Understanding Square Loss in Training Overparameterized Neural Network Classifiers

Tianyang Hu*
Huawei Noah's Ark Lab
hutianyang1@huawei.com

Jun Wang*
HKUST
jwangfx@connect.ust.hk

Wenjia Wang*
HKUST (GZ) and HKUST
wenjiawang@ust.hk

Zhenguo Li
Huawei Noah's Ark Lab
li.zhenguo@huawei.com

Abstract

Deep learning has achieved many breakthroughs in modern classification tasks. Numerous architectures have been proposed for different data structures but when it comes to the loss function, the cross-entropy loss is the predominant choice. Recently, several alternative losses have seen revived interests for deep classifiers. In particular, empirical evidence seems to promote square loss but a theoretical justification is still lacking. In this work, we contribute to the theoretical understanding of square loss in classification by systematically investigating how it performs for overparameterized neural networks in the neural tangent kernel (NTK) regime. Interesting properties regarding the generalization error, robustness, and calibration error are revealed. We consider two cases, according to whether classes are separable or not. In the general non-separable case, fast convergence rate is established for both misclassification rate and calibration error. When classes are separable, the misclassification rate improves to be exponentially fast. Further, the resulting margin is proven to be lower bounded away from zero, providing theoretical guarantees for robustness. We expect our findings to hold beyond the NTK regime and translate to practical settings. To this end, we conduct extensive empirical studies on practical neural networks, demonstrating the effectiveness of square loss in both synthetic low-dimensional data and real image data. Comparing to cross-entropy, square loss has comparable generalization error but noticeable advantages in robustness and model calibration. Code will be available at https://gitee.com/mindspore/models/tree/master/research/cv/sl_classification.

1 Introduction

The pursuit of better classifiers has fueled the progress of machine learning and deep learning research. The abundance of benchmark image datasets, e.g., MNIST, CIFAR, ImageNet, etc., provide test fields for all kinds of new classification models, especially those based on deep neural networks (DNN). With the introduction of CNN, ResNets, and transformers, DNN classifiers are constantly improving and catching up to the human-level performance. In contrast to the active innovations in model architecture, the training objective remains largely stagnant, with cross-entropy loss being the default choice. Despite its popularity, cross-entropy has been shown to be problematic in some applications. Among others, [1] argued that features learned from cross-entropy lack interpretability and proposed a new loss aiming for maximum coding rate reduction. [2] linked the use of cross-entropy to adversarial

*These authors contributed equally to this work. Corresponding author: Wenjia Wang.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

vulnerability and proposed a new classification loss based on latent space matching. [3] discovered that the confidence of most DNN classifiers trained with cross-entropy is not well-calibrated.

Recently, several alternative losses have seen revived interests for deep classifiers. In particular, many existing works have presented empirical evidence promoting the use of square loss over cross-entropy. [4] conducted large-scale experiments comparing the two and found that square loss tends to perform better in natural language processing related tasks while cross-entropy usually yields slightly better accuracy in image classification. Similar comparisons are also made in [5, 6, 7, 8]. [9] compared a variety of loss functions and output layer regularization strategies on the accuracy and out-of-distribution robustness, and found that square loss has greater class separation and better performance.

In comparison to the empirical investigation, theoretical understanding of square loss in training deep learning classifiers is still lacking. Through our lens, square loss has its uniqueness among classic classification losses, and we argue that it has great potentials for modern classification tasks. Below we list our motivations and reasons why.

Explicit feature modeling Deep learning's success can be largely attributed to its superior ability as feature extractors. For classification, the ideal features should be separated between classes and concentrated within classes. However, when optimizing cross-entropy loss, it is not obvious what the learned features should look like [1]. In the terminal stage of training, [10] proved that when cross-entropy is sufficiently minimized, the penultimate layers features will collapse to the scaled simplex structure. Such phenomenon is referred to as "neural collapse". Knowing this, would it be better to directly enforce the terminal solution by using square loss with the simplex coding [11], as Euclidean distance is probably most natural to measure the distance between samples' features and class-means? Unlike cross-entropy, square loss uses the label codings (one-hot, simplex etc.) as features, which can explicitly control class separations.

Model calibration An ideal classifier should not only give the correct class prediction, but also with the correct confidence. Calibration error measures the closeness of the predicted confidence to the underlying conditional probability η. Using square loss in classification can be essentially viewed as regression where it treats discrete labels as continuous code vectors. It can be shown that the optimal classifier under square loss is $2\eta - 1$, linear with the ground truth. This distinguishing property allows it to easily recover η. In comparison, the optimal classifiers under the hinge loss and cross-entropy are $\text{sign}(2\eta - 1)$ and $\log(\frac{\eta}{1-\eta})$, respectively [12]. Therefore, hinge loss doesn't provide reliable information on the prediction confidence, and cross-entropy can be problematic when η is close to 0 or 1, as stated in [12]. Hence, in terms of model calibration, square loss is a natural choice.

Connections to popular approaches Mixup [13] is a data augmentation technique where augmented data are constructed via convex combinations of inputs and their labels. Like in square loss, mixup treats labels as continuous and is shown to improve the generalization of DNN classifiers. In knowledge distillation [14], where a student classifier is trying to learn from a trained teacher, [15] proved that the "best" teacher with the ground truth conditional probabilities provides the lowest variance in student learning. Since classifiers trained using square loss is a consistent estimator of η, one can argue that it is a better teacher. In supervised contrastive learning [16], the optimal features are the same as those from square loss with simplex label coding [17] (details in Section 3.4).

Despite its lack of popularity in practice, square loss has many advantages that can be easily overlooked. In this work, we systematically investigate from a statistical estimation perspective, the properties of deep learning classifiers trained using square loss. Comparing to cross entropy, the square loss is much more theoretically tractable, which allows us to develop sharper results on the training process and generalization performance. The neural networks in our analysis are required to be sufficiently overparameterized in the neural tangent kernel (NTK) regime. Even though this restricts the implication of our results, it is a necessary first step towards a deeper understanding. In summary, our main contributions are:

- **Generalization error bound**: We consider two cases, according to whether classes are separable or not. In the general non-separable case, we adopt the classical binary classification setting with smooth conditional probability. Fast convergence rate is established for overparameterized neural network classifiers with Tsybakov's noise condition. If two

classes are separable with positive margin, we show that overparameterized neural network classifiers can provably reach zero misclassification error with probability *exponentially* tending to one. To the best of our knowledge, this is the *first* such result for separable but not linear separable classes. Furthermore, we bridge these two cases and offer a *unified* view by considering auxiliary random noise injection.

- **Adversarial robustness (margin property)**: When two classes are separable, the decision boundary is not unique and large-margin classifiers are preferred. The margin is naturally connected to adversarial robustness [18]. In the separable case, we show that the decision boundary of overparameterized neural network classifiers trained by square loss cannot be too close to the data support and the resulting margin is lower bounded away from zero, providing theoretical guarantees for adversarial robustness.

- **Calibration error**: We show that classifiers trained using square loss are inherently well-calibrated, i.e., the trained classifier provides consistent estimation of the ground-truth conditional probability in L_∞ norm. Such property doesn't hold for cross-entropy.

- **Empirical evaluation**: We corroborate our theoretical findings with empirical experiments in both synthetic low-dimensional data and real image data. Comparing to cross-entropy, square loss has comparable generalization error but noticeable advantages in robustness and model calibration.

This work contributes to the theoretical understanding of deep classifiers from a nonparametric estimation point of view, which has been a classic topic in statistics literature. Among others, [19] established the optimal convergence rate for 0-1 loss excess risk when the decision boundary is smooth. [12, 20] extended the analysis to various surrogate losses. [21, 22] studied the convergence rates for plug-in classifiers from local averaging estimators. [23] investigated the convergence rate for support vector machine using Gaussian kernels. We build on and extend classic results to neural networks in the NTK regime. There exist nonparametric results on deep classifiers, e.g., fast convergence rates have been derived for DNN classifiers that minimize the empirical 0-1 loss [24, 25], hinge loss [26] and cross-entropy loss [27], etc. Unlike aforementioned works that only concern the existence of a good classifier (with theoretical worst-case guarantee), in ignorance of the tremendous difficulty of neural network optimization, our results further incorporate the training algorithm and apply to trained classifiers, which relates better to practice. To the best of the authors' knowledge, similar attainable fast rates (faster than $n^{-\frac{1}{2}}$) have never been established for neural network classifiers.

We require the neural network to be overparameterized, which has been extensively studied recently via NTK. Most such results are in the regression setting with a handful of exceptions. [28] showed that only polylogarithmic width is sufficient for gradient descent to overfit the training data using logistic loss. [29] proved generalization error bound for regularized NTK in classification. [30, 31] provided optimization and generalization guarantees for overparameterized network trained with cross-entropy. In comparison, our results are sharper in the sense that we take the ground truth data assumptions into consideration. This allows a faster convergence rate, especially when the classes are separable, where the exponential convergence rate is attainable. The NTK framework greatly reduces the technical difficulty for our theoretical analysis. However, our results are mainly due to properties of the square loss itself and we expect them to hold for a wide range of classifiers.

There are other works investigating the use of square loss for training (deep) classifiers. [32] uncovered that the "neural collapse" phenomenon also occurs under the square loss. [33] compared classification and regression tasks in the overparameterized linear model with Gaussian features, illustrating different roles and properties of loss functions used at the training and testing phases. [34] made interesting observations on effects of popular regularization techniques such as batch normalization and weight decay on the gradient flow dynamics under square loss. These findings support our theoretical results' implication, which further strengthens our beliefs that the essence comes from the square loss and our analysis can go beyond NTK regime.

The rest of this paper is arranged as follows. Section 2 presents some preliminaries. Main theoretical results are in Section 3. The simplex label coding is discussed in Section 3.4 followed by numerical studies in Section 4 and conclusions in Section 5. Technical proofs and details of the numerical studies can be found in the Appendix.

2 Preliminaries

Notation For a function $f : \Omega \to \mathbb{R}$, let $\|f\|_\infty = \sup_{\boldsymbol{x}\in\Omega} |f(\boldsymbol{x})|$ and $\|f\|_p = (\int_\Omega |f(\boldsymbol{x})|^p d\boldsymbol{x})^{1/p}$. For a vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_p$ denotes its p-norm, for $1 \le p \le \infty$. L_p and l_p are used to distinguish function norms and vector norms. For two positive sequences $\{a_n\}_{n\in\mathbb{N}}$ and $\{b_n\}_{n\in\mathbb{N}}$, we write $a_n \lesssim b_n$ if there exists a constant $C > 0$ such that $a_n \le Cb_n$ for all sufficiently large n. We write $a_n \asymp b_n$ if $a_n \lesssim b_n$ and $b_n \lesssim a_n$. Let $[N] = \{1, \dots, N\}$ for $N \in \mathbb{N}$, $\mathbb{I}$ be the indicator function, and $\boldsymbol{I}_d$ be the $d \times d$ identity matrix. $N(\mu, \boldsymbol{\Sigma})$ represents Gaussian distribution with mean μ and covariance $\boldsymbol{\Sigma}$.

Classification problem settings Let P be an underlying probability measure on $\Omega \times \boldsymbol{Y}$, where $\Omega \subset \mathbb{R}^d$ is compact and $\boldsymbol{Y} = \{1, -1\}$. Let (X, Y) be a random variable with respect to P. Suppose we have observations $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \subset (\Omega \times Y)^n$ i.i.d. sampled according to P. The classification task is to predict the unobserved label y given a new input $\boldsymbol{x} \in \Omega$. Let η defined on Ω denote the conditional probability, i.e., $\eta(\boldsymbol{x}) = \mathbb{P}(y = 1|\boldsymbol{x})$. Let P_X be the marginal distribution of P on X.

Remark 1. In this work, we consider the fixed dimension settings, i.e., d is fixed. As pointed by a reviewer, [35, 36] discussed overparameterized neural networks in high dimensional settings under regression. The study in the high dimensional settings will be pursued in the future.

According to whether labels are deterministic, there are two scenarios of interest. If η only takes values from $\{0, 1\}$, i.e., labels are deterministic, we call this case the *separable case*. Note that in the separable case we consider, the classes are not limited to linearly separable but can be arbitrarily complicated, as long as the label is deterministic. Let $\Omega_1 = \{\boldsymbol{x}|\eta(\boldsymbol{x}) = 1\}$, $\Omega_2 = \{\boldsymbol{x}|\eta(\boldsymbol{x}) = 0\}$ and $\Omega = \Omega_1 \cup \Omega_2$. If the probability measure of $\{\boldsymbol{x}|\eta(\boldsymbol{x}) \in (0, 1)\}$ is non-zero, i.e., the labels contain randomness, we call this case the *non-separable case*. In the separable case, we further assume that there exists a positive margin, i.e., $\text{dist}(\Omega_1, \Omega_2) \ge 2\gamma > 0$, where γ is a constant, and $\text{dist}(\Omega_1, \Omega_2) = \inf_{\boldsymbol{x}\in\Omega_1, \boldsymbol{x}'\in\Omega_2} \|\boldsymbol{x} - \boldsymbol{x}'\|_2$. In the non-separable case, to quantify the difficulty of classification, we adopt the well-established Tsybakov's noise condition [21], which measures how large the "difficult region" is where $\eta(\boldsymbol{x}) \approx 1/2$.

Definition 2.1 (Tsybakov's noise condition)**.** Let $\kappa \in [0, \infty]$. We say P has Tsybakov noise exponent κ if there exists a constant $C, T > 0$ such that for all $0 < t < T$, $P_X(|2\eta(X) - 1| < t) \le C \cdot t^\kappa$.

A large value of κ implies the difficult region to be small. It is expected that a larger κ leads to a faster convergence rate of a neural network classifier. This intuition is verified for the overparameterized neural network classifier trained by square loss and ℓ_2 regularization. See Section 3 for details.

The key quantity of interest is the misclassification error, i.e., 0-1 loss. In the population level, the 0-1 loss can be written as

$$\begin{aligned} L(f) =& \mathbb{E}_{(X,Y)\sim P}\mathbb{I}\{\text{sign}(f(X)) \neq Y\} \\ =& \mathbb{E}_{X\sim P_X}[(1 - \eta(X))\mathbb{I}\{f(X) \ge 0\} + \eta(X)\mathbb{I}\{f(X) < 0\}]. \end{aligned} \tag{2.1}$$

One optimal minimizer of $L(f)$ is $2\eta - 1$, and we define $L^* = L(2\eta - 1)$. In fact, as long as the classifier has the same sign with $2\eta(\boldsymbol{x}) - 1$ for all $\boldsymbol{x} \in \Omega$, it is a optimal minimizer and achieves the minimum 0-1 loss.

Neural network setup We mainly focus on the one-hidden-layer ReLU neural network family $\mathcal{F}$ with m nodes in the hidden layer, denoted by $f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}) = m^{-1/2} \sum_{r=1}^m a_r \sigma(\boldsymbol{W}_r^\top \boldsymbol{x})$, where $\boldsymbol{x} \in \Omega$, $\boldsymbol{W} = (\boldsymbol{W}_1, \cdots, \boldsymbol{W}_m) \in \mathbb{R}^{d\times m}$ is the weight matrix in the hidden layer, $\boldsymbol{a} = (a_1, \cdots, a_m)^\top \in \mathbb{R}^m$ is the weight vector in the output layer, $\sigma(z) = \max\{0, z\}$ is the rectified linear unit (ReLU). The initial values of the weights are independently generated from $\boldsymbol{W}_r(0) \sim N(\boldsymbol{0}, \xi^2 \boldsymbol{I}_m), a_r \sim \text{unif}\{-1, 1\}, \forall r \in [m]$. Based on the observations $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, the goal of training a neural network is to find a solution to

$$\min_{\boldsymbol{W}} \sum_{i=1}^n l(f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i), y_i) + \mu\mathcal{R}(\boldsymbol{W}, \boldsymbol{a}), \tag{2.2}$$

where l is the loss function, $\mathcal{R}$ is the regularization, and $\mu \ge 0$ is the regularization parameter. Note in Equation 2.2 that we only consider training the weights $\boldsymbol{W}$. This is because $a\cdot\sigma(z) = \text{sign}(a)\cdot\sigma(|a|z)$,

which allows us to reparametrize the network to have all a_i's to be either 1 or -1. In this work, we consider square loss associated with ℓ_2 regularization, i.e., $l(f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i), y_i) = (f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i) - y_i)^2$ and $\mathcal{R}(\boldsymbol{W}, \boldsymbol{a}) = \|\boldsymbol{W}\|_2^2$.

A popular way to train the neural network is via gradient based methods. It has been shown that the training process of DNNs can be characterized by the neural tangent kernel (NTK) [37]. As is usually assumed in the NTK literature [38, 29, 39, 40], we consider data on the unit sphere $\mathbb{S}^{d-1}$, i.e., $\|\boldsymbol{x}_i\|_2 = 1, \forall i \in [n]$, and the neural network is highly overparameterized ($m \gg n$) and trained by gradient descent (GD). For details about NTK and GD in one-hidden-layer ReLU neural networks, we refer to Appendix A. In the rest of this work, $f_{\boldsymbol{W}(k),\boldsymbol{a}}$ denotes the GD-trained neural network classifier under square loss associated with ℓ_2 regularization, where k is the iteration number satisfying Assumption D.1 and $\boldsymbol{W}(k)$ is the weight matrix after k-th iteration.

