Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Quantifying the generalization error in deep learning in terms of data distribution and neural network smoothness

Pengzhan Jin[a,b,1], Lu Lu[c,1], Yifa Tang[a,b], George Em Karniadakis[c,*]

[a] LSEC, ICMSEC, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
[c] Division of Applied Mathematics, Brown University, Providence, RI 02912, USA

ARTICLE INFO

Article history:
Received 27 May 2019
Received in revised form 25 June 2020
Accepted 29 June 2020
Available online 3 July 2020

Keywords:
Neural networks
Generalization error
Learnability
Data distribution
Cover complexity
Neural network smoothness

ABSTRACT

The accuracy of deep learning, i.e., deep neural networks, can be characterized by dividing the total error into three main types: approximation error, optimization error, and generalization error. Whereas there are some satisfactory answers to the problems of approximation and optimization, much less is known about the theory of generalization. Most existing theoretical works for generalization fail to explain the performance of neural networks in practice. To derive a meaningful bound, we study the generalization error of neural networks for classification problems in terms of data distribution and neural network smoothness. We introduce the *cover complexity* (CC) to measure the difficulty of learning a data set and the *inverse of the modulus of continuity* to quantify neural network smoothness. A quantitative bound for expected accuracy/error is derived by considering both the CC and neural network smoothness. Although most of the analysis is general and not specific to neural networks, we validate our theoretical assumptions and results numerically for neural networks by several data sets of images. The numerical results confirm that the expected error of trained networks scaled with the square root of the number of classes has a linear relationship with respect to the CC. We also observe a clear consistency between test loss and neural network smoothness during the training process. In addition, we demonstrate empirically that the neural network smoothness decreases when the network size increases whereas the smoothness is insensitive to training dataset size.

1. Introduction

In the last 15 years, deep learning, i.e., deep neural networks (NNs), has been used very effectively in diverse applications, such as image classification (Krizhevsky et al., 2012), natural language processing (Maas et al., 2013), and game playing (Silver et al., 2016). Despite this remarkable success, our theoretical understanding of deep learning is lagging behind. The accuracy of NNs can be characterized by dividing the expected error into three main types: approximation (also called expressivity), optimization, and generalization (Bottou, 2010; Bottou & Bousquet, 2008), see Fig. 1. The well-known universal approximation theorem was obtained by Cybenko (1989) and Hornik et al. (1989) almost three decades ago stating that feed-forward neural nets can approximate essentially any function if their size is sufficiently large. In the past several years, there have been numerous studies that analyze the landscape of the non-convex objective functions, and the optimization process by stochastic gradient descent (SGD) (Allen-Zhu, Li and Song, 2018; Du et al., 2018; Lee et al., 2016; Liao & Poggio, 2017; Lu et al., 2019, 2018). Whereas there are some satisfactory answers to the problems of approximation and optimization, much less is known about the theory of generalization, which is the focus of this study.

The classical analysis of generalization is based on controlling the complexity of the function class, i.e., model complexity, by managing the bias–variance trade-off (Friedman et al., 2001). However, this type of analysis is not able to explain the small generalization gap between training and test performance of neural networks learned by SGD in practice, considering the fact that deep neural networks often have far more model parameters than the number of samples they are trained on, and have sufficient capacity to memorize random labels (Neyshabur et al., 2014; Zhang et al., 2016). To explain this phenomenon, several approaches have been recently developed by many researchers. The first approach is characterizing neural networks with some other low "complexity" instead of the traditional Vapnik–Chervonenkis (VC) dimension (Bartlett, Harvey et al., 2017) or Rademacher complexity (Bartlett & Mendelson, 2002), such as path-norm (Neyshabur et al., 2015), margin-based bounds (Bartlett, Foster et al., 2017; Neyshabur, Bhojanapalli and Srebro, 2017; Sokolić et al., 2017),

[*] Correspondence to: 182 George Street, Providence, RI 02912, USA.
E-mail address: george_karniadakis@brown.edu (G.E. Karniadakis).
[1] Pengzhan Jin and Lu Lu contributed equally to this work.

https://doi.org/10.1016/j.neunet.2020.06.024
0893-6080/© 2020 Elsevier Ltd. All rights reserved.

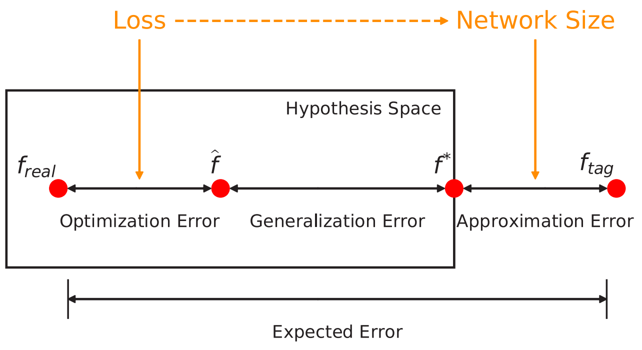

Fig. 1. Illustration of approximation error, optimization error, and generalization error. The total error consists of these three errors. f_{tag} is the target ground-truth function, f^* is the function closest to f_{tag} in the hypothesis space, $\hat{f}$ is a neural network whose loss is at a global minimum of the empirical loss, and f_{real} is the function returned by the training algorithm. Thus, the optimization error is correlated with the value of the empirical loss, while the approximation error depends on the network size. In addition, a small loss requires a large network size, which in turn leads to a small approximation error. Assuming a sufficiently small empirical loss, the expected error mainly depends on the generalization error.

Fisher–Rao norm (Liang et al., 2017), and more (Neyshabur et al., 2019; Wei & Ma, 2019). The second approach is to analyze some good properties of SGD or its variants, including its stability (Chen et al., 2018; Gonen & Shalev-Shwartz, 2017; Hardt et al., 2015; Kuzborskij & Lampert, 2017), robustness (Sokolic et al., 2016; Sokolić et al., 2017), implicit biases/regularization (Gunasekar et al., 2018; Nagarajan & Kolter, 2019b; Poggio et al., 2017; Soudry et al., 2018), and the structural properties (e.g., sharpness) of the obtained minimizers (Dinh et al., 2017; Keskar et al., 2016; Zhang et al., 2018). The third approach relies on over-parameterization, e.g., sufficiently overparameterized networks can learn the ground truth with a small generalization error using SGD from random initialization (Allen-Zhu, Li and Liang, 2018; Arora et al., 2019; Cao & Gu, 2019; Li & Liang, 2018). There are also other approaches, such as compression (Arora et al., 2018; Baykal et al., 2018; Cheng et al., 2018; Zhou et al., 2018), Fourier analysis (Rahaman et al., 2018; Xu et al., 2019), "double descent" risk curve (Belkin et al., 2018), PAC-Bayesian framework (Nagarajan & Kolter, 2019a; Neyshabur, Bhojanapalli and Srebro, 2017), and information bottleneck (Saxe et al., 2019; Shwartz-Ziv & Tishby, 2017).

However, most theoretical bounds fail to explain the performance of neural networks in practice (Arora et al., 2018; Neyshabur, Bhojanapalli, McAllester et al., 2017). To get non-vacuous and tight enough bounds to be practically meaningful, some problem-specific factors should be taken into consideration, such as the low complexity (i.e., data-dependent analysis) (Dziugaite & Roy, 2017; Kawaguchi et al., 2017), or properties of the trained neural networks (Arora et al., 2018; Sokolić et al., 2017; Wei & Ma, 2019). In this study, to achieve a practically meaningful bound, our analysis relies on the data distribution and the smoothness of the trained neural network. The analysis proposed in this study provides guarantees on the generalization error, and theoretical insights to guide the practical application.

As shown in Fig. 1, the optimization error is correlated with the loss value (for notation simplicity, the term "loss" indicates "empirical loss"), while the approximation error depends on the network size. In addition, a small loss requires a sufficient approximation ability, i.e., a large network size, which in turn leads to a small approximation error. If we assume a sufficiently small loss, which usually holds in practice, then the expected error mainly depends on the generalization error. Hence, we study the expected error/accuracy directly. In particular, we propose a mathematical framework to analyze the expected accuracy of neural networks for classification problems. We introduce the concepts of *total cover (TC)*, *self cover (SC)*, *mutual cover (MC)* and *cover difference (CD)* to represent the data distribution, and then we use the concept of *cover complexity (CC)* as a measure of the complexity of classification problems. On the other hand, the smoothness of a neural network f is characterized by the *inverse of the modulus of continuity* δ_f. Because computing δ_f is not tractable in general, we propose an estimation using the spectral norm of the weight matrices of the neural network. The main terminologies are illustrated in Fig. 2. By combining the properties of the data distribution and the smoothness of neural networks, we derive a lower bound for the expected accuracy, i.e., an upper bound for the expected classification error.

Subsequently, we test our theoretical bounds on several data sets, including MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky & Hinton, 2009), CIFAR-100 (Krizhevsky & Hinton, 2009), COIL-20 (Nene et al., 1996b), COIL-100 (Nene et al., 1996a), and SVHN (Netzer et al., 2011). Our numerical results not only confirm our theoretical bounds, but also provide insights into the optimization process and the learnability of neural networks. In particular, we find that:

- The best accuracy that can be achieved in practice (i.e., optimized by stochastic gradient descent) by fully-connected networks is approximately linear with respect to the cover complexity of the data set.
- The trend of the expected accuracy is consistent with the smoothness of the neural network, which provides a new "early stopping" strategy by monitoring the smoothness of the neural network.
- The neural network smoothness decreases when the network depth and width increases, with the effects of depth more significant than that of width.
- The neural network smoothness is insensitive to the training dataset size, and is bounded from below by a positive constant. This point makes our theoretical result (Theorem 3.4) specifically pertinent to deep neural networks.

The paper is organized as follows. After setting up notation and terminology in Section 2, we present the main theoretical bounds for the accuracy based on the data distribution and the smoothness of neural networks in Section 3, while all proofs are deferred to the Appendix. In Section 4, we provide the numerical results for several data sets. In Section 5 we include a discussion, and in Section 6 we summarize our findings.

2. Preliminaries

Before giving the main results, we introduce the necessary notation and terminology. Without loss of generality, we assume that the space we need to classify is

$$D = [0, 1]^d,$$

where d is the dimensionality, and the points in this space are classified into K categories, i.e., there are K labels $\{1, 2, \dots, K\}$. We denote the probability measure on D by μ, i.e., for a measurable set $A \subseteq D$, $\mu(A)$ is the probability of a random sample belonging to A.

2.1. Ideal label function

For the problem setup, we assume that every sample has at least one true label, and one sample may have multiple true labels. Taking image classification as an example, each image has at least one correct label. A fuzzy image or an image with more than one object in it may have multiple possible correct labels,

