A Statistical Analysis of Neural Networks

Yo Joong "YJ" Choe
Carnegie Mellon University
yjchoe@cmu.edu

1 Introduction

While neural networks gained huge empirical success in the recent years [1], the statistical theory behind their success remains to be seen. This project attempts to review some of the previous and current work that may give insights into analyzing neural networks from a statistical point of view. In particular, we review two sets of papers that analyze neural networks using generalization error bounds and minimax theory.

First, we follow an analysis of error bounds [2] and minimax rates [3] for feedforward sigmoidal neural networks with a single hidden layer. This leads to a minimax rate that appears to break the statistical curse of dimensionality by only requiring quadratically many data points to achieve the same error rate. The result should be viewed with caution, however, because the set of all neural networks considered in the analysis implicitly shrinks as dimension increases.

Next, we give a more recent analysis of generalization bounds [4] for convex neural networks [5] using nonnegative homogeneous activation functions such as the rectified linear unit (ReLU) [6]. Assuming a low-dimensional nonlinear structure, Bach shows that the mean generalization error (or excess risk) can be bounded by a rate that also appears to break the statistical curse and further demonstrates adaptivity to the low-dimensional subspace.

While a rigorous statistical analysis of neural networks containing multiple hidden layers is still largely an open problem, we hope that analyses of "shallow" neural networks can still provide insights to understanding the statistical properties of neural networks.

2 Notation and Assumptions

2.1 Problem Setup

Throughout the paper, we consider a regression problem given n data points $(X_1, Y_1), \ldots, (X_n, Y_n) \in \mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^d \times \mathbb{R}$. We further assume that the data comes from the model $Y_i = f(X_i) + \varepsilon_i$, where f is the true regression function from a function class $\mathcal{F}$ and the errors ε_i have zero mean and are independent from the inputs X_i. For Theorem 3.3, we further assume that X_i are independent and identically distributed to some distribution P.

Unless specified otherwise (e.g. proof of Theorem 3.3), we also assume that both the input and output spaces are bounded: there exist $r, r' > 0$ such that $\|x\|_2 \leq r$ and $|y| \leq r'$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. In such cases, we let $\mathcal{X} = B_r = \{x \in \mathbb{R}^d : \|x\|_2 \leq r\}$ for simplicity.

2.2 Neural Networks

In all of our results, we will consider the set of all *(feedforward) neural networks* with one hidden layer and an activation function. That is, we consider the set of all real-valued functions on $\mathbb{R}^d$ that have the form

$$f(x) = \eta_0 + \sum_{j=1}^{k} \eta_j \sigma(w_j^T x + b_j) \qquad \forall\, x \in \mathbb{R}^d \tag{1}$$

Technical Report (2016).

for *any* $k \in \mathbb{N}$, where $\eta = (\eta_0, \eta_1, \dots, \eta_k) \in \mathbb{R}^{k+1}$ and $(w_j, b_j) \in \mathbb{R}^d \times \mathbb{R}$ for each $j = 1, \dots, k$. We call $\sigma : \mathbb{R} \to [0, \infty)$ an *activation function* and the function $x \mapsto \sigma(w_j^T x + b_j)$ as the jth *neuron* or *unit*.

We say that an activation function σ is *sigmoidal* or *squashing* if it is a nondecreasing, bounded, and continuous function on $\mathbb{R}$ such that $\lim_{t\to-\infty} \sigma(t) = 0$ and $\lim_{t\to\infty} \sigma(t) = 1$. The sigmoidal activation function can be viewed as a continuous surrogate for the discontinuous *step function* $\sigma^* : \mathbb{R} \to \{0, 1\}$ defined by $\sigma^*(t) := I(t \geq 0)$, where I is the indicator function. A neural network of the form (1) with a sigmoidal activation function is called a *sigmoidal neural network*.

Alternatively, we say that an activation function σ is *homogeneous of order* $\alpha \in \{0, 1, \dots\}$ if $\sigma(t) = (t)_+^\alpha = \max\{t, 0\}^\alpha$. Note that, if $\alpha = 0$, we get the step function σ^*. If $\alpha = 1$, we also call the function as the *rectified linear unit (ReLU)* or the *rectifier*. For $\alpha \geq 1$, an homogeneous activation function σ is *not* sigmoidal because $\lim_{t\to\infty} \sigma(t) = \infty$. Rather, we have that $\lim_{t\to\infty} \sigma(t)/t^\alpha = 1$ for the homogeneous activation function of order α.