3 Theoretical results

In this section, we present our main theoretical results, which consist of three parts: generalization error, robustness, and calibration error. Throughout the analysis, we make the following assumptions on the data and the estimation model. Due to the page limit, we move the technical specification of the assumptions to Appendix D, with detailed discussions.

Data assumptions We assume the ground-truth $\eta(\boldsymbol{x})$ to be well-behaved (Assumption D.2). Optionally, the marginal density X is assumed to be upper bounded (Assumptions D.4) or both upper and lower bounded (Assumptions D.5). Assumptions D.4 and D.5 are standard in classical analysis of classification in statistics literature [21, 22], which covers a large class of density functions.

Model assumptions We require the ReLU neural network to be sufficiently overparameterized (with a finite width), and imposes conditions on the learning rate and iteration number (Assumption D.1); similar settings have been adopted by [38, 40]. We also assume that the solution to (2.2) is well-behaved, i.e., the complexity of the neural network estimator generated by the GD training is controlled (Assumption D.3).

3.1 Generalization error bound

In classification, the generalization error is typically referred to as the misclassification error, which can be quantified by $L(f)$ defined in Equation 2.1. In the non-separable case, the excess risk, defined by $L(f) - L^*$, is used to evaluate the quality of a classifier f, where $L^* = L(2\eta - 1)$, which is the minimum 0-1 loss, as stated after Equation 2.1. Theorem 3.1 states that the overparameterized neural network with GD and ℓ_2 regularization can achieve a small excess risk in the non-separable case.

Theorem 3.1 (Excess risk in the non-separable case)**.** Suppose Assumptions D.1, D.2, and D.4 hold. Assume the conditional probability $\eta(\boldsymbol{x})$ satisfies Tsybakov's noise condition with component κ. Let $\mu \asymp n^{\frac{d-1}{2d-1}}$. Then

$$L(f_{\boldsymbol{W}(k),\boldsymbol{a}}) = L^* + O_{\mathbb{P}}(n^{-\frac{d(\kappa+1)}{(2d-1)(\kappa+2)}}). \tag{3.1}$$

From Theorem 3.1, we can see that as κ becomes larger, the convergence rate becomes faster, which is intuitively true. Generalization error bounds in this setting is scarce. To the best of the authors' knowledge, [29] is the closest work (the labels are randomly flipped), where the bound is in the order of $O_{\mathbb{P}}(1/\sqrt{n})$. Our bound is faster, especially with larger κ. It is known that the optimal convergence rate under Assumptions D.2 (with slightly different notion of the smoothness) and D.4 is $O_{\mathbb{P}}(n^{-\frac{d(\kappa+1)}{d\kappa+4d-2}})$ [21]. The differences between (3.1) and the optimal convergence rate is the extra $(d-1)\kappa$ in the denominator of the convergence rate in (3.1) (since $n^{-\frac{d(\kappa+1)}{(2d-1)(\kappa+2)}} = n^{-\frac{d(\kappa+1)}{(d-1)\kappa+d\kappa+4d-2}}$). If the conditional probability η has a bounded Lipschitz constant, [22] showed that the convergence rate based on the plug-in kernel estimate is $O_{\mathbb{P}}(n^{-\frac{\kappa+1}{\kappa+3+d}})$, which is slower than the rate in Equation 3.1 if d is large.

Now we turn to the separable case. Since η only takes value from $\{0, 1\}$ in the separable case, η is bounded away from 1/2. Therefore, one can trivially take $\kappa \to \infty$ in Equation 3.1 and obtain the

convergence rate $O_{\mathbb{P}}(n^{-d/(2d-1)})$. However, this rate can be significantly improved in the separable case, as stated in the following theorem.

Theorem 3.2 (Generalization error in the separable case)**.** Suppose Assumptions D.1, D.3, and D.5 hold. Let $\mu = o(1)$. There exist positive constants C_1, C_2 such that the misclassification rate is 0% with probability at least $1 - \delta - C_1 \exp(-C_2 n)$, and δ can be arbitrarily small[2] by enlarging the neural network's width.

In Theorem 3.2, the regularization parameter can take any rate that converges to zero. In particular, μ can be zero, and the corresponding classifier overfits the training data. Theorem 3.2 states that the convergence rate in the separable case is exponential, if a sufficiently wide neural network is applied. This is because the observed labels are not corrupted by noise, i.e., $\mathbb{P}(y = 1|\boldsymbol{x})$ is either one or zero. Therefore, it is easier to classify separable data, which is intuitively true.

3.2 Robustness and calibration error

If two classes are separable with positive margin, the decision boundary is not unique. Practitioners often prefer the decision boundary with large margins, which are robust against possible perturbation on input points [41, 42]. The following theorem states that the square loss trained margin can be lower bounded by a positive constant. Recall that in the separable case, $\Omega = \Omega_1 \cup \Omega_2$, where $\Omega_1 = \{\boldsymbol{x}|\eta(\boldsymbol{x}) = 1\}$ and $\Omega_2 = \{\boldsymbol{x}|\eta(\boldsymbol{x}) = 0\}$.

Theorem 3.3 (Robustness in the separable case)**.** Suppose the assumptions of Theorem 3.2 are satisfied. Let $\mu = o(1)$. Then there exist positive constants C, C_1, C_2 such that $\min_{\boldsymbol{x}\in\mathcal{D}_T, \boldsymbol{x}'\in\Omega_1\cup\Omega_2} \|\boldsymbol{x} - \boldsymbol{x}'\|_2 \geq C$, and the misclassification rate is 0% with probability at least $1 - \delta - C_1 \exp(-C_2 n)$ for all n, where $\mathcal{D}_T$ is the decision boundary, and δ is as in Theorem 3.2.

Theorem 3.3 states that the square loss trained margin is robust in the sense that the predicted label will not change in case of any noise whose l_2 norm is smaller than C. Since $\|\boldsymbol{x} - \boldsymbol{x}'\|_\infty \geq \sqrt{d}\,\|\boldsymbol{x} - \boldsymbol{x}'\|_2$, Theorem 3.3 also indicates l_∞ robustness. To the best of our knowledge, similar theoretical robustness guarantee has not been provided for any other loss functions. The most relevant work is [43], but their result is not on the population and doesn't apply to ReLU networks trained via GD.

In the non-separable case, $\eta(\boldsymbol{x})$ varies within (0,1) and practitioners may not only want a classifier with a small excess risk, but also want to recover the underlying conditional probability η. Therefore, square loss is naturally preferred since it treats the classification problem as a regression problem. The following theorem states that, one can recover the conditional probability η by using an overparameterized neural network with ℓ_2 regularization and GD training.

Theorem 3.4 (Calibration error)**.** Suppose Assumptions D.1-D.4 are fulfilled. Let $\mu \asymp n^{\frac{d-1}{2d-1}}$. Then

$$\left\|(f_{\boldsymbol{W}(k),\boldsymbol{a}} + 1)/2 - \eta\right\|_{L_\infty} = O_{\mathbb{P}}(n^{-1/(4d-2)}).$$

Theorem 3.4 states that the underlying conditional probability in the non-separable case can be recovered by $(f_{\boldsymbol{W}(k),\boldsymbol{a}} + 1)/2$. The form $(f_{\boldsymbol{W}(k),\boldsymbol{a}} + 1)/2$ is to account for the $\{-1, 1\}$ label coding. Under {0,1} coding, the estimator would be $f_{\boldsymbol{W}(k),\boldsymbol{a}}$ itself. The L_∞ consistency doesn't hold for cross-entropy trained neural networks, due to the form of the optimal solution $\log(\frac{\eta}{1-\eta})$ [12]. With limited capacity, the network's confidence prediction is bounded away from 0 and 1. In practice, we want to control the complexity of the neural network thus it is usually the case that $\left\|f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_\infty < C$ for some constant C. Hence, following the results in [12], direct calculation shows that such a neural network with limited complexity cannot accurately estimate $\eta(\boldsymbol{x})$ when $\eta(\boldsymbol{x}) > \frac{e^C}{1+e^C}$ or $\eta(\boldsymbol{x}) < \frac{1}{1+e^C}$, which makes the calibration error under the cross-entropy loss always bounded away from zero. However, square loss does not have such a problem.

Notice that the calibration error bound in Theorem 3.4 does not depend on the Tsybakov's noise condition, and is slower than the excess risk. This is because a small calibration error is much stronger than a small excess risk, since the former requires the conditional probability estimation to

[2]The term δ only depends on the width of the neural network. A smaller δ requires a wider neural network. If $\delta = 0$, then the number of nodes in the hidden layer is infinity.

be *uniformly* accurate, not just matching the sign of $\eta - 1/2$. To be more specific, a good estimated $\widehat{\eta}$ can always lead to a low risk plug-in classifier $\widehat{f} = 2\widehat{\eta} - 1$, but not vice versa.

Remark 2 (Technical challenge)**.** Despite the similar forms of regression and classification using square loss, most of the regression analysis techniques cannot be directly applied to the classification problem, even if the supports of two classes are non-separable. Moreover, it is clear that classification problems in the separable case are completely different with regression problems.

3.3 Transition from separable to non-separable

The general non-separable case and the special separable case can be connected via Gaussian noise injection. In practice, data augmentation is an effective way to improve robustness and the simplest way is Gaussian noise injection [44]. In this section, we only consider it as an auxiliary tool for theoretical analysis purpose and not for actual robust training. Injecting Gaussian noise amounts to convoluting a Gaussian distribution $N(0, \upsilon^2 \boldsymbol{I}_d)$ to the marginal distribution P_X, which enlarges both Ω_1 and Ω_2 to $\mathbb{R}^d$ and a unique decision boundary $\mathcal{D}_\upsilon$ can be induced. Correspondingly, the "noisy" conditional probability, denoted as $\widetilde{\eta}_\upsilon$, is also smoothed to be continuous on $\mathbb{R}^d$. As $\upsilon \to 0$, $\|\widetilde{\eta}_\upsilon - \eta\|_\infty \to 0$ on Ω_1 and Ω_2 and the limiting $\widetilde{\eta}_0$ is a piecewise constant function with discontinuity at the induced decision boundary.

Lemma 3.5 (Tsybakov's noise condition under Gaussian noises)**.** Let the margin be $2\gamma > 0$, the noise be $N(0, \upsilon^2 \boldsymbol{I}_d)$. Then there exist some constants $T, C > 0$ such that

$$P_X(|2\widetilde{\eta}_\upsilon(X) - 1| < t) \leq (C\upsilon^2/\gamma)\exp(-\gamma^2/(2\upsilon^2))t, \forall t \in (0, T).$$

Theorem 3.6 (Exponential convergence rate)**.** Suppose the classes are separable with margin $2\gamma > 0$. No matter how complicated $\Omega_1 \cup \Omega_2$ are, the excess risk of the over parameterized neural network classifier satisfying Assumptions D.1 and D.4 has the rate $O_{\mathbb{P}}(e^{-n\gamma/7})$.

Although Theorem 3.6 is similar to that in [45], the techniques are different. Our analysis is directly from the general non-separable case (Theorem 3.1), by the addition of auxiliary noises (Lemma 3.5). The proof of Theorem 3.6 involves taking the auxiliary noise to zero, e.g., $\upsilon = \upsilon_n \asymp 1/\sqrt{n}$. The exponential convergence rate is a direct outcome of Lemma 3.5 and Theorem 3.1. Note that our exponential convergence rate is much faster than existing ones under the similar separable setting [28, 30, 31], which are all polynomial with n, e.g., $O_{\mathbb{P}}(1/\sqrt{n})$.

Remark 3 (Extension on NTK)**.** Although our analysis only concerns overparameterized one-hidden-layer ReLU neural networks, it can potentially apply to other types of neural networks. Recently, it has been shown that overparameterized multi-layer networks correspond to the Laplace kernel [46, 47]. As long as the trained neural networks can approximate the classifier induced by the NTK, our results can be naturally extended.

Remark 4. Theorems 3.4 and 3.6 share the same gist: the overparameterized neural network classifiers can have exponential convergence rate when data are separable with positive margin. The result of Theorem 3.6 is weaker than that of Theorem 3.4, but with milder conditions. Technically, Theorem 3.6 also bridges the non-separable case and separable case through auxiliary noise injection.

3.4 Multiclass Classification

In binary classification, the labels are usually encoded as -1 and 1. When there are $K > 2$ classes, the default label coding is one-hot. However, it is empirically observed that this vanilla square loss struggles when the number of classes are large, for which scaling tricks have been proposed [4, 7]. Another popular coding scheme is the simplex coding [11], which takes maximally separated K points on the sphere as label features. When $K = 2$, this reduces to the typical $-1, 1$ coding. Many advantages of the simplex coding have been discussed, including its relationship with cross-entropy loss and supervised contrastive learning [10, 32, 17, 48]. Even though the choice of label coding may have significant impact on optimization, it does not significantly affect our analysis of square loss from a theoretical perspective. In this work, we adopt the simplex coding[3].

Given the label coding, one can easily generalize the theoretical development in Section 3 by employing the following objective function

[3]More discussion and empirical comparison about the coding choices can be found in Appendix G.2

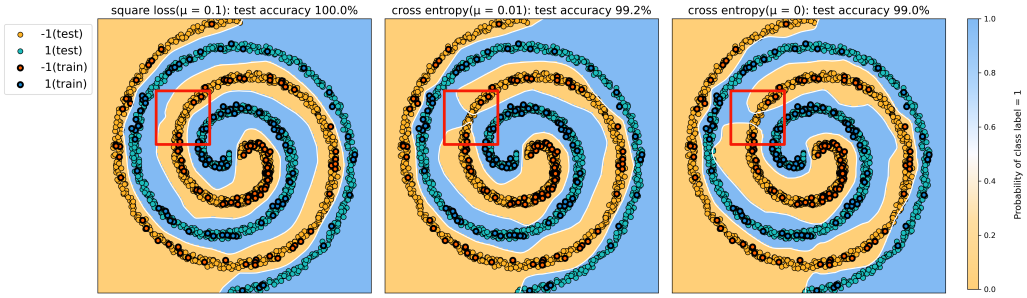

Figure 1: Test misclassification rates and decision boundaries predicted by: SL-ONN + ℓ_2 (Left); CE-ONN + ℓ_2 (Center); CE-ONN (Right) in the separable case.

$$\min_{\boldsymbol{W}} \sum_{j=1}^{K} \sum_{i=1}^{n} (f_{j,\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i) - y_{i,j})^2 + \mu \left\| \boldsymbol{W} \right\|_2^2,$$

where $f_{\boldsymbol{W},\boldsymbol{a}} : \Omega \mapsto \mathbb{R}^K$, and $\boldsymbol{y}_i = (y_{i,1}, ..., y_{i,K})^\top$ is the label of i-th observation.

Next proposition states a relationship between the simplex coding and the conditional probability, which is a generalization of the binary classification case.

Proposition 3.7 (Conditional probability)**.** Let $f^* : \Omega \to \mathbb{R}^K$ minimize the mean square error $\mathbb{E}_X(f^*(X) - \boldsymbol{v}_y)^2$, where $\boldsymbol{v}_y$ is the simplex coding vector of label y. Then

$$\eta_k(\boldsymbol{x}) := \mathbb{P}(y = k|\boldsymbol{x}) = \left((K-1) f^*(\boldsymbol{x})^\top \boldsymbol{v}_k + 1\right)/K. \tag{3.2}$$

Based on the relationship established by Proposition 3.7, one can show that the calibration error can be small in the multi-class classification case. Unlike the softmax function when using cross entropy, the estimated conditional probability using square loss is not guaranteed to be within 0 and 1. This will cause issues for adversarial attacks; see more discussions in Appendix G.2.

4 Numerical experiments

Although our theoretical results are for overparameterized neural network in the NTK regime, we expect our conclusions to generalize to practical network architectures. The focus of this section is not on improving the state-of-the-art performance for deep classifiers, but to illustrate the difference between cross-entropy and square loss. We provide experiment results on both synthetic and real data, to support our theoretical findings and illustrate the practical benefits of square loss in training overparameterized DNN classifiers. Compared with cross-entropy, the square loss has comparable generalization performance, but with stronger robustness and smaller calibration error.

4.1 Synthetic data

We consider the square loss based, cross-entropy based overparameterized neural networks (ONN) with ℓ_2 regularization, and the cross-entropy based ONN without ℓ_2 regularization, denoted as SL-ONN + ℓ_2, CE-ONN + ℓ_2, and CE-ONN respectively. For the separable case, we consider two separated classes with spiral curve like supports. Figure 1 shows one instance of the test misclassification rate and decision boundaries attained by three methods, where we can see that SL-ONN + ℓ_2 has a smaller test misclassification rate and a much smoother decision boundary. In particular, in the red region, where the training data are sparse, SL-ONN + ℓ_2 fits the correct data distribution best. For the non-separable case, we consider the calibration performance of SL-ONN + ℓ_2 and CE-ONN + ℓ_2, where the classifiers are denoted by $\widehat{f}_{l2}$ and $\widehat{f}_{ce}$, respectively. The results presented in Figure G.9 in the Appendix shows that $\widehat{f}_{l2}$ has the smaller mean and standard deviation than $\widehat{f}_{ce}$. More implementation details are in Appendix G.1.

4.2 Real data

To make a fair comparison, we adopt popular architectures, ResNet [49] and Wide ResNet [50] and evaluate them on the CIFAR image classification datasets [51], with only the training loss function changed, from cross-entropy (CE) to square loss with simplex coding (SL). Further, we don't employ any large scale hyper-parameter tuning and all the parameters are kept as default except for the learning rate (lr) and batch size (bs), where we are choosing from the better of (lr=0.01, bs=32) and (lr=0.1, bs=128). Each experiment setting is replicated 5 times and we report the average performance followed by its standard deviation in the parenthesis. (lr=0.01, bs=32) works better for the most cases except for square loss trained WRN-16-10 on CIFAR-100. More experiment details and additional results can be found in Appendix G.2.