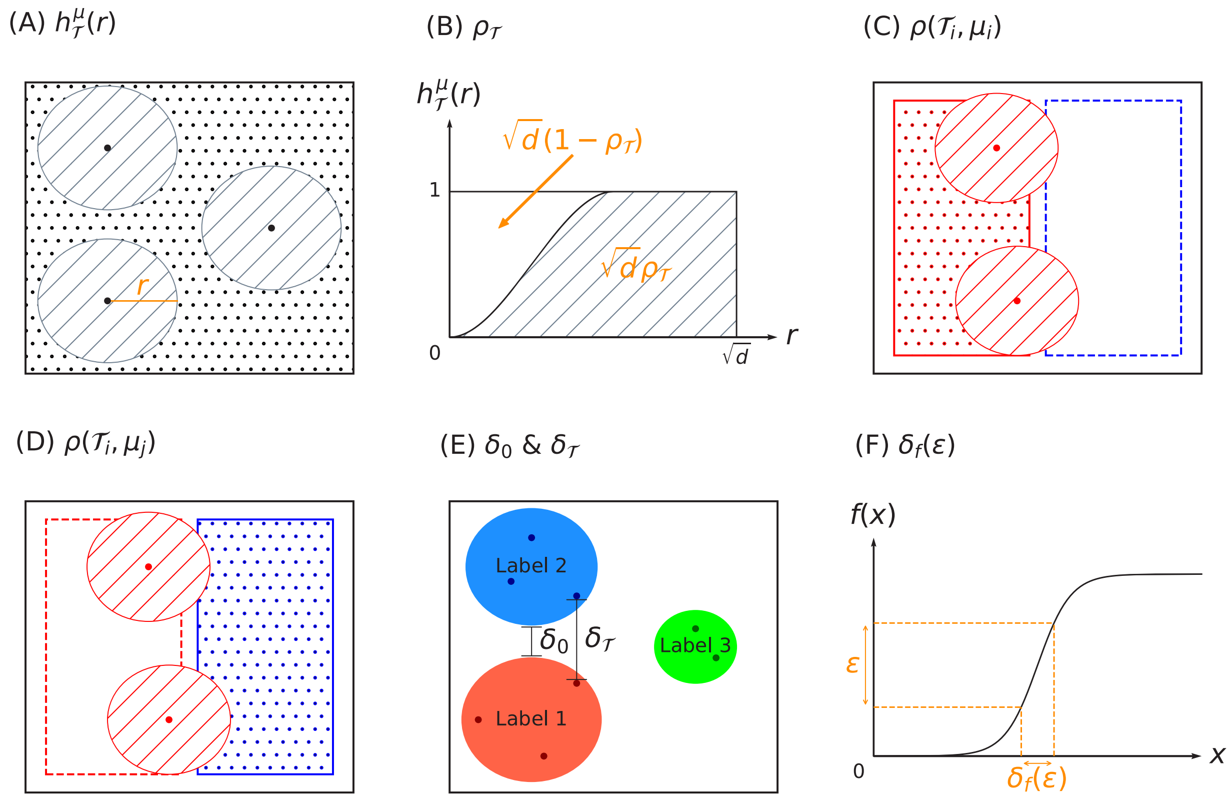
(A) $h_{\mathcal{T}}^{\mu}(r)$
r
(B) $\rho_{\mathcal{T}}$
$h_{\mathcal{T}}^{\mu}(r)$
$\sqrt{d}(1-\rho_{\mathcal{T}})$
1
0
$\sqrt{d}\rho_{\mathcal{T}}$
$\sqrt{d}$
r
(C) $\rho(\mathcal{T}_i,\mu_i)$
(D) $\rho(\mathcal{T}_i,\mu_j)$
(E) δ_0 & $\delta_{\mathcal{T}}$
Label 2
δ_0
$\delta_{\mathcal{T}}$
Label 3
Label 1
(F) $\delta_f(\varepsilon)$
$f(x)$
ε
0
$\delta_f(\varepsilon)$
x

Fig. 2. Illustrations of the main definitions and terminologies. (**A**) $h_{\mathcal{T}}^{\mu}(r)$ in Eq. (2): the probability of the neighborhood of the training set with radius of r. (**B**) $\rho_{\mathcal{T}}$ in Definition 2.5: total cover of training set $\mathcal{T}$. (**C**) $\rho(\mathcal{T}_i,\mu_i)$ in Definition 2.5: self cover of training set $\mathcal{T}$, which intuitively represents the aggregation of the data points of the same class. The red dots denote the probability distribution of the class "red", and the area of the disks intersected with the dotted rectangle represents the probability of the neighborhood of two "red" data points with respect to the distribution of the class "red" itself. (**D**) $\rho(\mathcal{T}_i,\mu_j)$ in Definition 2.5: mutual cover of training set $\mathcal{T}$, which intuitively represents the overlapping between the data points of two different classes. The blue dots denote the probability distribution of the class "blue", and the area of the disks intersected with the dotted rectangle represents the probability of the neighborhood of the two "red" data points with respect to the distribution of the class "blue". (**E**) δ_0 in Proposition 2.3: separation gap; $\delta_{\mathcal{T}}$ in Theorem 3.3: empirical separation gap. (**F**) δ_f in Definition 2.16: the inverse of the modulus of continuity.

and as long as the prediction is one of these labels, we consider the prediction to be correct.

It is intuitive that when two samples are close enough, they should have similar labels, which means that the ideal label function should be continuous. The continuity of a mapping depends on the topology of both domain and image space. For the domain of the ideal label function, we choose the standard topology induced by the Euclidean metric. As for the topology of the image space, we define it as follows. We first define the label set and the topology on it.

Definition 2.1 (*Topology*). Let

$$T = 2^{\{1,2,\dots,K\}}\backslash\{\varnothing\}$$

be the label set. Define the topology on T to be

$$\tau_T = \left\{ \bigcup_{k\in I} U_k \,\middle|\, I \subseteq T \right\},$$

where $U_k = \{V \in T | V \supseteq k\}$ for $k \in T$, and thus (T, τ_T) constitutes a topological space.

Analogous to the Euclidean metric topology, U_k is viewed as the open "ball" centered at k, and arbitrary unions of the "balls" U_k are defined as the open sets, see Appendix A for an example. With this choice of the topology, a function $f : D \to T$ is continuous if and only if

$$\forall x \in D \forall k \in f(x) \exists \delta > 0 \; s.t. \; k \in f(y) \; if \; \|y - x\| < \delta.$$

Next we give the definition of the ideal label function according to this topological space.

Definition 2.2 (*Ideal Label Function*). An idea label function is a continuous function

$$tag : D \to T$$

where D is equipped with the Euclidean metric topology and T with the topology τ_T from Definition 2.1. This continuity holds if and only if

$$\forall x \in D \forall k \in tag(x) \exists \delta > 0 \; s.t. \; k \in tag(y) \; if \; \|y - x\| < \delta. \tag{1}$$

Eq. (1) means that two neighboring points would have some common labels. Based on the topological space defined above, it is easy to show that Eq. (1) is equivalent to continuity. The reason why we consider a multi-label setup for classification problems is that it allows for the continuity property in Eq. (1), which is impossible in the setup of a single label set, unless the label function is constant. In addition, the multi-label setup introduces a smooth transition, i.e., a buffer domain, between two domains of different labels, while the transition is sharp in the single label setup. In the following proposition, we show that if two samples are close enough, they must share at least one common label.

Proposition 2.3 (*Separation Gap*). *$\exists \delta > 0$, $s.t.$ $tag(x) \cap tag(y) \neq \varnothing$ when $\|x-y\| < \delta$. We denote the supremum of δ as the separation gap δ_0, which is used in the sequel.*

Proof. The proof can be found in Appendix B. □

To understand the geometric interpretation of δ_0, we consider the following special case: the label of each sample is either a single label set, such as $\{1\}$, or the full label set $\{1, 2, \ldots, K\}$ if it is not uniquely identifiable.

Proposition 2.4 (*Geometric Interpretation of Separation Gap*)**.** *If the label of each sample is either a single label set or the full label set $\{1, 2, \ldots, K\}$, then δ_0 is the smallest distance between two different single label points, i.e.,*

$$\delta_0 = \inf\{ \|x - y\| \mid tag(x) \neq tag(y), \text{ and } \\ tag(x), tag(y) \in \{\{1\}, \{2\}, \ldots, \{K\}\} \}.$$

Proof. The proof can be found in Appendix C. □

2.2. Cover complexity of data set

In this subsection, we introduce a quantity to measure the difficulty of learning a training data set

$$\mathcal{T} = \{x_1, x_2, \ldots, x_n\} \subseteq D.$$

First, we give some notations and propositions.

With the measure μ, the probability of the neighborhood of the training set with radius of r is defined as

$$h^{\mu}_{\mathcal{T}}(r) := \mu\left(D \cap \bigcup_{x_i \in \mathcal{T}} B(x_i, r)\right), \tag{2}$$

where $B(x_i, r)$ is the open ball centered at x_i with radius of r, see Fig. 2A. Obviously, $h^{\mu}_{\mathcal{T}}(r)$ is a monotone non-decreasing function, $h^{\mu}_{\mathcal{T}}(0) = 0$ (since $B(x, 0) = \varnothing$), and $h^{\mu}_{\mathcal{T}}(r) = 1$ when $r > \sqrt{d}$, see Fig. 2B. To represent the global behavior of $h^{\mu}_{\mathcal{T}}(r)$, we use the integral of $h^{\mu}_{\mathcal{T}}(r)$ with respect to r:

$$\rho(\mathcal{T}, \mu) := \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} h^{\mu}_{\mathcal{T}}(r) dr.$$

Hence, $\rho(\mathcal{T}, \mu)$ considers both the number and location of the data points, and also the probability distribution of the space. The value $\rho(\mathcal{T}, \mu)$ is larger if the number of data points is increased and also if the probability distribution is more concentrated around $\mathcal{T}$, which we call the "coverability" of $\mathcal{T}$. We can increase $\rho(\mathcal{T}, \mu)$ by adding more data points or redistribute their locations. Next, we introduce the formal definition for the "coverability".

Definition 2.5 (*Coverability*)**.** Let $\mathcal{T}$ be a data set from a domain D with probability measure μ. We define the following for the coverability of $\mathcal{T}$.

(i) The total cover (TC) is

$$\rho_{\mathcal{T}} := \rho(\mathcal{T}, \mu).$$

Thus, $0 \leq \rho_{\mathcal{T}} \leq 1$.

(ii) The cover difference (CD) is

$$CD(\mathcal{T}) := \frac{1}{K} \sum_i \rho(\mathcal{T}_i, \mu_i) - \frac{1}{K(K-1)} \sum_{i \neq j} \rho(\mathcal{T}_i, \mu_j),$$

where K is the number of categories, and $\mathcal{T}_i \subset \mathcal{T}$ and μ_i represent the subset and probability measure of the label i respectively, i.e.,

$$\mathcal{T}_i := \{x \in \mathcal{T} | i \in tag(x)\}, \quad \mu_i(A) := \frac{\mu(A \cap D_i)}{\mu(D_i)}$$

with $D_i := \{x \in D | i \in tag(x)\}$. Here, $\rho(\mathcal{T}_i, \mu_i)$ is called self cover (SC), and $\rho(\mathcal{T}_i, \mu_j)$ is called mutual cover (MC).

(iii) The cover complexity (CC) is

$$CC(\mathcal{T}) := \frac{1 - \rho_{\mathcal{T}}}{CD(\mathcal{T})}$$

for $CD(\mathcal{T}) \neq 0$.

Remark 2.6. CD is defined as the difference between the mean of SC and the mean of MC, since each category occurs with the same probability ($\sim 1/K$) in the data sets mostly used in practice. If there are some categories occurring more frequently than others, then it is straightforward to extend this definition by using the mean weighted by the probability of each category.

In image classification, the dimension of the image space is very high, and thus the data points are quite sparse. However, due to the fact that images actually live on a manifold of low dimension, the probability density around $\mathcal{T}$ is actually high, which makes the TC to be meaningful. In our next result, we derive a lower bound of $h^{\mu}_{\mathcal{T}}(r)$ by $\rho_{\mathcal{T}}$.

Proposition 2.7. *Let $\mathcal{T}$ be a data set. $h^{\mu}_{\mathcal{T}}(r)$ and $\rho_{\mathcal{T}}$ are defined as above. Then we have*

$$h^{\mu}_{\mathcal{T}}(r) \geq 1 - \frac{\sqrt{d}}{r}(1 - \rho_{\mathcal{T}}), \quad 0 < r \leq \sqrt{d}.$$

Proof. The proof can be found in Appendix D. □

From this proposition, we know that for a fixed r, $h^{\mu}_{\mathcal{T}}(r)$ is close to 1 if $\rho_{\mathcal{T}}$ is large enough. However, the probability distribution is usually given in practice, and we can only control the number of samples. The following theorem shows that $\rho_{\mathcal{T}}$ can be arbitrary close to 1 when enough samples are available.

Theorem 2.8. *Let $\mathcal{T}$ be a data set of size n drawn according to μ. Then there exists a non-increasing function $\varrho(\epsilon)$ satisfying $\lim_{\epsilon \to 0} \varrho(\epsilon) = 1$, and for any $0 < \eta, \epsilon \leq \frac{1}{2}$, there exists an*

$$m = O\left(\frac{d+1}{\epsilon} \ln \frac{d+1}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\eta}\right),$$

such that

$$\rho_{\mathcal{T}} \geq \varrho(\epsilon)$$

holds with probability at least $1 - \eta$ when $n \geq m$.

Proof. The proof and some other results regarding TC can be found in Appendix E. □

The reason why CD is introduced is that TC does not consider the labels of the data points. However, data points of the same label should be clustered in an easily learnable data set. $CD(\mathcal{T})$ is the difference of self cover and mutual cover, which considers the distributions of each label. By normalizing TC with CD, the cover complexity $CC(\mathcal{T})$ is able to measure the difficulty of learning a data set. The difficulty of a problem should be translation-independent and scale-independent. It is easy to see that $CC(\mathcal{T})$ is independence of translation, and the following proposition shows that it is also scale-independent.

