Better Representations via Adversarial Training in Pre-Training: A Theoretical Perspective

Yue Xing
Michigan State University

Xiaofeng Lin
University of California, Los Angeles

Qifan Song
Purdue University

Yi Xu
Amazon Search-M5

Belinda Zeng
Amazon Search-M5

Guang Cheng
University of California, Los Angeles
Amazon Search-M5

Abstract

Pre-training is known to generate universal representations for downstream tasks in large-scale deep learning such as large language models. Existing literature, e.g., Kim et al. (2020), empirically observe that the downstream tasks can inherit the adversarial robustness of the pre-trained model. We provide theoretical justifications for this robustness inheritance phenomenon. Our theoretical results reveal that feature purification plays an important role in connecting the adversarial robustness of the pre-trained model and the downstream tasks in two-layer neural networks. Specifically, we show that (i) with adversarial training, each hidden node tends to pick only one (or a few) feature; (ii) without adversarial training, the hidden nodes can be vulnerable to attacks. This observation is valid for both supervised pre-training and contrastive learning. With purified nodes, it turns out that clean training is enough to achieve adversarial robustness in downstream tasks.

1 Introduction

Adversarial training is a popular way to improve the adversarial robustness of modern machine learning models. However, compared to clean training, the computation cost of adversarial training is much higher. For example, using a single GPU to train ResNet18 for CIFAR-10, clean training takes 1 hour, but adversarial training uses 20 hours for 200 epochs (Rice et al., 2020).

One possible way to train an adversarially robust neural network with a lower cost is to utilize pre-trained models. That is, we use a large amount of (possibly un-labelled) pre-training data to first train a "general-purpose" neural network model; then, for any specific downstream task, we only need to adapt the last one or two layers according to the (often labelled) downstream data. The computation burden is then moved from downstream users to the pre-training phase. Such a strategy has been widely adopted in the training of large language models such as the GPT series. Please see more discussions in the recent review on foundation model (Bommasani et al., 2021).

If the statistical properties of pre-trained models can be inherited, then the pre-training strategy can also greatly simplify the training of robust downstream models, as long as the pre-trained models possess proper adversarial robustness. Recent literature, e.g., Zhao et al. (2022) shows that clean pre-training can improve the sample efficiency of the downstream tasks, while for adversarial training, it is empirically observed such an inheritance of robustness from pre-training models to downstream task training (Shafahi et al., 2019; Chen et al., 2021; Salman et al., 2020; Deng et al., 2021b; Zhang et al., 2021; Kim et al., 2020; Fan et al., 2021). Unlike the existing works, this paper aims to provide theoretical validation for this robustness inheritance phenomenon.

While most theoretical studies of adversarial training are from statistical/optimization perspectives, Allen-Zhu and Li (2020) studies how adversarial training improves supervised learning in neural networks via *feature purification*. The observed data can be viewed

Proceedings of the 27$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume TBD.

as a mixture of semantic features, and the response is directly related to the features rather than the observed data. It is justified that in clean training, each node learns a mixture of features, i.e., *no feature purification*. Rather, the nodes will be purified in the adversarial training in the sense that each only learns one or a few features, i.e., *feature purification happens*.

Different from Allen-Zhu and Li (2020) which studies supervised learning, this work aims to know whether the benefit of feature purification appear in self-supervised pre-training methods, e.g., contrastive learning. In addition, after obtaining a pre-trained robust model, we wonder how the downstream task inherits the robustness using adversarial pre-training.

Our contributions are as follows:

(1) We provide a theoretical framework to verify that, the design of adversarial loss promotes feature purification. Beyond the work of Allen-Zhu and Li (2020) which studies the evolution of the training trajectory given a specific optimizer, we directly consider the optimal solution to focus on the best possible performance of adversarial training regardless of the optimization algorithm. For the class of neural networks we consider, there are many possible optimal models to minimize the clean population risk, but only those with minimal adversarial loss achieve feature purification. (Section 4)

(2) We extend our analysis to contrastive learning and verify that many non-robust models achieve the best clean performance while the robust ones have purified hidden nodes. An interesting observation is that adversarial training purifies the neural network via negative (dissimilar) pairs of data, and the loss of positive (similar) pairs of data is almost resistant to adversarial attack. This is a different observation compared to Allen-Zhu and Li (2020). (Section 5)

(3) Our results also show that when the pre-trained model perfectly purifies the hidden nodes, we can achieve good model robustness when the downstream tasks are trained using clean training. (Section 6)

2 Related Works

Feature Purification and Better Representation Some related literature touches on similar questions as our targets but with a different purpose from ours. Wen and Li (2021) shows that contrastive learning can purify features using RandomMask. A detailed discussion on the advantages/disadvantages of adversarial training and RandomMask can be found in Section 5.4. Another related work is Deng et al. (2021b), which shows that adversarial training helps select better features from individual tasks. This is different from ours as it does not work on nonlinear neural networks.

Adversarial training There are fruitful studies in the area of adversarial training. For methodology, there are many techniques, e.g., Goodfellow et al. (2015); Zhang et al. (2019); Wang et al. (2019b); Cai et al. (2018); Zhang et al. (2020a); Carmon et al. (2019); Gowal et al. (2021); Mo et al. (2022); Wang et al. (2022). Theoretical investigations have also been conducted for adversarial training from different perspectives. For instance, Chen et al. (2020a); Javanmard et al. (2020); Taheri et al. (2021); Yin et al. (2018); Raghunathan et al. (2019); Najafi et al. (2019); Min et al. (2020); Hendrycks et al. (2019); Dan et al. (2020); Wu et al. (2020b); Javanmard and Mehrabi (2021); Deng et al. (2021a); Javanmard and Soltanolkotabi (2022) study the statistical properties of adversarial training, Sinha et al. (2018); Wang et al. (2019a); Xing et al. (2021b,a); Xiao et al. (2022a,b) study the optimization aspect of adversarial training, Zhang et al. (2020b); Wu et al. (2020a); Xiao et al. (2021) work on theoretical issues related to adversarial training with deep learning.

Contrastive learning Contrastive learning is a popular self-supervised learning algorithm. It uses unlabeled images to train representations that distinguish different images invariant to non-semantic transformations (Mikolov et al., 2013; Oord et al., 2018; Arora et al., 2019; Dai and Lin, 2017; Chen et al., 2020b; Tian et al., 2020; Chen et al., 2020b; Khosla et al., 2020; HaoChen et al., 2021; Chuang et al., 2020; Xiao et al., 2020; Li et al., 2020). Beside empirical studies, there are also many theoretical studies, e.g., Saunshi et al. (2019); HaoChen et al. (2021, 2022); Shen et al. (2022); HaoChen and Ma (2022); Saunshi et al. (2022). Other related studies in adversarial training with contrastive learning can also be found in Alayrac et al. (2019); Ho and Nvasconcelos (2020); Jiang et al. (2020); Cemgil et al. (2019); Petrov and Kwiatkowska (2022); Nguyen et al. (2022).

3 Model Setups

This section defines data generation model, neural network, and adversarial training for supervised learning.

3.1 Data Generation Model

We consider the following data generation model. There exists some underlying **true features** $X \in \mathbb{R}^d$, such that $Z = MX + \xi$ for a unitary matrix M, and the response Y is directly determined by X. However, instead of observing X, we observe transformed noisy features $Z \in \mathbb{R}^d$ and the response $Y \in \mathbb{R}$. An illustration can be found in Figure 1.

The relationship between X and Y is as follows:

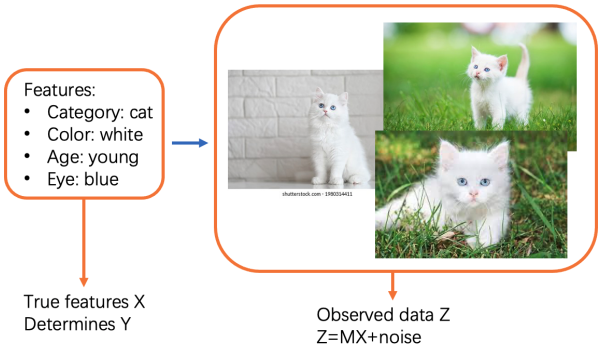

Figure 1: A proof-of-concept example of the Sparse Coding Model. For the categorical features, one can reshape it to a sparse feature vector.

(1) Regression: $Y = \theta_0^\top X + \varepsilon$ for Gaussian noise ε.

(2) Classification: $Y \sim Bern(1/(1+\exp(\theta_0^\top X)))$.

We impose the following assumption on the data.

Assumption 3.1 (Sparse coding model)**.** The model of (X, Z, Y) satisfies the following conditions:

(1) The coordinates of X are i.i.d. symmetric variables, and $|X_i| \in \{0\} \cup [1/\sqrt{k}, 1]$ for some sparsity parameter k. Moreover, $P(|X_i| \neq 0) = \Theta(k/d)$, $\mathbb{E}X_i^2 = \Theta(1/d)$, $\mathbb{E}|X_i| = \Theta(1/\sqrt{k})$, and $\mathbb{E}|X_i|^3 = \Theta(1/(d\sqrt{k}))$.

(2) The noise ξ follows i.i.d. $N(0, \zeta^2 I_d/d)$ for $\zeta > 0$.

(3) Each coordinate of θ_0 satisfies $(\theta_0)_i = \Theta(1)$.

In Assumption 3.1 (1), we assume X is a sparse signal. In general, there are $O(k)$ **active** features (i.e., non-zero X_i) in a realization of X. In the later results, we always assume $k \ll d$. In addition, together with (2), we have $\|X\| = O_p(1)$, $\|Z\| = O_p(1)$, and $\|\xi\| = O_p(1)$, i.e., the total magnitudes of the features, observed data, and noise are comparable. Assumption 3.1 (3) indicates that all features are important in determining Y.

Assumption 3.1 is similar to the model considered in Allen-Zhu and Li (2020). We impose a constraint on the third moment of X_i to use concentration bounds in Lemma F.2, and assume a symmetric distribution to ease the contrastive learning (Lemma 5.1). Similar sparse coding models have a long history in literature, e.g., Hyvarinen et al. (1998).

3.2 Two-Layer Neural Network

We use a two-layer neural network to fit the model. In particular, given an input z,

$$f_{W,b}(z) = \sum_{h=1}^{H} a_h \sigma(z^\top W_h, b_h),$$

where $a_h = 1$ for all h. In the pre-training stage, we use lazy training and do not update a_h. The vector $b = (b_1, b_2, \ldots, b_H)$ is the intercept[1] term in each node, and $W = (W_1 \mid \ldots \mid W_H)$ is the coefficient matrix. In later sections, besides using W_h as the hth column of W, we also define $W_{i,:}$ as the ith row and use $W_{:,h}$ as the hth column of W to avoid confusion when needed. Similar notations will be used for other matrices.

To simplify the derivation, we mainly consider the following activation function, for $v, e \in \mathbb{R}$,

$$\sigma(v, e) = v1\{|v| \geq e\}. \tag{1}$$

Compared to an identity mapping, (1) has an extra "gate parameter" e to screen out weak signals. When $|v| > e$, the hidden node is **activated**.

3.3 Adversarial Training

We consider $\mathcal{L}_2$ fast gradient attack (FGM) with attack strength ϵ, i.e., given the current model f and loss function l, for each sample (z, y), the attack is

$$\delta_2 = \epsilon(\partial l/\partial z)/\|\partial l/\partial z\|,$$

where $\|\cdot\|$ is the $\mathcal{L}_2$ norm. In the models we consider, when approaching the optimal solution to minimize clean/adversarial loss, the FGM is the best attack.

Besides the adversarial attack, we also define the corresponding adversarial loss as $l_\epsilon(z, y; f) = l(z+\delta, y; f)$. Besides $\mathcal{L}_2$ attack, some discussions can also be found in the appendix Section C if δ is the $\mathcal{L}_\infty$ attack (i.e., Fast Gradient Signed Method (FGSM)). When $\epsilon = 0$, the loss l_0 is reduced to l, and represents the clean loss. Details of contrastive learning are in Section 5.

For clean and adversarial training in this paper, we use "clean training" to minimize the clean loss and "adversarial training" to minimize the adversarial loss.

4 Feature Purification

This section aims to provide basic intuitions on (i) why the activation function (1) and ReLU are preferred over linear networks, and (ii) why adversarial training can purify features. While the high-level ideas are similar to Allen-Zhu and Li (2020), we restate them via different technical tools so that it can be carried over to the later sections of contrastive learning and downstream study.

4.1 Screen Out Noise

The basic rationale of why the activation function in (1) (or ReLU) is that it can screen out the noise ξ. Intuitively, the noise ξ in Z only contributes to a negligible noise in hidden nodes, which can be screened out by a proper "gate parameter" b_h.

[1]We use the term "intercept" to describe b to avoid confusion with "bias" in statistics.

To explain more details, we introduce the notations first. Based on the data generation model, assume z is a realization of Z, then we can define $U = M^\top W \in \mathbb{R}^{d\times H}$ and rewrite $f_{W,b}(z)$ as

$$f_{W,b}(z) = \sigma(z^\top W, b)a = \sigma(x^\top U + \xi W, b)a.$$

To interpret U, for each hidden node h, the column U_h represents the strength of each feature in the hidden node. Note that the noise $\xi^\top W_h \sim N(0, \zeta^2\|U_h\|^2/d)$. When $b_h \gg \|U_h\|/\sqrt{d}$, the noise alone is not able to activate the hidden node. On the other hand, for an active feature $X_i \neq 0$, we have $|U_{i,h}X_i| \geq |U_{i,j}|/\sqrt{k}$, which can be much larger than $\zeta\|U_h\|/\sqrt{d}$ for proper U_h. As a result, under a reasonably tuned b_h, the active features will survive the screening effect and activate the hidden node. Noise may pass through the screening of a node and corrupt the prediction only when this node also contains other active features, but the contribution of noise $(W_h^\top \xi)$ will be negligible compared with other active features.

To simplify our analysis, we impose the following assumption to focus on strong features:

Definition 4.1. Define $\mathcal{M}$ as the set of two-layer neural networks such that, for any node h,

(1) The intercept is within a proper range, i.e. $b_h \ll \|U_h\|/\sqrt{k\|U_h\|_0}$, and

(2) $b_h \gg \|U_h\|/\sqrt{k\|U_h\|_0}/\log d \gg \|U_h\|/\sqrt{d}$.

(3) There are at most m^* of features of X learned by each hidden node, i.e., $\|U_h\|_0 \leq m^*$ for all $h = 1, \ldots, H$, and $m^* = o(d/k)$. All hidden nodes are non-zero and $H \gg d$.

(4) Non-zero $U_{i,h}$'s have the same sign for the same i and $|U_{i,h}| = \Theta(\gamma)$.

The conditions (1) and (2) in Definition 4.1 match the intuition above to conduct screening. The conditions (3) and (4) are for the simplicity of the derivation.

For the ReLU activation function, it is similar to the activation we considered in 1, and the related discussion is postponed to Section B.

4.2 Purified Nodes Lead to Robustness

To intuitively understand why feature purification improves adversarial robustness, a graphical illustration can be found in Figure 2. With either purified/unpurified hidden nodes, the active features X_1 and X_3 will always be attacked. With purified features, adding an attack does not impact the inactive features. With unpurified features, the inactive features can also be attacked.