Generalization In both CIFAR-10 and CIFAR-100, the performance of cross-entropy and square loss with simplex coding are quite comparable, as observed in [4]. Cross-entropy tends to perform slightly better for ResNet, especially on CIFAR-100 with an advantage of less than 1%. There is a more significant gap with Wide ResNet where square loss outperforms cross-entropy by more than 1% on both CIFAR-10 and CIFAR-100. The details can be found in Table 1.

Table 1: Test accuracy on CIFAR datasets. Average accuracy larger than 0 but less than 0.1 is denoted as 0^* without standard deviation.

Dataset	Network	Loss	Clean acc %	*PGD-100 (l_∞-strength)*			*AutoAttack (l_∞-strength)*		
				$2/255$	$4/255$	$8/255$	$2/255$	$4/255$	$8/255$
CIFAR-10	ResNet-18	CE	**95.15 (0.11)**	8.81 (1.61)	0.65 (0.24)	0	2.74 (0.09)	0	0
		SL	95.04 (0.07)	**30.53 (0.92)**	**6.64 (0.67)**	**0.86 (0.24)**	**4.10 (0.50)**	$\mathbf{0}^*$	0
	WRN-16-10	CE	93.94 (0.16)	1.04 (0.10)	0	0	0.33 (0.06)	0	0
		SL	**95.02 (0.11)**	**37.47 (0.61)**	**23.16 (1.28)**	**7.88 (0.72)**	**5.37 (0.50)**	$\mathbf{0}^*$	0
CIFAR-100	ResNet-50	CE	**79.82 (0.14)**	2.31 (0.07)	0^*	0	0.99 (0.10)	0^*	0
		SL	78.91 (0.14)	**13.76 (1.30)**	**4.63 (1.20)**	**1.21 (0.80)**	**3.67 (0.60)**	**0.16 (0.05)**	0
	WRN-16-10	CE	77.89 (0.21)	0.83 (0.07)	0^*	0	0.42 (0.07)	0	0
		SL	**79.65 (0.15)**	**6.48 (0.40)**	**0.42 (0.04)**	$\mathbf{0}^*$	**2.73 (0.20)**	$\mathbf{0}^*$	0

Adversarial robustness Naturally trained deep classifiers are found to be adversarially vulnerable and adversarial attacks provide a powerful tool to evaluate classification robustness. For our experiment, we consider the black-box Gaussian noise attack, the classic white-box PGD attack [52] and the state-of-the-art AutoAttack [53], with attack strength level 2/255, 4/255, 8/255 in l_∞ norm. AutoAttack contains both white-box and black-box attacks and offers a more comprehensive evaluation of adversarial robustness. The Gaussian noises results are presented in Table G.3 in the Appendix. At different noise levels, square loss consistently outperforms cross-entropy, especially for WRN-16-10, with around 2-4% accuracy improvement. More details can be found in Appendix G.2. The PGD and AutoAttack results are reported in Table 1. Even though classifiers trained with square loss is far away from adversarially robust, it consistently gives significantly higher adversarial accuracy. The same margin can be carried over to standard adversarial training as well. Table 2 lists results from standard PGD adversarial training with CE and SL. By substituting cross-entropy loss to square loss, the robust accuracy increased around 3% while maintaining higher clean accuracy.

One thing to notice is that when constructing white-box attacks, square loss will not work well since it doesn't directly reflect the classification accuracy. More specifically, for a correctly classified image $(\boldsymbol{x}, y)$, maximizing the square loss may result in linear scaling of the classifier $f(\boldsymbol{x})$, which doesn't change the predicted class (see Appendix G.2 for more discussion). To this end, we consider a special attack for classifiers trained by square loss by maximizing the cosine similarity between $f(\boldsymbol{x})$ and $\boldsymbol{v}_y$. We call this angle attack and also utilize it for the PGD adversarial training paired with square loss in Table 2. In our experiments, this special attack rarely outperforms the standard PGD with cross-entropy and the reported PGD accuracy are from the latter settings. This property of square loss may be an advantage in defending adversarial attacks.

Model calibration The predicted class probabilities for square loss can be obtained from Equation 3.2. Expected calibration error (ECE) measures the absolute difference between predicted confidence and actual accuracy. Deep classifiers are usually over-confident [54]. Using ResNet as an example, we report the typical reliability diagram in Figure 2. On CIFAR-10 with ResNet-18,

Table 2: Performance on CIFAR-10 dataset for ResNet-18 under standard PGD adversarial training.

Loss	Acc (%)	PGD steps	Strength(l_∞)	AutoAttack
CE	86.87	3	8/255	37.08
	84.50	7	8/255	41.88
SL	**87.31**	3	8/255	**40.46**
	84.52	7	8/255	**44.76**

the average ECE for cross-entropy is 0.028 (0.002) while that for square loss is 0.0097 (0.001). On CIFAR-100 with ResNet-50, the average ECE for cross-entropy is 0.094 (0.005) while that for square loss is 0.068 (0.005). Square loss results are much more calibrated with significantly smaller ECE.

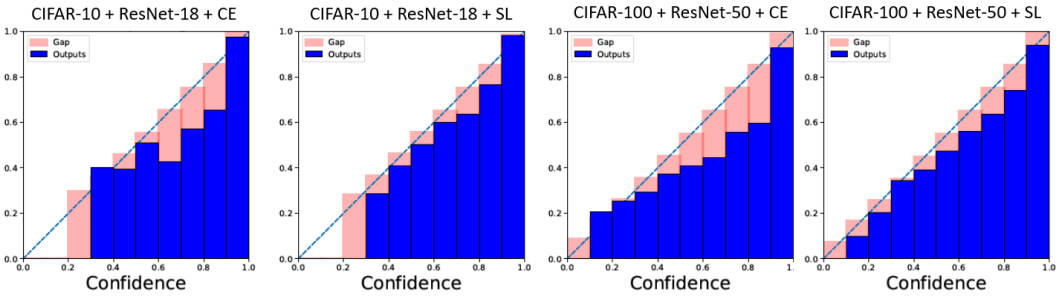

Figure 2: Reliability diagrams of ResNet-18 on CIFAR-10 and ResNet-50 on CIFAR-100. Square loss trained models behave more well-calibrated while cross-entropy trained ones tend to be visibly more over-confident.

5 Conclusions

Classification problems are ubiquitous in deep learning. As a fundamental problem, any progress in classification can potentially benefit numerous relevant tasks. Despite its lack of popularity in practice, square loss has many advantages that can be easily overlooked. Through both theoretical analysis and empirical studies, we identify several ideal properties of using square loss in training neural network classifiers, including provable fast convergence rates, strong robustness, and small calibration error. We encourage readers to try square loss in your own application scenarios.

6 Acknowledgements

We would like to thank reviewers for their valuable comments on the manuscript. Wenjia Wang and Jun Wang were supported by Foshan HKUST Projects FSUST20-FYTRI03B. We gratefully acknowledge the support of MindSpore for this research.

References

[1] Yaodong Yu, Kwan Ho Ryan Chan, Chong You, Chaobing Song, and Yi Ma. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. *Advances in Neural Information Processing Systems*, 33, 2020.

[2] Tianyu Pang, Kun Xu, Yinpeng Dong, Chao Du, Ning Chen, and Jun Zhu. Rethinking softmax cross-entropy loss for adversarial robustness. *arXiv preprint arXiv:1905.10626*, 2019.

[3] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[4] Like Hui and Mikhail Belkin. Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. *arXiv preprint arXiv:2006.07322*, 2020.

[5] Joern-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge. Excessive invariance causes adversarial vulnerability. In *International Conference on Learning Representations*, 2018.

[6] Kamil Nar, Orhan Ocal, S Shankar Sastry, and Kannan Ramchandran. Cross-entropy loss and low-rank features have responsibility for adversarial examples. *arXiv preprint arXiv:1901.08360*, 2019.

[7] Ahmet Demirkaya, Jiasi Chen, and Samet Oymak. Exploring the role of loss functions in multiclass classification. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–5. IEEE, 2020.

[8] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34:1256–1272, 2021.

[9] Simon Kornblith, Honglak Lee, Ting Chen, and Mohammad Norouzi. Demystifying loss functions for classification. 2020.

[10] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117 (40):24652–24663, 2020.

[11] Youssef Mroueh, Tomaso Poggio, Lorenzo Rosasco, and Jean-Jacques Slotine. Multiclass learning with simplex coding. *arXiv preprint arXiv:1209.1360*, 2012.

[12] Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, pages 56–85, 2004.

[13] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[15] Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. A statistical perspective on distillation. In *International Conference on Machine Learning*, pages 7632–7642. PMLR, 2021.

[16] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[17] Florian Graf, Christoph Hofer, Marc Niethammer, and Roland Kwitt. Dissecting supervised constrastive learning. In *International Conference on Machine Learning*, pages 3821–3830. PMLR, 2021.

[18] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2020.

[19] Enno Mammen and Alexandre B Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27(6):1808–1829, 1999.

[20] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[21] Jean-Yves Audibert and Alexandre B Tsybakov. Fast learning rates for plug-in classifiers. *The Annals of Statistics*, 35(2):608–633, 2007.

[22] Michael Kohler and Adam Krzyzak. On the rate of convergence of local averaging plug-in classification rules under a margin condition. *IEEE Transactions on Information Theory*, 53(5): 1735–1742, 2007.

[23] Ingo Steinwart, Clint Scovel, et al. Fast rates for support vector machines using Gaussian kernels. *The Annals of Statistics*, 35(2):575–607, 2007.

[24] Tianyang Hu, Zuofeng Shang, and Guang Cheng. Sharp rate of convergence for deep neural network classifiers under the teacher-student setting. *arXiv preprint arXiv:2001.06892*, 2020.

[25] Tianyang Hu, Ruiqi Liu, Zuofeng Shang, and Guang Cheng. Minimax optimal deep neural network classifiers under smooth decision boundary. *arXiv preprint arXiv:2207.01602*, 2022.

[26] Yongdai Kim, Ilsang Ohn, and Dongha Kim. Fast convergence rates of deep neural networks for classification. *arXiv preprint arXiv:1812.03599*, 2018.

[27] Michael Kohler and Sophie Langer. Statistical theory for image classification using deep convolutional neural networks with cross-entropy loss. *arXiv preprint arXiv:2011.13602*, 2020.

[28] Ziwei Ji and Matus Telgarsky. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow ReLU networks. *arXiv preprint arXiv:1909.12292*, 2019.

[29] Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *International Conference on Learning Representations*, 2020.

[30] Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in Neural Information Processing Systems*, 32:10836–10846, 2019.

[31] Yuan Cao and Quanquan Gu. Generalization error bounds of gradient descent for learning over-parameterized deep relu networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3349–3356, 2020.

[32] XY Han, Vardan Papyan, and David L Donoho. Neural collapse under mse loss: Proximity to and dynamics on the central path. *arXiv preprint arXiv:2106.02073*, 2021.

[33] Vidya Muthukumar, Adhyyan Narang, Vignesh Subramanian, Mikhail Belkin, Daniel Hsu, and Anant Sahai. Classification vs regression in overparameterized regimes: Does the loss function matter? *arXiv preprint arXiv:2005.08054*, 2020.

[34] Tomaso Poggio and Qianli Liao. Generalization in deep network classifiers trained with the square loss. Technical report, CBMM Memo No, 2019.

[35] Ben Adlam and Jeffrey Pennington. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *International Conference on Machine Learning*, pages 74–84. PMLR, 2020.

[36] Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75(4):667–766, 2022.

[37] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pages 8571–8580, 2018.

[38] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019.

[39] Alberto Bietti and Julien Mairal. On the inductive bias of neural tangent kernels. *arXiv preprint arXiv:1905.12173*, 2019.

[40] Tianyang Hu, Wenjia Wang, Cong Lin, and Guang Cheng. Regularization matters: A non-parametric perspective on overparametrized neural network. In *International Conference on Artificial Intelligence and Statistics*, pages 829–837. PMLR, 2021.

[41] Gamaleldin F Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. *arXiv preprint arXiv:1803.05598*, 2018.

[42] Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. *arXiv preprint arXiv:1812.02637*, 2018.

[43] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

[44] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–597, 2019.

[45] Atsushi Nitanda and Taiji Suzuki. Optimal rates for averaged stochastic gradient descent under neural tangent kernel regime. *arXiv preprint arXiv:2006.12297*, 2020.

[46] Amnon Geifman, Abhay Yadav, Yoni Kasten, Meirav Galun, David Jacobs, and Ronen Basri. On the similarity between the Laplace and neural tangent kernels. *arXiv preprint arXiv:2007.01580*, 2020.

[47] Lin Chen and Sheng Xu. Deep neural tangent kernel and Laplace kernel have the same RKHS. *arXiv preprint arXiv:2009.10683*, 2020.

[48] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences*, 118(43), 2021.

[49] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[50] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[51] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[52] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[53] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216. PMLR, 2020.

[54] Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön. Evaluating model calibration in classification. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3459–3467. PMLR, 2019.

[55] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[56] David Eric Edmunds and Hans Triebel. *Function Spaces, Entropy Numbers, Differential Operators*, volume 120. Cambridge University Press, 2008.

[57] Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. *The Annals of Statistics*, 48(4):1875–1897, 2020.

[58] Max H Farrell, Tengyuan Liang, and Sanjog Misra. Deep neural networks for estimation and inference. *Econometrica*, 89(1):181–213, 2021.

[59] Atsushi Nitanda, Geoffrey Chinot, and Taiji Suzuki. Gradient descent can learn less over-parameterized two-layer neural networks on classification problems. *arXiv preprint arXiv:1905.09870*, 2019.

[60] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pages 8157–8166, 2018.

[61] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, 1950.

[62] Colin Bennett and Robert C Sharpley. *Interpolation of Operators*. Academic press, 1988.

[63] Holger Wendland. *Scattered Data Approximation*. Cambridge University Press, 2004.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 3.
   (b) Did you describe the limitations of your work? [Yes] Our theoretical results apply to shallow overparameterized neural networks in the NTK regime. However, our findings can still hold beyond the assumptions as we conducted extensive experiments on practical neural networks in Section 4.
   (c) Did you discuss any potential negative societal impacts of your work? [No] We do not foresee any potential negative societal impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Appendix D.
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendix E.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The real data we use is the publicly available. The code is proprietary. However, we have provided implementation details of our experiments in Appendix G.1 and Appendix G.2
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] They are all discussed in detail either in the main submission or in appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Fig. G.3 and Fig. G.8
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] Our experiments are not computationally intensive.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.
   (b) Did you mention the license of the assets? [Yes] See Section 4.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A] Does not apply
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] Does not apply
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] Does not apply
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] Does not apply

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] Does not apply

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] Does not apply

Appendix

A Gradient Descent and Neural Tangent Kernel

Gradient Descent Since we consider the square loss and ℓ_2 regularization, the optimization problem in Equation 2.2 becomes

$$\min_{\boldsymbol{W}} \sum_{i=1}^{n} (f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i) - y_i)^2 + \mu \|\boldsymbol{W}\|_2^2 . \tag{A.1}$$

We consider the GD training of Equation A.1. Let

$$\Phi(\boldsymbol{W}) = \sum_{i=1}^{n} (f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i) - y_i)^2 + \mu \|\boldsymbol{W}\|_2^2$$

be the objective function in Equation A.1. The gradient of Φ with respect to $\boldsymbol{w}_r$ can be written as [38]

$$\frac{\partial \Phi(\boldsymbol{W})}{\partial \boldsymbol{w}_r} = \frac{2}{\sqrt{m}} a_r \sum_{i=1}^{n} (u_i - y_i) \mathbb{I}_{r,i} \boldsymbol{x}_i + 2\mu \boldsymbol{w}_r, \quad r \in [m],$$

where $u_i = f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i)$ and $\mathbb{I}_{r,i} = \mathbb{I}\{\boldsymbol{w}_r^\top \boldsymbol{x}_i \geq 0\}$. Then the GD update rule is

$$\boldsymbol{w}_r(k+1) = \boldsymbol{w}_r(k) - \zeta \frac{\partial \Phi(\boldsymbol{W})}{\partial \boldsymbol{w}_r} \Big|_{\boldsymbol{W} = \boldsymbol{W}(k)},$$

where $\boldsymbol{W}(k)$ is the weight matrix at iteration k, and ζ is the learning rate. Define $\mathbb{I}_{r,i}(k) = \mathbb{I}\{\boldsymbol{w}_r(k)^\top \boldsymbol{x}_i \geq 0\}$, $\boldsymbol{Z}(k) \in \mathbb{R}^{md \times n}$ as

$$\boldsymbol{Z}(k) = \frac{1}{\sqrt{m}} \begin{pmatrix} a_1 \mathbb{I}_{1,1}(k) \boldsymbol{x}_1 & \dots & a_1 \mathbb{I}_{1,n}(k) \boldsymbol{x}_n \\ \vdots & \ddots & \vdots \\ a_m \mathbb{I}_{m,1}(k) \boldsymbol{x}_1 & \dots & a_m \mathbb{I}_{m,n}(k) \boldsymbol{x}_n \end{pmatrix},$$

$\boldsymbol{H}(k) = \boldsymbol{Z}(k)^\top \boldsymbol{Z}(k)$, and $\boldsymbol{u}(k) = (\boldsymbol{u}_1(k), ..., \boldsymbol{u}_n(k))^\top$ with $\boldsymbol{u}_i(k) = f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}_i)$. Then the GD update rule with respect to $\boldsymbol{W}$ can be written as

$$\mathrm{vec}(\boldsymbol{W}(k+1)) = \mathrm{vec}(\boldsymbol{W}(k)) - 2\zeta \big(\boldsymbol{Z}(k)(\boldsymbol{u}(k) - \boldsymbol{y}) + \mu \mathrm{vec}(\boldsymbol{W}(k)) \big), \tag{A.2}$$

where $\mathrm{vec}(\boldsymbol{W}) = (\boldsymbol{w}_1^\top, \cdots, \boldsymbol{w}_m^\top)^\top \in \mathbb{R}^{md \times 1}$ is the vectorized weight matrix and $\boldsymbol{y} = (y_1, ..., y_n)^\top$.