Proposition 2.9 (*Scale Independence*)**.** *$CC(\mathcal{T})$ is scale-independent, i.e., if all the data points are scaled by the same factor less than 1, then $CC(\mathcal{T})$ is unchanged.*

Proof. The proof can be found in Appendix F. □

2.3. Setup for accuracy analysis

The setup for accuracy analysis is as follows.

Definition 2.10. If $f : D \to \mathbb{R}^K$ is a continuous mapping, then the mapping

$$\hat{f} : x \mapsto \frac{e^{f(x)}}{\sum_i e^{f_i(x)}}$$

is still continuous, where $f_i(x)$ represents the ith component of $f(x)$, and $e^{f(x)} \in \mathbb{R}^K$ is the exponential function applied componentwise to $f(x)$. We have $\hat{f}_i(x) > 0$, and $\sum_i \hat{f}_i(x) = 1$. For convenience, we directly consider the case that $f_i(x) > 0$ and $\sum f_i(x) = 1$, and we call such mapping the normalized continuous positive mapping.

Remark 2.11. A neural network with softmax nonlinearity is a normalized continuous positive mapping.

Different from the accuracy usually used in classification problems, we define a stronger accuracy called c-accuracy as follows.

Definition 2.12 (*c-accuracy at x*)**.** Let f be a normalized continuous positive mapping. For $0.5 \le c < 1$, we say that f is c-accurate at point x if

$$\exists i_{max} \in tag(x),\ s.t.\, f_{i_{max}}(x) > c.$$

Definition 2.13 (*c-accuracy on D*)**.** Let f be a normalized continuous positive mapping. The c-accuracy of f on a sample space D is defined as

$$p_c(f) := \frac{\mu(H_c^f)}{\mu(D)} = \mu(H_c^f),$$

where $H_c^f := \{x \in D | f \text{ is } c\text{-accurate at } x\}$.

Definition 2.14 (*c-accuracy on $\mathcal{T}$*)**.** Let f be a normalized continuous positive mapping. The c-accuracy of f on a data set $\mathcal{T}$ is defined as

$$p_c^{\mathcal{T}} := \frac{\rho_{\widetilde{\mathcal{T}}_c}}{\rho_{\mathcal{T}}},$$

where $\widetilde{\mathcal{T}}_c := \{x \in \mathcal{T} | f \text{ is } c\text{-accurate at } x\}$, and $\rho_{\widetilde{\mathcal{T}}_c}$ and $\rho_{\mathcal{T}}$ are the TC of $\widetilde{\mathcal{T}}_c$ and $\mathcal{T}$, respectively.

Definition 2.15 (*Expected Accuracy*)**.** Let f be a normalized continuous positive mapping. The expected accuracy of f on a sample space D is defined as

$$p(f) := \frac{\mu(H^f)}{\mu(D)} = \mu(H^f),$$

where $H^f := \{x \in D | \exists i_{max} \in tag(x),\ s.t.\, f_{i_{max}}(x) > f_i(x)\ \forall 1 \le i \le K,\ i \ne i_{max}\}$.

We note that the c-accuracy of f on D represents the expected c-accuracy, and the c-accuracy of f on $\mathcal{T}$ represents the empirical c-accuracy.

Finally, we define a non-decreasing function δ_f to describe the smoothness of f.

Definition 2.16 (*Smoothness*)**.** Let $f : D \to \mathbb{R}^K$ be a continuous mapping. Then f is uniformly continuous due to the compactness of D, i.e.

$$\forall \epsilon > 0,\ \exists \delta > 0,\ s.t.\ \|f(x) - f(y)\|_\infty < \epsilon \text{ when } \|x - y\| < \delta.$$

We denote the supremum of δ satisfying the above requirement by $\delta_f(\epsilon)$. It is easy to see that $\delta_f(\epsilon)$ is equal to the inverse of the modulus of continuity of f.

For low dimensional problems, we can directly compute δ_f by brute force. However, for high dimensional problems, it is intractable to compute δ_f, and thus we give the following lower bound of δ_f for a fully-connected ReLU-network f with softmax as the activation function in the last layer, which is also the main network structure considered through this work.

A fully-connected neural network is defined as follows:

$$\phi_i(x) = \sigma(W_i x + b_i),\ 1 \le i \le l - 1,$$

$$f(x) = softmax(W_l(\phi_{l-1} \circ \cdots \circ \phi_1(x)) + b_l),$$

$$W_i \in \mathbb{R}^{n_i \times n_{i-1}},\ b_i \in \mathbb{R}^{n_i},\ 1 \le i \le l,$$

where n_i is the number of neurons in the layer i ($n_0 = d$ and $n_l = K$), and σ is the activation function. Then for the ReLU activation function, we have

$$\delta_f(\epsilon) \ge \frac{\epsilon}{Lip(f)} \ge \frac{\epsilon}{\|W_1\|_2 \cdots \|W_l\|_2 \cdot Lip(softmax)}, \tag{3}$$

where $\|\cdot\|_2$ is the spectral norm, $Lip(f)$ and $Lip(softmax)$ represent the Lipschitz constants of f and $softmax$ mapping from $(D, \|\cdot\|_2)$ to $(\mathbb{R}^K, \|\cdot\|_\infty)$, respectively. $Lip(softmax)$ is a constant less than $1/2$, and thus is ignored in our numerical examples. We note that although the lower bound of $\delta_f(\epsilon)$ depends exponentially on the neural net depth, $\delta_f(\epsilon)$ itself does not necessarily scale exponentially with the network depth.

3. Lower bounds for the expected accuracy

In this section, we present a theoretical analysis of the lower bound for the expected accuracy as well as an upper bound for the expected error.

Proposition 3.1. *Let f be a normalized continuous positive mapping. Suppose that $\mathcal{T}$ is a single label training set, i.e. $tag(\mathcal{T}) \subseteq \{\{1\}, \{2\}, \ldots, \{K\}\}$. For any $0.5 \le c_2 < c_1 \le 1$, we have*

$$p_{c_2}(f) \ge 1 - \frac{\sqrt{d}}{\delta}(1 - p_{c_1}^{\mathcal{T}} \rho_{\mathcal{T}}),$$

where $\delta = \min(\delta_0, \delta_f(c_1 - c_2))$.

Proof. The proof can be found in Appendix G. □

Proposition 3.1 shows that the expected c_2-accuracy of f can be bounded by the empirical c_1-accuracy and the TC of the training set. We can see that $p_{c_2}(f)$ tends to 1 when $\rho_{\mathcal{T}}$ and $p_{c_1}^{\mathcal{T}}$ tend to 1. Next we derive a bound for the accuracy by taking into account the loss function.

Theorem 3.2 (*Lower Bound of c-accuracy*)**.** *Let f be a normalized continuous positive mapping. Suppose that $\mathcal{T} = \{x_1, \ldots, x_n\}$ is a single label training set, and $tag(x_i) = \{k_i\}$. For any $0.5 \le c < 1$, if the maximum cross entropy loss*

$$L_f^{max} := \max_{1 \le i \le n} \ell(f(x_i), k_i) = -\min_{1 \le i \le n} \ln(f_{k_i}(x_i)) < -\ln c,$$

then we have

$$p_c(f) \ge 1 - \frac{\sqrt{d}}{\delta}(1 - \rho_{\mathcal{T}}),$$

where ℓ is the cross entropy loss that $\ell(f(x), k) = -\ln(f_k(x))$, $\delta = \min(\delta_0, \delta_f(e^{-L_f^{max}} - c))$, and δ_0 is defined in Proposition 2.3.

Proof. The proof can be found in Appendix H. □

Theorem 3.2 reveals that the expected accuracy is related to the total cover $\rho_{\mathcal{T}}$, separation gap δ_0, neural network smoothness δ_f, and loss value L_f^{max}. We will show numerically in Section 4 that $\delta_f(e^{-L_f^{max}} - c)$ increases first and then decreases during the training of neural networks. The following theorem states that the maximum value of $\delta_f(e^{-L_f^{max}} - c)$ is bounded by the empirical separation gap.

Theorem 3.3 (*Empirical Separation Gap*). *Let f be a normalized continuous positive mapping. Suppose that $\mathcal{T} = \{x_1, \dots, x_n\}$ is a single label training set. For any $0.5 \le c < 1$, if $L_f^{max} < -\ln c$, then we have*

$$\delta_f(e^{-L_f^{max}} - c) \le \delta_{\mathcal{T}}/2,$$

where

$$\delta_{\mathcal{T}} := \min_{tag(x_i) \ne tag(x_j)} \|x_i - x_j\| \ge \delta_0$$

is called the empirical separation gap, i.e., the smallest distance between two differently labeled training points.

Proof. The proof can be found in Appendix I. □

Besides the upper bound, the lower bound of δ_f is also important to the accuracy. We have observed that in practice the low bound of δ_f exists (Figs. 5–8), which indicates the existence of κ in the following theorem. Based on this observation, we have the following theorem for the accuracy.

Theorem 3.4 (*Lower Bound of Accuracy*). *Assume that there exists a constant $\kappa > 0$, such that*

$$\kappa\delta_0 \le \kappa\delta_{\mathcal{T}} \le \delta_f(e^{-L_f^{max}} - 0.5) \le \delta_{\mathcal{T}}/2 \tag{4}$$

holds for any single label training set $\mathcal{T}$ and a corresponding suitable trained network f on $\mathcal{T}$ such that $L_f^{max}(\mathcal{T}) < \ln 2$, then we have the following conclusions for the expected accuracy $p(f)$ and the expected error $\mathcal{E}(f) = 1 - p(f)$:

(i) with the same condition of Theorem 3.2, $p(f) \ge 1 - \frac{\sqrt{d}}{\delta}(1 - \rho_{\mathcal{T}})$,

(ii) $\mathcal{E}(f) \le \frac{\sqrt{d}\cdot(1-\rho_{\mathcal{T}})}{\min(\delta_0, \kappa\delta_{\mathcal{T}})} = \alpha(\mathcal{T}) \cdot CC(\mathcal{T})$,

(iii) $\lim_{\rho_{\mathcal{T}} \to 1} p(f) = 1$,

where $\delta = \min(\delta_0, \delta_f(e^{-L_f^{max}} - 0.5))$, and $\alpha(\mathcal{T}) = \frac{\sqrt{d}\cdot CD(\mathcal{T})}{\min(\delta_0, \kappa\delta_{\mathcal{T}})}$.

Proof. (i) is the conclusion of Theorem 3.2. The proof of (ii) and (iii) can be found in Appendix J. □

Remark 3.5. We have the following remarks for this main theorem:

- In (ii), we rewrite $\frac{\sqrt{d}\cdot(1-\rho_{\mathcal{T}})}{\min(\delta_0, \kappa\delta_{\mathcal{T}})}$ to emphasize $CC(\mathcal{T})$ by introducing a coefficient term $\alpha(\mathcal{T})$. Although this seems artificial, our numerical experiments empirically show that in practice the linear scaling between $\mathcal{E}(f)$ and $CC(\mathcal{T})$ is indeed satisfied.
- This theorem is not specific to neural networks, but rather holds for any trained model f satisfying the assumptions required in the theorem. While the assumptions are not true for a general machine learning algorithm, we show numerically that in practice neural networks satisfy the assumptions.
- The current proof of the theorem relies on the assumption of the maximum cross entropy L_f^{max}, which could be hard to satisfy in practice, since it will be large even if only a single training sample is misclassified. However, the assumption of the maximum cross entropy is possible to be relaxed according to our experiments.

Here, the cover complexity $CC(\mathcal{T})$ consists of two parts, one represents the richness of the whole training set while the other part describes the degree of separation between different labeled subsets. As for $\alpha(\mathcal{T})$, both the denominator and numerator seem to have a positive correlation with respect to separation level. What we wish is that $\alpha(\mathcal{T})$ is almost close to a constant with high probability and the expected error $\mathcal{E}(f)$ is mainly determined by $CC(\mathcal{T})$, which approximately represents the complexity level of the data set. We will provide more information in detail in the section concerning the numerical results.

4. Numerical results

In this section, we use numerical simulations to test the accuracy of neural networks in terms of the data distribution (cover complexity), and neural network smoothness. In addition, we study the effects of the network size and training dataset size on the smoothness. The codes are published in GitHub (https://github.com/jpzxshi/generalization).