The following is extended from the above intuition:

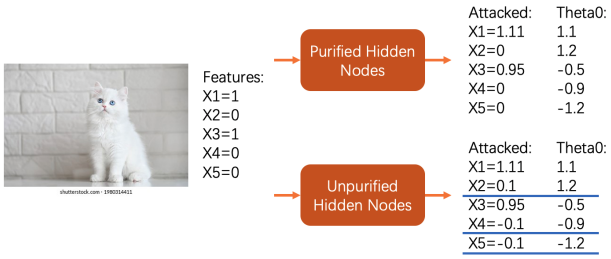

Figure 2: With purified hidden nodes, only the active features will be attacked, and the resulting adversarial loss is small. With unpurified hidden nodes, inactive features will also be impacted. Note that we transform the attack on the observable Z back to its features X, to compare with θ_0.

Lemma 4.2. *Assume $\epsilon = O(1/(\log(d)\sqrt{m^*k}))$, and $(W,b) \in \mathcal{M}$. Denote $\mathcal{X}$ as the set of coordinate i where $|X_i| > 0$. Assume $Ua = \theta$, $\|\theta\|_\infty = \Theta(1)$. With probability tending to 1 over the randomness of ξ and X,*

$$\Delta_{W,b}(z,y) = \epsilon\frac{\partial l}{\partial f_{W,b}}\left\|a^\top diag(\mathbb{I}(W^\top z, b))W^\top\right\|_2 + o, \quad (2)$$

where "o" represents a negligible term caused by the curvature of the loss. In probability,

$$\|\theta_{\mathcal{X}}\|_2 \leq \|a^\top diag(\mathbb{I}(U^\top X, b))U^\top\|_2 \leq \|\theta\|_2, \quad (3)$$

the $\theta_{\mathcal{X}}$ is the vector of the coordinates of the θ in $\mathcal{X}$.

*The **left** equation holds (i.e., highest robustness) only when the matrix U is sparse, i.e., $\|U_h\|_0 \leq 1$ for every hidden node h. When all hidden nodes are activated, the **right** equation holds.*

Lemma 4.2 illustrates how the effectiveness of attack (i.e.,.$\Delta_{W,b}$) is affected by the neural network. If the hidden nodes are purified, then the neural network is more robust, and the increase from l_0 to l_ϵ is small. If not, more hidden nodes are activated, leaking more weight information.

There are two key claims to prove Lemma 4.2: (i) to show equation (2), we show that in probability, every activated hidden node will not be deactivated by the attack (Lemma F.4), and (ii) to show equation (3), we show that in probability, every hidden node is activated as long as some of its learned features are non-zero (Lemma F.3). The proof for Lemma 4.2 and all the following theorems and propositions can be found in Appendix F.

4.3 Purification in Supervised Learning

We use square loss and absolute loss for regression and logistic loss for classification. Given the loss function as l, the task is to minimize $\mathbb{E}l_\epsilon(Z, Y; W, b)$.

Thanks to Lemma 4.2, we are able to study the clean and robust performance of neural networks. Since our

main focus is on contrastive learning, we provide an informal statement for supervised learning below, and postpone the formal theorems to Appendix A.

Theorem 4.3 (Informal Statement)**.** *For some $(W, b) \in \mathcal{M}$ satisfying $Ua = \theta_0$, for square loss, absolute loss, and logistic regression, we define a vanishing term ψ as*

$$\mathbb{E}l_0(Z, Y; W, b) = \mathbb{E}l_0(X, Y; \theta_0) + O(\psi).$$

There exists many $(W, b) \in \mathcal{M}$ such that the clean loss is $O(\psi)$-close to its minimum, while the adversarial loss is $\Theta(\epsilon\sqrt{m^*k})$-close to its minimum. When using adversarial training so that the adversarial loss is $O(\psi)$-close to its minimum, the clean loss is $\Theta(\epsilon\sqrt{k})$-close to its minimum, and at most $o(1)$ proportion of hidden nodes learn more than 1 feature.

5 Purification in Contrastive Learning

In this section, we show that in contrastive learning, clean training does not intend to purify the neural networks, and adversarial training does.

5.1 Model Setup

The contrastive learning aims to learn a $g : \mathbb{R}^d \otimes \mathbb{R}^d \to \mathbb{R}$ to minimize the following loss

$$\begin{aligned} & \mathbb{E}_Z \mathbb{E}_Y l(Z, Z'(Y), Y; g) \\ := \quad & \mathbb{E}_Z \mathbb{E}_Y \log\left(1 + \exp[-Y g(Z, Z'(Y))]\right) \end{aligned} \tag{4}$$

where $Z'(Y) := Z' = MX' + \xi'$ for a noise ξ' that is i.i.d. to ξ, and Y determines whether the pair (Z, Z') is similar or not, i.e., if $Y = 1$, $X' = X$, otherwise, X' is an independent copy of X. In other words, when $Y = 1$, Z and Z' share the same true features, can be interpreted as two views of the same object X; when $Y = -1$, Z and Z' are independent and correspond to different true features X and X'. Note that the label $Y = \pm 1$ in contrastive learning is not the class label in the original data set. It is an artificial label, manually generated following marginal distribution $P(Y = 1) = P(Y = -1) = 0.5$. The label Y represents whether the two views correspond to the same sample or not. Given a neural network parameterized by W, b and A that outputs multiple responses, the loss function l is in the format of

$$l(z, z', y; W, b) = \log\left(1 + \exp[-y g_{W,b}(z, z')]\right),$$

where

$$\begin{aligned} & g_{W,b}(z, z') \\ = \quad & \left(\sum_{h=1}^{H} A_{h,:}\sigma(W_h^\top z, b_h)\right)^\top \left(\sum_{h=1}^{H} A_{h,:}\sigma(W_h^\top z', b_h)\right) \end{aligned} \tag{5}$$

with the output layer $A \in \mathbb{R}^{H \times d}$ with the same output dimension as the data dimension. Note that parameter A is not a trainable parameter since we will consider a lazy training scenario. The details will be discussed later. Unlike the supervised task where the neural network outputs a single value, in contrastive learning, the neural network outputs a vector.

For adversarial attack, we again consider the FGM attack, i.e., $\delta_2 = \epsilon(\partial l/\partial z)/\|\partial l/\partial z\|_2$, and the corresponding adversarial loss can be written as

$$l_\epsilon(z, z', y; W, b) = l(z + \delta_2, z', y; W, b).$$

5.2 Optimal Solution and Lazy Training

The optimal solutions for supervised learning loss and contrastive learning loss are different. But for contrastive learning, by Tosh et al. (2021), the optimal solution of contrastive learning (4) is

$$g^*(z, z') = \log\left(\frac{f_{Z,Z'}(z, z')}{f_Z(z) f_{Z'}(z')}\right),$$

where f_Z, $f_{Z'}$, and $f_{Z,Z'}$ are the marginal and joint density functions and are not linear functions. Thus, under our Definition 4.1, which imposes restrictions on W and b such that $\sigma(W_h^\top z, b_h)$ has a linear function behavior, we cannot achieve good contrastive loss with the two-layer network modeling of $g_{W,b}$.

Based on the following lemma, the best solution of contrastive loss, among linear networks Tx, still enjoys a nice and tractable form under simple settings.

Lemma 5.1 (Basic Properties of Contrastive Learning)**.** *Consider the class of functions $g_T(x, x') = x^\top T^\top T x'$ using the ground-truth feature X for some matrix $T^\top T = PDP^\top$ with an orthonormal matrix P and a diagonal matrix D. Assuming $tr(D)$ is fixed, then the best model to minimize contrastive loss $\mathbb{E}_X \mathbb{E}_Y \log\left(1 + \exp[-Y g_T(X, X')]\right)$ satisfies $D \propto I_d$.*

Lemma 5.1 motivates us to utilize a new lazy-training method in contrastive learning to simplify the analysis. Unlike supervised pre-training, where we fix weight $a \equiv 1$, in contrastive learning, we would ensure that $M^\top W A A^\top W^\top M \propto I_d$ is fixed, i.e., $WAA^\top W^\top \propto I_d$. As a result, instead of completely fixing last layer parameters, in contrastive learning, we take $A = \tau W^+$ for a fixed τ while updating weight matrix W, where W^+ is the pseudo inverse of W.

5.3 Similar vs Dissimlar Pairs

For the above setting, adversarial training will help feature purification. However, different from supervised adversarial training where the attack of all samples contributes to feature purification, in contrastive learning, only the attack on dissimilar data pairs are affected by feature purification.

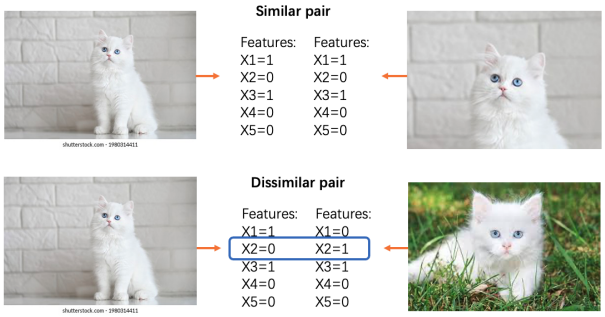

Figure 3: Adversary attacks on dissimilar pairs, but have little effect on similar pairs.

Intuition To explain why adversarial training affects more on the loss of dissimilar pairs, we use Figure 3 as an example. In Figure 3, we attack the left data and keep the right data unchanged. Suppose that the attack changes the inactive features from 0 to another value of the left image, Given a similar pair (i.e., two views of the same data via different data augmentation and their underlying features are the same), the change of the left features is canceled when multiplying the zero feature of the right data. However, for dissimilar pairs, there is a mismatch between the features. For example, the attack changes X_2 of the left data to another value, when multiplying with $X_2 = 1$ on the right data, the product gets changed, leading to a big change in the loss value.

Simulation We also conduct a toy simulation and plot the result in the left panel of Figure 4.

To generate X, we consider the following distribution. First, each coordinate of X is independent of each other, and has k/d probability to be non-zero. Second, given X_i is non-zero, it has $1/2$ probability to be positive, and we take the distribution as $\min(1, |\varepsilon|/\sqrt{k} + 1/\sqrt{k})$ with $\varepsilon \sim N(0,1)$. The distribution is symmetric to $X_i < 0$.

To generate Z, we randomly generate a unitary matrix M, and take $Z = MX + \xi$, with $\xi \sim N(0, \zeta^2 I_d)$. To generate M, we use library `pracma` in `R`. We take $(d, k, \zeta) = (1000, 10, 0.005)$, and generate 1000 samples in each simulation and repeat 30 times to obtain an average. To generate Y for supervised learning, we take $\theta_0 = \mathbf{1}$, and $Y = X^\top\theta + N(0, \sigma^2 I_d)$ with $\sigma = 0.1$. And in terms of the neural network, we take $H = 10000$ hidden nodes.

We control the average $\|U_h\|_0$ and evaluate the clean and adversarial loss. We plot four curves, representing the change of clean and adversarial contrastive losses for similar data pairs (i.e., Z and $Z'(1)$) and dissimilar data pairs (i.e., Z and $Z'(-1)$), as the number of features in each hidden node increases. As the number of features in each hidden node gets larger, the adversarial loss for dissimilar pairs gets larger. The detailed setup, numbers in the figure, and standard errors can be found in Appendix E.

Theory Based on the above simulation observation and intuition, the following theorem demonstrates how an adversarial attack impacts contrastive learning.

Theorem 5.2. *Assume that (W, b) satisfies Definition 4.1 and $|\mathcal{X}| = \Theta(k)$ where $\mathcal{X}$ denotes the set $\{i : |X_i| > 0\}$. Let $A = \tau W^+$ for some fixed $\tau > 0$. If*

$$U_{i,:}\, diag(\mathbb{I}(U_{\mathcal{X},:}, \boldsymbol{0}))U^\top (UU^\top)^{-1}_{:,j} = \Theta(\alpha)$$

for $i \in \mathcal{X}^c$ and $j \in \mathcal{X}$, and

$$U_{i,:}\, diag(\mathbb{I}(U_{\mathcal{X},:}, \boldsymbol{0}))U^\top (UU^\top)^{-1}_{:,j} = \Theta(\alpha^2)$$

for $i \neq j$ and $i, j \in \mathcal{X}^c m$ and furthermore $\alpha = o(1/\sqrt{d})$, then when $\epsilon = \Theta(1/(\log(d)\sqrt{m^* k}))$,

$$\begin{aligned} &\mathbb{E} l_\epsilon(Z, Z'(1), 1; W, b) \qquad (6)\\ = &\min_{(W',b')\in\mathcal{M}} \mathbb{E} l_0(Z, Z'(1), 1; W', b') + \Theta(\epsilon) + O(\psi), \end{aligned}$$

and

$$\begin{aligned} &\mathbb{E} l_\epsilon(Z, Z'(-1), -1; W, b) \qquad (7)\\ = &\min_{(W',b')\in\mathcal{M}} \mathbb{E} l_0(Z, Z'(-1), -1; W', b') + O(\psi)\\ &+\Theta(\epsilon k^{3/2}/d) + \Theta(\epsilon\alpha^2\sqrt{d}). \end{aligned}$$

To make a connection between α and the level of purification of U, we perform a simulation in Figure 4 and calculate the average $U_{i,:}\text{diag}(\mathbb{I}(U_{\mathcal{X}}, \mathbf{0}))U^\top(UU^\top)^{-1}_{:,j}$ for $(i \in \mathcal{X}^c, j \in \mathcal{X})$ and $(i \neq j,\ i, j \in \mathcal{X}^c)$ respectively, and denote γ_1 and γ_2 as the corresponding average value. From the right panel of Figure 4, one can see that $\log(\gamma_1)$ and $\log(\gamma_2)$ are approximately linearly increasing functions of $\log(m)$. With a larger m, α will be larger. In addition, one can also see that $\log(\gamma_2) \approx \log(\gamma_1^2)$, which validates the appropriateness of our assumption in Theorem 5.2.

To connect Theorem 5.1 and the intuition in Figure 4, when designing an attack on Z, $Z'(1)$ carries the information of true features X. As a result, the best attack on Z that aims to make $Z + \delta$ dissimilar to $Z'(1)$, corresponds to the active features in X. For similar pairs, α, which quantifies the associations between active and non-active features, will only have negligible effect.

On the other hand, the effect of the adversarial attack is different in (7) for dissimilar pairs. When α gets larger, $\epsilon\alpha^2\sqrt{d}$ can dominate the fixed $\epsilon k^{3/2}/d$, indicating that the neural network is more vulnerable to adversarial attack for dissimilar data pairs.

The above theorems and simulation evidences together answer our question: contrastive learning can also benefit from adversarial training.