Neural Tangent Kernel (NTK) It has been shown that the following NTK

$$h(\boldsymbol{s}, \boldsymbol{t}) = \mathbb{E}_{\boldsymbol{w} \sim N(0, \boldsymbol{I}_d)} \left(\boldsymbol{s}^\top \boldsymbol{t} \, \mathbb{I}\{\boldsymbol{w}^\top \boldsymbol{s} \geq 0, \boldsymbol{w}^\top \boldsymbol{t} \geq 0\} \right) = \frac{\boldsymbol{s}^\top \boldsymbol{t} (\pi - \arccos(\boldsymbol{s}^\top \boldsymbol{t}))}{2\pi} \tag{A.3}$$

plays an important role in the study of one-hidden-layer ReLU neural networks, where $\boldsymbol{s}, \boldsymbol{t}$ are d-dimensional vectors [55, 40]. Since h is positive definite on the unit sphere $\mathbb{S}^{d-1}$ [39], by Mercer's theorem, it possesses a Mercer decomposition as $h(\boldsymbol{s}, \boldsymbol{t}) = \sum_{j=0}^{\infty} \lambda_j \varphi_j(\boldsymbol{s}) \varphi_j(\boldsymbol{t})$, where $\lambda_1 \geq \lambda_2 \geq ... \geq 0$ are the eigenvalues, and $\{\varphi_j\}_{j=1}^{\infty}$ is an orthonormal basis. The asymptotic behavior of the eigenvalues is described in the following lemma.

Lemma A.1 (Lemma 3.1 of [40])**.** Let λ_j be the eigenvalues of NTK h defined above. Then we have $\lambda_j \asymp j^{-\frac{d}{d-1}}$.

Let $\mathcal{N}$ denote the reproducing kernel Hilbert space (RKHS) generated by h on $\mathbb{S}^{d-1}$, equipped with norm $\|\cdot\|_{\mathcal{N}}$. As a corollary of Lemma A.1, it can be shown that the (L_∞) entropy number of a unit ball in $\mathcal{N}$, denoted by $\mathcal{N}(1)$, can be controlled. The relationship between the eigenvalues and entropy numbers has been well studied; see [56].

Lemma A.2. The entropy number of $\mathcal{N}(1)$, denoted by $H(\mathcal{N}(1), \delta, \|\cdot\|_{L_\infty})$, is bounded by $H(\mathcal{N}(1), \delta, \|\cdot\|_{L_\infty}) \lesssim \delta^{-2(d-1)/d}$.

There are extensive works studying the generalization error bounds under NTK regime. For regression, [45, 40] show the optimal convergence rates when using overparameterized one-hidden-layer neural networks, where the square loss is used. [38] provides generalization error bounds and provable learning scenarios for noiseless data. In the NTK regime, the neural network as a regressor is linked with the nonparametric regression via NTK. There are also other works studying the generalization performance of the neural network as a nonparametric regressor, out of the NTK regime; see [57, 58]. For classification, most of the existing results are established based on the separable data; see [28, 30, 59] and references therein. In particular, [29] consider classification with noisy labels (labels are randomly flipped) and propose to use the square loss with ℓ_2 regularization. Besides the generalization error bounds, another important research direction is to bridge the gap between NTK and finite-width overparameterized neural networks via GD training; see [55, 38, 60, 40], among others.

B Overview of Reproducing Kernel Hilbert Space

We provide here a brief overview of reproducing kernel Hilbert space (RKHS).

Definition B.1 (Positive Definite Kernel)**.** A function $k : \Omega \times \Omega \mapsto \mathbb{R}$ is said to be a *positive definite kernel*, if $k(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) = k(\widetilde{\boldsymbol{x}}, \boldsymbol{x})$ for all $\boldsymbol{x}, \widetilde{\boldsymbol{x}} \in \Omega$, and

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \beta_i \beta_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) > 0,$$

for all $n \in \mathbb{N}$, $\beta_1, \ldots, \beta_n \in \mathbb{R}$ such that at least one $\beta_j \neq 0$, and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \Omega$.

For a positive definite kernel k, define a linear space

$$\mathcal{N}_k^0 := \left\{ \sum_{i=1}^{n} \beta_i k(\cdot, \boldsymbol{x}_i) : n \in \mathbb{N},\ \beta_1, \ldots, \beta_n \in \mathbb{R},\ \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \Omega \right\},$$

and equip this space with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{N}_k^0}$ by

$$\left\langle \sum_{i=1}^{n} \beta_i k(\cdot, \boldsymbol{x}_i), \sum_{j=1}^{\widetilde{n}} \widetilde{\beta}_j k(\cdot, \widetilde{\boldsymbol{x}}_i) \right\rangle_{\mathcal{N}_k^0} := \sum_{i=1}^{n}\sum_{j=1}^{\widetilde{n}} \beta_i \widetilde{\beta}_j k(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j).$$

The norm of $g \in \mathcal{N}_k^0$ is defined by $\|g\|_{\mathcal{N}_k^0}^2 := \langle g, g \rangle_{\mathcal{N}_k^0}$. Then the RKHS induced by k, denoted by $\mathcal{N}_k(\Omega)$, is defined as the closure of $\mathcal{N}_k^0(\Omega)$ with respect to the norm $\|\cdot\|_{\mathcal{N}_k^0(\Omega)}$.

For a subset $\Omega_0 \subset \Omega$, define the *restriction* of $\mathcal{N}_k$ on Ω_0 as

$$\mathcal{N}_k(\Omega_0) := \{ g : \Omega_0 \mapsto \mathbb{R} : g = h|_{\Omega_0} \text{ for some } h \in \mathcal{N}_k \},$$

where $g = h|_{\Omega_0}$ means $g(\boldsymbol{x}) = h(\boldsymbol{x})$ for all $\boldsymbol{x} \in \Omega_0$. We equip $\mathcal{N}_k(\Omega_0)$ with norm

$$\|g\|_{\mathcal{N}_k(\Omega_0)} := \inf_{\{h \in \mathcal{N}_k : h|_{\Omega_0} = g\}} \|h\|_{\mathcal{N}_k}.$$

Then, $\mathcal{N}_k(\Omega_0)$ is a RKHS with norm $\|\cdot\|_{\mathcal{N}_k(\Omega_0)}$ (see Aronszajn 61, page 351).

C Simplex Coordinates

In simplex label coding, the one-hot labels are replaced by the simplex vertices of a $(K-1)$-simplex. The vertices of a regular $(K-1)$-simplex centered on the origin can be written as:

$$\boldsymbol{v}_0 = \frac{1}{\sqrt{2K}} \cdot (1, \ldots, 1)$$

and for $1 \leq i \leq K-1$,

$$\boldsymbol{v}_i = \frac{1}{\sqrt{2}} \boldsymbol{e}_i - \frac{1}{(K-1)\sqrt{2}} \left(1 + \frac{1}{\sqrt{K}}\right) \cdot (1, \ldots, 1).$$

The pairwise angle between vertices is $\arccos(-1/(K-1))$ and as $K \to \infty$, the angle converges to $90°$.

The vertices of a $(K-1)$-simplex can be viewed as maximally separated K points on a sphere. In theory, the radius of the sphere doesn't matter but in practice, we recommend scaling it for larger number of classes, e.g., radius = K for K-class classification. We find that such scaling empirically outperforms the default radius 1 in our experiments. More details can be found in Appendix G.2.

D Assumptions

In this work, we impose the following assumptions. In the rest of the Appendix, we use $\mathrm{poly}(t_1, t_2, \ldots)$ to denote some polynomial function with arguments $t_1, t_2, \ldots$.

Assumption D.1. Let $\lambda_{\min}(\boldsymbol{H}^\infty)$ be the minimum eigenvalue of the symmetric matrix $\boldsymbol{H}^\infty$, where $\boldsymbol{H}^\infty = (h(\boldsymbol{x}_i, \boldsymbol{x}_j))_{n\times n}$ ($\boldsymbol{H}^\infty$ is usually called the NTK matrix). Let λ_0 be the largest number such that with probability at least $1-\delta_n$, $\lambda_{\min}(\boldsymbol{H}^\infty) \geq \lambda_0$, and $\delta_n \to 0$ as n goes to infinity[4]. For sufficiently large n, the regularization parameter $\mu \asymp n^{\frac{d-1}{2d-1}}$, the learning rate $\zeta = o(n^{-\frac{3d-1}{2d-1}})$, the variance of initialization $\xi^2 = O(1)$, the number of nodes in the hidden layer $m \geq \xi^{-2}\mathrm{poly}(n, \lambda_0^{-1})$, and the iteration number k satisfies $\log\left(\mathrm{poly}_1(n, \xi, 1/\lambda_0)\right) \lesssim \zeta\mu k \lesssim \log\left(\mathrm{poly}_2(\xi, 1/n, \sqrt{m})\right)$.

Assumption D.2. The conditional probability in the non-separable case satisfies $\eta \in \mathcal{N}$.

Assumption D.3. The solution to Equation 2.2 satisfies $\left\|f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{\mathcal{N}} \leq C$, where C is a constant not depending on n.

Remark 5. Assumption D.3 can be replaced by an alternative assumption, that is, $f_{\boldsymbol{W}(k),\boldsymbol{a}}$ has a bounded Lipschitz constant, and the constant does not depend on n.

Assumption D.4. The probability density function of the marginal distribution P_X, denoted by $p(\boldsymbol{x})$, is continuous on Ω, and there exists a positive constant c_0 such that

$$p(\boldsymbol{x}) \leq c_0, \forall \boldsymbol{x} \in \Omega.$$

Assumption D.5. The probability density function of the marginal distribution $p(\boldsymbol{x})$ is continuous on Ω, and there exist positive constants $c_1 \leq c_2$ such that

$$c_1 \leq p(\boldsymbol{x}) \leq c_2, \forall \boldsymbol{x} \in \Omega.$$

Assumption D.1 is related to the neural network and GD training, where similar settings have been adopted by [38, 40]. From the results in [38, 40], the width of the neural network depends on the minimum eigenvalue of the NTK matrix $\lambda_{\min}(\boldsymbol{H}^\infty)$, where a smaller $\lambda_{\min}(\boldsymbol{H}^\infty)$ leads to a wider neural network. Therefore, it is desired that λ_0 is as large as possible. However, the consistency requires that the probability is tending to one; thus, we require $\delta_n \to 0$ as $n \to \infty$. As n becomes larger, with probability tending to one, the distance of the two nearest points in n input points converges to zero, thus making $\boldsymbol{H}^\infty$ close to a degenerate matrix, and the minimum eigenvalue of $\boldsymbol{H}^\infty$ converges to zero. Therefore, inevitably, $\lambda_0 \to 0$ (but $\lambda_{\min}(\boldsymbol{H}^\infty)$ is strictly larger than 0 for all n with probability one). The requirements of the regularization parameter, the learning rate, the variance of initialization, the number of nodes in the hidden layer and the iteration number are all the same as those in [40].

Assumption D.2 imposes conditions on the underlying true conditional probability in the *non-separable* case. This assumption basically requires that the conditional probability is within the function class generated by the GD-trained neural networks we consider (thus can be calibrated). Given that the neural networks are highly flexible, we believe that most of the functions are within the function class generated by the neural networks.

Assumption D.3 is a technical assumption, which requires that the solution to Equation 2.2 is well-behaved, i.e., the solution is within a ball in $\mathcal{N}$ with a certain radius. Roughly speaking, Assumption D.3 requires that the *complexity* of the neural network estimator generated by the GD training is controlled. Since the step size is relatively small and the iteration number is not large (only

[4]Potential dependency of λ_0 on n is suppressed for notational simplicity.

$\log(\text{poly}(n, \xi, , \lambda_0^{-1}))$, we believe it is a mild assumption. It is worth noting that although $f_{\mathbf{W}(k),\mathbf{a}}$ is close to the solution of Equation E.1 under the L_2 metric, we cannot confirm whether this closeness still holds under a stronger RKHS metric. Therefore, even if the RKHS norm of the solution of Equation E.1 is bounded, whether the RKHS norm of $f_{\mathbf{W}(k),\mathbf{a}}$ is bounded remains unknown. Thus, we make it as an assumption, and leave it as a future work. Nevertheless, we point out that Theorem 3.1 does not depend on Assumption D.3.

Assumption D.4 only requires the probability to be upper bounded from infinity, while Assumption D.5 requires the probability to be upper bounded from infinity and lower bounded away from zero on the support Ω. They are standard assumptions used in the classical analysis of classification in statistics; see [21, 22] for example. Clearly, uniform distribution satisfies Assumptions D.4 and D.5. In [21], Assumption D.4 is called mild density assumption and Assumption D.5 is called strong density assumption.

E Proofs of Main Results

This section includes the proofs of main results in the paper.

E.1 Proof of Theorem 3.1

We first introduce some lemmas that are used in the proof of Theorem 3.1.

Let $l_1(y_i, f(\boldsymbol{x}_i)) = (1 - y_i f(\boldsymbol{x}_i))^2 = (y_i - f(\boldsymbol{x}_i))^2$ be the square loss on a training point $(\boldsymbol{x}_i, y_i)$, the l_1-risk of f be $\mathcal{R}_{l_1}(f) = \mathbb{E}_{X,Y\sim P} l_1(Y, f(X))$, and $\mathcal{R}_{l_1} = \min_{f\in\mathcal{N}} \mathbb{E}_{X,Y\sim P} l_1(Y, f(X))$. Let $L_1(f, \boldsymbol{x}, y) = \mu \|f\|_{\mathcal{N}}^2 + l_1(y, f(\boldsymbol{x}))$ and the L_1-risk of f be $\mathcal{R}_{L_1}(f) = \mathbb{E}_{X,Y\sim P} L_1(f, X, Y)$. Let $f_n = \arg\min_{f\in\mathcal{N}} \mathcal{R}_{L_1}(f)$.

Lemma E.1 is (a weaker version of) Theorem 5.6 of [23], which provides a bound on the deviation between the empirical minimizer and true minimizer. Lemma E.2 is used to verify that one of the conditions of Lemma E.1 is fulfilled. Lemma E.3 shows that under certain conditions, the solution to

$$\min_{f\in\mathcal{N}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2 + \frac{\mu}{n} \|f\|_{\mathcal{N}}^2 \tag{E.1}$$

is closely related to the estimator given by the overparameterized neural networks $f_{\boldsymbol{W}(k),\boldsymbol{a}}$. Lemma E.3 can be obtained by merely repeating the proof of Theorem 5.2 of [40], since we require that the probability density function $p(\boldsymbol{x})$ of P_X is upper bounded by a positive constant by Assumption D.4. Therefore, the only difference is that we replace $\|\cdot\|_2$ (which corresponds to the uniform distribution) to the L_2 norm corresponding to the probability measure P_X; thus the proof is omitted. Note also that the second statement of Lemma E.3 corresponds to the noiseless case.

Lemma E.1. Let $Z = \Omega \times \{-1, 1\}$. Let $\mathcal{F}$ be a convex set of bounded measurable functions from Z to $\mathbb{R}$ and let $L : \mathcal{F} \times Z \to [0, \infty)$ be a convex and continuous loss function. For a probability measure P on Z, define

$$\mathcal{G} := \{L \circ f - L \circ f_{P,\mathcal{F}} : f \in \mathcal{F}\},$$

where $f_{P,\mathcal{F}}$ is a minimizer of $\mathbb{E}_{Z\sim P} L(f, Z)$. Suppose that there are constants $c \geq 0$, $0 < \alpha \leq 1$, $\delta \leq 0$ and $B > 0$ such that $\mathbb{E}_{Z\sim P} g^2 \leq c(\mathbb{E}_{Z\sim P} g)^\alpha + \delta$ and $||g||_\infty \leq B$ for all $g \in \mathcal{G}$. Furthermore, assume that $\mathcal{G}$ is separable with respect to $\|\cdot\|_\infty$ and that there are constants $a \geq 1$ and $0 < \alpha < 2$ with

$$\sup_{T\in Z^n} H(B^{-1}\mathcal{G}, \epsilon, \|\cdot\|_{L_2(T)}) \leq a\epsilon^{-\beta} \tag{E.2}$$

for all $\epsilon > 0$, where $H(B^{-1}\mathcal{G}, \epsilon, \|\cdot\|_{L_2(T)})$ is the entropy number of the set $B^{-1}\mathcal{G}$, and $\|f\|_{L_2(T)}^2 = \frac{1}{n}\sum_{i=1}^n f(\boldsymbol{x}_i, y_i)^2$ is the empirical norm. Then there exists a constant $c_\beta > 0$ depending only on β such that for all $n \geq 1$ and all $t \geq 1$ we have

$$\mathbb{P}(T \in Z^n : \mathcal{R}_{L,P}(f_{T,\mathcal{F}}) > \mathcal{R}_{L,P}(f_{P,\mathcal{F}}) + c_\beta \varepsilon(n, a, B, c, \delta, t)) \leq e^{-t},$$

where

$$
\begin{aligned}
&\varepsilon(n,a,B,c,\delta,t)\\
=&B^{2\beta/(4-2\alpha+\alpha p)}c^{(2-\beta)/(4-2\alpha+\alpha\beta)}\left(\frac{a}{n}\right)^{2/(4-2\alpha+\alpha\beta)}+B^{\beta/2}\delta^{(2-\beta)/4}\left(\frac{a}{n}\right)^{1/2}\\
&+B\left(\frac{a}{n}\right)^{2/(2+\beta)}+\sqrt{\frac{\delta t}{n}}+\left(\frac{ct}{n}\right)^{1/(2-\alpha)}+\frac{Bt}{n},
\end{aligned}
\tag{E.3}
$$

and $f_{T,\mathcal{F}}$ is the minimizer with respect to the empirical measure.

