(d,\alpha,\beta)y_{N+1}\sum_{i=1}^d x_{N+1,i} \tag{A210}$$

$$= \begin{cases} \alpha+(d-1)\beta & (\text{w.p.} \quad p) \\ -\alpha+(d-1)\beta & (\text{w.p.} \quad 1-p) \end{cases}. \tag{A211}$$

Adversarially Trained Transformer. Now,

$$\frac{1}{N}\mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^N}[\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top]$$

$$= \begin{bmatrix} \mathbb{E}[(y\boldsymbol{x})(y\boldsymbol{x}^\top)] & \mathbb{E}[y\boldsymbol{x}] \\ \mathbb{E}[y\boldsymbol{x}^\top] & 1 \end{bmatrix} \tag{A212}$$

$$= \begin{bmatrix} \alpha^2 & (2p-1)\alpha\beta & \cdots & (2p-1)\alpha\beta & (2p-1)\alpha \\ (2p-1)\alpha\beta & \beta^2 & \cdots & \beta^2 & \beta \\ (2p-1)\alpha\beta & \beta^2 & \cdots & \beta^2 & \beta \\ \vdots & & & & \\ (2p-1)\alpha\beta & \beta^2 & \cdots & \beta^2 & \beta \\ (2p-1)\alpha & \beta & \cdots & \beta & 1 \end{bmatrix}. \tag{A213}$$

Thus,

$$\frac{1}{N}\boldsymbol{b}^\top\mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^N}[\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top]\boldsymbol{A} = \begin{bmatrix} \alpha\{\alpha+(d-1)(2p-1)\beta+(2p-1)\} \\ \beta\{(2p-1)\alpha+(d-1)\beta+1\} \\ \vdots \\ \beta\{(2p-1)\alpha+(d-1)\beta+1\} \end{bmatrix}^\top. \tag{A214}$$

Therefore,

$$\frac{1}{N}\boldsymbol{b}^\top\mathbb{E}_{\{(\boldsymbol{x}_n,y_n)\}_{n=1}^N}[\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top]\boldsymbol{A}y_{N+1}\boldsymbol{x}_{N+1}$$

$$= \begin{bmatrix} \alpha\{\alpha+(d-1)(2p-1)\beta+(2p-1)\} \\ \beta\{(2p-1)\alpha+(d-1)\beta+1\} \\ \vdots \\ \beta\{(2p-1)\alpha+(d-1)\beta+1\} \end{bmatrix}^\top \begin{bmatrix} y_{N+1}x_{N+1,1} \\ \beta \\ \vdots \\ \beta \end{bmatrix} \tag{A215}$$

$$= \begin{cases} \alpha^2\{\alpha+(d-1)(2p-1)\beta+(2p-1)\} \\ \quad +(d-1)\beta^2\{(2p-1)\alpha+(d-1)\beta+1\} & (\text{w.p.} \quad p) \\ -\alpha^2\{\alpha+(d-1)(2p-1)\beta+(2p-1)\} \\ \quad +(d-1)\beta^2\{(2p-1)\alpha+(d-1)\beta+1\} & (\text{w.p.} \quad 1-p) \end{cases}. \tag{A216}$$

In particular,

$$
\begin{aligned}
&-\alpha^2\{\alpha+(d-1)(2p-1)\beta+(2p-1)\}+(d-1)\beta^2\{(2p-1)\alpha+(d-1)\beta+1\} \\
&=\{(2p-1)\alpha+(d-1)\beta+1\}(-C\alpha^2+(d-1)\beta^2), \qquad \text{(A217)}
\end{aligned}
$$

where

$$
\begin{aligned}
C=\frac{\alpha+(d-1)(2p-1)\beta+(2p-1)}{(2p-1)\alpha+(d-1)\beta+1} &> \frac{(2p-1)^2\alpha+(d-1)(2p-1)\beta+(2p-1)}{(2p-1)\alpha+(d-1)\beta+1} \qquad \text{(A218)}\\
&=2p-1. \qquad \text{(A219)}
\end{aligned}
$$

□

G Proof of Theorem G.1 (Need for Larger Sample Size)

Theorem G.1 (Need for Larger Sample Size)**.** Assume the same assumptions in Theorem 3.7. Then,

$$\mathbb{E}_{\boldsymbol{x}_{N+1},y_{N+1}}[y_{N+1}[f(\boldsymbol{Z_0};\boldsymbol{P}^{\text{std}},\boldsymbol{Q}^{\text{std}})]_{d+1,N+1}]>0 \qquad (\text{w.p. at least } 1-e^{-pN}). \qquad \text{(A220)}$$

In addition, suppose that there exists a constant $0<C<1$ such that $(d-1)\beta+1<C\alpha$. Moreover, assume that N is an even number. Then, as $p\to\frac{1}{2}$ with $p>\frac{1}{2}$, for $4\le N\le\frac{2}{C}$,