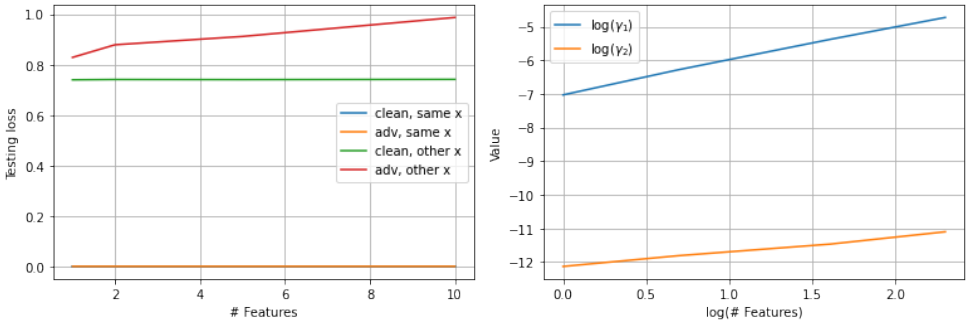

Figure 4: Left: Clean/adversarial contrastive testing loss under different levels of purification of the hidden nodes, for similar data pairs (i.e., $Y = 1$) and dissimilar data pairs (i.e., $Y = -1$). Note that the blue and yellow curves overlap. Right: How α is related to m. The values of γ_1 and γ_2 are assumed to be in $\Theta(\alpha)$ and $\Theta(\alpha^2)$ respectively in Theorem 5.2.

5.4 Discussion

While we study how adversarial training purifies features in contrastive learning, another work, Wen and Li (2021), studies how random data augmentation improves feature purification. This augmentation is simpler to implement with a smaller computation cost, but there are two advantages of adversarial training.

First, for random data augmentation, the intercept term b in the hidden node is taken to purify features only. For adversarial robustness, when ϵ gets larger, we need a larger b to avoid the adversarial attack activate/deactivate hidden nodes. The intercept b in adversarial training can better improve the robustness.

Second, the random augmentation purifies features via decoupling the features in similar pairs, rather than in dissimilar pairs as in adversarial training. In Figure 4, the loss for similar pairs is smaller than dissimilar pairs, implying that adversarial training is more sensitive in purification.

The data augmentation in Wen and Li (2021) is also used in our experiments, and the clean-trained contrastive models are vulnerable to adversarial attack.

6 Robustness in Downstream Tasks

After obtaining the pre-trained model (W, b), we further utilize it in a downstream supervised task.

The downstream training aims to minimize the clean loss of downstream data $(Z_{\text{down}}, Y_{\text{down}})$ w.r.t. a given pre-trained weights (W, b)

$$L^{W,b}(a) := \mathbb{E}L(\sigma(Z_{\text{down}}^\top W, b)a, Y_{\text{down}}), \tag{8}$$

where the loss function L can be different from the one in pre-training, $Z_{\text{down}} = MX_{\text{down}} + \xi_{\text{down}}$ uses the same M but possibly different X_{down} satisfying the sparse coding model, Y_{down} can also be different from Y. Denote L_ϵ as the corresponding adversarial loss, and a^* as the corresponding optimal solution.

The following proposition indicates that the robustness in pre-training can be inherited.

Proposition 6.1. *When* $\epsilon = \Theta(1/(\log(d)\sqrt{m^*k}))$,

(1) There exists (W, b) *that minimizes pre-training clean (supervised or contrastive) loss s.t.*

$$L_\epsilon^{W,b}(a^*) - L_0^{W,b}(a^*) \gg \epsilon\sqrt{k} + O(\psi).$$

(2) Assume $(W, b) \in \mathcal{M}$ *minimizes the adversarial pre-training loss and* $\sup_h \|U_h\|_0 = 1$*, then*

$$L_\epsilon^{W,b}(a^*) - L_0^{W,b}(a^*) = \Theta(\epsilon\sqrt{k}) + O(\psi).$$

The proof of Proposition 6.1 is similar to Theorem A.1.

Proposition 6.1 illustrates two observations. First, using clean loss in the pre-training, since one cannot purify the neural network, the corresponding downstream training is not robust. Second, if we obtain a purified neural network, the downstream model is robust.

7 Real-Data Experiments

Our experiments aim to justify (1) the robustness inheritance phenomenon in Section 6; (2) Adversarial training purifies the features (Section 4.3 and 5).

7.1 Experimental Setups

We perform supervised learning Rice et al. (2020)[2] and contrastive learningKim et al. (2020)[3] pre-training (i.e., pre-training consists of a clean training phase, followed by an adversarial training phase) to verify that the hidden nodes are purified. After pre-training the neural network, we remove its last layer, train a new

[2] https://github.com/locuslab/robust_overfitting
[3] https://github.com/Kim-Minseon/RoCL

last layer using a supervised task (clean training), and test the adversarial robustness.

Our tests are conducted on ResNet-18 He et al. (2016). The attack method used for training and evaluation is PGD under l_∞ norm and $\epsilon = 8/255$. We use CIFAR-10, CIFAR-100, or Tiny-Imagenet in pre-training and CIFAR-10 for downstream training and testing. Details on training configurations are in Table 3 in the appendix, and we retain the same configuration used by the original GitHub repositories. We perform the training on a RTX-2080 GPU with 12GB RAM.

Pre-train	Pre-train	Down	Acc	Robust
CIFAR10	Clean	Clean	0.955	0.001
	Clean	Adv Sup	0.477	0.109
	Adv Sup	-	0.810	0.495
	Adv Sup	Clean	0.847	0.429
	Adv Sup	Adv Sup	0.836	0.484
	Adv Contra	Clean	0.831	0.393
	Adv Contra	Adv Sup	0.807	0.462

Table 1: Robustness and accuracy in CIFAR-10 downstream task for different pre-training setups. "Pre-train" and "Downstream" indicate the method of pre-training and the downstream task. "Adv" stands for adversarial training. "Sup" and "Contra" stands for supervised and constrastive learning.

Table 1 shows the training results for CIFAR-10. For the results of using CIFAR-100 or Tiny-Imagenet in the pre-training, we postpone them to Table 2 in the appendix due to the page limit. In Table 1, we evaluate the clean accuracy (Acc) and robust accuracy (Robust) in the testing dataset. For both supervised and contrastive adversarial pre-training training + clean downstream training, we observe higher robustness against PGD attacks than clean pre-training, despite minor losses in standard accuracy. This verifies the robustness inheritance phenomenon.

We also provide benchmarks for comparison. First, clean pre-training + clean downstream training together result in near-zero robustness. Second, clean pre-training + adversarial downstream training increased robustness by 10%, but at the cost of drastically decreased clean accuracy (-47.8%), since the learning capacity of downstream linear layer is limited. Third, when the downstream tasks are also trained in an adversarial manner, compared with clean downstream training, the robustness increases by 5.5% in the task following supervised adversarial pre-training and 6.9% in the task following adversarial contrastive pre-training, which means that we are not losing too much from using clean training in the downstream tasks. Finally, the robustness is only slightly higher compared to adversarial training from scratch (49.5%).

Table 1 also provides the results when using CIFAR-100 in the pre-training. The observations are similar to the case of CIFAR-10. Similar results can be found in Table 4 in the appendix for a different input layer kernel size. In contrast, Section D.3 shows that data augmentation method (Wen and Li, 2021) solely cannot effectively improve robustness.

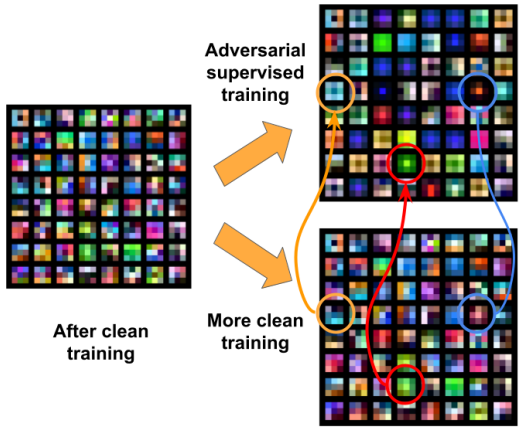

Figure 5: Learned features in the input convolutional layer trained on CIFAR-10.

In addition to the numerical robustness result, we also visualize the trained neural networks to demonstrate the feature purification effect. Figure 5 visualizes the features in the input convolutional layer learned from adversarial and clean pre-training. The features learned from adversarial training tend to have fewer types of colors in one cell, showing purification effects. Features in multiple filters (see the blocks marked with circles for examples) become highly concentrated, reducing the small perturbations around the center points. More figures of purification can be found in Figure 6, 7 and 8 in Appendix D.

8 Conclusion

In this study, we consider the feature purification effect of adversarial training in supervised/self-supervised pre-training, and the robustness inheritance in the downstream clean trained task. Both theory and experiments demonstrate the feature purification phenomenon. As for future direction, while we consider adversarial pre-training and clean fine-tuning, it can still be burdensome for the pre-trained model provider to train a robust model. Thus, it is interesting to study the performance of clean pre-training and adversarial fine-tuning, which is needed when the pre-training is expensive, e.g., foundation models. As mentioned in Table 1, when simply using clean pre-training with adversarial fine-tuning, the robustness cannot be effectively improved. Other methods may be considered to improve the robustness.

References

Alayrac, J.-B., Uesato, J., Huang, P.-S., Fawzi, A., Stanforth, R., and Kohli, P. (2019), "Are labels required for improving adversarial robustness?" *Advances in Neural Information Processing Systems*, 32.

Allen-Zhu, Z. and Li, Y. (2020), "Feature Purification: How Adversarial Training Performs Robust Deep Learning," *arXiv preprint arXiv:2005.10190*.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. (2019), "A theoretical analysis of contrastive unsupervised representation learning," *arXiv preprint arXiv:1902.09229*.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021), "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*.

Cai, Q.-Z., Du, M., Liu, C., and Song, D. (2018), "Curriculum adversarial training," *arXiv preprint arXiv:1805.04807*.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., and Liang, P. S. (2019), "Unlabeled data improves adversarial robustness," in *Advances in Neural Information Processing Systems*, pp. 11192–11203.

Cemgil, T., Ghaisas, S., Dvijotham, K. D., and Kohli, P. (2019), "Adversarially robust representations with smooth encoders," in *International Conference on Learning Representations*.

Chen, D., Hu, H., Wang, Q., Yinli, L., Wang, C., Shen, C., and Li, Q. (2021), "CARTL: Cooperative Adversarially-Robust Transfer Learning," in *International Conference on Machine Learning*, PMLR, pp. 1640–1650.

Chen, L., Min, Y., Zhang, M., and Karbasi, A. (2020a), "More data can expand the generalization gap between adversarially robust and standard models," in *International Conference on Machine Learning*, PMLR, pp. 1670–1680.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020b), "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, PMLR, pp. 1597–1607.

Chuang, C.-Y., Robinson, J., Yen-Chen, L., Torralba, A., and Jegelka, S. (2020), "Debiased contrastive learning," *arXiv preprint arXiv:2007.00224*.

Dai, B. and Lin, D. (2017), "Contrastive learning for image captioning," *arXiv preprint arXiv:1710.02534*.

Dan, C., Wei, Y., and Ravikumar, P. (2020), "Sharp Statistical Guaratees for Adversarially Robust Gaussian Classification," in *International Conference on Machine Learning*, PMLR, pp. 2345–2355.

Deng, Z., Zhang, L., Ghorbani, A., and Zou, J. (2021a), "Improving adversarial robustness via unlabeled out-of-domain data," in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 2845–2853.

Deng, Z., Zhang, L., Vodrahalli, K., Kawaguchi, K., and Zou, J. (2021b), "Adversarial Training Helps Transfer Learning via Better Representations," *arXiv preprint arXiv:2106.10189*.

Fan, L., Liu, S., Chen, P.-Y., Zhang, G., and Gan, C. (2021), "When Does Contrastive Learning Preserve Adversarial Robustness from Pretraining to Finetuning?" *Advances in Neural Information Processing Systems*, 34, 21480–21492.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015), "Explaining and Harnessing Adversarial Examples," in *3rd International Conference on Learning Representations*.

Gowal, S., Rebuffi, S.-A., Wiles, O., Stimberg, F., Calian, D. A., and Mann, T. A. (2021), "Improving Robustness using Generated Data," *Advances in Neural Information Processing Systems*, 34.

Grigor'eva, M. and Popov, S. (2012), "An upper bound for the absolute constant in the nonuniform version of the Berry-Esseen inequalities for nonidentically distributed summands," in *Doklady Mathematics*, Springer, vol. 86, pp. 524–526.

HaoChen, J. Z. and Ma, T. (2022), "A Theoretical Study of Inductive Biases in Contrastive Learning," *arXiv preprint arXiv:2211.14699*.

HaoChen, J. Z., Wei, C., Gaidon, A., and Ma, T. (2021), "Provable guarantees for self-supervised deep learning with spectral contrastive loss," *Advances in Neural Information Processing Systems*, 34, 5000–5011.

HaoChen, J. Z., Wei, C., Kumar, A., and Ma, T. (2022), "Beyond separability: Analyzing the linear transferability of contrastive representations to related subpopulations," *arXiv preprint arXiv:2204.02683*.

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

Hendrycks, D., Lee, K., and Mazeika, M. (2019), "Using pre-training can improve model robustness and uncertainty," *arXiv preprint arXiv:1901.09960*.

Ho, C.-H. and Nvasconcelos, N. (2020), "Contrastive learning with adversarial examples," *Advances in Neural Information Processing Systems*, 33, 17081–17093.

Hyvarinen, A., Oja, E., Hoyer, P., and Hurri, J. (1998), "Image feature extraction by sparse coding and independent component analysis," in *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, IEEE, vol. 2, pp. 1268–1273.

Javanmard, A. and Mehrabi, M. (2021), "Adversarial robustness for latent models: Revisiting the robust-standard accuracies tradeoff," *arXiv preprint arXiv:2110.11950*.

Javanmard, A. and Soltanolkotabi, M. (2022), "Precise statistical analysis of classification accuracies for adversarial training," *The Annals of Statistics*, 50, 2127–2156.

Javanmard, A., Soltanolkotabi, M., and Hassani, H. (2020), "Precise tradeoffs in adversarial training for linear regression," in *Conference on Learning Theory*, PMLR, pp. 2034–2078.

Jiang, Z., Chen, T., Chen, T., and Wang, Z. (2020), "Robust pre-training by adversarial contrastive learning," *Advances in neural information processing systems*, 33, 16199–16210.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. (2020), "Supervised contrastive learning," *arXiv preprint arXiv:2004.11362*.

Kim, M., Tack, J., and Hwang, S. J. (2020), "Adversarial Self-Supervised Contrastive Learning," in *Advances in Neural Information Processing Systems*.

Li, J., Zhou, P., Xiong, C., and Hoi, S. C. (2020), "Prototypical contrastive learning of unsupervised representations," *arXiv preprint arXiv:2005.04966*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013), "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, pp. 3111–3119.

Min, Y., Chen, L., and Karbasi, A. (2020), "The curious case of adversarially robust models: More data can help, double descend, or hurt generalization," *arXiv preprint arXiv:2002.11080*.

Mo, Y., Wu, D., Wang, Y., Guo, Y., and Wang, Y. (2022), "When Adversarial Training Meets Vision Transformers: Recipes from Training to Architecture," *arXiv preprint arXiv:2210.07540*.