Lemma E.2. Assume the conditions of Theorem 3.1 hold. Define $C := 8||(2\eta-1)^{-1}||_{\kappa,\infty}+32$, where $\|\cdot\|_{\kappa,\infty}$ is the norm of Lorentz space $L_{\kappa,\infty}$ [62]. Let $\mu>0$ and $0<\gamma\le n^{1/2}\mu^{-1/2}$, then for all $f\in\gamma\mathcal{N}(1)$ we have

$$\mathbb{E}_{X,Y\sim P}(L_1\circ f-L_1\circ f_n)^2\le C(K\gamma+1)^2(\mathbb{E}_{X,Y\sim P}(L_1\circ f-L_1\circ f_n))+2C(K\gamma+1)^2a(\mu),$$

where $a(\mu)$ is the approximation error function given by

$$a(\mu)=\inf_{f\in\mathcal{N}}(n^{-1}\mu||f||_{\mathcal{N}}^2+\mathcal{R}_{l_1}(f)-\mathcal{R}_{l_1}).$$

Lemma E.3. Suppose Assumptions D.1 and D.4 hold. Then we have

$$\mathbb{E}_{X\sim P_X}(f_{\boldsymbol{W}(k),\boldsymbol{a}}(X)-\widehat{f}(X))^2=O_{\mathbb{P}}(n^{-\frac{d}{2d-1}}),$$

where $\widehat{f}$ is the solution to Equation E.1. Furthermore, if there exists a function $f\in\mathcal{N}$ that does not depend on n and $f(\boldsymbol{x}_i)=y_i$ for all $i=1,...,n$, then we can set $\mu=o(1)$ and obtain

$$\mathbb{E}_{X\sim P_X}(f_{\boldsymbol{W}(k),\boldsymbol{a}}(X)-\widehat{f}(X))^2=o_{\mathbb{P}}(1).$$

Remark 6. According to the proof in [40], the probability in $o_{\mathbb{P}}(1)$ of Lemma E.3 only relates to the width of the one-hidden-layer neural network, which can be arbitrarily small by enlarging the neural network's width.

Now we are ready to prove Theorem 3.1. Let $L=L_1$ in Lemma E.1, which is clearly continuous. Let $\widehat{f}$ be the solution to Equation E.1. The key idea in this proof is using $\widehat{f}$ to bridge two functions $2\eta-1$ and $f_{\boldsymbol{W}(k),\boldsymbol{a}}$.

Since $\widehat{f}$ is the solution to Equation E.1, it can be seen that

$$
\begin{aligned}
\frac{1}{n}\sum_{i=1}^n(y_i-\widehat{f}(\boldsymbol{x}_i))^2+\frac{\mu}{n}\left\|\widehat{f}\right\|_{\mathcal{N}}^2\le&\frac{1}{n}\sum_{i=1}^n(y_i-(2\eta(\boldsymbol{x}_i)-1))^2+\frac{\mu}{n}\left\|2\eta(\boldsymbol{x}_i)-1\right\|_{\mathcal{N}}^2\\
\le&\frac{2}{n}\sum_{i=1}^n(y_i^2+(2\eta(\boldsymbol{x}_i)-1)^2)+\frac{\mu}{n}\left\|2\eta(\boldsymbol{x}_i)-1\right\|_{\mathcal{N}}^2\\
\le&C_1,
\end{aligned}
\tag{E.4}
$$

where the second inequality is by the Cauchy-Schwarz inequality, and the third inequality is because $y_i^2=1$ and $\eta(\boldsymbol{x})$ is bounded.

The reproducing property implies that

$$\widehat{f}(\boldsymbol{x})=\langle\widehat{f},h(\boldsymbol{x},\cdot)\rangle_{\mathcal{N}}\le\left\|\widehat{f}\right\|_{\mathcal{N}}\|h(\boldsymbol{x},\cdot)\|_{\mathcal{N}}=\left\|\widehat{f}\right\|_{\mathcal{N}}\sqrt{h(\boldsymbol{x},\boldsymbol{x})},\forall\boldsymbol{x}\in\Omega,$$

which yields

$$\left\|\widehat{f}\right\|_{L_\infty}\le C_2\left\|\widehat{f}\right\|_{\mathcal{N}}.$$

Together with Equation E.4, we obtain

$$\left\|\widehat{f}\right\|_{L_\infty}\le C_3\left\|\widehat{f}\right\|_{\mathcal{N}}\le C_4(\mu/n)^{-1/2}.\tag{E.5}$$

Thus, we can take $B = C_4(\mu/n)^{-1/2}$ in Lemma E.1. The entropy condition can be verified via Lemma A.2, which allows us to take $\beta = 2(d-1)/d$. Equation E.5, together with Lemma E.2, also suggests that we can take $c = C(KB+1)^2$, $\alpha = 1$, and $\delta = 2C(KB+1)^2 a(\mu)$.

Next, we provide an upper bound on $a(\mu)$. The definition of $a(\mu)$ implies

$$\begin{aligned} a(\mu) =& n^{-1}\mu \|f_n\|_{\mathcal{N}}^2 + R_{l_1}(f_n) - R_{l_1} \\ =& n^{-1}\mu \|f_n\|_{\mathcal{N}}^2 + \mathbb{E}_{X\sim P_X}(2\eta(X) - 1 - f_n(X))^2 \\ \leq& n^{-1}\mu \|2\eta - 1\|_{\mathcal{N}}^2, \end{aligned} \tag{E.6}$$

where we use the relationship $\mathcal{R}_{l_1,P}(f) - \mathcal{R}_{l_1,P} = \mathbb{E}_{X\sim P_X}(2\eta(X) - 1 - f(X))^2$.

Plugging all the terms into Equation E.3, together with Lemma E.1, yields that

$$\mathcal{R}_{L_1,P}(\widehat{f}) = \mathcal{R}_{L_1,P}(f_n) + O_{\mathbb{P}}(\varepsilon(n,a,B,c,\delta)), \tag{E.7}$$

where

$$\begin{aligned} \varepsilon(n,a,B,c,\delta) =& B^{\frac{4}{2+\beta}} n^{-\frac{2}{2+\beta}} + B(\mu/n)^{\frac{2-\beta}{4}} n^{-\frac{1}{2}} \|2\eta - 1\|_{\mathcal{N}}^{\frac{2-\beta}{2}} + B^2 n^{-1} \\ =& B^{\frac{4d}{4d-2}} n^{-\frac{2d}{4d-2}} + B\mu^{\frac{1}{2d}} n^{-\frac{1}{2}-\frac{1}{2d}} \|2\eta - 1\|_{\mathcal{N}}^{\frac{1}{d}} + B^2 n^{-1}. \end{aligned} \tag{E.8}$$

Since $\mathcal{R}_{l_1,P}(f) - \mathcal{R}_{l_1,P} = \mathbb{E}_{X\sim P_X}(2\eta(X) - 1 - f(X))^2$, we subtract $\mathcal{R}_{l_1,P}$ on both sides of Equation E.7 and get

$$\begin{aligned} & \mathbb{E}_{X\sim P_X}(2\eta(X) - 1 - \widehat{f}(X))^2 + n^{-1}\mu \left\|\widehat{f}\right\|_{\mathcal{N}}^2 \\ =& \mathbb{E}_{X\sim P_X}(2\eta(X) - 1 - f_n(X))^2 + n^{-1}\mu \|f_n\|_{\mathcal{N}}^2 + O_{\mathbb{P}}(\varepsilon(n,a,B,c,\delta)) \\ =& O_{\mathbb{P}}(n^{-1}\mu \|2\eta - 1\|_{\mathcal{N}}^2 + \varepsilon(n,a,B,c,\delta)), \end{aligned} \tag{E.9}$$

where the last equality (with big O notation) is by Equation E.6. Combining Equation E.9 and Equation E.5 implies

$$\left\|\widehat{f}\right\|_{L_\infty}^2 \leq C_3^2 \left\|\widehat{f}\right\|_{\mathcal{N}}^2 = O_{\mathbb{P}}(1 + n\mu^{-1}\varepsilon(n,a,B,c,\delta)).$$

In the following, we will show that by taking $\mu \asymp n^{\frac{d-1}{2d-1}}$,

$$\mathbb{E}_{X\sim P_X}(2\eta(X) - 1 - \widehat{f}(X))^2 + n^{-1}\mu \left\|\widehat{f}\right\|_{\mathcal{N}}^2 = O_{\mathbb{P}}(n^{-\frac{d}{2d-1}} \max(1, \|2\eta - 1\|_{\mathcal{N}}^{\frac{2}{d}})) = O_{\mathbb{P}}(n^{-\frac{d}{2d-1}}). \tag{E.10}$$

If $n\mu^{-1}\varepsilon(n,a,B,c,\delta) \lesssim 1$, then $\varepsilon(n,a,B,c,\delta) \lesssim \mu/n$, and Equation E.10 holds. Otherwise, we can replace B^2 by its upper bound $O_{\mathbb{P}}(n\mu^{-1}\varepsilon(n,a,B,c,\delta))$ in Equation E.8 and obtain that

$$\varepsilon = O_{\mathbb{P}}(\varepsilon^{\frac{d}{2d-1}} \mu^{-\frac{d}{2d-1}} + \varepsilon^{\frac{1}{2}} \mu^{-\frac{d-1}{2d}} n^{-\frac{1}{2d}} \|2\eta - 1\|_{\mathcal{N}}^{\frac{1}{d}}),$$

where we set $\varepsilon = \varepsilon(n,a,B,c,\delta)$ for notational simplicity. Let us hereby denote $I_1 = \varepsilon^{\frac{d}{2d-1}} \mu^{-\frac{d}{2d-1}}$ and $I_2 = \varepsilon^{\frac{1}{2}} \mu^{-\frac{d-1}{2d}} n^{-\frac{1}{2d}} \|2\eta - 1\|_{\mathcal{N}}^{\frac{1}{d}}$, and consider the following two cases.

Case 1: $I_1 \geq I_2$, then we have

$$\varepsilon = O_{\mathbb{P}}(\varepsilon^{\frac{d}{2d-1}} \mu^{-\frac{d}{2d-1}}).$$

Solving this equality leads to

$$\varepsilon = O_{\mathbb{P}}(\mu^{-\frac{d}{d-1}}). \tag{E.11}$$

Plugging Equation E.11 into Equation E.9 and minimize the right-hand side of Equation E.9 with respect to μ gives us $\mu \asymp n^{\frac{d-1}{2d-1}}$; thus Equation E.10 holds.

Case 2: $I_1 < I_2$, then we have

$$\varepsilon = O_{\mathbb{P}}(\varepsilon^{\frac{1}{2}} \mu^{-\frac{d-1}{2d}} n^{-\frac{1}{2d}} \|2\eta - 1\|_{\mathcal{N}}^{\frac{1}{d}}),$$

which leads to

$$\varepsilon = O_{\mathbb{P}}(\mu^{-\frac{d-1}{d}} n^{-\frac{1}{d}} \|2\eta - 1\|_{\mathcal{N}}^{\frac{2}{d}}). \tag{E.12}$$

Similarly, we plug Equation E.12 into Equation E.9 and minimize the right-hand side of Equation E.9 with respect to μ and obtain $\mu \asymp n^{\frac{d-1}{2d-1}}$, which also leads to Equation E.10.

Now we can obtain an upper bound on the excess risk. For the notation simplicity, let $f = f_{\boldsymbol{W}(k),\boldsymbol{a}}$. The excess risk can be bounded by

$$\begin{aligned} L(f) - L^* \leq & \mathbb{E}_{X\sim P_X} \mathbb{I}\{(2\eta(X)-1)f(X) \leq 0, |\eta(X) - 0.5| < \delta\}|2\eta(X) - 1| \\ & + \mathbb{E}_{X\sim P_X} \mathbb{I}\{(2\eta(X)-1)f(X) \leq 0, |\eta(X) - 0.5| \geq \delta\}|2\eta(X) - 1|. \end{aligned} \tag{E.13}$$

The first term can be bounded via Tsybakov's noise condition as

$$\begin{aligned} & \mathbb{E}_{X\sim P_X} \mathbb{I}\{(2\eta(X)-1)f(X) \leq 0, |\eta(X) - 0.5| < \delta\}|2\eta(X) - 1| \leq 2\delta \mathbb{E}[\mathbb{I}\{|\eta(X) - 0.5| < \delta\}] \\ = & 2\delta \mathbb{P}(|\eta(X) - 0.5| < \delta) \leq 2C\delta^{\kappa+1}. \end{aligned} \tag{E.14}$$

It remains to bound the second term in Equation E.13. If $p(\boldsymbol{x})$ is continuous, then by the fact that $|2\eta(X) - 1| \leq |2\eta(X) - 1 - f(X)|$ if $(2\eta(X)-1)f(X) \leq 0$, we have

$$\begin{aligned} & \mathbb{E}_{X\sim P_X} \mathbb{I}\{(2\eta(X)-1)f(X) \leq 0, |\eta(X) - 0.5| \geq \delta\}|2\eta(X) - 1| \\ \leq & 2\delta^{-1} \mathbb{E}_{X\sim P_X} \mathbb{I}\{(2\eta(X)-1)f(X) \leq 0, |\eta(X) - 0.5| \geq \delta\}|2\eta(X) - 1|^2 \\ \leq & 2\delta^{-1} \mathbb{E}_{X\sim P_X} \mathbb{I}\{|\eta(X) - 0.5| \geq \delta\}(2\eta(X) - 1 - f(X))^2 \\ \leq & 2\delta^{-1} \mathbb{E}_{X\sim P_X} (2\eta(X) - 1 - f(X))^2 \\ \leq & 4\delta^{-1} \mathbb{E}_{X\sim P_X} (2\eta(X) - 1 - \widehat{f}(X))^2 + 4\delta^{-1} \mathbb{E}_{X\sim P_X} (f(X) - \widehat{f}(X))^2 \\ = & O_{\mathbb{P}}(\delta^{-1} n^{-\frac{d}{2d-1}}), \end{aligned} \tag{E.15}$$

where the fourth inequality is by the Cauchy-Schwarz inequality, and the last equality (with big O notation) is by Equation E.10 and Lemma E.3. Taking $\delta = n^{-\frac{d}{(2d-1)(\kappa+2)}}$, and plugging Equation E.14 and Equation E.15 into Equation E.13 leads to

$$L(f) = L^\star + O_{\mathbb{P}}(n^{-\frac{d(\kappa+1)}{(2d-1)(\kappa+2)}}).$$

This finishes the proof.

E.2 Proof of Theorem 3.2

We first present a lemma.

Lemma E.4. Suppose two sets are separable with a positive margin $\gamma > 0$. Then there exists a function f_T satisfying

$$f_T(\boldsymbol{x}) = 1, \forall \boldsymbol{x} \in \Omega_1, \quad f_T(\boldsymbol{x}) = -1, \forall \boldsymbol{x} \in \Omega_2.$$

Proof of Theorem 3.2. By the equivalence of the RKHS generated by the Laplace kernel and $\mathcal{N}$ [46, 47], it can be shown that $\mathcal{N}$ can be embedded into the Sobolev space W_2^ν for some $\nu > d/2$. Consider the Hölder space $C_b^{0,\alpha}$ for $0 < \alpha \leq 1$ equipped with the norm

$$\|f\|_{C_b^{0,\alpha}} := \sup_{\boldsymbol{x},\boldsymbol{x}' \in \Omega, \boldsymbol{x} \neq \boldsymbol{x}'} \frac{|f(\boldsymbol{x}) - f(\boldsymbol{x}')|}{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^\alpha}. \tag{E.16}$$

By the Sobolev embedding theorem, we have the embedding relationship

$$\|f\|_{C_b^{0,\tau}} \leq C_1 \|f\|_{W_2^\nu} \leq C_2 \|f\|_{\mathcal{N}} \tag{E.17}$$

for all $f \in \mathcal{N}$, where $\tau = \min(\nu - d/2, 1)$.