4.1. Data distribution

In this subsection, we explore how $CC(\mathcal{T})$ affects the expected error $\mathcal{E}(f)$. In our experiments, we test several data sets, including MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky & Hinton, 2009), CIFAR-100 (Krizhevsky & Hinton, 2009), COIL-20 (Nene et al., 1996b), COIL-100 (Nene et al., 1996a), SVHN (Netzer et al., 2011). In addition to the original data set, we also create some variants: (1) the images of gray color, (2) the images extracted from a convolutional layer after training the original data set using a convolutional neural network (CNN), (3) combine several categories into one category to reduce the number of total categories, see Table 1 and details in Appendix K.

For a training data set $\mathcal{T}$, we estimate $h_{\mathcal{T}}^{\mu}(r)$ by the proportion of the test data points within the balls with radius r centered at training data points, i.e.,

$$h_{\mathcal{T}}^{\mu}(r) \approx \frac{\text{\# Test points within radius-}r\text{ balls of training points}}{\text{\# Test data points}},$$

and then $\rho_{\mathcal{T}}$ is obtained by Definition 2.5. Similarly, we estimate $CD(\mathcal{T})$ and then compute $CC(\mathcal{T})$. Next for each data set, we train fully-connected neural networks with different hyperparameters, and record the best error we observed, see the details in Appendix K. The cover complexity and the best error for each data set are shown in Table 1.

These data sets are divided into three groups according to their output dimensions. For each group of the same output dimension, the error is almost linearly correlated with $CC(\mathcal{T})$, see Fig. 3A, regardless of the input dimension. In addition, we find that all the cases collapse into a single line when normalizing the error $\mathcal{E}(f)$ by a factor of $1/\sqrt{K}$, see Fig. 3B.

It is noteworthy that the $CC(\mathcal{T})$ of convolutional variants of data sets is much smaller than that of the original data sets, and hence the expected accuracy increases. The results confirm the importance of data distribution.

Next, we consider the most difficult data set, i.e., data with random labels. We choose MNIST and then assign each image a random label. We repeat this process 50 times, and compute each $CC(\mathcal{T})$. The distribution of $|CC(\mathcal{T})|$ is shown in Fig. 4. The smallest $|CC(\mathcal{T})|$ is $\sim$300, which is much larger than that of the original data sets with $CC(\mathcal{T}) < 20$. This extreme example again confirms that $CC(\mathcal{T})$ is a proper measure of the difficulty of classifying a data set.

Table 1
Cover complexity $CC(\mathcal{T})$, best error $\mathcal{E}(f)$, and normalized error $\frac{\mathcal{E}(f)}{\sqrt{K}}$ of different data sets. Different variants of data sets are used, including the original RGB or gray images (Original), gray images (Gray), and images extracted from a CNN (Conv). Images in CIFAR-100 have two variants: 100 categories (fine) and 20 categories (coarse). See Appendix K for details.

Data set	Variants	Input dim (d)	Output dim (K)	$\rho_{\mathcal{T}}$	$CD(\mathcal{T})$	$CC(\mathcal{T})$	$\mathcal{E}(f)$	$\frac{\mathcal{E}(f)}{\sqrt{K}}$
MNIST	Original	784	10	.8480	.1053	1.442	.01	.0032
CIFAR-10	Original	3072	10	.8332	.0163	10.23	.45	.1423
CIFAR-10	Gray	1024	10	.8486	.0125	12.11	.53	.1676
CIFAR-10	Conv	1024	10	.9505	.0094	5.280	.18	.0569
SVHN	Original	3072	10	.9034	.0076	12.68	.49	.1550
SVHN	Gray	1024	10	.9117	.0084	10.48	.56	.1771
SVHN	Conv	1024	10	.9632	.0123	2.995	.23	.0727
CIFAR-100	Original (coarse)	3072	20	.8337	.0185	9.012	.62	.1386
CIFAR-100	Gray (coarse)	1024	20	.8541	.0132	11.08	.72	.1610
CIFAR-100	Conv (coarse)	1024	20	.9626	.0070	5.326	.40	.0894
COIL-20	Original	16 384	20	.9176	.2385	.3453	.03	.0067
CIFAR-100	Original (fine)	3072	100	.8337	.0270	6.149	.73	.0730
CIFAR-100	Gray (fine)	1024	100	.8541	.0198	7.380	.81	.0810
CIFAR-100	Conv (fine)	1024	100	.9457	.0136	4.000	.52	.0520
COIL-100	Original	49 152	100	.9430	.1944	.2930	.01	.0010

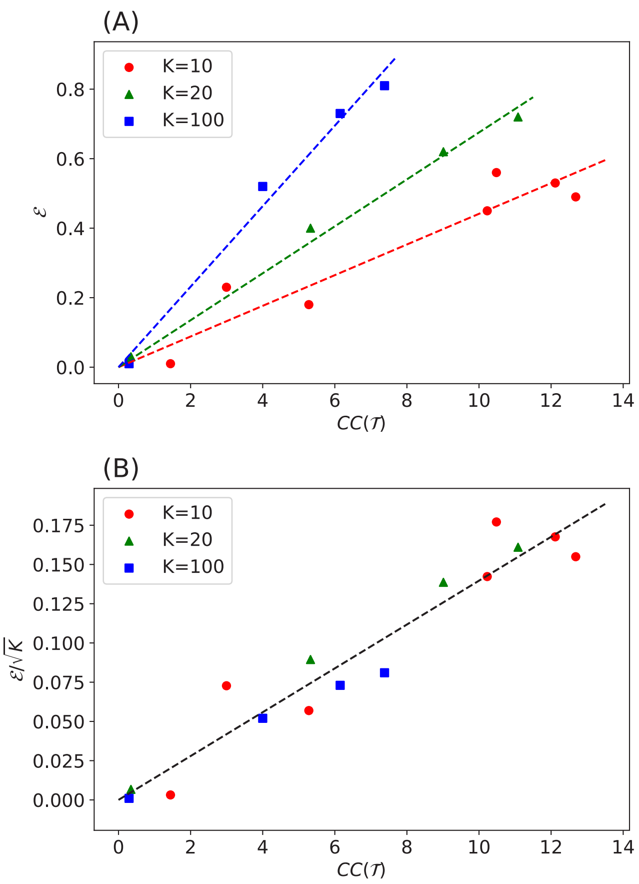

Fig. 3. Relationship between cover complexity and error achieved by fully-connected neural networks. (**A**) Linear relationship between cover complexity $CC(\mathcal{T})$ and error $\mathcal{E}(f)$ for different data sets with different category number K. The coefficients of determination of the linear regression (with the constraint that the line must pass through the origin) are $R^2 \approx 0.89, 0.99, 0.98$ for $K = 10, 20, 100$, respectively. (**B**) Linear relationship between cover complexity $CC(\mathcal{T})$ and normalized error $\frac{\mathcal{E}(f)}{\sqrt{K}}$ for different data sets. Red, green and blue points represent data sets with output dimension of 10, 20 and 100, respectively. The line is fitted as $\frac{\mathcal{E}(f)}{\sqrt{K}} \approx 0.014CC(\mathcal{T})$, and the coefficient of determination is $R^2 \approx 0.92$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

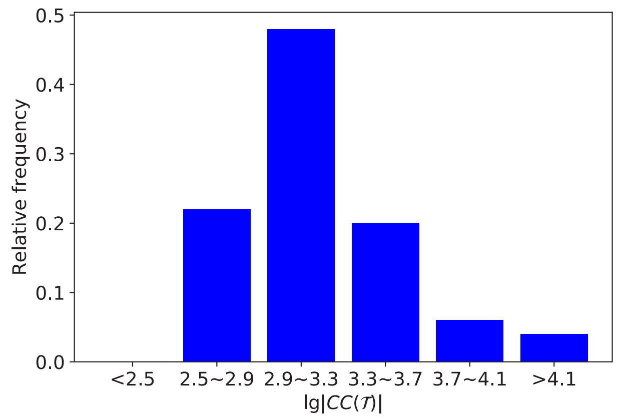

Fig. 4. Distribution of $CC(\mathcal{T})$ of randomly labeled MNIST data set. We assign each image in MNIST a random label and compute its $CC(\mathcal{T})$. The smallest $|CC(\mathcal{T})|$ is ~300. The distribution is obtained from 50 random data sets.

4.2. Neural network smoothness

In this subsection, we will investigate the relationship between the neural network smoothness $\delta_f(e^{-L_f^{max}} - c)$ and the accuracy, and the effects of network size (depth and width) on the smoothness. We first show results for one- and two-dimensional problems, where $\delta_f(e^{-L_f^{max}} - c)$ can be computed accurately by brute force. Subsequently, we consider the high dimensional setting of the MNIST data set, where we estimate $\delta_f(e^{-L_f^{max}} - c)$ by Eq. (3).

4.2.1. One- and two-dimensional problems

We first consider a one-dimensional case and a two-dimensional case. For the one-dimensional case, we choose the sample space $D = [0, 1]$, $K = 2$, and the ideal label function as

$$tag(x) = \begin{cases} \{1\} & \text{if } x \in [0, 0.5 - \frac{\delta_0}{2}] \\ \{1, 2\} & \text{if } x \in (0.5 - \frac{\delta_0}{2}, 0.5 + \frac{\delta_0}{2}) \\ \{2\} & \text{if } x \in [0.5 + \frac{\delta_0}{2}, 1], \end{cases}$$

with separation gap $\delta_0 = 0.1$. We use n equispaced points ($n \geq 4$ is an even number) on $D \setminus (0.5 - \frac{\delta_0}{2}, 0.5 + \frac{\delta_0}{2})$ as the training set,

Table 2
Comparison between c-accuracy $p_c(f)$ with $c = 0.5$ and the lower bounds for different training set sizes for the one-dimensional problem. Here δ is indicated in Theorem 3.2. The neural network is trained for 1000 iterations by the Adam optimizer.

n	L_f^{max}	$\rho_\mathcal{T}$	δ	$p_c(f)$	$h_\mathcal{T}(\delta)$	$1-\frac{\sqrt{d}}{\delta}(1-\rho_\mathcal{T})$
10	.285	.972	.045	1.0	0.80	0.38
20	.246	.988	.041	1.0	1.00	0.69
40	.182	.994	.041	1.0	1.00	0.85
80	.127	.997	.038	1.0	1.00	0.92

i.e., $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$, where

$$\mathcal{T}_1 = \left\{0, \frac{1-\delta_0}{n-2}, \frac{1-\delta_0}{n-2}\cdot 2, \dots, \frac{1}{2} - \frac{\delta_0}{2}\right\},$$

$$\mathcal{T}_2 = \left\{\frac{1}{2} + \frac{\delta_0}{2}, \dots, 1 - \frac{1-\delta_0}{n-2}, 1\right\}.$$

We choose 10 000 equispaced points on D as the test data.

For the two-dimensional case, we choose the sample space $D = [0, 1]^2$, $K = 2$, and the ideal label function as

$$tag(\mathbf{x}) = \begin{cases} \{1\} & \text{if } \|\mathbf{x} - (0.5, 0.5)\| \leq 0.4 - \frac{\delta_0}{2} \\ \{2\} & \text{if } \|\mathbf{x} - (0.5, 0.5)\| \geq 0.4 + \frac{\delta_0}{2} \\ \{1, 2\} & \text{otherwise,} \end{cases}$$

with $\delta_0 = 0.1$. For the training set, we first choose $n = m^2$ equispaced points, i.e., $\mathcal{T} = \{0, \frac{1}{m-1}, \frac{2}{m-1}, \dots, \frac{m-2}{m-1}, 1\}^2$, and then remove the points with label $\{1, 2\}$ to ensure that all samples are of single label. We choose 1000^2 equispaced points on D as the test data.

In our experiments, we use a 3-layer fully-connected NN with ReLU activation and 30 neurons per layer. The neural network is trained for 1000 iterations by the Adam optimizer (Kingma & Ba, 2015) for the one-dimensional problem, and 2000 iterations for the two-dimensional problem. For the one-dimensional problem, the c-accuracy $p_c(f)$ with $c = 0.5$ and lower bounds for different numbers of training points are listed in Table 2. We can see that the bounds become tighter when n is larger.

During the training process of the neural network, the test loss first decreases and then increases, while δ_f first increases and then decreases, see Fig. 5A for the one-dimensional problem ($n = 20$) and Fig. 5B for the two-dimensional problem ($n = 400$). δ_f is bounded by $\delta_\mathcal{T}/2$, as proved in Theorem 3.3. We also observe that the trends of test loss and δ_f coincide, and thus we should stop the training when δ_f begins to decrease to prevent overfitting.