Najafi, A., Maeda, S.-i., Koyama, M., and Miyato, T. (2019), "Robustness to adversarial perturbations in learning from incomplete data," in *Advances in Neural Information Processing Systems*, pp. 5542–5552.

Nguyen, A. T., Lim, S. N., and Torr, P. (2022), "Task-Agnostic Robust Representation Learning," *arXiv preprint arXiv:2203.07596*.

Oord, A. v. d., Li, Y., and Vinyals, O. (2018), "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*.

Petrov, A. and Kwiatkowska, M. (2022), "Robustness of Unsupervised Representation Learning without Labels," *arXiv preprint arXiv:2210.04076*.

Raghunathan, A., Xie, S. M., Yang, F., Duchi, J. C., and Liang, P. (2019), "Adversarial training can hurt generalization," *arXiv preprint arXiv:1906.06032*.

Rice, L., Wong, E., and Kolter, J. Z. (2020), "Overfitting in adversarially robust deep learning," *arXiv preprint arXiv:2002.11569*.

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. (2020), "Do adversarially robust imagenet models transfer better?" *arXiv preprint arXiv:2007.08489*.

Saunshi, N., Ash, J., Goel, S., Misra, D., Zhang, C., Arora, S., Kakade, S., and Krishnamurthy, A. (2022), "Understanding contrastive learning requires incorporating inductive biases," *arXiv preprint arXiv:2202.14037*.

Saunshi, N., Plevrakis, O., Arora, S., Khodak, M., and Khandeparkar, H. (2019), "A theoretical analysis of contrastive unsupervised representation learning," in *International Conference on Machine Learning*, PMLR, pp. 5628–5637.

Shafahi, A., Saadatpanah, P., Zhu, C., Ghiasi, A., Studer, C., Jacobs, D., and Goldstein, T. (2019), "Adversarially robust transfer learning," *arXiv preprint arXiv:1905.08232*.

Shen, K., Jones, R. M., Kumar, A., Xie, S. M., HaoChen, J. Z., Ma, T., and Liang, P. (2022), "Connect, not collapse: Explaining contrastive learning for unsupervised domain adaptation," in *International Conference on Machine Learning*, PMLR, pp. 19847–19878.

Sinha, A., Namkoong, H., and Duchi, J. (2018), "Certifying some distributional robustness with principled adversarial training," .

Taheri, M., Xie, F., and Lederer, J. (2021), "Statistical guarantees for regularized neural networks," *Neural Networks*, 142, 148–161.

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. (2020), "What makes for good views for contrastive learning?" *arXiv preprint arXiv:2005.10243*.

Tosh, C., Krishnamurthy, A., and Hsu, D. (2021), "Contrastive learning, multi-view redundancy, and linear models," in *Algorithmic Learning Theory*, PMLR, pp. 1179–1206.

Wang, Q., Wang, Y., Zhu, H., and Wang, Y. (2022), "Improving Out-of-Distribution Generalization by Adversarial Training with Structured Priors," *arXiv preprint arXiv:2210.06807*.

Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., and Gu, Q. (2019a), "On the convergence and robustness of adversarial training," in *International Conference on Machine Learning*, pp. 6586–6595.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., and Gu, Q. (2019b), "Improving adversarial robustness requires revisiting misclassified examples," in *International Conference on Learning Representations*.

Wen, Z. and Li, Y. (2021), "Toward understanding the feature learning process of self-supervised contrastive learning," in *International Conference on Machine Learning*, PMLR, pp. 11112–11122.

Wu, B., Chen, J., Cai, D., He, X., and Gu, Q. (2020a), "Does Network Width Really Help Adversarial Robustness?" *arXiv preprint arXiv:2010.01279*.

Wu, D., Wang, Y., and Xia, S.-t. (2020b), "Adversarial Weight Perturbation Helps Robust Generalization," *arXiv preprint arXiv:2004.05884*.

Xiao, J., Fan, Y., Sun, R., and Luo, Z.-Q. (2021), "Adversarial Rademacher Complexity of Deep Neural Networks," .

Xiao, J., Fan, Y., Sun, R., Wang, J., and Luo, Z.-Q. (2022a), "Stability analysis and generalization bounds of adversarial training," *arXiv preprint arXiv:2210.00960*.

Xiao, J., Qin, Z., Fan, Y., Wu, B., Wang, J., and Luo, Z.-Q. (2022b), "Adaptive Smoothness-weighted Adversarial Training for Multiple Perturbations with Its Stability Analysis," *arXiv preprint arXiv:2210.00557*.

Xiao, T., Wang, X., Efros, A. A., and Darrell, T. (2020), "What should not be contrastive in contrastive learning," *arXiv preprint arXiv:2008.05659*.

Xing, Y., Song, Q., and Cheng, G. (2021a), "On the Algorithmic Stability of Adversarial Training," *Advances in Neural Information Processing Systems*, 34.

— (2021b), "On the generalization properties of adversarial training," in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 505–513.

Yin, D., Ramchandran, K., and Bartlett, P. (2018), "Rademacher complexity for adversarially robust generalization," *arXiv preprint arXiv:1810.11914*.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. (2019), "Theoretically Principled Trade-off between Robustness and Accuracy," in *Proceedings of the 36th International Conference on Machine Learning*, PMLR, vol. 97 of *Proceedings of Machine Learning Research*, pp. 7472–7482.

Zhang, J., Sang, J., Yi, Q., Yang, Y., Dong, H., and Yu, J. (2021), "Pre-training also Transfers Non-Robustness," *arXiv preprint arXiv:2106.10989*.

Zhang, J., Xu, X., Han, B., Niu, G., Cui, L., Sugiyama, M., and Kankanhalli, M. (2020a), "Attacks which do not kill training make adversarial learning stronger," in *International Conference on Machine Learning*, PMLR, pp. 11278–11287.

Zhang, Y., Plevrakis, O., Du, S. S., Li, X., Song, Z., and Arora, S. (2020b), "Over-parameterized Adversarial Training: An Analysis Overcoming the Curse of Dimensionality," *arXiv preprint arXiv:2002.06668*.

Zhao, Y., Chen, J., and Du, S. S. (2022), "Blessing of Class Diversity in Pre-training," *arXiv preprint arXiv:2209.03447*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. **Yes**.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. **Not Applicable**.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. **No**, we are using Github repositories from other literature.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. **Yes**.
   (b) Complete proofs of all theoretical results. **Yes**.
   (c) Clear explanations of any assumptions. **Yes**.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). **No**.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). **No**, we are using default settings in the existing code and highlighting the changes in our paper.
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). **Yes**.
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). **No**, we only use a single GPU, and the computation is not expensive.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. **Yes**.
   (b) The license information of the assets, if applicable. **Not Applicable**.
   (c) New assets either in the supplemental material or as a URL, if applicable. **Not Applicable**.
   (d) Information about consent from data providers/curators. **Not Applicable**.
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. **Not Applicable**.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. **Not Applicable**.
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. **Not Applicable**.
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. **Not Applicable**.

Below is a list of the contents in this appendix:

- Section A: detailed theorem for supervised learning.
- Section B: discussion on potential relaxations of the assumptions.
- Section C: discussion when using $\mathcal{L}_\infty$ attack.
- Section D: real-data experiments.
- Section E: simulation details.
- Section F: the proof for theorems and lemmas using $\mathcal{L}_2$ attack.

A Details for Supervised Learning

Clean training does not purify features The following theorem indicates that clean training can achieve good clean performance without feature purification.

Theorem A.1. *For some $(W, b) \in \mathcal{M}$ satisfying $Ua = \theta_0$, for square loss, absolute loss, and logistic regression,*

$$\mathbb{E}l_0(Z, Y; W, b) = \mathbb{E}l_0(X, Y; \theta_0) + O(\psi),$$

where ψ is a vanishing term induced by the noise ξ and the activation gate, i.e., the discrepancy between $\mathbb{I}(x^\top U_h, 0)$ and $\mathbb{I}(x^\top U_h + \xi^\top W_h, b_h)$. If $Hk^3(m^*)^3 = O(d^{2-\varepsilon})$ for some $\varepsilon > 0$, then $\psi \to 0$.

*There are many choices of $(W, b) \in \mathcal{M}$ with **good clean performance**, i.e.,*

$$\mathbb{E}l_0(Z, Y; W, b) = \min_{(W', b') \in \mathcal{M}} \mathbb{E}l_0(Z, Y; W', b') + O(\psi) = \mathbb{E}l_0(X, Y; \theta_0) + O(\psi). \tag{9}$$

*Meanwhile, **their robustness is poor**: When taking $\epsilon = \Theta(1/(\log(d)\sqrt{m^*k}))$,*

$$\mathbb{E}l_\epsilon(Z, Y; W, b) - \min_{(W', b') \in \mathcal{M}} \mathbb{E}l_\epsilon(Z, Y; W', b') = O(\psi) + \Theta(\epsilon\sqrt{m^*k}),$$

The notation Θ belongs to the family of Big-O notation, and it is the same as $\asymp$. For two sequences $\{a_n\}, \{b_n\}$, $b_n = \Theta(a_n)$ (or $b_n \asymp a_n$) means that when $n \to \infty$, there exists some constants $c_0, c_1 > 0$ so that $c_0 a_n \le b_n \le c_1 a_n$.

Note that when m^* and k are small enough, and $H \gg d$ in a suitable range, $\epsilon\sqrt{m^*k} \gg O(\psi)$.

The proof of Theorem A.1 and the following Theorem A.2 mainly utilize Lemma 4.2. In Lemma 4.2, the results hold in probability. To prove Theorem A.1 and A.2, the main goal is to quantify the effect when the exceptions happen.

Adversarial training purifies features Based on the idea in Lemma 4.2, the following theorem shows how adversarial training improves robustness and how purification happens. One can purify the neural network using adversarial training while achieving a good performance in both clean and adversarial testing.

Theorem A.2. *Assume $\epsilon = \Theta(1/(\sqrt{km^*}\log d))$ and $H = o(\epsilon d^{3/2})$, then if $W, b \in \mathcal{M}$ leads to **a small adversarial loss**, i.e.,*

$$\mathbb{E}l_\epsilon(Z, Y; W, b) = \min_{W', b' \in \mathcal{M}} \mathbb{E}l_\epsilon(Z, Y; W', b') + O(\psi),$$

*then (1) **its clean performance is also good**:*

$$\mathbb{E}l_0(Z, Y; W, b) - \mathbb{E}l_0(X, Y; \theta_0) = O(\psi) + O(\epsilon\sqrt{k}),$$

and (2) when $d/H \gg \psi$, $(1 - o(1))H$ hidden nodes satisfy $\|U_h\|_0 = 1$.

B Potential Relaxations

Below is a list on the potential relaxations in the theory:

For the activation function, our choice (1) simplifies the analysis to highlight the feature purification. For other activation functions, e.g., ReLU, if they can work as a gate to screen out noise, the idea of feature purification still works.

In terms of the architecture of the neural network, under the sparse coding model, as long as the first layer purifies the features, the neural network is always robust to adversarial attacks regardless of the number of layers. Thus, one can extend our analysis to multi-layer neural networks.

For the sparse coding model, this is a key assumption of the feature purification phenomenon. If all features are always active, there is no need to purify them in the hidden nodes to minimize adversarial loss. All features will contribute to the adversarial loss together. For future study, one may consider better connecting sparse models with real data distribution. In addition, one may also relax the linear model assumption between X and Y. Intuitively, if the features are not purified, the attacker will attack the weights of the inactive features. Thus from this perspective, feature purification will also work beyond linear data models.

C Using $\mathcal{L}_\infty$ Attack

We consider fast gradient sign attack (FGSM)

$$\delta_\infty = \epsilon \operatorname{sgn}(\partial l/\partial z).$$

We have

$$\begin{aligned}
f_{W,b}(z+\delta_\infty) &= \sigma\left(\left(z+\epsilon \operatorname{sgn}\left(\frac{\partial l}{\partial f}\right)\operatorname{sgn}\left(MU\operatorname{diag}(\mathbb{I}(x^\top U+\xi^\top W,b))a\right)\right)^\top W,b\right)a \\
&= \sigma\left(\left(x+\xi^\top M+\epsilon \operatorname{sgn}\left(\frac{\partial l}{\partial f}\right)\operatorname{sgn}\left(MU\operatorname{diag}(\mathbb{I}(x^\top U+\xi^\top W,b))a\right)\right)^\top U,b\right)a \\
&= \left(x+\xi^\top M\right)^\top U\operatorname{diag}(\mathbb{I}((z+\delta_2)^\top W,b))a \\
&\quad +\epsilon \operatorname{sgn}\left(\frac{\partial l}{\partial f}\right)\operatorname{sgn}\left(MU\operatorname{diag}(\mathbb{I}(x^\top U+\xi^\top W,b))a\right)^\top MU\operatorname{diag}(\mathbb{I}((z+\delta_2)^\top W,b))a \\
&= f_{W,b}(z)+\epsilon \operatorname{sgn}\left(\frac{\partial l}{\partial f}\right)\|MU\operatorname{diag}(\mathbb{I}(x^\top U+\xi^\top W,b))a\|_1.
\end{aligned}$$

Assume the first coordinate of x is non-zero. Since with probability tending to 1 (Lemma F.3), all the hidden nodes receiving x_1 are activated, we have

$$a^\top\operatorname{diag}(\mathbb{I}(U^\top x+\xi^\top W,b))U_{1,:}=a^\top U_{1,:}=\theta_1.$$

Assume the second coordinate of x is zero, since we minimized $\|U\|_F$, each non-zero element of $U_{2,:}$ has the same sign as θ_2, and

$$0\le|a^\top\operatorname{diag}(\mathbb{I}(U^\top x+\xi^\top W,b))U_{2,:}|\le|\theta_2|,$$

and the left/right equation is satisfied if every node containing x_2 is not/is activated.

For $\mathcal{L}_2$ attack, $\|MU\operatorname{diag}(\mathbb{I}(x^\top U+\xi^\top W,b))a\|_2$ becomes $\|U\operatorname{diag}(\mathbb{I}(x^\top U+\xi^\top W,b))a\|_2$, so the attack is directly related to the each coordinate of θ.

For $\mathcal{L}_\infty$ attack, we want to investigate $\|MD\theta\|_1$ where D is a diagonal matrix with $D_{i,i}=a^\top\operatorname{diag}(\mathbb{I}(U^\top x+\xi^\top W,b))U_{i,:}/\theta_i$.

One can see that the relationship between D and $\|MD\theta\|_1$ is more complicated because of the existence of M. To discuss about $\|MD\theta\|_1$, one need some information about M.

- Assume M is the identity matrix, then similar to $\mathcal{L}_2$ attack case, we have

$$\|\theta_{\mathcal{X}}\|_1 \leq \|D\theta\|_1 \leq \|\theta\|_1.$$

- If the unitary matrix M satisfies $\|MD\theta\|_1 = \Theta(\|D\theta\|_1)$ for all D, θ, then although we cannot claim $\sup \|U_{:,h}\|_0 \leq 1$ lead to the minimal $\Delta_{W,b}$, we can still claim that a constant $\sup \|U_{:,h}\|_0$ is preferred than dense mixtures.