Without loss of generality, let us consider $\boldsymbol{x} \in \Omega_1$. The case of $\boldsymbol{x} \in \Omega_2$ can be proved similarly. For any $\boldsymbol{x} \in \Omega_1$, take $\boldsymbol{x}' = \arg\min_{\boldsymbol{x}_i} \|\boldsymbol{x}_i - \boldsymbol{x}\|_2$. Thus, the definition of the Hölder space and Equation E.17 imply

$$|f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}) - f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}')| \leq C_2 \left\|f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{\mathcal{N}} \|\boldsymbol{x}' - \boldsymbol{x}\|_2^\tau \leq C_3 \|\boldsymbol{x}' - \boldsymbol{x}\|_2^\tau, \tag{E.18}$$

where the last inequality is by Assumption D.3.

Let $\widehat{f}$ be the solution to Equation E.1, and let $\widehat{f}_1$ be the solution to Equation E.1 with $\mu = 0$. Let f_T be as in Lemma E.4. Note that $\widehat{f}_1$ satisfies $\widehat{f}_1(\boldsymbol{x}_i) = f_T(\boldsymbol{x}_i)$. Thus, by the identity $\left\|\widehat{f}_1\right\|_{\mathcal{N}}^2 + \left\|\widehat{f}_1 - f_T\right\|_{\mathcal{N}}^2 = \left\|\widehat{f}_T\right\|_{\mathcal{N}}^2$ [63], we have $\left\|\widehat{f}_1\right\|_{\mathcal{N}} \leq \|f_T\|_{\mathcal{N}}$. Since $\widehat{f}$ is the solution to Equation E.1, we have

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{f}(\boldsymbol{x}_i))^2 + \frac{\mu}{n}\left\|\widehat{f}\right\|_{\mathcal{N}}^2 \leq \frac{1}{n}\sum_{i=1}^{n}(y_i - f_T(\boldsymbol{x}_i))^2 + \frac{\mu}{n}\|f_T\|_{\mathcal{N}}^2 = \frac{\mu}{n}\|f_T\|_{\mathcal{N}}^2, \tag{E.19}$$

which implies $\left\|\widehat{f}\right\|_{\mathcal{N}} \leq \|f_T\|_{\mathcal{N}}$, where we utilize $y_i = f_T(\boldsymbol{x}_i)$ in the separable case.

Direct computation shows

$$\begin{aligned} f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}') =& \widehat{f}_1(\boldsymbol{x}') - (\widehat{f}_1(\boldsymbol{x}') - \widehat{f}(\boldsymbol{x}')) - (\widehat{f}(\boldsymbol{x}') - f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}')) \\ =& 1 - I_1 - I_2, \end{aligned} \tag{E.20}$$

where we use $\widehat{f}_1(\boldsymbol{x}_i) = 1$ for any $\boldsymbol{x}_i \in \Omega$; thus $\widehat{f}_1(\boldsymbol{x}') = 1$.

By the representer theorem, $\widehat{f}$ and $\widehat{f}_1$ can be expressed as

$$\widehat{f}_1(\boldsymbol{x}) = h(\boldsymbol{x}, \boldsymbol{X})(\boldsymbol{H}^\infty + \mu \boldsymbol{I}_n)^{-1}\boldsymbol{y}, \widehat{f}(\boldsymbol{x}) = h(\boldsymbol{x}, \boldsymbol{X})(\boldsymbol{H}^\infty)^{-1}\boldsymbol{y},$$

where $h(\boldsymbol{x}, \boldsymbol{X}) = (h(\boldsymbol{x}, \boldsymbol{x}_1), ..., h(\boldsymbol{x}, \boldsymbol{x}_n)) \in \mathbb{R}^{1\times n}$, $\boldsymbol{H}^\infty = (h(\boldsymbol{x}_i, \boldsymbol{x}_j))_{n\times n}$, and $\boldsymbol{y} = (y_1, ..., y_n)^\top = (f_T(\boldsymbol{x}_1), ..., f_T(\boldsymbol{x}_n))^\top$. Thus, the first term I_1 in Equation E.20 can be bounded by

$$\begin{aligned} |I_1| =& |\widehat{f}_1(\boldsymbol{x}') - \widehat{f}(\boldsymbol{x}')| = |h(\boldsymbol{x}', \boldsymbol{X})(\boldsymbol{H}^\infty)^{-1}\boldsymbol{y} - h(\boldsymbol{x}', \boldsymbol{X})(\boldsymbol{H}^\infty + \mu\boldsymbol{I}_n)^{-1}\boldsymbol{y}| \\ =& |\mu h(\boldsymbol{x}', \boldsymbol{X})(\boldsymbol{H}^\infty)^{-1}(\boldsymbol{H}^\infty + \mu\boldsymbol{I}_n)^{-1}\boldsymbol{y}| \\ \leq& \mu\sqrt{h(\boldsymbol{x}', \boldsymbol{X})(\boldsymbol{H}^\infty)^{-1}(\boldsymbol{H}^\infty + \mu\boldsymbol{I}_n)^{-1}(\boldsymbol{H}^\infty)^{-1}h(\boldsymbol{x}', \boldsymbol{X})^\top\boldsymbol{y}^\top(\boldsymbol{H}^\infty + \mu\boldsymbol{I}_n)^{-1}\boldsymbol{y}} \\ =& \mu\sqrt{h(\boldsymbol{x}', \boldsymbol{X})(\boldsymbol{H}^\infty)^{-1}(\boldsymbol{H}^\infty + \mu\boldsymbol{I}_n)^{-1}(\boldsymbol{H}^\infty)^{-1}h(\boldsymbol{x}', \boldsymbol{X})^\top}\left\|\widehat{f}_1\right\|_{\mathcal{N}} \\ \leq& \sqrt{\mu}\sqrt{h(\boldsymbol{x}', \boldsymbol{X})(\boldsymbol{H}^\infty)^{-2}h(\boldsymbol{x}', \boldsymbol{X})^\top}\|f_T\|_{\mathcal{N}} = \sqrt{\mu}\|f_T\|_{\mathcal{N}}, \end{aligned} \tag{E.21}$$

where the first inequality is by the Cauchy-Schwarz inequality, the second inequality is because $(\boldsymbol{H}^\infty + \mu\boldsymbol{I}_n)^{-1} \preceq \mu^{-1}\boldsymbol{I}_n$, and the last equality is because for any $\boldsymbol{x}_i$, $(\boldsymbol{H}^\infty)^{-1}h(\boldsymbol{x}_i, \boldsymbol{X})^\top = \boldsymbol{e}_i$. Therefore, I_1 converges to zero as $n \to \infty$ since $\mu = o(1)$. Specifically, there exists an n_1 such that when $n \geq n_1$, $|I_1| \leq 1/4$.

The second term I_2 in Equation E.20 can be bounded by

$$\begin{aligned} |I_2| \leq& \left\|\widehat{f} - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_\infty \leq C_4\left\|\widehat{f} - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{\mathcal{N}}^{\frac{d-1}{d}}\left\|\widehat{f} - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_2^{\frac{1}{d}} \\ \leq& C_4(\left\|\widehat{f}\right\|_{\mathcal{N}} + \|f_{\boldsymbol{W}(k),\boldsymbol{a}}\|_{\mathcal{N}})^{\frac{d-1}{d}}\left\|\widehat{f} - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_2^{\frac{1}{d}} \\ \leq& C_5\left(\mathbb{E}_{X\sim P_X}(\widehat{f}(X) - f_{\boldsymbol{W}(k),\boldsymbol{a}}(X))^2\right)^{\frac{1}{2d}}, \end{aligned} \tag{E.22}$$

which converges to zero by Lemma E.3. In Equation E.22, the second inequality is by the interpolation inequality, the third inequality is by the triangle inequality, and the last inequality is because of Assumption D.5. Therefore, there exists an n_2 such that when $n \geq n_2$, with probability at least $1 - \delta$, $|I_2| \leq 1/4$.

Take $n_0 = \max(n_1, n_2)$. For $n \geq n_0$, Equation E.20 gives us $f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}') \geq 1/2$ with probability at least $1 - \delta$. Therefore, by Equation E.18, as long as

$$\|\boldsymbol{x}' - \boldsymbol{x}\|_2 = \min_{\boldsymbol{x}_i}\|\boldsymbol{x}_i - \boldsymbol{x}\|_2 \leq (4C_3)^{-1/\tau} := C_6, \forall \boldsymbol{x} \in \Omega_1, \tag{E.23}$$

we have $f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}) \geq 1/4$ for all $\boldsymbol{x} \in \Omega_1$, which implies that the missclassification rate is zero.

Let $\mathrm{N}(\delta, \Omega_1, \|\cdot\|_2)$ be the covering number of Ω_1 and $\mathrm{N}_0 = \mathrm{N}(C_6/2, \Omega_1, \|\cdot\|_2)$. Since Ω_1 is compact and $C_6 > 0$, N_0 is finite (and is a constant). Therefore, Ω_1 can be covered by N_0 balls with radius $C_6/2$ (denoted by $\mathbf{B}$), and as long as for each ball, there exists one point $\boldsymbol{x}_j$ in this ball, Equation E.23 is satisfied. Since $\boldsymbol{x}_k$ has a probability density function with lower bound c_1, it remains to upper bound the probability that there exists one ball such that there is no point in it. Define this event as $\mathcal{A}$. The union bound of probability implies that for $n > n_0$,

$$\mathbb{P}(\mathcal{A}) \leq \mathrm{N}_0 \left(1 - \frac{c_1 Vol(\mathbf{B})}{Vol(\Omega_1)}\right)^n \leq \mathrm{N}_0 \exp(-C_7 n),$$

where $C_7 = -\log\left(1 - \frac{c_1 Vol(\mathbf{B})}{Vol(\Omega_1)}\right)$ is a positive constant. Clearly, we can adjust the constants such that the results in Theorem 3.2 holds for all n. This finishes the proof.

E.3 Proof of Theorem 3.3

Note that $f_{\boldsymbol{W}(k),\boldsymbol{a}}$ is a classifier, and the decision boundary is defined by $\mathcal{D}_T := \{\boldsymbol{x} | f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}) = 0\}$. Take any point $\boldsymbol{x}'$ in $\mathcal{D}_T$. The definition of the Hölder space and Equation E.17 imply

$$\frac{|f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}) - f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}')|}{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^\tau} \leq C_2 \left\|f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{\mathcal{N}} \leq C_3, \forall \boldsymbol{x} \in \Omega, \tag{E.24}$$

which is the same as

$$\|\boldsymbol{x} - \boldsymbol{x}'\|_2^\tau \geq |f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x})|/C_3, \forall \boldsymbol{x} \in \Omega, \tag{E.25}$$

where the last inequality in Equation E.24 is because of Assumption D.3. Therefore, it suffices to provide a lower bound of $|f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x})|$. Without loss of generality, let $\boldsymbol{x} \in \Omega_1$, because the case $\boldsymbol{x} \in \Omega_2$ can be proved similarly. However, this has already been proved in the proof of Theorem 3.2, where we showed that with probability at least $1 - \delta - C_4 \exp(-C_5 n)$, $f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}) \geq 1/4$ for all $\boldsymbol{x} \in \Omega_1$.

E.4 Proof of Theorem 3.4

By applying the interpolation inequality, the L_∞ norm of $2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}$ can be bounded by

$$\begin{aligned}
\left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_\infty \leq & C_0 \left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_2^{\frac{1}{d}} \left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{W^\nu}^{\frac{d-1}{d}} \\
\leq & C_1 \left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_2^{\frac{1}{d}} \left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{\mathcal{N}}^{\frac{d-1}{d}} \\
\leq & C_2 \left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_2^{\frac{1}{d}} \left(\|2\eta - 1\|_{\mathcal{N}} + \left\|f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_{\mathcal{N}}\right)^{\frac{d-1}{d}} \\
\leq & C_3 \left\|2\eta - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}\right\|_2^{\frac{1}{d}} \\
\leq & C_4 \left(\mathbb{E}_{X \sim P_X} (2\eta(X) - 1 - f_{\boldsymbol{W}(k),\boldsymbol{a}}(X))^2\right)^{\frac{1}{2d}} = O_{\mathbb{P}}(n^{-\frac{1}{4d-2}})
\end{aligned} \tag{E.26}$$

where the second equality is by the equvilance of the Sobolev space W^ν and the RKHS $\mathcal{N}$; the third inequality is by the triangle inequality; the fourth inequality is by Assumptions D.2 and D.3; the fifth inequality is because of Assumption D.5; and the last equality (with big O notation) is because of Equation E.15. This finishes the proof.

E.5 Proof of Lemma 3.5

Let's first consider the simplest $d = 1$ case where $\Omega_1 = \{\gamma\}$ and $\Omega_2 = \{-\gamma\}$. Let ϕ denote the standard normal $N(0, 1)$ density. By injecting Gaussian noises $N(0, \upsilon^2)$, the induced conditional probability can be written as

$$\widetilde{\eta}_\upsilon(x) = \frac{\phi(\frac{x-\gamma}{\upsilon})}{\phi(\frac{x-\gamma}{\upsilon}) + \phi(\frac{x+\gamma}{\upsilon})} = \frac{1}{1 + \exp(-\frac{2\gamma x}{\upsilon^2})}.$$

For small enough $1/2 > t > 0$, direct calculation yields $\{x \in \mathbb{R} : |2\widetilde{\eta}_\upsilon(x) - 1| < t\} = (-x_t, x_t)$ where

$$x_t = \frac{\upsilon^2}{2\gamma} \log\left(\frac{1+t}{1-t}\right) \leq \frac{2\upsilon^2}{\gamma} t.$$

Hence,

$$P_X(|2\widetilde{\eta}_\upsilon(x)-1|<t) = P_X(-x_t < x < x_t) \le 2x_t\phi((\gamma+x_t)/\upsilon)$$
$$\le \frac{C\upsilon^2}{\gamma}\exp\left(-\frac{\gamma^2}{2\upsilon^2}\right)\cdot t.$$

In general cases, notice that Tsybakov's noise condition measures the separation between classes. Therefore, the bottleneck for the inequality is where Ω_1 and Ω_2 are the closest, i.e., where margin 2γ is attained. Let $\boldsymbol{x}_+ \in \Omega_1$ and $\boldsymbol{x}_- \in \Omega_2$ satisfy $\|\boldsymbol{x}_+ - \boldsymbol{x}_-\|_2 = 2\gamma$ (which can be attained since Ω is compact). Consider the delta distribution at $\boldsymbol{x}_+$ and $\boldsymbol{x}_-$, which is less separated than the original distribution. Then, it reduces to the simplest case.

E.6 Proof of Theorem 3.6

A closer look at the proof of Theorem 3.1 reveals that the convergence rate depends polynomially on the constant in Tsybakov's noise condition. Specifically, it can be checked that $\|\widetilde{\eta}_\upsilon\|_{\mathcal{N}} = O(\text{poly}(1/\upsilon))$. Let $D^\alpha f := \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1}\cdots x_d^{\alpha_d}} f$ denote the α-th (weak) derivative of a function f with $|\alpha| = \alpha_1 + \ldots + \alpha_d$ for a multi-index $\alpha = (\alpha_1, \cdots, \alpha_d) \in \mathbb{N}_0^d$. The Sobolev embedding theorem implies that $\|\widetilde{\eta}_\upsilon\|_{\mathcal{N}}$ is bounded by the $\sum_{|\alpha|\le 1}\|D^\alpha\widetilde{\eta}_\upsilon\|_{L_2}$, which is polynomial with $1/\upsilon$. Furthermore, it can be checked that μ converges to zero polynomially with $\upsilon \to 0$. Under Tsybakov's noise condition, the convergence rate can be obtained via the proof of Theorem 3.1 as

$$L(\widehat{f}) - L^* = O_{\mathbb{P}}(C^{\frac{1}{\kappa+2}} n^{-\frac{d(\kappa+1)}{(2d-1)(\kappa+2)}}) = O_{\mathbb{P}}\left(\text{poly}\left(\frac{1}{\upsilon}\right) C^{\frac{1}{\kappa+2}} n^{-\frac{d(\kappa+1)}{(2d-1)(\kappa+2)}}\right).$$

In the Gaussian noise injection case, if we choose $\upsilon = \upsilon_n = n^{-1/2}$, applying Lemma 3.5 yields

$$L(\widehat{f}) - L^* = O_{\mathbb{P}}(e^{-n\gamma/6}\text{poly}\,(n)) = O_{\mathbb{P}}(e^{-n\gamma/7}).$$

E.7 Proof of Theorem 3.7

Direct computation implies that

$$f_i^*(\boldsymbol{x}) = \sum_{j=1}^K \eta_j(\boldsymbol{x}) v_{ji},$$

which implies

$$f^*(\boldsymbol{x}) = (\boldsymbol{v}_1, ..., \boldsymbol{v}_K)\eta(\boldsymbol{x}), \tag{E.27}$$

where v_{ji} is the i-th element of $\boldsymbol{v}_j$. Let $\boldsymbol{V} = (\boldsymbol{v}_1, ..., \boldsymbol{v}_K)$. Multiplying $\boldsymbol{V}^\top$ on both sides of Equation E.27 leads to

$$\begin{aligned}\boldsymbol{V}^\top f^*(\boldsymbol{x}) =& \boldsymbol{V}^\top(\boldsymbol{v}_1, ..., \boldsymbol{v}_K)\eta(\boldsymbol{x}) = \left(\frac{K}{K-1}\boldsymbol{I} - \frac{1}{K-1}\mathbf{1}\mathbf{1}^\top\right)\eta(\boldsymbol{x}) \\ =& \frac{K}{K-1}\eta(\boldsymbol{x}) - \frac{1}{K-1}\mathbf{1}\mathbf{1}^\top\eta(\boldsymbol{x}) \\ =& \frac{K}{K-1}\eta(\boldsymbol{x}) - \frac{1}{K-1}\mathbf{1},\end{aligned} \tag{E.28}$$

where $\mathbf{1} = (1, ..., 1)^\top$. In Equation E.28, the second equality is because $\boldsymbol{v}_i^\top\boldsymbol{v}_j = -1/(K-1)$ if $i \ne j$ and $\boldsymbol{v}_i^\top\boldsymbol{v}_i = 1$; and the last equality is because $\sum_{i=1}^n \eta_i(\boldsymbol{x}) = 1$. By Equation E.28, it can be seen that

$$\eta_j(\boldsymbol{x}) = \frac{(K-1)\boldsymbol{v}_j^\top f^*(\boldsymbol{x}) + 1}{K},$$

which finishes the proof.