4.2.2. High-dimensional problem

In the high-dimensional problem of MNIST, we consider the average loss L_f instead of the maximum loss L_f^{max}, which is very sensitive to extreme points. As shown in Eq. (3), we use the following quantity to bound $\delta_f(e^{-L_f} - c)$:

$$\Delta_f = \frac{e^{-L_f} - c}{\|W_1\|_2 \cdots \|W_l\|_2}.$$

Because we use the c-accuracy to approximate the true accuracy, for the classification problems with two categories, $p_{0.5}(f)$ and $p(f)$ are equivalent. However, they are not equal for problems with more than two categories, where the best c depends on the properties of the data set, such as the easiness of learning to classify the data set. If the data set is easy to classify, such as MNIST, the best c should be close to 1. In our example, we choose $c = 0.9$. We train MNIST using a 3-layer fully-connected NN with ReLU activation and 100 neurons per layer for 100 epochs. In Fig. 6, we can also see the consistency between the test loss and neural network smoothness, as we observed in the low-dimensional problems.

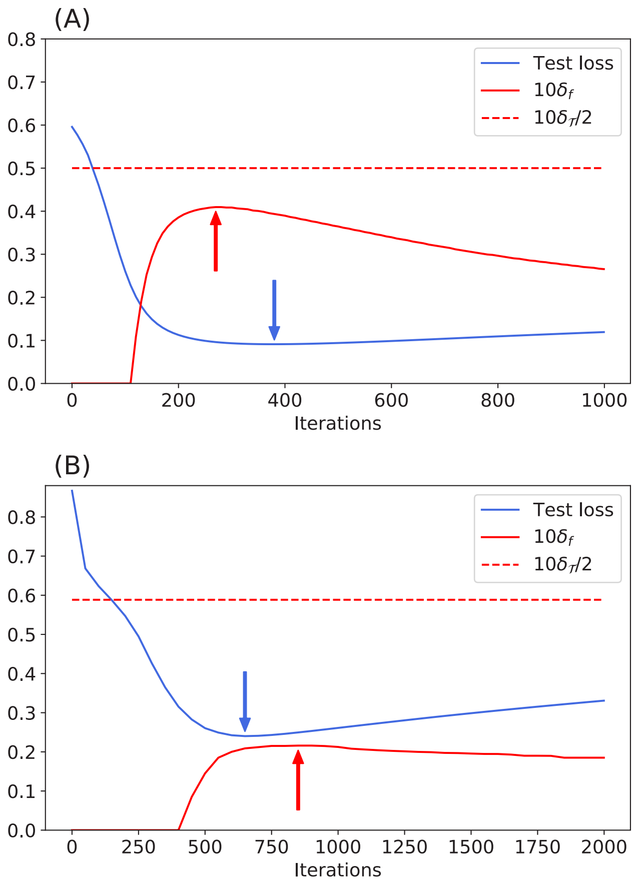

Fig. 5. Consistency between the test loss and neural network smoothness $\delta_f(e^{-L_f^{max}} - \frac{1}{2})$ during the training process of the neural network. (**A**) One-dimensional problem of $n = 20$. (**B**) Two-dimensional problem of $n = 400$. δ_f is bounded by $\delta_\mathcal{T}/2$. The arrows indicate the minimum of the test loss and the maximum of δ_f.

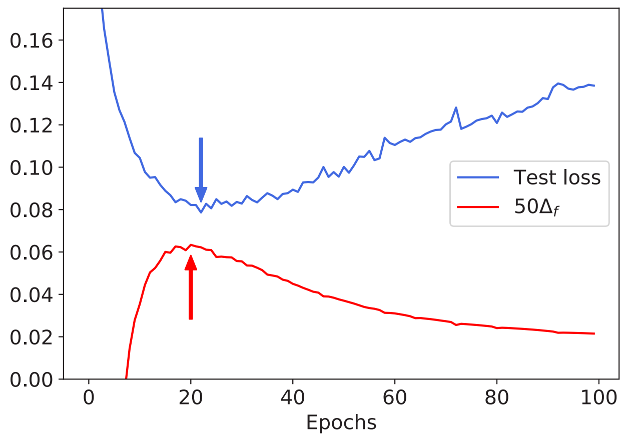

Fig. 6. Consistency between test loss and neural network smoothness during the training of the neural network for MNIST. The arrows indicate the minimum of the test loss and the maximum of Δ_f.

4.2.3. Effects of the network size and training dataset size on the smoothness

We have demonstrated that network smoothness $\delta_f(e^{-L_f^{max}} - c)$ is an important factor to the accuracy. Next, we investigate the effects of network size (depth and width) on the smoothness for

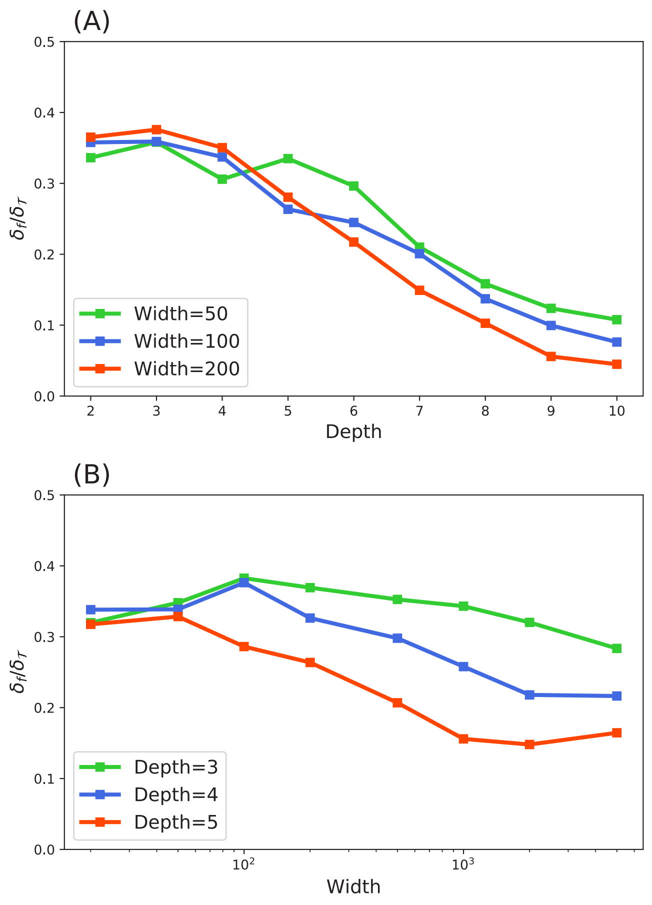

Fig. 7. Effects of network depth and width on the normalized smoothness $\delta_f/\delta_\mathcal{T}$. Recall that $\delta_f = \delta_f(e^{-L_f^{max}} - \frac{1}{2})$. (**A**) The $\delta_f/\delta_\mathcal{T}$ decreases fast when the network depth increases. (**B**) The $\delta_f/\delta_\mathcal{T}$ decreases relatively slow when the network width increases. The results are averaged from 50 independent experiments.

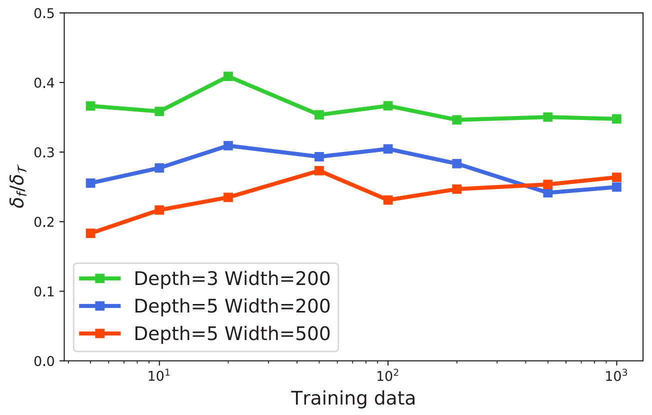

Fig. 8. Effects of the training data size on the normalized $\delta_f/\delta_\mathcal{T}$. Recall that $\delta_f = \delta_f(e^{-L_f^{max}} - \frac{1}{2})$. The $\delta_f/\delta_\mathcal{T}$ is insensitive to the training data size, and is bounded from below by a positive constant. The results are averaged from 50 independent experiments.

binary classification problems, which are explained as follows. We consider the one-dimensional sample space $D = [0, 1]$, and choose n equispaced points on D as the training data locations. To avoid the effects of the choice of target true functions, we always repeat experiments with different target functions, and in each experiment we generate a random target function. Specifically, to generate a random target function, we first sample two random functions $g_1(x)$ and $g_2(x)$ from a Gaussian process with the radial basis function kernel of a length scale 0.2, and then assign a point x as category 1 if $g_1(x) > g_2(x)$, otherwise assign this point as category 2. When training neural networks, we monitor the value of $\delta_f(e^{-L_f^{max}} - c)$ and stop the training once δ_f begins to decrease as shown in Figs. 5 and 6. We first choose the dataset size $n = 10$, and we show that the normalized smoothness $\delta_f/\delta_\mathcal{T}$ decreases as the network depth or width increases (Fig. 7). We also show that the effects of depth is more significant than that of width.

Our main theorem (Theorem 3.4) requires the assumption (Eq. (4)), which would not be true for a general machine learning algorithms. Here, we verify this assumption for neural networks by numerical experiments. Specifically, we train a fully-connected neural networks using training datasets of different size n. We show that $\delta_f/\delta_\mathcal{T}$ is insensitive to training dataset size, and is always bounded from below by a positive constant κ (Fig. 8). This result reveals that the neural networks would fit a dataset in a relatively smooth way during the training process.

5. Discussion

When neural networks are used to solve classification problems, we expect that the accuracy is dependent on some properties of the data set. It is still quite surprising, however, that there is a linear relationship between the accuracy and the cover complexity of the data set, as we have seen in Section 4.1. Theorem 3.4(ii) provides an upper bound of the error, but a lower bound is missing. To fully explain this observation, two conjectures of the learnability of fully-connected neural networks are proposed: when a neural network f is trained on a data set $\mathcal{T}$ in such a way that $L_f^{max}(\mathcal{T}) < \ln 2$ and $\delta_f(e^{-L} - c) \asymp \delta_\mathcal{T}$, then we have

- $\mathcal{E}(f) \approx c(K) \cdot CC(\mathcal{T})$, where $c(K)$ is a constant depending only on K.
- $\frac{\mathcal{E}(f)}{\sqrt{K}} \approx c \cdot CC(\mathcal{T})$, where $c \approx 0.014$ is a constant.

On the other hand, the theoretical and numerical results provide a better understanding of the generalization of neural network from the training procedure. The smoothness $\delta_f(e^{-L} - c)$ of neural networks plays a key role, where L is the maximum cross entropy loss L_f^{max} or the average cross entropy loss L_f. We can see that:

- $\delta_f(e^{-L} - c)$ depends on both the regularity of f and the loss value L (which also depends on f). Large $\delta_f(e^{-L} - c)$ requires good regularity and large $e^{-L} - c$, i.e., small L. However, small L could correspond to bad regularity of f. Thus, there is a trade-off between the loss value L and the regularity of f.
- Due to this trade-off, $\delta_f(e^{-L} - c)$ increases first and then decrease during the training process. Hence, we should not optimize neural networks excessively. Instead, we should stop the training early when $\delta_f(e^{-L} - c)$ begins to decrease, which leads to another "early stopping" strategy to prevent overfitting.

We also note that the lower bound of $\delta_f(e^{-L} - c)$ in Eq. (3) relates to the norm of weight matrices of neural networks:

$$\delta_f(e^{-L} - c) \gtrsim \frac{e^{-L} - c}{\|W_1\|_2 \cdots \|W_l\|_2}.$$

There have been some works to study the norm-based complexity of neural networks (see the Introduction), and these bounds typically scale with the product of the norms of the weight matrices, e.g., (Neyshabur, Bhojanapalli, McAllester et al., 2017)

$$\frac{1}{\gamma_{margin}^2}\prod_{i=1}^{l} h_i\|W_i\|_2^2,$$

where h_i and W_i are the number of nodes and the weight matrix in layer i of a network with l-layers, and γ_{margin} is the margin quantity, which describes the goodness of fit of the trained network to the data. The product of the matrix norms depends exponentially on the depth, while some recent works show that the generalization bound could scale polynomially in depth under some assumptions (Nagarajan & Kolter, 2019a; Wei & Ma, 2019). The exploration of the dependence of $\delta_f(e^{-L}-c)$ on depth is left for future work.