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{x}_{N+1},y_{N+1}}[y_{N+1}[f(\boldsymbol{Z_0};\boldsymbol{P}^{\text{adv}},\boldsymbol{Q}^{\text{adv}})]_{d+1,N+1}]>0\\
&\left(\text{w.p. at most } 1-\frac{0.483}{\sqrt{N}}<1-e^{-pN}\right). \qquad \text{(A221)}
\end{aligned}
$$

Proof. Using $\boldsymbol{b}$ and $\boldsymbol{A}$ defined in Appendix D, we can calculate

$$\mathbb{E}_{\boldsymbol{x}_{N+1},y_{N+1}}[y_{N+1}[f(\boldsymbol{Z_0};\boldsymbol{P},\boldsymbol{Q})]_{d+1,N+1}]=\frac{1}{N}\boldsymbol{b}^\top\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\boldsymbol{A}\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}]. \qquad \text{(A222)}$$

Now,

$$
\begin{aligned}
&\frac{1}{N}\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\\
&=\begin{bmatrix}
\alpha^2 & \frac{\beta}{N}\sum_{n=1}^N y_n x_{n,1} & \cdots & \frac{\beta}{N}\sum_{n=1}^N y_n x_{n,1} & \frac{1}{N}\sum_{n=1}^N y_n x_{n,1}\\
\frac{\beta}{N}\sum_{n=1}^N y_n x_{n,1} & \beta^2 & \cdots & \beta^2 & \beta\\
\frac{\beta}{N}\sum_{n=1}^N y_n x_{n,1} & \beta^2 & \cdots & \beta^2 & \beta\\
\vdots & & & & \\
\frac{\beta}{N}\sum_{n=1}^N y_n x_{n,1} & \beta^2 & \cdots & \beta^2 & \beta\\
\frac{1}{N}\sum_{n=1}^N y_n x_{n,1} & \beta & \cdots & \beta & 1
\end{bmatrix}. \qquad \text{(A223)}
\end{aligned}
$$

Standard Transformer. From the configuration of $\boldsymbol{b}$ and $\boldsymbol{A}$, all the entries of $\boldsymbol{b}^\top\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\boldsymbol{A}$ are the same. Since all the entries of $\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}]$ are positive, with some positive function $g(d,\alpha,\beta)>0$,

$$\frac{1}{N}\boldsymbol{b}^\top\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\boldsymbol{A}\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}]=g(d,\alpha,\beta)\frac{1}{N}\boldsymbol{1}_{d+1}^\top\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\boldsymbol{1}_{d+1}. \qquad \text{(A224)}$$

Now,

$$
\begin{aligned}
&\frac{1}{N}\boldsymbol{1}_{d+1}^\top\boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top\boldsymbol{1}_{d+1}\\
&=(d-1)^2\beta^2+2(d-1)\beta+1+\alpha^2+\frac{2}{N}\sum_{n=1}^N y_n x_{n,1}+2(d-1)\frac{\beta}{N}\sum_{n=1}^N y_n x_{n,1} \qquad \text{(A225)}\\
&=\{(d-1)\beta+1\}^2+\alpha^2+\frac{2\{(d-1)\beta+1\}}{N}\sum_{n=1}^N y_n x_{n,1} \qquad \text{(A226)}
\end{aligned}
$$

$$= [\{(d-1)\beta+1\}-\alpha]^2 + \frac{2\{(d-1)\beta+1\}}{N}\sum_{n=1}^{N}(\alpha+y_n x_{n,1}) \tag{A227}$$

$$> 0 \qquad (\text{w.p. at least } 1-(1-p)^N > 1-e^{-pN}). \tag{A228}$$

Adversarially Trained Transformer. Note that $\mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}] = [(2p-1)\alpha \;\; \beta \;\; \cdots \;\; \beta]$. Thus,

$$\begin{aligned}
&\frac{1}{N}\mathbf{1}_{d+1}^\top \boldsymbol{Z_0}\boldsymbol{M}\boldsymbol{Z_0}^\top \boldsymbol{I}_d \mathbb{E}[y_{N+1}\boldsymbol{x}_{N+1}] \\
&= (2p-1)\alpha\left(\alpha^2 + (d-1)\frac{\beta}{N}\sum_{n=1}^{N} y_n x_{n,1} + \frac{1}{N}\sum_{n=1}^{N} y_n x_{n,1}\right) \\
&\quad + (d-1)\beta\left(\frac{\beta}{N}\sum_{n=1}^{N} y_n x_{n,1} + (d-1)\beta^2 + \beta\right) \qquad (A229) \\
&= [(2p-1)\alpha^3 + (d-1)\beta^2\{(d-1)\beta+1\}] \\
&\quad + [(2p-1)\alpha\{(d-1)\beta+1\} + (d-1)\beta^2]\frac{1}{N}\sum_{n=1}^{N} y_n x_{n,1}. \qquad (A230)
\end{aligned}$$