F Proof of Lemmas in the Appendix

F.1 Proof of Lemma E.2

We follow the approach in the proof of Lemma 6.1 and Proposition 6.3 in [23]. Note that $f_{l,P} = 2p-1$ minimizes $\mathcal{R}_{l,P}$. We first show that for all $f \in \mathcal{F}$ and all $\alpha \geq 0$,

$$\mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_{l,P})^2 \leq C_{\eta,\kappa}(\|f\|_\infty + 1)^{\frac{2\kappa+4\alpha}{\kappa+\alpha}} \left\|(2\eta-1)^{-1}\right\|_{q,\infty}^{\frac{\alpha}{\kappa+\alpha}} \mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_{l,P})^{\frac{\kappa}{\kappa+\alpha}}, \tag{F.1}$$

where $C_{\eta,\kappa} := ||(2\eta-1)^{-1}||_{\kappa,\infty} + 4$. In particular, one can take $\alpha = 0$ and obtain

$$\mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_{l,P})^2 \leq C_{\eta,\kappa}(\|f\|_\infty + 1)^2 \mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_{l,P}). \tag{F.2}$$

Clearly, Tsybakov's noise condition implies that $\left\|(2\eta-1)^{-1}\right\|_{\kappa,\infty}$ exists. For $\boldsymbol{x} \in \Omega$, let $p := \mathbb{P}(Y=1|\boldsymbol{x})$ and $t := f(\boldsymbol{x})$. Without loss of generality, let $p > 1/2$. Additionally, we denote

$$\begin{aligned} v(p,t) =& p\left(l(1,t) - l\left(1, f_{l,P}(\boldsymbol{x})\right)\right)^2 + (1-p)\left(l(-1,t) - l\left(-1, f_{l,P}(\boldsymbol{x})\right)\right)^2, \\ m(p,t) =& p\left(l(1,t) - l\left(1, f_{l,P}(\boldsymbol{x})\right)\right) + (1-p)\left(l(-1,t) - l\left(-1, f_{l,P}(\boldsymbol{x})\right)\right). \end{aligned} \tag{F.3}$$

Note $f_{l,P} = 2p-1$ implies $l\left(1, f_{l,P}(\boldsymbol{x})\right) = 4(p-1)^2$ and $l\left(-1, f_{l,P}(\boldsymbol{x})\right) = 4p^2$. Plugging them into Equation F.3, it can be checked that

$$\begin{aligned} m(p,t) =& 1 + t^2 + 2(1-2p)t - 4p(1-p) = (1+t-2p)^2, \\ v(p,t) =& (1+t-2p)^2((t+1)^2 + 12p - 4pt - 12p^2). \end{aligned}$$

By taking

$$\alpha \geq \frac{\log 4 - \log(12p - 12p^2 - 4pt + 2 - (t-1)^2)}{\log|2p-1|}, \tag{F.4}$$

it can be shown that

$$v(p,t) \leq \left(2t^2 + \frac{4}{|2p-1|^\alpha}\right) m(p,t). \tag{F.5}$$

Since

$$\frac{\log 4 - \log(12p - 12p^2 - 4pt + 2 - (t-1)^2)}{\log|2p-1|} \leq \frac{\log 4 - \log(-\frac{2}{3}t^2 + 4)}{\log|2p-1|} \leq 0,$$

it suffices to take $\alpha \geq 0$. We further define

$$\begin{aligned} g(y,\boldsymbol{x}) &:= l(y, f(\boldsymbol{x})) - l(y, f_{l,P}(\boldsymbol{x})), \\ h_1(\boldsymbol{x}) &:= \eta(\boldsymbol{x})g(1,\boldsymbol{x}) + (1-\eta(\boldsymbol{x}))g(-1,\boldsymbol{x}), \\ h_2(\boldsymbol{x}) &:= \eta(\boldsymbol{x})g^2(1,\boldsymbol{x}) + (1-\eta(\boldsymbol{x}))g^2(-1,\boldsymbol{x}). \end{aligned}$$

Therefore, Equation F.5 implies $h_2(\boldsymbol{x}) \leq (2\|f\|_\infty^2 + \frac{4}{|2\eta(\boldsymbol{x})-1|^\alpha})h_1(\boldsymbol{x})$ for all $\boldsymbol{x}$ with $\eta(\boldsymbol{x}) \neq 1/2$. Hence, we obtain

$$\begin{aligned} \mathbb{E}_{X,Y\sim P}g^2 =& \int_{\{\boldsymbol{x}||2\eta(\boldsymbol{x})-1|^{-1}<t\}} h_2(\boldsymbol{x})dP_X + \int_{\{\boldsymbol{x}|\infty>|2\eta(\boldsymbol{x})-1|^{-1}\geq t\}} h_2(\boldsymbol{x})dP_X \\ \leq& (2\|f\|_\infty^2 + 4t^\alpha)\int_{\{\boldsymbol{x}||2\eta(\boldsymbol{x})-1|^{-1}<t\}} h_1(\boldsymbol{x})dP_X + \int_{\{\boldsymbol{x}|\infty>|2\eta(\boldsymbol{x})-1|^{-1}\geq t\}} (\|f\|_\infty + 1)^4 dP_X \\ \leq& 4(\|f\|_\infty^2 + t^\alpha)\mathbb{E}_{X,Y\sim P}g + (\|f\|_\infty + 1)^4 \left\|(2\eta-1)^{-1}\right\|_{q,\infty} t^{-\kappa} \\ \leq& 4t^\alpha(\|f\|_\infty + 1)^2\mathbb{E}_{X,Y\sim P}g + (\|f\|_\infty + 1)^4 \left\|(2\eta-1)^{-1}\right\|_{q,\infty} t^{-\kappa} \\ \leq& 4t^\alpha(\|f\|_\infty + 1)^2\mathbb{E}_{X,Y\sim P}g + (\|f\|_\infty + 1)^4 \left\|(2\eta-1)^{-1}\right\|_{\kappa,\infty} t^{-\kappa} \\ \leq& C_{\eta,\kappa}(\|f\|_\infty + 1)^{\frac{2\kappa+4\alpha}{\kappa+\alpha}} \left\|(2\eta-1)^{-1}\right\|_{\kappa,\infty}^{\frac{\alpha}{\kappa+\alpha}} \mathbb{E}_{X,Y\sim P}g^{\frac{\kappa}{\kappa+\alpha}}, \end{aligned}$$

where the last equality is implied by taking $t^{\kappa+\alpha} := (||f||_\infty + 1)^2(\mathbb{E}_{X,Y\sim P}g)^{-1}$. This shows Equation F.1 holds.

Based on Equation F.1, we can show that Lemma E.1 holds. To see this, let $\widehat{C} := (K\gamma + 1)^{(2\kappa+4\alpha)/(\kappa+\alpha)}$ and fix an $f \in \gamma B_{\mathcal{N}}$. The term $\mathbb{E}_{X,Y\sim P}(L_1 \circ f - L_1 \circ f_n)^2$ can be bounded by

$$
\begin{aligned}
&\mathbb{E}_{X,Y\sim P}(L_1 \circ f - L_1 \circ f_n)^2 \\
\leq& 2\mu^2 n^{-2}||f||^4 + 2\mu^2 n^{-2}||f_n||^4 + 2\mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_n)^2 \\
\leq& 4\mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_{l,P})^2 + 4\mathbb{E}_{X,Y\sim P}(l_1 \circ f_{l,P} - l_1 \circ f_n)^2 + 2\mu^2 n^{-2}||f||^4 + 2\mu^2 n^{-2}||f_n|| \\
\leq& 8C_{\eta,\kappa}\widehat{C}(\mathbb{E}_{X,Y\sim P}(l_1 \circ f - l_1 \circ f_{l,P}) + \mathbb{E}_{X,Y\sim P}(l \circ f_n - l \circ f_{l,P}))^{\kappa/(\kappa+\alpha)} + 2\mu^2 n^{-2}||f||^4 + 2\mu^2||f_n||^4 \\
\leq& C\widehat{C}(\mathbb{E}_{X,Y\sim P}(l \circ f - l \circ f_{l,P}) + \mathbb{E}_{X,Y\sim P}(l \circ f_n - l \circ f_{l,P}) + \mu^2 n^{-2}||f||^4 + \mu^2 n^{-2}||f_n||^4)^{\kappa/(\kappa+\alpha)} \\
\leq& C\widehat{C}(\mathbb{E}_{X,Y\sim P}(L_1 \circ f - L_1 \circ f_n) + 2\mathbb{E}_{X,Y\sim P}(l \circ f_n - l \circ f_{l,P}) + 2\mu n^{-1}||f_n||^2)^{\kappa/(\kappa+\alpha)} \\
\leq& C\widehat{C}(\mathbb{E}_{X,Y\sim P}(L_1 \circ f - L_1 \circ f_n))^{\kappa/(\kappa+\alpha)} + 2C\widehat{C}a^{\kappa/(\kappa+\alpha)}(\mu). \qquad \text{(F.6)}
\end{aligned}
$$

In Equation F.6 the first and second inequalities is because of the Cauchy-Schwarz inequality; the third inequality is because of Equation F.1 and $a^p + b^p < 2(a+b)^p$ for all $a, b \geq 0, 0 < p \leq 1$; the fourth inequality follows from $a^p + b^p < 2(a+b)^p$ for all $a, b \geq 0, 0 < p \leq 1$; the fifth inequality is because $n^{-1}\mu||f||^2 \leq 1$ and $n^{-1}\mu||f_n||^2 \leq 1$; the last inequality follows $(a+b)^p < a^p + b^p$ for all $a, b \geq 0, 0 < p \leq 1$. This finishes the proof of Lemma E.1.

F.2 Proof of Lemma E.4

Since there is a positive margin between Ω_1 and Ω_2, we can always find two sets $\widetilde{\Omega}_1$ and $\widetilde{\Omega}_2$ with infinitely smooth boundaries such that $\Omega_1 \subset \widetilde{\Omega}_1$, and $\Omega_2 \subset \widetilde{\Omega}_2$. Then the result follows from the Sobolev extension theorem.

G Appendix for Detailed Experiments

G.1 Synthetic Data

We consider the square loss based and cross-entropy based overparameterized neural networks (ONN) with ℓ_2 regularization, denoted as SL-ONN + ℓ_2 and CE-ONN + ℓ_2, respectively. The chosen ONNs are two-hidden-layer ReLU neural networks with 500 neurons for each layer, and the parameter μ is selected via a validation set. During the neural network training, we use the popular RMSProp optimizer with the default settings, and select the tuning parameter μ for SL-ONN + ℓ_2 and CE-ONN + ℓ_2 by a validation set.

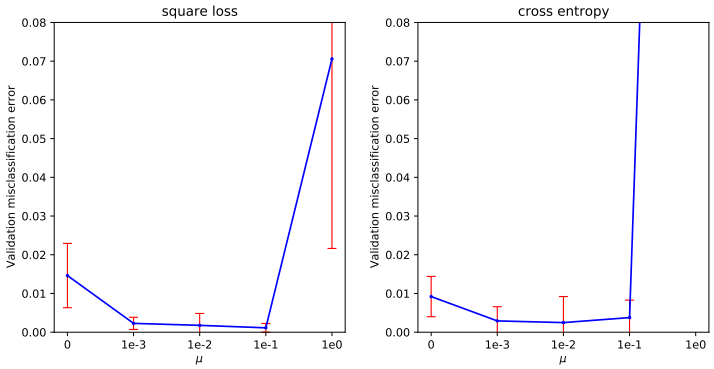

Figure G.3: The errorbar plot of validation misclassification rate with respect to different μ in the separable case.

Separable case In the separable case, we consider a two-dimension distribution $P = (\rho \sin\theta + 0.04, \rho\cos\theta)$ where $\rho = (\theta/4\pi)^{4/5} + \epsilon$ with selected θ from $(0, 4\pi]$ and $\epsilon \sim \text{unif}([-0.03, 0.03])$. We draw 100 positive and 100 negative training samples from $-P$ and P, respectively. We select the tuning parameter μ for SL-ONN + ℓ_2 and CE-ONN + ℓ_2 by minimizing the validation misclassification rate, where the candidate set of μ is $\{0, 0.001, 0.01, 0.1, 1\}$. For each μ, we generate 40 replications to estimate the mean and standard deviation of validation misclassification rate. We observe that SL-ONN + ℓ_2 and CE-ONN + ℓ_2 have the least mean and least standard deviation for the validation misclassification rate at $\mu = 0.1$ and $\mu = 0.01$, respectively. The errorbar plot for each μ is shown in Figure G.3.

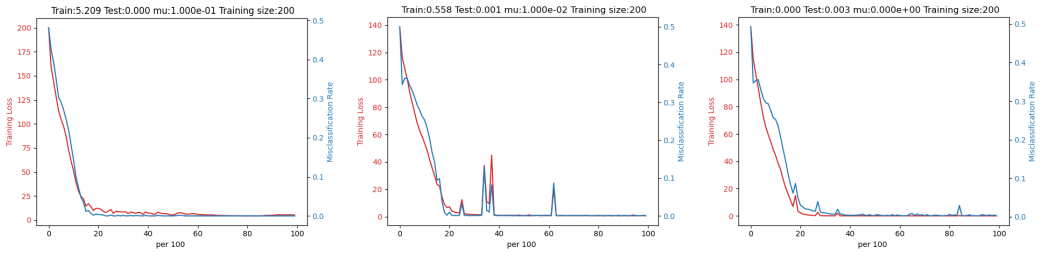

Figure G.4: An instance about the training process of SL-ONN + ℓ_2 *(Left)*, CE-ONN + ℓ_2 *(Center)* and CE-ONN *(Right)* in the separable case.

We also consider the cross-entropy loss based ONN without ℓ_2 regularization (CE-ONN). All three models are trained for 10000 iterations and achieve 0% training misclassification rate. In Figure G.5, we present five more examples about the decision boundary prediction and test accuracy of SL-ONN + ℓ_2, CE-ONN + ℓ_2 and CE-ONN. We can find that SL-ONN + ℓ_2 still beats CE-ONN + ℓ_2 and CE-ONN in almost all the cases. SL-ONN + ℓ_2 attains the smallest misclassification rate and depicts the largest margin decision boundary which separates the positive and negative samples best. In addition, we can observe that CE-ONN + ℓ_2 outperforms CE-ONN in all cases, although the ℓ_2 regularization term bring some oscillations to the training of CE-ONN + ℓ_2, as shown in Figure G.4.

For providing concrete numerical verification, we also conduct an experiment to compare the margins of decision boundaries trained by the square loss, as shown in Figure G.7. Moreover, Figure G.7 also presents the margin of the decision boundary predicted by the cross entropy with adversarial training, where we adapt the PGD attack with the attack strength level 0.05. It is obvious that the margins of square loss is larger than that of cross entropy, and is comparable to that of cross entropy with adversarial training. Meanwhile, the decision boundary of cross entropy with adversarial training is more smooth than that of cross entropy. We note that hereby the decision boundary of cross entropy with adversarial training can be regarded as the optimal decision boundary. Hence, Figure G.7 verifies the robustness of the square loss and demonstrates that the decision boundary predicted by square loss is closer to the optimal one.

Non-separable case We consider the conditional probability $\eta(\boldsymbol{x}) = \sin(\sqrt{2}\pi||\boldsymbol{x}||_2), \boldsymbol{x} \in [-1, 1]^2$, and the calibration performance of SL-ONN + ℓ_2 and CE-ONN + ℓ_2, where the classifiers are denoted by $\widehat{f}_{l2}$ and $\widehat{f}_{ce}$, respectively. The training data points $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n$ are i.i.d. sampled from $\text{unif}([-1, 1]^2)$ and the training labels $y_1, \cdots, y_n$ are sampled according to $Bernoulli(\eta(\boldsymbol{x}_i))$, where $\eta(\boldsymbol{x}) = \sin(\sqrt{2}\pi||\boldsymbol{x}||_2)$, and $n = 8000$. The 3-dimensional plot of $\eta(\boldsymbol{x})$ is presented in Figure G.6.

We select the tuning parameter μ for SL-ONN + ℓ_2 and CE-ONN + ℓ_2 via a validation set, and the candidate set of μ is $\{0, 0.001, 0.01, 0.1, 1\}$. For each μ, we run 40 replications to estimate the mean and standard deviation of validation misclassification rate. The iteration number of training is 2000. We find SL-ONN + ℓ_2 and CE-ONN + ℓ_2 have the smallest mean and standard deviation for the validation misclassification rate at $\mu = 0.1$ and $\mu = 0.1$, respectively. The error bar plot [5] for μ equaling to 0, 0.001, 0.01, 0.1 and 1 is shown in Figure G.8.