6. Conclusion

In this paper, we study the generalization error of neural networks for classification problems in terms of the data distribution and neural network smoothness. We first establish a new framework for classification problems. We introduce the *cover complexity* (CC) to measure the difficulty of learning a data set, an accuracy measure called *c-accuracy* which is stronger than the standard classification accuracy, and *the inverse of the modulus of continuity* to quantify neural network smoothness. Subsequently, we derive a quantitative bound for the expected accuracy/error in Theorem 3.4, which considers both the cover complexity and neural network smoothness.

We validate our theoretical results by several data sets of images. Our numerical results demonstrate that CC is a reliable measure for the difficulty of learning to classify a data set. On the other hand, we observe a clear consistency between test loss and neural network smoothness during the training process. We also show that neural network smoothness decreases when the network depth and width increases, and the effects of depth is more significant than that of width, while the smoothness is insensitive to training dataset size.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the DOE PhILMs project (No. de-sc0019453), the AFOSR grant FA9550-17-1-0013, and the DARPA AIRA grant HR00111990025. The work of P. Jin and Y. Tang is partially supported by the Major Project on New Generation of Artificial Intelligence from MOST of China (Grant No. 2018AAA0101002), and the National Natural Science Foundation of China (Grant No. 11771438).

Appendix A. Example of topology

Example A.1. Given

$$T = 2^{\{1,2,3\}}\backslash\{\varnothing\} = \{\ \{1\},\ \{2\},\ \{3\},\ \{1,2\},\ \{1,3\},\ \{2,3\},\ \{1,2,3\}\ \},$$

$$\begin{aligned}U = \{\ &\{\{1\},\{1,2\},\{1,3\},\{1,2,3\}\},\ \{\{2\},\{2,1\},\{2,3\},\{1,2,3\}\},\\ &\{\{3\},\{3,1\},\{3,2\},\{1,2,3\}\},\ \{\{1,2\},\{1,2,3\}\},\\ &\{\{1,3\},\{1,2,3\}\},\ \{\{2,3\},\{1,2,3\}\},\ \{\{1,2,3\}\}\ \},\end{aligned}$$

then τ_T is the topology generated by U.

In this example, $\{\{1\},\{1,2\},\{1,3\},\{1,2,3\}\}$ is an open set, since it consists of all elements containing label $\{1\}$, and $\{\{2,3\},\{1,2,3\}\}$ is also an open set with common part $\{2,3\}$. Besides open sets from base U, $\{\{1\},\{1,2\},\{1,3\},\{2,3\},\{1,2,3\}\}$ is still an open set as the union of the two shown above.

Appendix B. Proof of Proposition 2.3

Proof. We use the proof by contradiction. Assume that the result does not hold, then

$$\exists\{x_n\},\{y_n\}\subseteq D,\ s.t.\ \|x_n-y_n\|\to 0(n\to\infty),\ \text{and}\quad tag(x_n)\cap tag(y_n)=\varnothing.$$

Because D is compact, there exist $x^*\in D$ and a subsequence $\{x_{k_n}\}$ of $\{x_n\}$ such that $x_{k_n}\to x^*$. As $\|x_n-y_n\|\to 0$, also $y_{k_n}\to x^*$. Choose any $i_0\in tag(x^*)$, then there exists a sufficient large k_{n_0} such that $i_0\in tag(x_{k_{n_0}})$, $i_0\in tag(y_{k_{n_0}})$. Therefore $tag(x_{k_{n_0}})\cap tag(y_{k_{n_0}})\neq\varnothing$, which contradicts the assumption. □

Appendix C. Proof of Proposition 2.4

Proof. Let δ_0 be as defined in this proposition. For any two different points x, y with distance less than δ_0, either $tag(x) = tag(y)$, or at least one of the two is a full label point, in both cases $tag(x)\cap tag(y)\neq\varnothing$. For any $\delta>\delta_0$, according to the definition of δ_0, there exist two points x_0, y_0 satisfying

$$\|x_0-y_0\|<\delta,\quad tag(x_0)\cap tag(y_0)=\varnothing.$$

The two facts imply that δ_0 is the supremum of δ satisfying Proposition 2.3. □

Appendix D. Proof of Proposition 2.7

Proof. According to the definition,

$$\begin{aligned}\sqrt{d}\rho_{\mathcal{T}} &= \int_0^{\sqrt{d}} h_{\mathcal{T}}^{\mu}(t)dt\\ &= \int_0^{r} h_{\mathcal{T}}^{\mu}(t)dt + \int_r^{\sqrt{d}} h_{\mathcal{T}}^{\mu}(t)dt\\ &\le r\cdot h_{\mathcal{T}}^{\mu}(r) + (\sqrt{d}-r)\cdot 1\\ &= \sqrt{d}-r(1-h_{\mathcal{T}}^{\mu}(r)),\end{aligned}$$

thus

$$h_{\mathcal{T}}^{\mu}(r)\ge 1-\frac{\sqrt{d}}{r}(1-\rho_{\mathcal{T}}).\quad □$$

Appendix E. Estimate of total cover

In this section, we estimate the TC by the number of samples in the training set. The notations, such as D, d, μ, $\rho_{\mathcal{T}}$, as well as training set

$$\mathcal{T}=\{x_1,x_2,\dots,x_n\}\subseteq D$$

are the same as before. Note that samples in $\mathcal{T}$ are drawn according to μ. Before presenting the analysis, we first collect the following auxiliary notions and results (Definitions E.1–E.4, Theorem E.5) which are easily found in Mitzenmacher and Upfal (2017) (Definitions 14.1–14.3, Definition 14.5, and Theorem 14.8):

Definition E.1. A range space is a pair $(X,\mathcal{R})$ where:

1. X is a (finite or infinite) set of points;
2. $\mathcal{R}$ is a family of subsets of X, called ranges.

Definition E.2. Let $(X, \mathcal{R})$ be a range space and let $Y \subseteq X$. The projection of $\mathcal{R}$ on Y is

$$\mathcal{R}_Y = \{R \cap Y | R \in \mathcal{R}\}.$$

Definition E.3. Let $(X, \mathcal{R})$ be a range space. A set $Y \subseteq X$ is shattered by $\mathcal{R}$ if $|\mathcal{R}_Y| = 2^{|Y|}$. The Vapnik–Chervonenkis (VC) dimension of a range space $(X, \mathcal{R})$ is the maximum cardinality of a set $Y \subseteq X$ that is shattered by $\mathcal{R}$. If there are arbitrarily large finite sets that are shattered by $\mathcal{R}$, then the VC dimension is infinite.

Definition E.4. Let $(X, \mathcal{R})$ be a range space, and let $\mathcal{F}$ be a probability distribution on X. A set $N \subseteq X$ is an ϵ-net for X with respect to $\mathcal{F}$ if for any set $R \in \mathcal{R}$ such that $Pr_{\mathcal{F}}(R) \geq \epsilon$, the set R contains at least one point from N, i.e.,

$$\forall R \in \mathcal{R}, Pr_{\mathcal{F}}(R) \geq \epsilon \Rightarrow R \cap N \neq \varnothing.$$

Theorem E.5. *Let $(X, \mathcal{R})$ be a range space with VC dimension d_{vc} and let $\mathcal{F}$ be a probability distribution on X. For any $0 < \eta, \epsilon \leq 1/2$, there is an*

$$m = O\left(\frac{d_{vc}}{\epsilon} \ln \frac{d_{vc}}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\eta}\right)$$

such that a random sample from $\mathcal{F}$ of size greater than or equal to m is an ϵ-net for X with probability at least $1 - \eta$.

Now let

$$B_D = \{B(x, r) \cap D | x \in \mathbb{R}^d, r > 0\},$$

we first show (D, B_D) is a range space with VC dimension $d + 1$.

Lemma E.6. *The VC dimension of range space (D, B_D) is $d + 1$.*

Proof. The proof can be found in Dudley (1979). □

Set

$$B_D^\epsilon(r) = \{A \in B_D | A = B(x, s) \cap D, 0 < s \leq r, \mu(A) \geq \epsilon\},$$

and

$$\varrho(\epsilon) = \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \mu\left(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A\right) dr,$$

we have the following lemmas.

Lemma E.7. $\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A \subseteq D \cap \bigcup_{x_i \in \mathcal{T}} B(x_i, r)$ *when $\mathcal{T}$ is an ϵ-net for (D, B_D).*

Proof. For any $x \in \bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A$, we have $x \in A^* = B(x^*, s^*) \cap D$ for certain $s^* \leq \frac{1}{2}r$, with $\mu(A^*) \geq \epsilon$. Since $\mathcal{T}$ is an ϵ-net and $\mu(A^*) \geq \epsilon$, we know $\mathcal{T} \cap A^* \neq \varnothing$. Thus there exists $x_t \in \mathcal{T}$ such that $\|x_t - x^*\| < s^* \leq \frac{1}{2}r$. Therefore

$$\|x - x_t\| \leq \|x - x^*\| + \|x^* - x_t\| < \frac{1}{2}r + \frac{1}{2}r = r.$$

The above inequality shows that

$$x \in D \cap B(x_t, r) \subseteq D \cap \bigcup_{x_i \in \mathcal{T}} B(x_i, r). \quad \square$$

Lemma E.8. $\lim_{\epsilon \to 0} \varrho(\epsilon) = 1$.

Proof. For any positive decreasing sequence $\{\epsilon_i\}$ which satisfies $\lim_{i \to \infty} \epsilon_i = 0$, it leads to an increasing chain

$$\left(\bigcup_{A \in B_D^{\epsilon_i}(\frac{1}{2}r)} A\right) \subseteq \left(\bigcup_{A \in B_D^{\epsilon_{i+1}}(\frac{1}{2}r)} A\right), \quad \lim_{i \to \infty} \left(\bigcup_{A \in B_D^{\epsilon_i}(\frac{1}{2}r)} A\right) = \widetilde{D} \subseteq D,$$

where $\widetilde{D} = \bigcup_{A \in B_D^+(\frac{1}{2}r)} A$ for $B_D^+(\frac{1}{2}r) = \{A \in B_D | A = B(x, s) \cap D, 0 < s \leq \frac{1}{2}r, \mu(A) > 0\}$. Let us consider a series of open balls $\{B_i\}$ of radius at most $\frac{1}{2}r$ that cover D, and we divide them into two parts $\{B_i\} = \{B_i^+\} \cup \{B_i^0\}$ such that $\mu(B_i^+ \cap D) > 0$ and $\mu(B_i^0 \cap D) = 0$. Then $\mu(\widetilde{D}) \geq \mu(D \cap \bigcup_i B_i^+) \geq \mu(D) - \mu(D \cap \bigcup_i B_i^0) = 1 - 0 = 1$, and thus $\mu(\widetilde{D}) = 1$. Therefore, we have $\lim_{\epsilon \to 0} \mu(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A) = \mu(\widetilde{D}) = 1$.

Since

$$\lim_{\epsilon \to 0} \mu\left(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A\right) = 1 \text{ and } \mu\left(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A\right) \leq 1,$$

by dominated convergence theorem, we have

$$\begin{aligned}
\lim_{\epsilon \to 0} \varrho(\epsilon) &= \lim_{\epsilon \to 0} \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \mu\left(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A\right) dr \\
&= \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \lim_{\epsilon \to 0} \mu\left(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A\right) dr \\
&= \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} 1 dr \\
&= 1. \quad \square
\end{aligned}$$

According to the aforementioned lemmas, we deduce the following theorem.

Theorem E.9. *Let $\mathcal{T} = \{x_1, x_2, \ldots, x_n\}$ be the training set drawn according to μ, then for any $0 < \eta, \epsilon \leq \frac{1}{2}$, there exists an*

$$m = O\left(\frac{d+1}{\epsilon} \ln \frac{d+1}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\eta}\right)$$

such that

$$\rho_{\mathcal{T}} \geq \varrho(\epsilon)$$

holds with probability at least $1 - \eta$ when $n \geq m$. Note that $\varrho(\epsilon) \to 1$ when $\epsilon \to 0$.