Under these two cases, all the observations for $\mathcal{L}_2$ apply to $\mathcal{L}_\infty$ attack.

D Real-Data Experiments

D.1 Experiment Results

Additional results can be found in Table 2, Figure 6, 7, 8. The settings of the experiments can be found in 3.

Pre-train	Pre-train	Down	Acc	Robust
CIFAR10	Clean	Clean	0.955	0.001
	Clean	Adv Sup	0.477	0.109
	Adv Sup	-	0.810	0.495
	Adv Sup	Clean	0.847	0.429
	Adv Sup	Adv Sup	0.836	0.484
	Adv Contra	Clean	0.831	0.393
	Adv Contra	Adv Sup	0.807	0.462
CIFAR100	Clean	Clean	0.786	0.000
	Adv Sup	Clean	0.649	0.108
	Adv Contra	Clean	0.749	0.185
Tiny-Imagenet	Clean	Clean	0.840	0.001
	Adv Sup	Clean	0.323	0.131
	Adv Contra	Clean	0.774	0.150

Table 2: Robustness inheritance.

Further, Figure 9 illustrates how the learned features evolve during the contrastive pre-training. After 30 epochs of adversarial training, the features show the same purification effects as in supervised learning. This purifying process continues throughout the adversarial training.

D.2 Experiment on Resnet-18 with a different input layer

To better visualize the purification effect in the learned features, we repeat our real data test with an input convolutional layer with a larger kernel size. Specifically, the input layer in this test has Kernel Size = 7, Stride=2, Padding=3. All other layers used the same configuration. We pre-train this modified network using clean training and then adversarial supervised learning on CIFAR-10 dataset. Then we fine-tune the downstream task on CIFAR-10 dataset.

Table 4 shows the standard and robust accuracy. We observe the same inherited robustness in the downstream tasks. Figure 10 shows the learned features in the filter. In addition to the reduction of the number of colors, the shapes of features are also simplified, often from multiple parallel lines to one single line, demonstrating purification effects (see blocks marked with blue circles, in which a feature with 4 lines becomes 2 lines.)

D.3 Effect of Augmentation on Inherited Robustness

Our primary experiments employ crop&resize and color distortion augmentations, as highlighted in Wen and Li (2021), to enhance feature learning. We evaluate their impact on downstream robustness by comparing test outcomes both with and without these augmentations. As demonstrated in Table 5, although omitting augmentations diminishes robustness, their presence alone does not substantially improve it. This underscores the pivotal role of adversarial training in achieving robustness.

	Supervised	Contrastive
Clean Epochs	200	200
Adv Epochs	200	1,200
Downstream Epochs	200	200
Batch Size	128	256
Transform	Croppring+Horizontal Flipping	Croppring+Horizontal Flipping+Color Jittering
Learning Rate	0.1	0.1
LR Updating Schedule	Divide by 10 after 50% / 75% of total epochs	Cosine learning rate annealing (SGDR) and learning rate warmup. Warmup epoch = 10
Downstream Learning Rate	0.1	0.1
Attack Iteration	10	20
Attack Initialization	The original point	Within a random ball near the original point

Table 3: Detail configurations of different adversarial training experiments. To maximize robustness, the clean training use the "Supervised" configuration when pre-training on CIFAR10, and the "Contrastive" configuration when pre-training on CIFAR100.

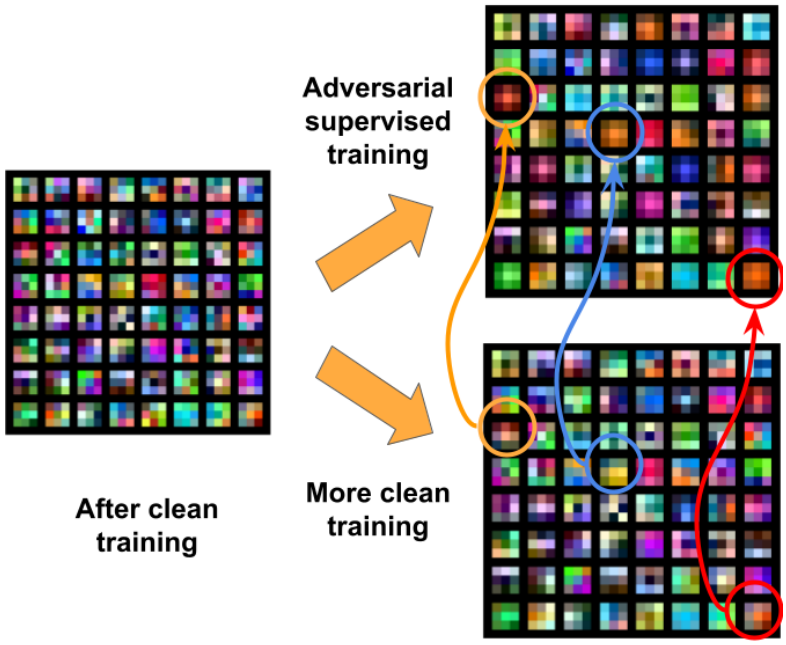

Figure 6: Learned features in the first convolutional layer with and without the adversarial supervised pre-training. The training is performed on CIFAR-100 dataset. Parameters in each filter are normalized to [0,1] separately.

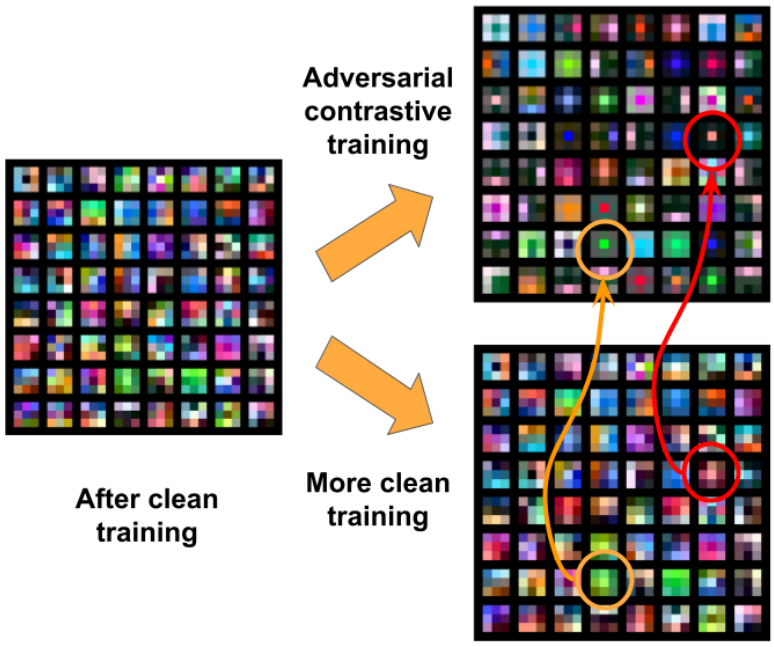

Figure 7: Learned features in the input convolutional layer with and without the adversarial contrastive pre-training. The training is performed on CIFAR-10 dataset. Parameters in each filter are normalized to [0,1] separately.

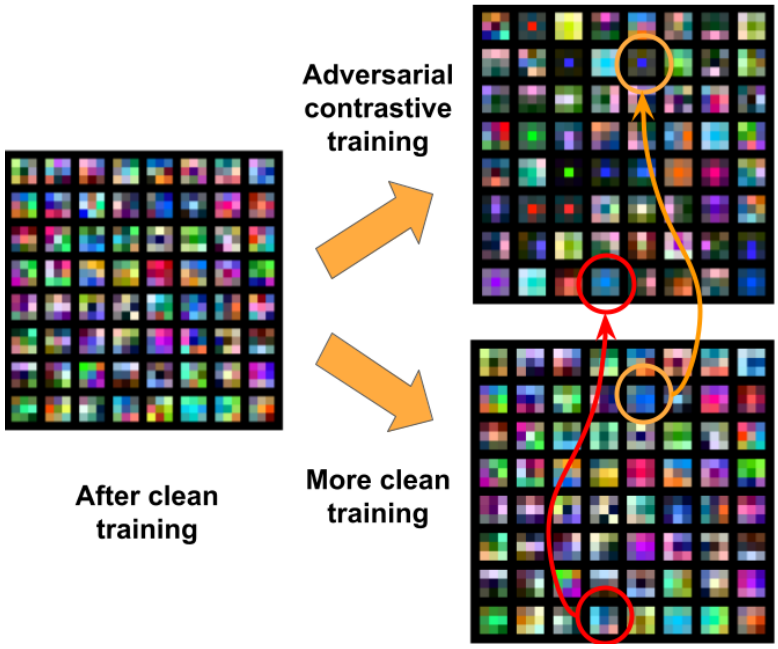

Figure 8: Learned features in the input convolutional layer with and without the adversarial contrastive pre-training. The training is performed on CIFAR-100 dataset. Parameters in each filter are normalized to [0,1] separately.

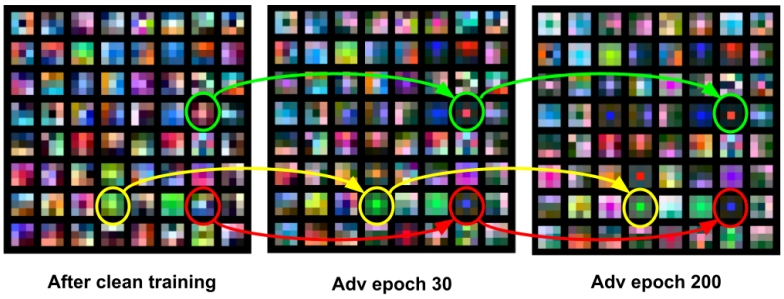

Figure 9: Changes of learned features in the input convolutional layer over adversarial contrastive training on CIFAR-10. Parameters in each filter are normalized to [0,1] separately.

E Simulation Studies

E.1 Controlling Feature Purification

Neural network for supervised learning To control the average number of features m in each hidden node, we

- Calculate the number of times each coordinate of X appears: Hm/d.
- For each coordinate of X, we randomly pick $H * m/d$ hidden nodes out of the total H hidden nodes, and take the corresponding elements in U as $d/(Hm)$.
- Transform U to W via $W = M^\top U$.
- Take b as $(\zeta(\log d)/\sqrt{d})(d\sqrt{m}/H)$, where $\zeta(\log d)/\sqrt{d}$ is a probability bound to screen out ξ, and $d\sqrt{m}/H$ is the adjustment based on the strength of the features.

Pre-train Data	Pre-train	Downstream	Acc	Robust
CIFAR-10	Clean	Clean	0.888	0.024
	Adv Sup	Clean	0.783	0.363
	Adv Sup	Adv Sup	0.789	0.401

Table 4: Adversarial robustness and accuracy in CIFAR-10 downstream task. In this test, the input convolutional layer is changed to have Kernel Size = 7, Stride=2, Padding=3.

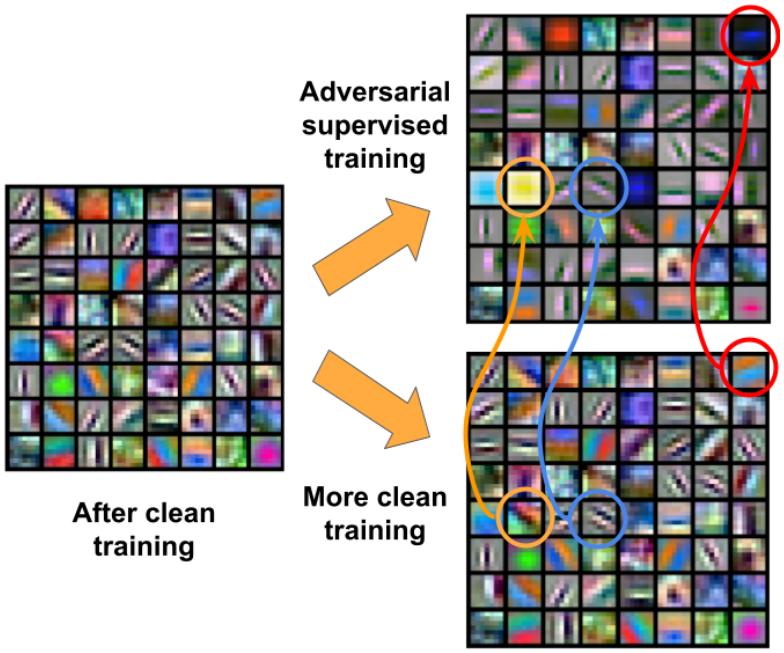

Figure 10: Learned features in the input convolutional layer with and without the adversarial contrastive pre-training. The training is performed on CIFAR-100 dataset. Parameters in each filter are normalized to [0,1] separately.

Neural network for contrastive learning To control the average number of features m in contrastive learning, we

- Follow the above procedure the generate the hidden layer.
- Calculate A as the pseudo inverse of W, and take $A = \sqrt{5}A$.

E.2 Detailed Numbers for Figure 4

We list all the exact numbers (both average and the corresponding standard error) of Figure 4 in Table 6, 7, and 8.

F Proof for $\mathcal{L}_2$ Attack

In this section, we present the proofs using $\mathcal{L}_2$ adversarial training for all the theorems and lemmas in Section 4, 4.3, 5, and 6.

Pre-train Data	Pre-train	Augmentation	Acc	Robust
CIFAR-10	Clean	✓	0.955	0.001
	Adv Contra	✓	0.831	0.393
	Adv Contra	×	0.822	0.389
	Clean Contra	✓	0.897	0.004
	Clean Contra	×	0.889	0.001
CIFAR-100	Clean	✓	0.786	0.000
	Adv Contra	✓	0.749	0.185
	Adv Contra	×	0.741	0.167
	Clean Contra	✓	0.801	0.005
	Clean Contra	×	0.797	0.000

Table 5: Downstream Task Robustness on Different Augmentation Settings.

# Features	Clean Loss, Similar	Adv Loss, Similar	Clean Loss, Dissimilar	Adv Loss, Dissimilar
1	0.05	0.08	73.98	82.86
2	0.05	0.08	74.11	87.87
5	0.05	0.08	74.06	91.17
10	0.05	0.08	74.17	98.69

Table 6: The exact average loss (×100) corresponding to Figure 4.

# Features	Clean Std, Similar	Adv Std, Similar	Clean Std, Dissimilar	Adv Std, Dissimilar
1	0.02	0.03	1.98	2.13
2	0.02	0.03	1.92	2.04
5	0.02	0.03	1.80	1.92
10	0.02	0.03	2.00	2.64

Table 7: The exact standard error (×100) corresponding to Figure 4.

# Features	Average γ_1	Average γ_2	Std γ_1	Std γ_2
1	8.86E-04	5.39E-06	8.56E-05	1.81E-06
2	1.87E-03	7.42E-06	1.15E-04	2.42E-06
5	4.62E-03	1.05E-05	1.32E-04	3.52E-06
10	8.87E-03	1.51E-05	3.32E-04	5.52E-06

Table 8: The average and standard error of γ_1 and γ_2.