The calibration error results are presented in Figure G.9. The error bar plot of the test calibration error shows that $\widehat{f}_{l2}$ has the smaller mean and standard deviation than $\widehat{f}_{ce}$. It suggests that square

[5] In an error bar plot, the center of each plot is the mean, and the upper and lower red dashes denote (mean+one standard deviation) and (mean − one standard deviation), respectively.

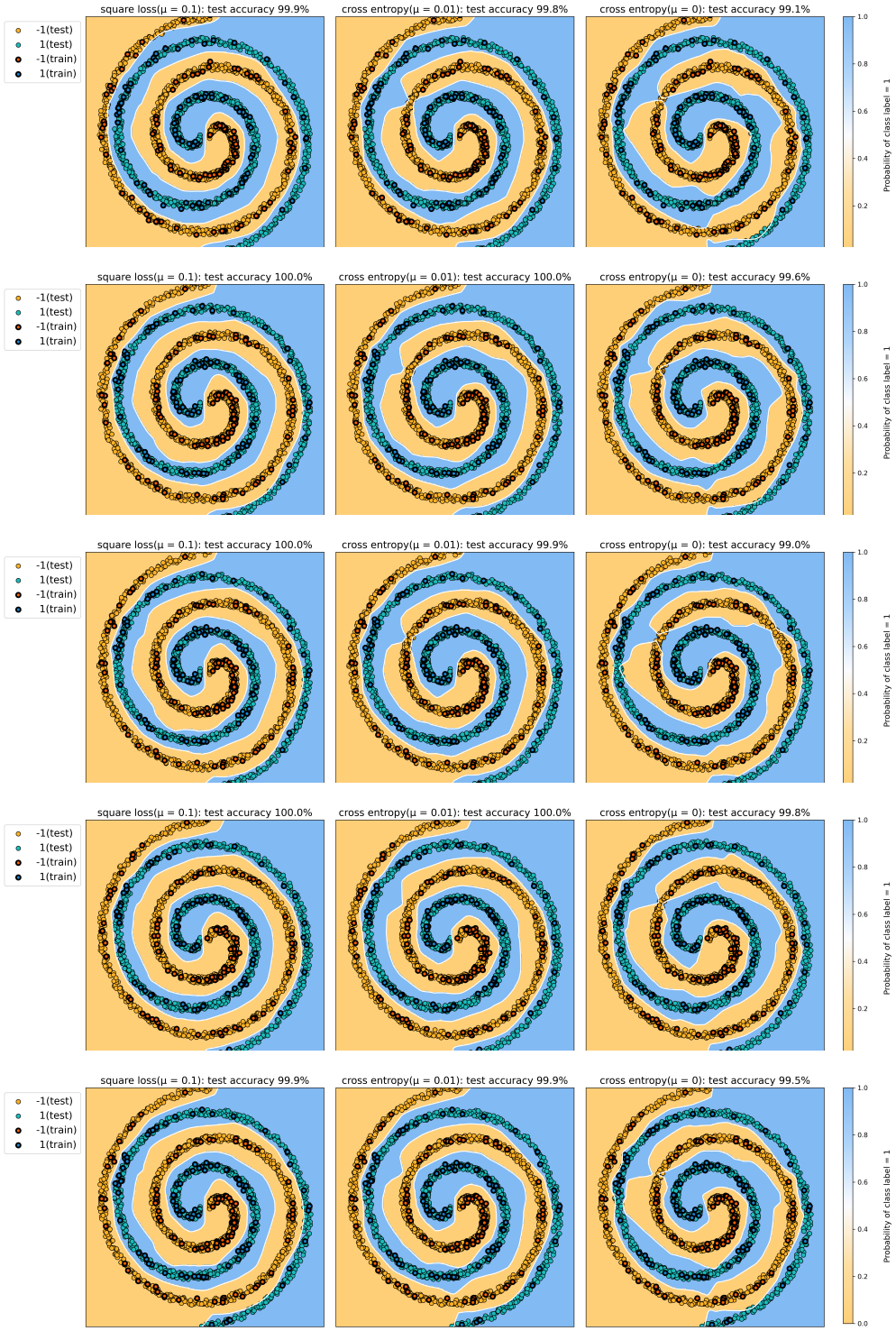

Figure G.5: Five examples in the separable case.

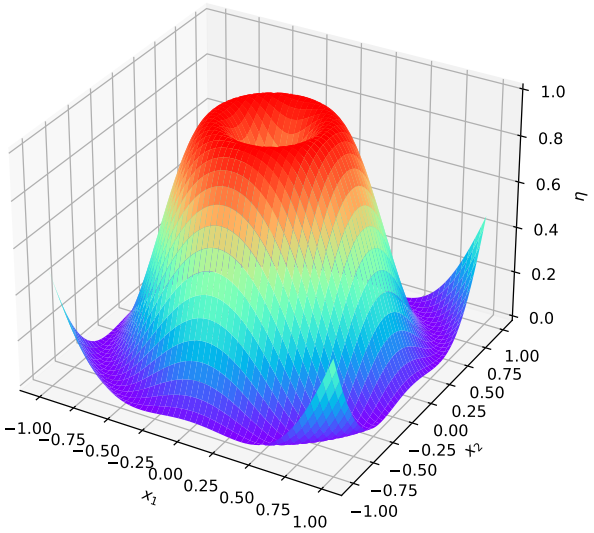

Figure G.6: The 3-dimensional plot of $\eta(\boldsymbol{x})$ in the non-separable case.

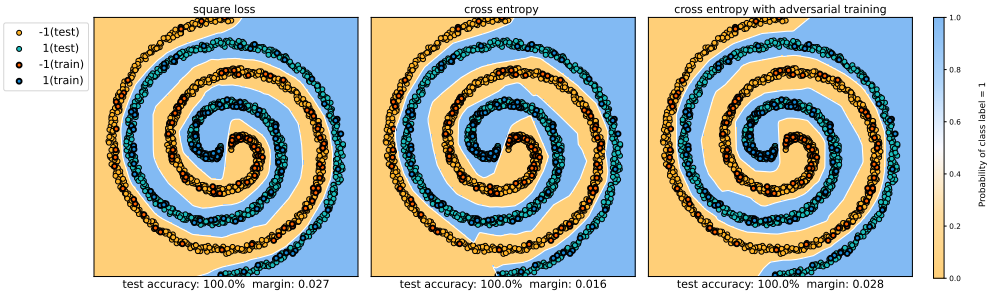

Figure G.7: Test misclassification rates, decision boundaries and margins predicted by: SL-ONN + ℓ_2 (Left); CE-ONN + ℓ_2 (Center); CE-ONN with adversarial training in the separable case.

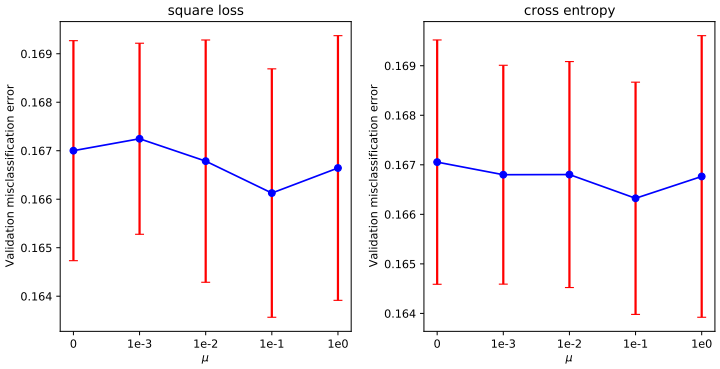

Figure G.8: The errorbar plot of validation misclassification rate with respect to different μ in the non-separable case.

loss generally outperforms cross entropy in calibration. The histogram and kernel density estimation of the test calibration errors for one case show that the pointwise calibration errors on the test points of $\widehat{f}_{l2}$ are more concentrated around zero than those of $\widehat{f}_{ce}$. Moreover, despite a comparable misclassification rate with $\widehat{f}_{ce}$, $\widehat{f}_{l2}$ has a smaller calibration error. Figure G.9 demonstrates that SL-ONN + ℓ_2 recovers η much better than CE-ONN + ℓ_2.

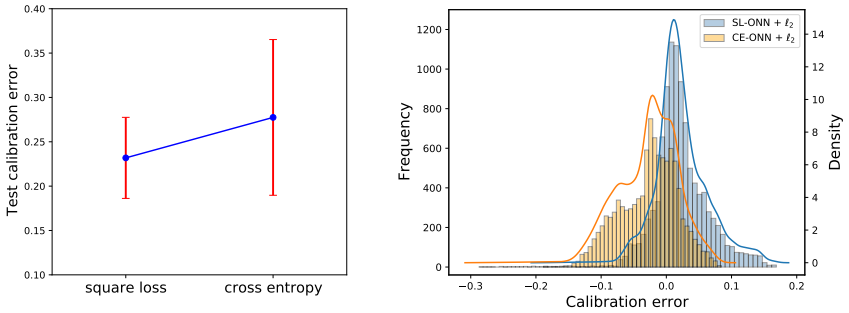

Figure G.9: (Left) The error bar plot of test calibration errors for the 40 replicated runs. (Right) The histogram and kernel density estimation of test calibration errors in one instance from 40 replications. In this instance, $||(\widehat{f}_{l2}+1)/2-\eta||_{L_\infty}$ = 0.188, $||(\widehat{f}_{ce}+1)/2-\eta||_{L_\infty}$ = 0.287, and the test misclassification rate for $\widehat{f}_{l2}$ and $\widehat{f}_{ce}$ are 0.167 and 0.164, respectively.

G.2 Real Data

Data and network architecture We use the popular CIFAR-10 and CIFAR-100 datasets, with training and testing split of 50000 and 10000. The data loader is from `torch.utils.data`. As typically employed in practice, our training includes data augmentations, a composition of random crop and horizontal flip. We trained two types of neural networks, ResNet [49] and Wide ResNet [50]. To be more specific, we used the default ResNet-18 and ResNet-50 for CIFAR-10 and CIFAR-100 respectively, and the default WRN-16-10 for both CIFAR-10 and CIFAR-100. All experiments are run in PyTorch version 1.9.0 and cuda 10.2.

Training details The training algorithm is the default SGD with momentum (0.9) and weight decay (0.0005). The learning rate scheduler is the `StepLR()` from `torch.optim.lr_scheduler` with step size 50. In our experiment, the only parameters that we tuned are the learning rate (lr) and batch size (bs), with only two options, (lr=0.01, bs=32) and (lr=0.1, bs=128). We find that (lr=0.01, bs=32) performs better for most cases except for square loss trained WRN-16-10 on CIFAR-100, where the average accuracy for (lr=0.01, bs=32) is 77.96%, around 1.5% less than that for (lr=0.1, bs=128). Meanwhile, for cross-entropy trained WRN-16-10, (lr=0.1, bs=128) yields an average accuracy of 76.83%, around 1% less than that for (lr=0.01, bs=32). The two training settings perform quite comparable for WRN-16-10 on CIFAR-10. For consistency, we stick with (lr=0.01, bs=32) in this case.

Adversarial robustness For square loss, training deep classifiers is the same as regression. When attacking classifiers trained with square loss, the default way of constructing adversarial examples doesn't work well. To be more specific, for a correctly classified training image $(\boldsymbol{x}, y)$, the adversarial examples are typically generated by

$$\max_{\|\boldsymbol{\delta}\|_\infty=\alpha} L(f(\boldsymbol{x}+\boldsymbol{\delta}), y).$$

Such an attacking scheme works fine for cross-entropy, where

$$L(f(\boldsymbol{x}), y) = -\log(\mathrm{softmax}(f(\boldsymbol{x}))) = -\log\left(\frac{\exp(f_y(\boldsymbol{x}))}{\sum_{k\neq y}\exp(f_k(\boldsymbol{x}))}\right),$$

but is problematic for regression losses such as square loss. The fundamental reason lies in Proposition 3.7 and its proof. Recall that the conditional probability for square loss consists of projections of

the classifier outputs to all the simplex vertices, some of which are sure to be non-positive. The sum of the class probabilities from Equation 3.2 is always 1 but unlike that from softmax function, the summand can be negative. By maximizing the square loss, the resulting "adversarial" image can stay the same class but more confidently. To illustrate, if $f(\boldsymbol{x}) = \boldsymbol{v}_y$, the predicted confidence for label y will be 100% and 0 for other classes. The "adversarial" image may be such that $f(\boldsymbol{x} + \delta) = 2\boldsymbol{v}_y$, where the predicted label remains unchanged but with an updated confidence of $2 - 1/K$ for label y and $(1/K - 1)/(K - 1) < 0$ for all other classes. This is obviously not a successful attack.

To this end, we devise a special attacking scheme for classifier trained with square loss and simplex coding. The key idea is to choose attack directions tangent to the sphere inscribed by the simplex. Instead of

$$L(f(\boldsymbol{x}), y) = \|f(\boldsymbol{x}) - \boldsymbol{v}_y\|_2^2,$$

we choose

$$L(f(\boldsymbol{x}), y) = \theta(f(\boldsymbol{x}), \boldsymbol{v}_y),$$

where $\theta(\boldsymbol{v}_1, \boldsymbol{v}_2)$ denotes the cosine similarity between $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. We refer to this attack as angle attack.

Empirically, we found our angle attack to significantly outperform the naive attack by maximizing the square loss. For square loss, let the predicted probabilities from Equation 3.2 be $\widehat{\boldsymbol{p}}$. Similar to cross entropy, we have also tried two cases of $L(f(\boldsymbol{x}), y)$, which corresponds to

$$L_1(f(\boldsymbol{x}), y) = -\log(\mathrm{softmax}(\widehat{p}_y(\boldsymbol{x}))) \quad \text{and} \quad L_2(f(\boldsymbol{x}), y) = -\log(\widehat{p}_y(\boldsymbol{x})).$$

Interestingly for PGD-100, L_1 performs the best, beating angle attack for the majority cases, except for attacking WRN-16-10 on CIFAR-100 with strength 2/255. The reported adversarial accuracy for square loss trained classifiers in Table 1 is by $L_1(f(\boldsymbol{x}), y) = -\log(\mathrm{softmax}(\widehat{p}_y(\boldsymbol{x})))$.

The PGD attack results may be further improved for square loss. Nonetheless, the AutoAttack still provides convincing results, as it includes both white-box and black-box attacks. We used the standard version which includes 4 types of attacks, APGD-CE, APGD-DLR, FAB and Square Attack as in [53].

Robustness to Gaussian Noise To make the robustness evaluation more comprehensive, beyond the adversarial robustness, we also investigate the classifier's robustness to Gaussian noise injections. With the image pixels' value normalized to 0 and 1, we consider injecting Gaussian noises to test images and report the test accuracy. The noise standard deviation ranges from 0.1 to 0.4. The test accuracy results for both CIFAR-10 and CIFAR-100 are listed in Table G.3.

Table G.3: Black-box Gaussian noise robustness results. The reported accuracy is the average of 5 replications.

Dataset	Network	Loss	*Gaussian noise standard deviation*				
			0.00	0.10	0.20	0.30	0.40
CIFAR-10	ResNet-18	SL	95.04	**90.07**	**70.16**	**42.13**	**25.38**
		CE	**95.15**	**90.03**	69.71	41.08	24.66
	WRN-16-10	SL	**95.02**	**88.49**	**60.91**	**35.78**	**24.04**
		CE	93.94	84.78	56.63	33.70	22.41
CIFAR-100	ResNet-50	SL	**78.91**	**63.06**	**36.64**	**17.78**	**9.47**
		CE	79.82	62.72	34.42	16.69	9.11
	WRN-16-10	SL	**79.65**	**62.01**	**30.69**	**15.11**	**8.88**
		CE	77.89	60.14	26.47	10.26	5.57

Simplex coding vs. one-hot coding The one-hot coding is the usual choice for applying square loss to classification. However, it is empirically observed to struggle when the number of classes are large. For a single training data point $\boldsymbol{x}$ and label k, [4] proposed to modify the training objective from the typical $(f_k(\boldsymbol{x}) - 1)^2 + \sum_{i \neq k} f_i(\boldsymbol{x})^2$ to $J \cdot (f_k(\boldsymbol{x}) - M)^2 + \sum_{i \neq k} f_i(\boldsymbol{x})^2$, where J, M are hyperparameters to make f_k more prominent in the loss. Similar modification is also proposed in [7]. The scaling trick involves two hyperparameters, which can be hard to tune. We evaluate the two coding schemes in our experiment setting and the results are summarized in Table G.4. The test accuracy for scaled one-hot coding scheme performs comparably for ResNet-18 on CIFAR-10 and ResNet-50 on CIFAR-100. For WRN-16-10, the simplex coding performs better.

Table G.4: Test accuracy for square loss with one-hot coding (scaled) (OC) vs. simplex coding (SC). Accuracy with an asteroid sign (*) denotes cases where the training accuracy doesn't overfit after 200 training epochs.

Dataset	Network	One-hot scaling	SGD parameters	OC clean acc(%)	SC clean acc(%)
CIFAR-10	ResNet-18	k=1, M=1	lr=0.01, bs=32	94.95	95.04
			lr=0.1, bs=128	10^*	10^*
	WRN-16-10	k=1,M =1	lr=0.01, bs=32	89.75^*	95.02
			lr=0.1, bs=128	88.43^*	95.03
CIFAR-100	ResNet-50	k=5, M=15	lr=0.01, bs=32	79.06	78.91
			lr=0.1, bs=128	1^*	1^*
	WRN-16-10	k=5, M=15	lr=0.01, bs=32	78.42	78.06
			lr=0.1, bs=128	78.39	79.65