Proof. Theorem E.5 shows that $\mathcal{T}$ is an ϵ-net for range space (D, B_D) with probability at least $1 - \eta$ when $n \geq m$. By Lemma E.7, we have

$$\begin{aligned}
\rho_{\mathcal{T}} &= \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \mu\left(D \cap \bigcup_{x_i \in \mathcal{T}} B(x_i, r)\right) dr \\
&\geq \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \mu\left(\bigcup_{A \in B_D^\epsilon(\frac{1}{2}r)} A\right) dr \\
&= \varrho(\epsilon). \quad \square
\end{aligned}$$

From this theorem, we know that a large number of samples lead to a sufficiently large $\rho_{\mathcal{T}}$ with a high probability.

In the previous sections, there is an assumption that every training point has only one single correct label, so we will naturally consider this special case in the sequel.

Denote

$$D_{sin} = \{x \in D | \; tag(x) \in \{\{1\}, \{2\}, \ldots, \{K\}\}\},$$

$$B_{D_{sin}} = \{B(x, r) \cap D_{sin} | x \in \mathbb{R}^d, r > 0\},$$

$$\mu_{sin}(A) = \frac{\mu(A \cap D_{sin})}{\mu(D_{sin})},$$

and $(D_{sin}, B_{D_{sin}})$ is a range space with VC dimension at most $d+1$. Let

$$\mathcal{T} = \{x_1, \ldots, x_n\} \subseteq D_{sin},$$

that is, the samples in $\mathcal{T}$ are drawn according to μ_{sin}. As before, denote

$$B^{\epsilon}_{D_{sin}}(r) = \{A \in B_D | A = B(x, s) \cap D, 0 < s \leq r, \mu_{sin}(A) \geq \epsilon\},$$

$$\varrho_{sin}(\epsilon) = \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \mu\left(\bigcup_{A \in B^{\epsilon}_{D_{sin}}(\frac{1}{2}r)} A\right) dr.$$

We have the following lemmas.

Lemma E.10. *$\bigcup_{A \in B^{\epsilon}_{D_{sin}}(\frac{1}{2}r)} A \subseteq D \cap \bigcup_{x_i \in \mathcal{T}} B(x_i, r)$ when $\mathcal{T}$ is an ϵ-net for $(D_{sin}, B_{D_{sin}})$.*

Lemma E.11. *$\lim_{\epsilon \to 0} \varrho_{sin}(\epsilon) = c_{sin}$, here*

$$c_{sin} = \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} \mu\left(\bigcup_{A \in C_{D_{sin}}(\frac{1}{2}r)} A\right) dr \geq \mu(D_{sin}),$$

$$C_{D_{sin}}(r) = \{A \in B_D | A = B(x, s) \cap D, 0 < s \leq r, \mu_{sin}(A) > 0\}.$$

From these two lemmas we deduce the following theorem.

Theorem E.12. *Let $\mathcal{T} = \{x_1, x_2, \ldots, x_n\}$ be the training set drawn according to μ_{sin}, then for any $0 < \eta, \epsilon \leq \frac{1}{2}$, there exists an*

$$m = O\left(\frac{d+1}{\epsilon} \ln \frac{d+1}{\epsilon} + \frac{1}{\epsilon} \ln \frac{1}{\eta}\right)$$

such that

$$\rho_{\mathcal{T}} \geq \varrho_{sin}(\epsilon)$$

holds with probability at least $1 - \eta$ when $n \geq m$. Note that $\varrho_{sin}(\epsilon) \to c_{sin}$ when $\epsilon \to 0$ for $c_{sin} \geq \mu(D_{sin})$.

The proofs for Lemmas E.10–E.11 and Theorem E.12 are very similar to those for Lemmas E.7–E.8 and Theorem E.9, respectively. We omit them here. It is noteworthy that c_{sin} is intuitively very close to 1, even equal to 1. At worst, c_{sin} is at least greater than $\mu(D_{sin})$ which may be quite large in practice, and the proof is similar to what we show in the Lemma in Appendix E.8.

Appendix F. Proof of Proposition 2.9

Proof. Let $\mathcal{T}$ be the training set and λ be a positive constant greater than 1, $\widetilde{\mathcal{T}} = \mathcal{T}/\lambda$, $\widetilde{\mu}(A) = \mu(\lambda A)$, then

$$\begin{aligned}
1 - \rho_{\widetilde{\mathcal{T}}} &= \frac{1}{\sqrt{d}} \int_0^{\sqrt{d}} (1 - h^{\widetilde{\mu}}_{\widetilde{\mathcal{T}}}(r)) dr \\
&= \frac{1}{\sqrt{d}} \int_0^{\infty} (1 - h^{\widetilde{\mu}}_{\widetilde{\mathcal{T}}}(r)) dr \\
&= \frac{1}{\sqrt{d}} \int_0^{\infty} (1 - h^{\mu}_{\mathcal{T}}(\lambda r)) dr \\
&= \frac{1}{\lambda\sqrt{d}} \int_0^{\infty} (1 - h^{\mu}_{\mathcal{T}}(r)) dr \\
&= (1 - \rho_{\mathcal{T}})/\lambda.
\end{aligned}$$

For the same reason,

$$\begin{aligned}
CD(\widetilde{\mathcal{T}}) &= \frac{1}{K(K-1)} \sum_{i \neq j} (1 - \rho(\widetilde{\mathcal{T}}_i, \widetilde{\mu}_j)) - \frac{1}{K} \sum_i (1 - \rho(\widetilde{\mathcal{T}}_i, \widetilde{\mu}_i)) \\
&= \frac{1}{\lambda K(K-1)} \sum_{i \neq j} (1 - \rho(\mathcal{T}_i, \mu_j)) - \frac{1}{\lambda K} \sum_i (1 - \rho(\mathcal{T}_i, \mu_i)) \\
&= CD(\mathcal{T})/\lambda,
\end{aligned}$$

therefore

$$CC(\widetilde{\mathcal{T}}) = CC(\mathcal{T}). \quad \square$$

Appendix G. Proof of Proposition 3.1

Proof. Denote $\widetilde{\mathcal{T}}_{c_1} = \{x_i \in \mathcal{T} \mid f \text{ is } c_1\text{-accurate at } x_i\}$, $\delta = \min(\delta_0, \delta_f(c_1 - c_2))$. For any $x_i \in \widetilde{\mathcal{T}}_{c_1}$, choose $k_i \in \{1, \ldots, K\}$ such that $tag(x_i) = \{k_i\}$. From the definition of $\widetilde{\mathcal{T}}_{c_1}$ we know that $f_{k_i}(x_i) > c_1$.

For any $x \in B(x_i, \delta) \cap D$, from Proposition 2.3 we know that $tag(x) \cap tag(x_i) \neq \varnothing$, and hence $k_i \in tag(x)$. On the other hand, $\|f(x) - f(x_i)\|_\infty < c_1 - c_2$, so we have $|f_{k_i}(x) - f_{k_i}(x_i)| < c_1 - c_2$, therefore

$$f_{k_i}(x) > f_{k_i}(x_i) - (c_1 - c_2) > c_1 - (c_1 - c_2) = c_2,$$

which means that f is c_2—accurate at x, that is to say

$$H^f_{c_2} \supseteq D \cap \bigcup_{x_i \in \widetilde{\mathcal{T}}_{c_1}} B(x_i, \delta).$$

Then

$$\begin{aligned}
p_{c_2}(f) &= \mu(H^f_{c_2}) \\
&\geq \mu\left(D \cap \bigcup_{x_i \in \widetilde{\mathcal{T}}_{c_1}} B(x_i, \delta)\right) \\
&= h^{\mu}_{\widetilde{\mathcal{T}}_{c_1}}(\delta) \\
&\geq 1 - \frac{\sqrt{d}}{\delta}(1 - \rho_{\widetilde{\mathcal{T}}_{c_1}}) \\
&= 1 - \frac{\sqrt{d}}{\delta}(1 - p^{\mathcal{T}}_{c_1} \rho_{\mathcal{T}}).
\end{aligned}$$

The second inequality can be derived from Proposition 2.7. $\square$

Appendix H. Proof of Theorem 3.2

Proof. Define $\delta := \min(\delta_0, \delta_f(e^{-L^{max}_f} - c))$. Note that if $x_i \in \mathcal{T}$, and $x \in B(x_i, \delta) \cap D$, then $\|x - x_i\| < \delta_0$, and hence $tag(x) \cap tag(x_i) \neq \varnothing$, so that $k_i \in tag(x)$.

Because of

$$\|x - x_i\| < \delta_f(e^{-L^{max}_f} - c),$$

we have

$$\|f(x) - f(x_i)\|_\infty < e^{-L^{max}_f} - c,$$

so that $|f_{k_i}(x) - f_{k_i}(x_i)| < e^{-L^{max}_f} - c$. Therefore

$$f_{k_i}(x) > f_{k_i}(x_i) - e^{-L^{max}_f} + c \geq e^{-L^{max}_f} - e^{-L^{max}_f} + c = c,$$

Table K.3
The results in this table were obtained as follows: We used the ReLU as the activation function and Adam as the optimizer. Furthermore, we chose several different network structures and learning rates. We then trained the networks for 10 000 iterations with a batch size of 300, and selected the best test error as the final result for each training procedure. Each of the columns *i*–*xii* stands for a specific choice of network architecture and learning rate, as described in Table K.4.

Data set	Version	*i*	*ii*	*iii*	*iv*	*v*	*vi*	*vii*	*viii*	*ix*	*x*	*xi*	*xii*	Best error
MNIST	Original	.02	.02	.05	.02	.02	.04	.02	.02	.04	.01	.02	.03	.01
CIFAR-10	Original	.47	.46	.52	.48	.46	.51	.47	.45	.50	.47	.45	.49	.45
CIFAR-10	Gray	.55	.55	.63	.55	.54	.62	.54	.53	.61	.54	.53	.59	.53
CIFAR-10	Conv	.18	.18	.19	.19	.18	.19	.18	.18	.18	.18	.18	.18	.18
SVHN	Original	.80	.59	.49	.80	.73	.60	.80	.69	.51	.80	.72	.64	.49
SVHN	Gray	.80	.64	.56	.80	.76	.66	.80	.64	.58	.80	.75	.66	.56
SVHN	Conv	.27	.23	.23	.31	.24	.23	.31	.24	.23	.69	.25	.24	.23
CIFAR-100	Original(coarse)	.64	.64	.69	.64	.63	.68	.63	.62	.67	.63	.62	.66	.62
CIFAR-100	Gray(coarse)	.74	.73	.79	.74	.73	.78	.74	.72	.78	.74	.72	.77	.72
CIFAR-100	Conv(coarse)	.40	.41	.45	.41	.41	.44	.40	.41	.44	.41	.40	.43	.40
COIL-20	Original	.08	.05	.05	.07	.06	.05	.08	.05	.05	.07	.05	.03	.03
CIFAR-100	Original(fine)	.75	.75	.82	.75	.74	.81	.74	.73	.80	.75	.73	.79	.73
CIFAR-100	Gray(fine)	.83	.83	.90	.83	.83	.90	.82	.81	.88	.83	.81	.87	.81
CIFAR-100	Conv(fine)	.53	.54	.64	.53	.54	.63	.52	.52	.61	.53	.52	.59	.52
COIL-100	Original	.02	.01	.01	.03	.02	.01	.03	.02	.01	.02	.02	.01	.01

so that f is c-accurate at x. Overall we obtain

$$\begin{aligned} p_c(f) &= \mu(H_c^f) \\ &\geq \mu\left(D \cap \bigcup_{x_i\in\mathcal{T}} B(x_i,\delta)\right) \\ &= h_{\mathcal{T}}^{\mu}(\delta) \\ &\geq 1 - \frac{\sqrt{d}}{\delta}(1-\rho_{\mathcal{T}}). \end{aligned}$$

The second inequality can be derived from Proposition 2.7. □

Appendix I. Proof of Theorem 3.3

Proof. We will prove it by contradiction. Consider two points x_1 and x_2 with different labels, and $\|x_1 - x_2\| = \delta_{\mathcal{T}}$.