F.1 Some Lemmas and Probability Bounds

Lemma F.1. *Denote a vector $m \in \mathbb{R}^d$ with $\|m\|_2 = 1$ and a random vector $\xi \sim N(0, I_d/d)$, then for any $t > 1$,*

$$P\left(|m^\top \xi| > t\sqrt{\frac{1}{d}}\right) \leq \sqrt{\frac{2}{\pi}}\frac{1}{t}\exp(-t^2/2), \tag{10}$$

and

$$\mathbb{E}\left[|m^\top \xi| \middle| |m^\top \xi| > t\sqrt{\frac{1}{d}}\right] = \frac{2}{\sqrt{d}}\frac{\phi(t)}{\Phi(t)}. \tag{11}$$

Proof of Lemma F.1. Observe that $\sqrt{d}m^\top \xi$ follows $N(0,1)$. One can directly bound $|m^\top \xi|$ using Gaussian tail bound. □

F.2 Proof for Section 4

To prove Lemma 4.2, we show the following things:

- The distribution of $X^\top U_h$. (Lemma F.2)
- Without attack, whether the hidden nodes are activated as long as corresponding features are non-zero. (Lemma F.3)
- Further adding attack, whether the nodes are not activated/deactivated additionally by the attack. (Lemma F.4)
- After proving Lemma F.2, F.3, and F.4, we finally present the proof of Lemma 4.2.

Lemma F.2. *Consider the case where $\zeta = 0$, i.e., Z is a linear transformation of X. Denote $m = \sup_h \|U_h\|_0 \le m^*$. Given at least one feature received by the hidden node h is non-zero, for $v = o(\|U_h\|_\infty/\sqrt{k})$, the conditional distribution of $X^\top U_h$ satisfies*

$$P\left(|X^\top U_h| < v \mid X_1 = x_1, |U_{1,h}| > 0\right) = \begin{cases} 0 & m = 1 \\ O\left(\frac{vk^{3/2}}{d\|U_h\|} + \left(\frac{k}{d}\right)^2\right) & m = \Theta(1) \\ \Phi\left(\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right) + c_u' \frac{1}{1+\left|\frac{v-U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right|^3}\sqrt{\frac{d}{mk}} & m \to \infty \end{cases}.$$

As a result,

$$P(\exists h = 1, \dots, H,\ s.t. |X^\top U_h| \in (0, v/\|U_h\|)\ and\ \exists |X_i U_{i,h}| > 0) = \begin{cases} 0 & m = 1 \\ O\left(vH\frac{k^{5/2}}{d^2} + H\left(\frac{k}{d}\right)^3\right) & m = \Theta(1) \\ O\left(\frac{mkH}{d}\Phi\left(\frac{v\|U_h\| - U_{1,h}/\sqrt{k}}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right) + c_u' \frac{H}{1+\left|\frac{v\|U_h\| - U_{1,h}/\sqrt{k}}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right|^3}\sqrt{\frac{mk}{d}}\right) & m \to \infty \end{cases}, \tag{12}$$

and when taking v such that $v/\|U_h\| \gg 1/\sqrt{d}$ and $v = o(1/\sqrt{mk})$, if $H = o(d^2/(k^2m^3))$ when $m \to \infty$,

$$P(\exists h = 1, \dots, H,\ s.t. |X^\top U_h| \in (0, v/\|U_h\|)\ and\ \exists |X_i U_{i,h}| > 0) \to 0.$$

Proof. We consider three regimes: (1) $\|U_h\|_0 = 1$, (2) $\|U_h\|_0 = m$ for some constant m, and (3) $\|U_h\|_0 \to \infty$.

Case 1, $\|U_h\|_0 = 1$ If a node only contains a single feature, then the conditional distribution of $X^\top U_h$ given the feature is non-zero is a single value.

Case 2, $\|U_h\|_0 = m$ If there are m features for some constant m, then the probability that all features are zero is $(1 - k/d)^m$, and the probability that only one of the features is non-zero is in $O(mk/d)$.

Given at least one feature is non-zero in the node, the probability of two or more features being non-zero is $1 - (1 - k/d)^{m-1}$, and the probability of exactly two features being non-zero is $(m-1)(1-k/d)^{m-2}(k/d)$.

When there are two features activated, denoting $\mathcal{X}$ as the set of non-zero features, under Assumption 4.1, the probability of $|X^\top U_h| < v$ for $v = o(\|(U_h)_\mathcal{X}\|)$ is in $O(v\sqrt{k}/\|U_h\|)$.

As a result, for $v = o(\|U_h\|)$, since m is a constant, we have

$$\begin{aligned} & P(|X^\top U_h| < v \mid \text{At least one feature is non-zero}) \\ \le\ & P(\text{ Two features are non-zero }, |X^\top U_h| < v \mid \text{At least one feature is non-zero}) \\ & + P(\text{ Three or more features are non-zero} \mid \text{At least one feature is non-zero}) \\ =\ & O\left(\frac{v\sqrt{k}}{\|U_h\|}(m-1)\frac{k}{d}\left(1-\frac{k}{d}\right)^{m-2}\right) + 1 - \left(1-\frac{k}{d}\right)^{m-1} - (m-1)\frac{k}{d}\left(1-\frac{k}{d}\right)^{m-2} \\ =\ & O\left(\frac{vk^{3/2}}{d\|U_h\|} + \left(\frac{k}{d}\right)^2\right). \end{aligned}$$

Case 3, $\|U_h\|_0 \to \infty$ If there are $m \to \infty$ features and $m = o(d/k)$, then assuming the first coordinate $X_1 = x_1$ is nonzero and $U_{1,h} \neq 0$, $U_h^\top X$ conditionally approximately follows a normal distribution $N(U_{1,h}x_1, \sigma^2\|U_{-1,h}\|^2/d)$. In this case, denoting Φ and the probability function of the standard Gaussian distribution, using non-uniform Berry-Esseen bound in Grigor'eva and Popov (2012), for some universal constant c_u,

$$\begin{aligned} & P(X^\top U_h < v \mid X_1 = x_1, U_{1,h}x_1 > 0) \\ \leq\ & \Phi\left(\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right) + c_u \frac{\sum \mathbb{E}|X_i U_{i,h}|^3}{\left(1 + \left|\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right|^3\right)} \frac{1}{(\sum \mathbb{E}|X_i U_{i,h}|^2)^{3/2}}, \end{aligned}$$

where

$$\begin{aligned} \sum_{i>1} \mathbb{E}|X_i U_{i,h}|^2 &= \sigma^2\|U_{-1,h}\|^2/d, \\ \sum_{i>1} \mathbb{E}|X_i U_{i,h}|^3 &= O\left(\sum_{i>1} |U_{i,h}|^3 \frac{1}{d\sqrt{k}}\right). \end{aligned}$$

Under Assumption 4.1, we have

$$\sum_{i>1} |U_{i,h}|^3 = O\left(m\left(\frac{\|U_{-1,h}\|^2}{m}\right)^{3/2}\right) = O\left(\frac{\|U_{-1,h}\|^3}{\sqrt{m}}\right).$$

As a result, we conclude that for some constant $c_u' > 0$,

$$\begin{aligned} & P(X^\top U_h < v \mid X_1 = x_1, U_{1,h}x_1 > 0) \\ \leq\ & \Phi\left(\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right) + c_u' \frac{\|U_{-1,h}\|^3/(d\sqrt{km})}{\left(1 + \left|\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right|^3\right)} \frac{1}{(\|U_{-1,h}\|^2/d)^{3/2}} \\ =\ & \Phi\left(\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right) + c_u' \frac{1}{\left(1 + \left|\frac{v - U_{1,h}x_1}{\sqrt{\sigma^2\|U_{-1,h}\|^2/d}}\right|^3\right)} \sqrt{\frac{d}{mk}}. \end{aligned}$$

We use Berry-Esseen bound rather than Hoeffding/Berstein bounds because the latter ones involve the range $M = \sup|X^\top U_h| - \inf|X^\top U_h|$ which is too broad.

The final (12) is a union bound taken for the m features in each node for all H hidden nodes. □

After discussing the distribution of $X^\top U_h$, we further add the noise ξ (Lemma F.3) and the attack (Lemma F.4) into the model.

Lemma F.3. *For any $v \geq 1/\sqrt{d}$,*

$$P\left(\sup_h |\xi^\top W_h|/\|W_h\| > v\right) = O\left(\frac{H\sqrt{d}}{v}\exp(-v^2 d/(2\zeta^2))\right).$$

When taking $v \gg \sqrt{(\log d)/d}$ and $H = poly(d)$, the probability bound goes to zero.

Furthermore, under the conditions of Lemma F.2, when taking b_h such that $b_h \gg \sqrt{(\log d)/d}\|U_h\|$ and $b_h = o(1/\sqrt{k \sup_h \|U_h\|_0})$,

$$P(\exists h = 1, \ldots, H,\ s.t.\ Node\ h\ is\ activated\ by\ \xi\ or\ deactivated) \to 0.$$

Proof of Lemma F.3. Since $\xi \sim N(\mathbf{0}, \zeta^2 I_d/d)$, we have $\zeta^\top W_h/\|W_h\| \sim N(0, \zeta^2/d)$. From Lemma F.1, we obtain

$$P\left(\sup_h |\xi^\top W_h|/\|W_h\| > v\right) \le HP(|\xi^\top W_h|/\|W_h\| > v) = O\left(\frac{H\sqrt{d}}{v}\exp(-v^2 d/(2\zeta^2))\right), \tag{13}$$

which provides the tail distribution of $|\xi^\top W_h|/\|W_h\|$. When taking $v \gg \sqrt{(\log d)/d}$, the above probability goes to zero, i.e., with probability tending to 1, no extra nodes are activated due to ξ.

Furthermore, for the nodes which are activated by non-zero features, we have

$$P(\exists h = 1, \dots, H, \ s.t., \text{ Node } h \text{ is deactivated}) = O(HP(\text{Node } h \text{ is deactivated})),$$

and

$$\begin{aligned} & P(\text{Node } h \text{ is deactivated}) \\ \le\ & P(\text{Node } h \text{ is deactivated} \mid \text{One feature in node } h \text{ is non-zero}) \\ & \qquad \times P(\text{One feature in node } h \text{ is non-zero}) \\ & + P(\text{Two or more features in node } h \text{ is non-zero}), \end{aligned}$$

where taking $m = \|U_h\|_0$,

$$\begin{aligned} & P(\text{Node } h \text{ is deactivated} \mid \text{One feature in node } h \text{ is non-zero}) \\ =\ & O\left(mP(|X^\top U_h| < b_h + |\xi^\top U_h| \mid X_1 = x_1, U_{1,h}x_1 > 0)\right) \\ =\ & O\left(mP(|X^\top U_h| < 2b_h \mid X_1 = x_1, U_{1,h}x_1 > 0)\right) + O\left(mP(|\xi^\top U_h| > b_h)\right). \end{aligned}$$

Therefore, taking $b_h \gg \sqrt{(\log d)/d}\|U_h\|$ and $b_h = o(\|U_h\|/\sqrt{k \sup_h \|U_h\|_0})$, when taking proper H, one can show that

$$\begin{aligned} & P(\exists h = 1, \dots, H, \ s.t. \text{ Node } h \text{ is activated by } \xi \text{ or deactivated}) \\ =\ & P(\exists h = 1, \dots, H, \ s.t., \text{ Node } h \text{ is activated by } \xi) + P(\exists h = 1, \dots, H, \ s.t., \text{ Node } h \text{ is deactivated}) \\ \le\ & O\left(\frac{H\sqrt{d}\|U_h\|}{b_h}\exp(-b_h^2 d/(2\zeta^2\|U_h\|^2))\right) \\ & + O\left(HmP(|X^\top U_h| < 2b_h \mid X_1 = x_1, U_{1,h}x_1 > 0)\right) + O\left(\frac{Hm\sqrt{d}\|U_h\|}{b_h}\exp(-b_h^2 d/(2\zeta^2\|U_h\|^2))\right) \\ & + O(HP(\text{Two or more features in node } h \text{ is non-zero in node } h)), \end{aligned}$$

where all the four terms go to zero based on Lemma F.2 and (13). □

Lemma F.4. *Under the conditions of Lemma F.2, when taking b_h such that $b_h \gg \sqrt{(\log d)/d}\|U_h\|$ and $b_h = o(1/\sqrt{k \sup_h \|U_h\|_0})$, and $\epsilon = o(\inf_h b_h/\|U_h\|)$ for $\mathcal{L}_2$ attack,*

$$P(\exists h = 1, \dots, H, \ s.t. \text{ Node } h \text{ is activated by } \xi \text{ and the attack or deactivated}) \to 0.$$

For $\mathcal{L}_\infty$ attack, when $\epsilon = o(\inf_h b_h/\|U_h\|/\sqrt{\|U_h\|_0})$, the above inequality also holds.

Proof of Lemma F.4. Since $\epsilon = o(\inf_h b_h/\|U_h\|)$, if all the features in node h are zero, then we have

$$P\left(\sup_h |\xi^\top W_h|/\|W_h\| + \epsilon > v\right) = O\left(\frac{H\sqrt{d}}{v-\epsilon}\exp(-(v-\epsilon)^2 d/(2\zeta^2))\right) = O\left(\frac{H\sqrt{d}}{v}\exp(-v^2 d/(2\zeta^2))\right),$$

Thus with probability tending to 1, all the nodes will not be additionally activated by ξ and the attack.

Furthermore, if a node h is activated by non-zero features, then we have the following decomposition:

$$P(\text{Node } h \text{ is deactivated})$$

$$\begin{aligned}
\leq\ & P(\text{Node } h \text{ is deactivated} \mid \text{One feature in node } h \text{ is non-zero}) \\
& \qquad \times P(\text{One feature in node } h \text{ is non-zero}) \\
& + P(\text{Two or more features in node } h \text{ is non-zero}),
\end{aligned}$$

where taking $m = \|U_h\|_0$,

$$\begin{aligned}
& P(\text{Node } h \text{ is deactivated} \mid \text{One feature in node } h \text{ is non-zero}) \\
=\ & O\left(mP(|X^\top U_h| < b_h + \epsilon\|U_h\| + |\xi^\top U_h| \mid X_1 = x_1, U_{1,h}x_1 > 0)\right) \\
=\ & O\left(mP(|X^\top U_h| < 2b_h \mid X_1 = x_1, U_{1,h}x_1 > 0)\right) \\
& + O\left(mP(\epsilon\|U_h\| + |\xi^\top U_h| > b_h)\right),
\end{aligned}$$

and the final steps are the same as in Lemma F.3.