This indicates $\mathbb{E}_{\boldsymbol{x}_{N+1},y_{N+1}}[y_{N+1}[f(\boldsymbol{Z_0};\boldsymbol{P}^{\text{adv}},\boldsymbol{Q}^{\text{adv}})]_{d+1,N+1}] > 0$ only if

$$\frac{1}{N}\sum_{n=1}^{N} y_n x_{n,1} > -\frac{(2p-1)\alpha^3 + (d-1)\beta^2\{(d-1)\beta+1\}}{(2p-1)\alpha\{(d-1)\beta+1\} + (d-1)\beta^2}. \tag{A231}$$

Representing $y_n x_{n,1} = \alpha(2X_n - 1)$ with X_n taking 1 with probability p and 0 with probability $1-p$,

$$\begin{aligned}
&\frac{1}{N}\sum_{n=1}^{N}\alpha(2X_n-1) > -\frac{(2p-1)\alpha^3 + (d-1)\beta^2\{(d-1)\beta+1\}}{(2p-1)\alpha\{(d-1)\beta+1\} + (d-1)\beta^2} \\
\iff &\sum_{n=1}^{N} X_n > \frac{N}{2}\left(1 - \frac{1}{\alpha}\frac{(2p-1)\alpha^3 + (d-1)\beta^2\{(d-1)\beta+1\}}{(2p-1)\alpha\{(d-1)\beta+1\} + (d-1)\beta^2}\right). \qquad (A232)
\end{aligned}$$

Let $Y \sim B(N,p)$, where $B(N,p)$ is the Binomial distribution. Consider the following probability:

$$\mathbb{P}_{Y\sim B(N,p)}\left[Y > \frac{N}{2}\left(1 - \frac{1}{\alpha}\frac{(2p-1)\alpha^3 + (d-1)\beta^2\{(d-1)\beta+1\}}{(2p-1)\alpha\{(d-1)\beta+1\} + (d-1)\beta^2}\right)\right]. \tag{A233}$$

When $p \to 1/2$,

$$\begin{aligned}
&\mathbb{P}_{Y\sim B(N,p)}\left[Y > \frac{N}{2}\left(1 - \frac{1}{\alpha}\frac{(2p-1)\alpha^3 + (d-1)\beta^2\{(d-1)\beta+1\}}{(2p-1)\alpha\{(d-1)\beta+1\} + (d-1)\beta^2}\right)\right] \\
&\to \mathbb{P}_{Y\sim B(N,1/2)}\left[Y > \frac{N}{2}\left(1 - \frac{(d-1)\beta+1}{\alpha}\right)\right] \qquad (A234) \\
&\le \mathbb{P}_{Y\sim B(N,1/2)}\left[Y > \frac{N}{2}(1-C)\right] \qquad (A235) \\
&\le \mathbb{P}_{Y\sim B(N,1/2)}\left[Y > \frac{N}{2} - 1\right]. \qquad (A236)
\end{aligned}$$

From Ash (1990), for an integer $0 < k < N/2$,

$$\mathbb{P}_{Y\sim B(N,1/2)}[Y \le k] \ge \frac{1}{\sqrt{8N\frac{k}{N}(1-\frac{k}{N})}}\exp\left(-ND\left(\frac{k}{N}\,/\!/\,\frac{1}{2}\right)\right), \tag{A237}$$

where D is the Kullback–Leibler divergence. Substituting $k = \frac{N}{2} - 1$,

$$\mathbb{P}_{Y\sim B(N,1/2)}\left[Y \le \frac{N}{2} - 1\right]$$

$$\geq \frac{1}{\sqrt{8N(\frac{1}{2}-\frac{1}{N})\{1-(\frac{1}{2}-\frac{1}{N})\}}} \exp\left(-ND\left(\frac{1}{2}-\frac{1}{N} \,/\!/\, \frac{1}{2}\right)\right) \tag{A238}$$

$$= \frac{1}{\sqrt{2(1-\frac{4}{N^2})}} \frac{1}{\sqrt{N}} \exp\left(-ND\left(\frac{1}{2}-\frac{1}{N} \,/\!/\, \frac{1}{2}\right)\right). \tag{A239}$$

Note that

$$D\left(\frac{1}{2}-\frac{1}{N} \,/\!/\, \frac{1}{2}\right) = \frac{1}{2}\left\{\left(1-\frac{2}{N}\right)\ln\left(1-\frac{2}{N}\right)+\left(1+\frac{2}{N}\right)\ln\left(1+\frac{2}{N}\right)\right\}. \tag{A240}$$

For $N \geq 4$,

$$\frac{1}{\sqrt{2(1-\frac{4}{N^2})}} \exp\left(-ND\left(\frac{1}{2}-\frac{1}{N} \,/\!/\, \frac{1}{2}\right)\right) > 0.483. \tag{A241}$$

In summary,

$$\mathbb{P}_{Y\sim B(N,1/2)}\left[Y > \frac{N}{2}-1\right] = 1-\mathbb{P}_{Y\sim B(N,1/2)}\left[Y \leq \frac{N}{2}-1\right] \leq 1-\frac{0.483}{\sqrt{N}}. \tag{A242}$$

□