If $\delta_f(e^{-L_f^{max}} - c) > \delta_{\mathcal{T}}/2$, then by the definition of $\delta_f(e^{-L_f^{max}} - c)$, $\|f(\frac{x_1+x_2}{2}) - f(x_1)\|_\infty < e^{-L_f^{max}} - c$, since $\|\frac{x_1+x_2}{2} - x_1\| = \|\frac{x_2-x_1}{2}\| = \delta_{\mathcal{T}}/2$. Similarly, $\|f(\frac{x_1+x_2}{2}) - f(x_2)\|_\infty < e^{-L_f^{max}} - c$. Then we have

$$\|f(x_1) - f(x_2)\|_\infty < 2(e^{-L_f^{max}} - c) \leq 2e^{-L_f^{max}} - 1.$$

On the other hand, by the definition of L_f^{max}, $-\ln f_{k_1}(x_1) \leq L_f^{max}$ and $-\ln f_{k_2}(x_2) \leq L_f^{max}$, so $f_{k_1}(x_1) \geq e^{-L_f^{max}}$, $f_{k_2}(x_2) \geq e^{-L_f^{max}}$ and $f_{k_1}(x_2) \leq 1 - f_{k_2}(x_2) \leq 1 - e^{-L_f^{max}}$, therefore

$$\begin{aligned} \|f(x_1) - f(x_2)\|_\infty &\geq f_{k_1}(x_1) - f_{k_1}(x_2) \\ &\geq e^{-L_f^{max}} - (1 - e^{-L_f^{max}}) \\ &= 2e^{-L_f^{max}} - 1. \quad \square \end{aligned}$$

Appendix J. Proof of Theorem 3.4

Proof. From Theorem 3.2 and assumption we know that

$$\begin{aligned} p(f) \geq p_{0.5}(f) &\geq 1 - \frac{\sqrt{d}}{\min(\delta_0, \delta_f(e^{-L_f^{max}} - 0.5))}(1-\rho_{\mathcal{T}}) \\ &\geq 1 - \frac{\sqrt{d}}{\min(\delta_0, \kappa\delta_{\mathcal{T}})}(1-\rho_{\mathcal{T}}) \\ &\geq 1 - \frac{\sqrt{d}}{\kappa\delta_0}(1-\rho_{\mathcal{T}}), \end{aligned}$$

Table K.4
Detailed setup for each case.

Structure	Learning rate		
	10^{-3}	10^{-4}	10^{-5}
[Input 256 256 Output]	*i*	*ii*	*iii*
[Input 256 256 256 Output]	*iv*	*v*	*vi*
[Input 512 512 Output]	*vii*	*viii*	*ix*
[Input 512 512 512 Output]	*x*	*xi*	*xii*

which implies $\lim_{\rho_{\mathcal{T}}\to 1} p(f) = 1$. Note that κ is less than 0.5 and δ_0 is only determined by the classification problem itself. The above inequality is easy to convert into form (ii). □

Appendix K. Detailed information of data and parameters for training

First, we list the information concerning the data selection.

1. MNIST: Last 55 000 samples of the training set for training and all the 10 000 samples of the test set for testing.
2. CIFAR-10: First 49 000 samples of the training set for training and all the 10 000 samples of the test set for testing.
3. CIFAR-100: First 49 000 samples of the training set for training and all the 10 000 samples of the test set for testing.
4. COIL-20: 1200 samples whose end numbers of the figure names are not multiples of 6 for training and 240 samples whose end numbers are multiples of 6 for testing.
5. COIL-100: 6000 samples whose end numbers of the figure names are not multiples of 30 for training and 1200 samples whose end numbers are multiples of 30 for testing.
6. SVHN: First 50 000 samples of the training set for training and first 10 000 samples of the test set for testing.

Parameters for networks are listed in Table K.3.

For generating convolution data, we choose the following structure

$$\begin{aligned} &conv[128] - relu - batchnorm - conv[256] - relu- \\ &batchnorm - pool - conv[512] - relu - batchnorm- \\ &conv[256] - relu - batchnorm - conv[64] - relu- \\ &batchnorm - pool - (extract\ data) - dense[512]- \\ &batchnorm - dropout - dense[128] - batchnorm- \\ &dropout - dense[output] \end{aligned}$$

with kernel size 3 × 3 (strides 1) and pool size 2 × 2 (strides 2), then train this CNN with batch size 64, learning rate 0.001 and optimizer RMSProp for 5 epochs. After that, extract new data at location mentioned in above structure by feeding the data to the trained network.

References

Allen-Zhu, Z., Li, Y., & Liang, Y. (2018). Learning and generalization in over-parameterized neural networks, going beyond two layers. arXiv preprint arXiv:1811.04918.

Allen-Zhu, Z., Li, Y., & Song, Z. (2018). A convergence theory for deep learning via over-parameterization. arXiv preprint arXiv:1811.03962.

Arora, S., Du, S., Hu, W., Li, Z., & Wang, R. (2019). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. arXiv preprint arXiv:1901.08584.

Arora, S., Ge, R., Neyshabur, B., & Zhang, Y. (2018). Stronger generalization bounds for deep nets via a compression approach. arXiv preprint arXiv:1802.05296.

Bartlett, P., Foster, D., & Telgarsky, M. (2017). Spectrally-normalized margin bounds for neural networks. In *Advances in neural information processing systems* (pp. 6240–6249).

Bartlett, P., Harvey, N., Liaw, C., & Mehrabian, A. (2017). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. arXiv preprint arXiv:1703.02930.

Bartlett, P., & Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research (JMLR)*, *3*(Nov), 463–482.

Baykal, C., Liebenwein, L., Gilitschenski, I., Feldman, D., & Rus, D. (2018). Data-dependent coresets for compressing neural networks with applications to generalization bounds. arXiv preprint arXiv:1804.05345.

Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2018). Reconciling modern machine learning and the bias-variance trade-off. arXiv preprint arXiv:1812.11118.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010* (pp. 177–186). Springer.

Bottou, L., & Bousquet, O. (2008). The tradeoffs of large scale learning. In *Advances in neural information processing systems* (pp. 161–168).

Cao, Y., & Gu, Q. (2019). A generalization theory of gradient descent for learning over-parameterized deep ReLU networks. arXiv preprint arXiv:1902.01384.

Chen, Y., Jin, C., & Yu, B. (2018). Stability and convergence trade-off of iterative optimization algorithms. arXiv preprint arXiv:1804.01619.

Cheng, Y., Wang, D., Zhou, P., & Zhang, T. (2018). Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, *35*(1), 126–136.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, *2*(4), 303–314.

Dinh, L., Pascanu, R., Bengio, S., & Bengio, Y. (2017). Sharp minima can generalize for deep nets. In *Proceedings of the 34th international conference on machine learning-Volume 70* (pp. 1019–1028). JMLR. org.

Du, S., Lee, J., Li, H., Wang, L., & Zhai, X. (2018). Gradient descent finds global minima of deep neural networks. arXiv preprint arXiv:1811.03804.

Dudley, R. (1979). Balls in $\mathbb{R}^k$ do not cut all subsets of k + 2 points. *Advances in Mathematics*, *31*(3), 306–308.

Dziugaite, G., & Roy, D. (2017). Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. arXiv preprint arXiv:1703.11008.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning, Vol. 1*. New York: Springer series in statistics.

Gonen, A., & Shalev-Shwartz, S. (2017). Fast rates for empirical risk minimization of strict saddle problems. arXiv preprint arXiv:1701.04271.

Gunasekar, S., Lee, J., Soudry, D., & Srebro, N. (2018). Implicit bias of gradient descent on linear convolutional networks. In *Advances in neural information processing systems* (pp. 9461–9471).

Hardt, M., Recht, B., & Singer, Y. (2015). Train faster, generalize better: Stability of stochastic gradient descent. arXiv preprint arXiv:1509.01240.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366.

Kawaguchi, K., Kaelbling, L., & Bengio, Y. (2017). Generalization in deep learning. arXiv preprint arXiv:1710.05468.

Keskar, N., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint arXiv:1609.04836.

Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International conference on learning representations*.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images: Technical Report*, Citeseer.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, *25*, http://dx.doi.org/10.1145/3065386.

Kuzborskij, I., & Lampert, C. (2017). Data-dependent stability of stochastic gradient descent. arXiv preprint arXiv:1703.01678.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, J., Simchowitz, M., Jordan, M., & Recht, B. (2016). Gradient descent converges to minimizers. arXiv preprint arXiv:1602.04915.

Li, Y., & Liang, Y. (2018). Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in neural information processing systems* (pp. 8157–8166).

Liang, T., Poggio, T., Rakhlin, A., & Stokes, J. (2017). Fisher-Rao metric, geometry, and complexity of neural networks. arXiv preprint arXiv:1711.01530.

Liao, Q., & Poggio, T. (2017). Theory II: Landscape of the empirical risk in deep learning. arXiv preprint arXiv:1703.09833.

Lu, L., Shin, Y., Su, Y., & Karniadakis, G. (2019). Dying reLU and initialization: Theory and numerical examples. arXiv preprint arXiv:1903.06733.

Lu, L., Su, Y., & Karniadakis, G. (2018). Collapse of deep and narrow neural nets. arXiv preprint arXiv:1808.04947.

Maas, A., Hannun, A., & Ng, A. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. Icml* (p. 3).

Mitzenmacher, M., & Upfal, E. (2017). *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis* (2nd ed.). New York, NY, USA: Cambridge University Press.

Nagarajan, V., & Kolter, J. (2019a). Deterministic PAC-Bayesian generalization bounds for deep networks via generalizing noise-resilience. In *International conference on learning representations*.

Nagarajan, V., & Kolter, J. (2019b). Generalization in deep networks: The role of distance from initialization. arXiv preprint arXiv:1901.01672.

Nene, S., Nayar, S., & Murase, H. (1996a). *Columbia object image library (coil-100)*. Citeseer.

Nene, S., Nayar, S., & Murase, H. (1996b). *Columbia object image library (coil-20): Technical report CUCS-005-96*.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. (2011). Reading digits in natural images with unsupervised feature learning. In *Advances in neural information processing systems*.

Neyshabur, B., Bhojanapalli, S., McAllester, D., & Srebro, N. (2017). Exploring generalization in deep learning. In *Advances in neural information processing systems* (pp. 5947–5956).

Neyshabur, B., Bhojanapalli, S., & Srebro, N. (2017). A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. arXiv preprint arXiv:1707.09564.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., & Srebro, N. (2019). The role of over-parametrization in generalization of neural networks. In *International conference on learning representations*.

Neyshabur, B., Salakhutdinov, R., & Srebro, N. (2015). Path-SGD: Path-normalized optimization in deep neural networks. In *Advances in neural information processing systems* (pp. 2422–2430).

Neyshabur, B., Tomioka, R., & Srebro, N. (2014). In search of the real inductive bias: On the role of implicit regularization in deep learning. arXiv preprint arXiv:1412.6614.

Poggio, T., Kawaguchi, K., Liao, Q., Miranda, B., Rosasco, L., Boix, X., Hidary, J., & Mhaskar, H. (2017). Theory of deep learning III: explaining the non-overfitting puzzle. arXiv preprint arXiv:1801.00173.

Rahaman, N., Arpit, D., Baratin, A., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., & Courville, A. (2018). On the spectral bias of deep neural networks. arXiv preprint arXiv:1806.08734.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., & Cox, D. D. (2019). On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, *2019*(12), 124020.

Shwartz-Ziv, R., & Tishby, N. (2017). Opening the black box of deep neural networks via information. arXiv preprint arXiv:1703.00810.

Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., & Lanctot, M. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, *529*(7587), 484.

Sokolic, J., Giryes, R., Sapiro, G., & Rodrigues, M. (2016). Generalization error of invariant classifiers. arXiv preprint arXiv:1610.04574.

Sokolić, J., Giryes, R., Sapiro, G., & Rodrigues, M. (2017). Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, *65*(16), 4265–4280.

Soudry, D., Hoffer, E., Nacson, M., Gunasekar, S., & Srebro, N. (2018). The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research (JMLR)*, *19*(1), 2822–2878.

Wei, C., & Ma, T. (2019). Data-dependent sample complexity of deep neural networks via Lipschitz augmentation. arXiv preprint arXiv:1905.03684.

Xu, Z., Zhang, Y., Luo, T., Xiao, Y., & Ma, Z. (2019). Frequency principle: Fourier analysis sheds light on deep neural networks. arXiv preprint arXiv:1901.06523.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530.

Zhang, C., Liao, Q., Rakhlin, A., Miranda, B., Golowich, N., & Poggio, T. (2018). Theory of deep learning IIb: Optimization properties of SGD. arXiv preprint arXiv:1801.02254.

Zhou, W., Veitch, V., Austern, M., Adams, R., & Orbanz, P. (2018). Compressibility and generalization in large-scale deep learning. arXiv preprint arXiv:1804.05862.