For $\mathcal{L}_\infty$ attack, $|\delta_\infty^\top U_h| = \epsilon\|U_h\|_1 = O(\epsilon\|U_h\|\sqrt{m})$, and one can replace the ϵ in the derivations of $\mathcal{L}_2$ attack case with $\epsilon\sqrt{m}$ to go through the proof. □

Proof of Lemma 4.2. For $\mathcal{L}_2$ attack, since we consider the FGM attack, we first calculate the gradient of l w.r.t. z. Denote $f_{W,b}$ as the non-linear neural network, then

$$\frac{\partial}{\partial z} l_0(z, y; W, b) = \frac{\partial l_0}{\partial f}\frac{\partial f}{\partial z},$$

where

$$\frac{\partial f}{\partial z} = \frac{\partial}{\partial z}\sigma(z^\top W, b)a = W\text{diag}(\mathbb{I}(z^\top W, b))a = MU\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a.$$

As a result, the attack becomes

$$\delta_2(z, y, f_{W,b}, l) = \epsilon\,\text{sgn}\left(\frac{\partial l}{\partial f}\right)\frac{MU\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a}{\|MU\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a\|},$$

and

$$\delta_2(z, y, f_{W,b}, l) = \epsilon\,\text{sgn}\left(\frac{\partial l}{\partial f}\right)\text{sign}\big[MU\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a\big].$$

Using $\mathcal{L}_2$ attack, the attacked fitted value becomes

$$\begin{aligned}
f_{W,b}(z + \delta_2) &= \sigma\left(\left(z + \epsilon\,\text{sgn}\left(\frac{\partial l}{\partial f}\right)\frac{MU\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a}{\|MU\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a\|}\right)^\top W, b\right)a \\
&= \sigma\left(\left(x + \xi^\top M + \epsilon\,\text{sgn}\left(\frac{\partial l}{\partial f}\right)\frac{U\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a}{\|U\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a\|}\right)^\top U, b\right)a \\
&= \left(x + \xi^\top M + \epsilon\,\text{sgn}\left(\frac{\partial l}{\partial f}\right)\frac{U\text{diag}(\mathbb{I}(x^\top U + \xi^\top M, b))a}{\|U\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a\|}\right)^\top U\text{diag}(\mathbb{I}((z + \delta_2)^\top W, b))a.
\end{aligned}$$

In order to cancel some terms in the above representation, we need that $\mathbb{I}((z+\delta_2)^\top W, b) = \mathbb{I}(z^\top W, b)$ in probability when $\epsilon = o(b^*)$, which has been shown in Lemma F.4.

As a result, with probability tending to 1,

$$f_{W,b}(z + \delta_2) = f_{W,b}(z) + \epsilon\,\text{sgn}\left(\frac{\partial l}{\partial f}\right)\|U\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))a\|.$$

Thus we have

$$\Delta_{W,b}(z, y) = l_\epsilon(z, y; W, b) - l_0(z, y; W, b)$$

$$\begin{aligned} &= l_0(z+\delta_2, y; W, b) - l_0(z, y; W, b) \\ &= \frac{\partial l_0}{\partial f_{w,b}}(f_{W,b}(z+\delta_2) - f_{W,b}(z)) + O(\epsilon^2) \\ &= \epsilon \frac{\partial l}{\partial f_{W,b}} \left\| a^\top \mathrm{diag}(\mathbb{I}(W^\top z, b)) W^\top \right\|_2 + o, \end{aligned}$$

where o represents the remainder term.

Assume the first coordinate of x is non-zero. Since with probability tending to 1 (Lemma F.3), all the hidden nodes receiving x_1 are activated, we have

$$a^\top \mathrm{diag}(\mathbb{I}(U^\top x + \xi^\top W, b)) U_{1,:} = a^\top U_{1,:} = \theta_1.$$

Assume the second coordinate x_2 of x is zero, when $(W, b) \in \mathcal{M}$, all non-zero elements in $U_{2,:}$ have the same sign as θ_2, and

$$0 \le |a^\top \mathrm{diag}(\mathbb{I}(U^\top x + \xi^\top W, b)) U_{2,:}| \le |\theta_2|,$$

and the left/right equation holds if every node containing x_2 is not/is activated.

As a result, we conclude that, with probability tending to 1,

$$\|\theta_{\mathcal{X}}\|_2 \le \|a^\top \mathrm{diag}(\mathbb{I}(U^\top X + \xi^\top W, b)) U^\top\|_2 \le \|\theta\|_2.$$

□

F.3 Proof for Supervised Pre-training

Denote

$$\psi = \frac{H m^3 k^3 \log^2 k}{d^2} + \sqrt{\frac{k}{d}}. \tag{14}$$

Before we start the proof of the theorems, we provide an additional lemma to characterize $X^\top \theta_0$. Different from the results in Section 4, since we directly work on the risk, rather than probability bounds, we need to know the distribution of $X^\top \theta_0$.

Lemma F.5. *Under Assumption 3.1,*

$$P(|X^\top \theta_0 - \mathbb{E}(X^\top \theta_0)| > v) = O\left(\Phi(-v)\right) + O\left(\frac{1}{\sqrt{k}(1+v^3)}\right).$$

Proof of Lemma F.5. Using Berry-Esseen bound, we have

$$\begin{aligned} & P(X^\top \theta_0 - \mathbb{E}(X^\top \theta_0) < v) \\ \le\ & \Phi\left(\frac{v-0}{\sqrt{\sigma^2 \|\theta_0\|^2/d}}\right) + c_u \frac{\sum \mathbb{E}|X_i \theta_i|^3}{\left(1 + \left|\frac{v}{\sqrt{\sigma^2 \|\theta_0\|^2/d}}\right|^3\right)} \frac{1}{(\sigma^2 \|\theta_0\|^2/d)^{3/2}}, \end{aligned}$$

where

$$\sum \mathbb{E}|X_i \theta_i|^3 = \Theta\left(d \frac{k}{d} \frac{1}{k^{3/2}}\right) = \Theta\left(\frac{1}{\sqrt{k}}\right).$$

As a result, for $v < 0$,

$$P(X^\top \theta_0 - \mathbb{E}(X^\top \theta_0) < v) = O\left(\Phi(v)\right) + O\left(\frac{1}{\sqrt{k}(1+v^3)}\right).$$

Similarly, we also have for $v > 0$,

$$P(X^\top\theta_0 - \mathbb{E}(X^\top\theta_0) > v) = O\left(\Phi(-v)\right) + O\left(\frac{1}{\sqrt{k}(1+v^3)}\right),$$

and merging the two sides we have

$$P(|X^\top\theta_0 - \mathbb{E}(X^\top\theta_0)| > v) = O\left(\Phi(-v)\right) + O\left(\frac{1}{\sqrt{k}(1+v^3)}\right).$$

□

Proof of Theorem A.1. The proof idea is to design some (W, b) such that $(W, b) \in \mathcal{M}$ with good clean performance but poor adversarial performance.

There are two claims for clean performance in Theorem A.1. In the proof, we merge the two proofs together, and finally figure out what is ψ.

Construction Assume in each node, there are m learned features. Here we can take any $m \le m^*$. Then since there are H hidden nodes, every feature appears in Hm/d nodes.

To design a (W, b), we split the total d features into groups of m features. For each group, we randomly pick Hm/d hidden nodes and assign non-zero weights for all features in this group. Thus $\|U_h\| = \Theta(d/(H\sqrt{m}))$. The intercept b_h can be determined correspondingly.

Through the above construction, one can obtain a neural network with good clean performance. The adversarial performance is related to m.

Clean performance We first analyze how the noise ξ affects the performance.

Based on Lemma F.3, when taking all b_h as the same value $b_h = t\zeta\sqrt{(\log d)/d}\|U_h\| = t\zeta\sqrt{d\log d}/(H\sqrt{m})$ such that $t^2/2 > 1$, we have

$$P\left(\sup_h |U_h^\top\xi| > b_h\right) = O\left(\frac{H}{d^{t^2/2}}\right).$$

Besides the noise ξ, another error is caused by the event that the hidden nodes are deactivated when more than one feature is active. Based on Lemma F.2, since we are constructing neural networks whose $m \to \infty$, we have

$$P(\exists h = 1, \ldots, H,\ s.t.\ |X^\top U_h| \in (0, b_h) \text{ while } \exists |X_i U_{i,h}| > 0) = O\left(\frac{k^2m^2H}{d^2}\right),$$

while we want the above union bound to be small enough, we also require $km/d \to 0$ so that for each node, the probability goes to zero.

Denote $E(h) = 1\{|\xi^\top W_h| > b_h \text{ or } (|X^\top U_h| \in (0, 2b_h) \text{ while } \exists |X_i U_{i,h}| > 0)\}$. If node h is activated/deactivated by noise or the non-zero features cancel with each other, we always have $E(h) = 1$. As a result, we use $E(h)$ as the upper bound of the event $\mathbb{I}(Z^\top W_h, b_h) \neq \mathbb{I}(X^\top U_h, 0)$. When taking t large enough, we have

$$P(\exists h = 1, \ldots, H,\ s.t.\ E(h) = 1) = O\left(\frac{k^2m^2H}{d^2}\right) + O\left(\frac{H}{d^{t^2/2}}\right) = O\left(\frac{k^2m^2H}{d^2}\right).$$

For each hidden node, if $E(h) = 1$, then it leads to at most b_h of error. For square loss, when there is only one or several nodes activated/deactivated by the noise, then there will only be $\Theta(b_h)$ error in the fitted value, which is negligible and leads to $O(b_h)$ increase in loss. In the worst case, when all hidden nodes are activated/deactivated by the noise, the fitted value could involve Δ error, leading to an increase of $(\Delta + Y)^2 - Y^2 = \Delta(\Delta + 2Y)$ in the loss.

Since each hidden node has at most m features and $m \ll d$, when there are k features are nonzero and $E(h) = 1$ for some h, there are $O(kHm/d)$ hidden nodes which are mistakenly deactivated in the worst case. We ignore the effect of the noise ξ because it is negligible.

We have

$$\begin{aligned}
& \mathbb{E}l_0(Z,Y;W,b) \\
\leq\ & \mathbb{E}l_0(Z,Y;W,b)1\left\{\forall h=1,\dots,H,\ s.t.\ E(h)=0\right\} \\
& +\mathbb{E}l_0(Z,Y;W,b)1\left\{\exists h=1,\dots,H,\ s.t.\ E(h)=1\right\}1\{\exists O(k\log k) \text{ choices of } i,\ s.t.\ X_i\neq 0\} \\
& +\mathbb{E}l_0(Z,Y;W,b)1\{\exists \geq k\log k \text{ choices of } i,\ s.t.\ X_i\neq 0\}1\{|Y|<d\} \\
& +\mathbb{E}l_0(Z,Y;W,b)1\{|Y|>d\}.
\end{aligned}$$

We bound the above terms one by one. Denote $b_{\max}=\max_h b_h$. Since we use the same upper bound for all hidden nodes to bound the probability that the hidden node gets unexpected zero/nonzero, we have

$$\begin{aligned}
& \mathbb{E}l_0(Z,Y;W,b)1\left\{\forall h=1,\dots,c,\ s.t.\ E(h)=0\right\} \\
& +\mathbb{E}l_0(Z,Y;W,b)1\left\{\exists h=1,\dots,H,\ s.t.\ E(h)=1\right\}1\{\exists O(k\log k) \text{ choices of } i,\ s.t.\ X_i\neq 0\} \\
\leq\ & \mathbb{E}l_0(X,Y;\theta_0)+\underbrace{O(\sqrt{k/d})}_{\text{Noise in the active features}} \\
& +O\left(\mathbb{E}(|X^\top(\theta_0-Ua)|+k\log k(Hm/d)b_{\max})^2 1\{\exists O(k\log k) \text{ choices of } i,\ s.t.\ X_i\neq 0\}\right)+o \\
=\ & \mathbb{E}l_0(X,Y;\theta_0)+O(\sqrt{k/d})+O(\|\theta_0-Ua\|/\sqrt{d})+O\left(b_{\max}^2\left(\frac{k^2m^2H}{d^2}\right)\left(\frac{Hmk\log k}{d}\right)^2\right)+o,
\end{aligned}$$

where o is a negligible term and is caused by the noise ξ. We ignore this term in the following derivations.

Second,

$$\mathbb{E}l_0(Z,Y;W,b)1\{\exists \geq k\log k \text{ choices of } i,\ s.t.\ X_i\neq 0\}1\{|Y|<d\}=o.$$

And finally,

$$\begin{aligned}
& \mathbb{E}l_0(Z,Y;W,b)1\{|Y|>d\} \\
=\ & O(\mathbb{E}(b_{\max}H+Y)^2 1\{|Y|>d\}) \\
=\ & O\left(\mathbb{E}_X\mathbb{E}_Y\left[(b_{\max}H+X^\top\theta_0+(Y-X^\top\theta_0))^2 1\{|Y|>d\}\middle| X=x\right]\right) \\
=\ & O\left(\mathbb{E}(b_{\max}H+\|\theta_0-Ua\|+X^\top\theta_0)^2 1\{|X^\top\theta_0|>d\}\right) \\
=\ & O\left(\mathbb{E}(b_{\max}H+\|\theta_0\|_1)^2 1\{|X^\top\theta_0|>d\}\right) \\
=\ & O\left((b_{\max}H+d)^2\frac{1}{\sqrt{k}(1+d^3)}\right),
\end{aligned}$$

where the second line is because of the distribution of the noise $Y-X^\top\theta_0$, and the last line is based on Lemma F.5.

Since $b_{\max}=o(d/(H\sqrt{mk}))$ when $k\gg\log d$, we have

$$\begin{aligned}
& \mathbb{E}l_0(Z,Y;W,b) \\
=\ & \mathbb{E}l_0(X,Y;\theta_0)+O(\|\theta_0-Ua\|/\sqrt{d})+O(\sqrt{k/d})+O\left(b_{\max}^2\left(\frac{k^2m^2H}{d^2}\right)\left(\frac{Hmk\log k}{d}\right)^2\right) \\
& +O\left(\frac{(b_{\max}H+d)^2}{\sqrt{k}d^3}\right)+o \\
=\ & \mathbb{E}l_0(X,Y;\theta_0)+O(\sqrt{k/d})+O(\|\theta_0-Ua\|/\sqrt{d})+O\left(b_{\max}^2\left(\frac{k^2m^2H}{d^2}\right)\left(\frac{Hmk\log k}{d}\right)^2\right)+o,
\end{aligned}$$

and

$$\mathbb{E}l_0(Z,Y;W,b)=\mathbb{E}l_0(X,Y;\theta_0)+O(\|\theta_0-Ua\|/\sqrt{d})+O\left(\frac{Hm^3k^3\log^2 k}{d^2}\right)+O(\sqrt{k/d}), \tag{15}$$

from which we define ψ.

For absolute loss and logistic regression, the error ϕ still holds.

Adversarial performance There are on average $\Theta(k)$ active features in each data point, which is far less than the total d features. As a result, each data point on average activates $\Theta(kHm/d)$ hidden nodes, and

$$\mathbb{E}\|a^\top \mathrm{diag}(\mathbb{I}(U^\top X, b))U^\top\|_2 = \Theta(\mathbb{E}\sqrt{m}\|\theta_{\mathcal{X}}\|_2) = \Theta(\sqrt{mk}).$$

One the other hand, when we bound the error in clean model, we consider $|Y| < d$ and $|Y| > d$ cases. In the worse case, the increase of loss caused by the attack is $\epsilon\|\theta\|$, which is much smaller than d. As a result, whether or not we have the attack or not does not affect ψ.

As a result,

$$\mathbb{E}l_\epsilon(X, Y; W, b) = \mathbb{E}l_0(X, Y; \theta_0) + O(\psi) + \Theta(\epsilon\sqrt{mk}).$$

On the other hand, taking $m = 1$, one can also design a neural network such that

$$\inf_{W',b'\in\mathcal{M}} \mathbb{E}l_\epsilon(X, Y; W', b') \le \mathbb{E}l_\epsilon(X, Y; W, b) = \mathbb{E}l_0(X, Y; \theta_0) + O(\epsilon\sqrt{k}) + O(\psi), \tag{16}$$

which finally indicates that

$$\mathbb{E}l_\epsilon(X, Y; W, b) = \inf_{W',b'\in\mathcal{M}} \mathbb{E}l_\epsilon(X, Y; W', b') + O(\psi) + \Theta(\epsilon\sqrt{mk}).$$

□

Proof of Theorem A.2. When $(W, b) \in \mathcal{M}$, the minimal non-zero value of $|U_{i,j}|$ is in $\Theta(d/(Hm^*))$. Assume on average there are m features in each hidden node, then on average, there are $\Theta(Hkm/d)$ nodes are activated. In addition to the activated features, there are $\Theta(Hkm(m-1)/d)$ elements of $U_{i,h}$ leaked to the attacker.

When these additional elements are all from different features, the increase of the loss is the smallest, which means that

$$\epsilon\left\|\sum_{\text{activated h}} U_h\right\| \ge \Theta\left(\epsilon\sqrt{k + \left(\frac{d}{Hm}\right)^2\left(\frac{Hkm(m-1)}{d}\right)}\right) = \Theta\left(\epsilon\left(\sqrt{k} + \frac{1}{2}\frac{d(m-1)}{Hm}\right)\right) + o.$$

When $H = o(\epsilon d^{3/2})$, $\epsilon d/H \gg \psi$. As a result, when an solution (W, b) has an adversarial loss $O(\psi)$-close to $\min_{W',b'\in\mathcal{M}} \mathbb{E}l_\epsilon(Z, Y; M', b')$, it also purifies most features, i.e, $m = 1 + o(1)$.

In terms of the clean performance, the result holds as

$$\mathbb{E}l_0(Z, Y; W, b) \le \min_{W',b'\in\mathcal{M}} \mathbb{E}l_\epsilon(Z, Y; W', b') + \Theta(\epsilon\sqrt{k}) + O(\psi) \le \mathbb{E}l_0(X, Y; \theta_0) + \Theta(\epsilon\sqrt{k}) + O(\psi) + o.$$

□

F.4 Proof for Contrastive Learning

Proof of Lemma 5.1. Given $g(z, z') = x_1^\top T^\top T x_2$, the contrastive loss becomes

$$\begin{aligned} & \mathbb{E}\log(1 + \exp(-X^\top T^\top T X)) + \mathbb{E}\log(1 + \exp(X^\top T^\top T X')) \\ = \; & \underbrace{\mathbb{E}\log(1 + \exp(-X^\top P D P^\top X))}_{:=V_1} + \underbrace{\mathbb{E}\log(1 + \exp(X^\top P D P^\top X'))}_{:=V_2}. \end{aligned}$$

To prove Lemma 5.1, the key is to show that, fixing $tr(D)$, both V_1 and V_2 are minimized when $D \propto I_d$.

For V_1, as $\log(1 + \exp(-v))$ is a convex function w.r.t. v, to show that $D \propto I_d$, we would like to show that for any $v \ge 0$,

$$\mathbb{E}\left[X^\top P D P^\top X \middle| \|P^\top X\|^2 = v\right] = tr(D)v/d. \tag{17}$$

Based on the distribution of X, i.e., the distribution of each coordinate is symmetric and identical, we have

$$\begin{aligned}\mathbb{E}\left[X^\top PDP^\top X \middle| \|P^\top X\|^2 = v\right] &= \mathbb{E}\left[\sum_{i=1}^d X_i^2 (PDP^\top)_{i,i} \middle| \|P^\top X\|^2 = v\right] \\ &= \mathbb{E}\left[X_1^2 \sum_{i=1}^d (PDP^\top)_{i,i} \middle| \|P^\top X\|^2 = v\right] \\ &= \mathbb{E}\left[X_1^2 tr(D) \middle| \|P^\top X\|^2 = v\right] \\ &= tr(D)v/d.\end{aligned}$$

Consequently,

$$\begin{aligned}\mathbb{E}\left[\log(1+\exp(-X^\top PDP^\top X)) \middle| \|X\|^2 = v\right] &\geq \log\left(1+\exp\left(\mathbb{E}\left[-X^\top PDP^\top X \middle| \|X\|^2 = v\right]\right)\right) \\ &= \log\left(1+\exp\left(-tr(D)v/d\right)\right),\end{aligned}$$

the equation holds when $D \propto I_d$.

Similarly, for V_2, $\log(1+\exp(v))$ is a convex function w.r.t. v, and

$$\begin{aligned}\mathbb{E}\left[\log(1+\exp(X^\top PDP^\top X')) \middle| X^\top X' = v\right] &\geq \log\left(1+\exp\left(\mathbb{E}\left[X^\top PDP^\top X' \middle| X^\top X' = v\right]\right)\right) \\ &= \log\left(1+\exp\left(tr(D)v/d\right)\right).\end{aligned}$$

As a result, fixing $tr(D)$, both V_1 and V_2 are minimized when taking $D \propto I_d$. □

Proof of Theorem 5.2, loss for similar pairs. For $A = \tau W^+$, we have

$$A = \tau W^+ = \tau W^\top (WW^\top)^{-1} = \tau U^\top M^\top (MUU^\top M^\top)^{-1} = \tau U^\top (UU^\top)^{-1} M^\top,$$

which indicates that with probability tending to 1, $\sigma(Z^\top W, b)A = \tau X^\top UU^\top (UU^\top)^{-1} M^\top + o = \tau X^\top M^\top + o$, where the term o is negligible and is caused by the noise ξ.

With probability tending to 1, we have

$$\begin{aligned}& g_{W,b}(z+\delta_2, z') \\ =\ & g_{W,b}(z,z') + \epsilon \operatorname{sgn}\left(\frac{\partial l}{\partial f}\right) \|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))(U^\top x + M^\top \xi)\|,\end{aligned}$$

where with probability tending to 1, $\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b)) = \operatorname{diag}(\mathbb{I}(x^\top U, \mathbf{0}))$, and

$$\begin{aligned}& \|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))(U^\top x + W^\top \xi)\| \\ \leq\ & \|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))U^\top x\| \\ & + \|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))W^\top \xi\| \\ =\ & \|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMx\| + \|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))W^\top \xi\|.\end{aligned}$$

Now we look into $\|U\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMx\|$.

Active features Assume the first coordinate of x is non-zero. Since with probability tending to 1, all hidden nodes involving x_1 are activated, we have $U_{1,:}\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b)) = U_{1,:}$, and

$$U_{1,:}\operatorname{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMx = U_{1,:}AMx = \tau x_1. \tag{18}$$

Inactive features Assume the second coordinate of x is zero. We have for any active feature $i \in \mathcal{X}$,

$$\begin{aligned} & |U_{2,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AM_{:,i}x_i| \\ = \; & |\tau U_{2,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))U^\top (UU^\top)^{-1}M^\top M_{:,i}x_i| \\ = \; & |\tau U_{2,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))U^\top (UU^\top)^{-1}_{:,i}x_i| \\ = \; & \Theta(\tau |x_i| \alpha). \end{aligned} \tag{19}$$

For $i \in \mathcal{X}^c$, the value of the ith element of $U_{2,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))$ does not matter because $x_i = 0$, i.e.,

$$U_{2,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AM_{:,i}x_i \equiv 0. \tag{20}$$

As a result, we have

$$|U_{2,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))Ax| = O(\tau\alpha),$$

and thus the adversarial loss does not affected by whether the neural network is purified or not.

Noise We now look into $\|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))W^\top \xi\|$.

From the assumptions, we know that

$$U_{i,:}\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))A_{:,j} = \begin{cases} \tau & i = j \in \mathcal{X} \\ 0 & i \neq j, i \in \mathcal{X} \\ \Theta(\alpha) & i \neq j, i \in \mathcal{X}^c, j \in \mathcal{X} \\ \Theta(\alpha^2) & \text{otherwise} \end{cases}. \tag{21}$$

Consequently, taking proper value of α, we get

$$\mathbb{E}\|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))W^\top \xi\|^2 = o(k),$$

which indicates that the attack is dominated by $\|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top Mx\|$.

Next, we examine the performance of $g_{W,b}(z, z')$. We know that

$$\begin{aligned} g_{W,b}(z,z') \;=\; & (x^\top U + \xi^\top W)\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + (\xi')^\top W, b))(U^\top x + W^\top \xi') \\ = \; & x^\top U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + (\xi')^\top W, b))U^\top x \\ & + \xi^\top W\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + (\xi')^\top W, b))W^\top \xi' \\ & + \xi^\top W\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + (\xi')^\top W, b))U^\top x \\ & + x^\top \mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + (\xi')^\top W, b))W^\top \xi'. \end{aligned}$$

Based on (21), one can see that with probability tending to one over the randomness of (x, ξ, ξ'),

$$g_{W,b}(z,z') = x^\top U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}(x^\top U + (\xi')^\top W, b))U^\top x + o.$$

Finally, when $\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b)) \neq \mathrm{diag}(\mathbb{I}(x^\top U, \mathbf{0}))$, if the eigenvalues of $UU^\top$ are bounded and bounded away from zero, we have

$$\mathbb{E}g_{W,b}(Z, Z')1\{\exists E(h) = 1\} = O(\|UU^\top\| P\{\exists E(h) = 1\}) = O(\psi).$$

Different from supervised learning, in contrastive learning, we only care about the nodes which are related to the non-zero features in x', so we only need to consider $O(Hk/d)$ hidden nodes rather than all the H nodes. As a result, the value of ψ gets smaller.

□

Proof of Theorem 5.2, loss for dissimilar pairs. Similar to Theorem 5.2, loss for similar pairs, we have

$$\begin{aligned} & \|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}((x')^\top U + (\xi')^\top W, b))(U^\top x' + W^\top \xi')\| \\ \leq \; & \|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}((x')^\top U + (\xi')^\top W, b))U^\top x'\| \\ & + \|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}((x')^\top U + (\xi')^\top W, b))W^\top \xi'\| \\ = \; & \|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMx'\| + \|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}((x')^\top U + (\xi')^\top W, b))W^\top \xi'\|. \end{aligned}$$

We look into $\|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMx'\|$.

Active features Assume the first coordinate of x is non-zero. Since with probability tending to 1, all hidden nodes involving x_1 are activated, we have $U_{1,:}\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b)) = U_{1,:}$, and

$$U_{1,:}\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMx' = U_{1,:}AMx' = \tau x'_1.$$

Consequently,

$$\begin{aligned}
\mathbb{E}_{X'}\|U_{\mathcal{X}}\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMX'\|/\tau &= \mathbb{E}\sqrt{\sum_{i\in\mathcal{X}}|X'_i|^2} \\
&= \mathbb{P}(\text{Only one } X'_i \text{ for } i \in \mathcal{X} \text{ is nonzero})/\sqrt{k} \\
&\quad +\sqrt{2}\mathbb{P}(\text{Only two } X'_i \text{ for } i \in \mathcal{X} \text{ are nonzero})/\sqrt{k} \\
&\quad +\sqrt{3}\mathbb{P}(\text{Only three } X'_i \text{ for } i \in \mathcal{X} \text{ are nonzero})/\sqrt{k} \\
&\quad +\dots,
\end{aligned}$$

As a result,

$$\begin{aligned}
\mathbb{E}\sqrt{\sum_{i\in\mathcal{X}}|X'_i|^2} &\geq \mathbb{P}(\text{Only one } X'_i \text{ for } i \in \mathcal{X} \text{ is nonzero})/\sqrt{k} \\
&\geq \frac{1}{c_l\sqrt{k}}\frac{k^2}{d} \\
&= \Theta(k^{3/2}/d),
\end{aligned}$$

where $c_l > 0$ is some constant number. Meanwhile,

$$\begin{aligned}
\mathbb{E}\sqrt{\sum_{i\in\mathcal{X}}|X'_i|^2} &= \mathbb{P}(\text{Only one } X'_i \text{ for } i \in \mathcal{X} \text{ is nonzero})/\sqrt{k} \\
&\quad +\sqrt{2}\mathbb{P}(\text{Only two } X'_i \text{ for } i \in \mathcal{X} \text{ are nonzero})/\sqrt{k} \\
&\quad +\sqrt{3}\mathbb{P}(\text{Only three } X'_i \text{ for } i \in \mathcal{X} \text{ are nonzero})/\sqrt{k} \\
&\quad +\dots, \\
&\leq \mathbb{P}(\text{Only one } X'_i \text{ for } i \in \mathcal{X} \text{ is nonzero})/\sqrt{k} \\
&\quad +2\mathbb{P}(\text{Only two } X'_i \text{ for } i \in \mathcal{X} \text{ are nonzero})/\sqrt{k} \\
&\quad +3\mathbb{P}(\text{Only three } X'_i \text{ for } i \in \mathcal{X} \text{ are nonzero})/\sqrt{k} \\
&\quad +\dots, \\
&\leq \frac{1}{\sqrt{k}}\left(\frac{k^2}{d} + 2\left(\frac{k^2}{d}\right)^2 + 3\left(\frac{k^2}{d}\right)^3 + \dots\right) \\
&= \frac{1}{\sqrt{k}}\frac{k^2/d}{1-k^2/d} + \frac{1}{\sqrt{k}}\frac{(k^2/d)^2}{1-k^2/d} + \dots \\
&= \Theta(k^{3/2}/d).
\end{aligned}$$

Since both the upper bound and lower bound are in $\Theta(k^{3/2}/d)$, we have $\mathbb{E}_{X'}\|U_{\mathcal{X}}\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMX'\| = \Theta(\tau k^{3/2}/d)$.

Inactive features Assume the second coordinate of x is zero. From the assumption, we have

$$(U\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))U^\top(UU^\top)^{-1})_{i,j} = \begin{cases} 1 & i,j \in \mathcal{X} \\ O(\alpha) & i \neq j, i \in \mathcal{X}^c, j \in \mathcal{X} \\ O(\alpha^2) & \text{otherwise} \end{cases}.$$

As a result, given x,

$$U_{2,:}\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMX' = O_p\left(\frac{k}{d}\alpha + \alpha^2\right),$$

and $\|U_{\mathcal{X}^c}\text{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AMX'\| = O_p(\alpha^2\sqrt{d} + \alpha k/\sqrt{d})$.

Noise Taking a proper value of α, we get $\mathbb{E}\|U\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b))AA^\top \mathrm{diag}(\mathbb{I}((x')^\top U + (\xi')^\top W, b))W^\top \xi'\|^2$ is negligible.

When $\mathrm{diag}(\mathbb{I}(x^\top U + \xi^\top W, b)) \neq \mathrm{diag}(\mathbb{I}(x^\top U, \mathbf{0}))$, the bound follows that same as Theorem 5.2.

□

F.5 Proof for Downstream Task

Proof of Proposition 6.1 . The arguments for Lemma 4.2 holds for any θ, not limited to θ_0. As a result, when applying the neural network in new tasks, feature purification still preserves the adversarial robustness. □