Finally, we define a set of target functions in terms of the absolute mean of Fourier transformations. Specifically, for any $C > 0$, define $\Gamma^C = \{f : \mathbb{R}^d \to \mathbb{R} \mid \int_{\mathbb{R}^d} |\omega| \, |\tilde{f}(\omega)| d\omega \leq C\}$, where $\tilde{f}$ denotes the Fourier transform of f.

2.3 Convex Neural Networks

Convex neural networks [5] provide an interesting theoretical approach to analyzing single-hidden-layer neural networks. By letting the number of neurons $k \to \infty$ in (1), we can obtain functions of the form

$$f(x) = \int_{\mathcal{V}} \sigma(w^T x + b) \eta(w, b) d\tau(w, b) = \int_{\mathcal{V}} \varphi_v(x) \eta(v) d\tau(v) \tag{2}$$

where $\eta : \mathcal{V} \to \mathbb{R}$ is now a function of the weight and bias vector $v = (w, b)$, over which some measure τ is defined. We assume that $\mathcal{V} = B_l = \{v = (w, b) \in \mathbb{R}^d \times \mathbb{R} : \|v\|_1 = \|(w, b)\|_1 \leq l\}$ for some $l > 0$, and for each $v = (w, b) \in \mathcal{V} \subseteq \mathbb{R}^d \times \mathbb{R}$, we will denote a neuron by $\varphi_v : x \mapsto \sigma(w^T x + b) = \sigma(v^T z)$, where $z = (x, 1)$. We recover (1) by letting $\eta d\tau = \eta_0 + \sum_{j=1}^k \eta_j \mathrm{Dirac}_{(w_j, b_j)}$ in (2), where Dirac is the Dirac delta function.

With convex neural networks, we will restrict ourselves to the case when σ is a homogeneous activation function such as the rectified linear unit.

2.4 Function Norms and Spaces

For any function f defined on the bounded domain $\mathcal{X} = B_r \subseteq \mathbb{R}^d$ and an arbitrary probability measure P defined on B_r, we define the *(squared) functional L^2 norm* as

$$\|f\|_2^2 = \int_{\mathcal{X}} f^2 dP = \int_{\mathcal{X}} f^2(x) dP(x)$$

The closure of set of all functions that have finite functional L^2 norm defines the Hilbert space $L^2(\mathcal{X})$.

We also define norms that give constraints on the function space we consider. For a neural network f of the form (1), we define the *L^1 variation norm* as $\gamma_1(f) = \|\eta\|_1$ and the *L^2 RKHS norm* as $\gamma_2(f) = \|\eta\|_2$. For $p = 1, 2$, we let $\mathcal{N}_p^\delta$ be the closure (in the functional L^2 norm) of the set of all neural networks f of the form (1) such that $\gamma_p(f) \leq \delta$. We will use $\mathcal{N}_p$ to denote the analogous set with the constraint that $\gamma_p(f) < \infty$.

More generally, we can extend this definition to convex neural networks. For a convex neural network of the form (2), define

$$\gamma_p(f) = \left(\int_{\mathcal{V}} |\eta(w, b)|^p \, d\tau(w, b) \right)^{\frac{1}{p}}$$

and the definitions of $\mathcal{N}_p^\delta$ and $\mathcal{N}_p$ can be extended analogously (we will denote these convex analogues as $\mathcal{F}_p^\delta$ and $\mathcal{F}_p$ respectively). One can show that this is a well-defined norm on $\mathcal{F}_p$, provided that $\mathcal{V}$ is a compact topological space. Note that this definition of the norm is equivalent to what we defined in the previous paragraph.

Since convex neural networks are generalizations of neural networks, we get $\mathcal{N}_p^\delta \subseteq \mathcal{F}_p^\delta$ for each $\delta \in (0, \infty]$. Also, one can prove using Jensen's inequality that $\gamma_1(f) \le \gamma_2(f)$ for each (convex) neural network f, which in turn implies that $\mathcal{N}_2^\delta \subseteq \mathcal{N}_1^\delta$ and $\mathcal{F}_2^\delta \subseteq \mathcal{F}_1^\delta$ for each $\delta \in (0, \infty]$.

Finally, since the form (2) depends on the choice of the activation function σ, which we assume to be homogeneous of order α (i.e. $\sigma(t) = (t)_+^\alpha$), both γ_p and $\mathcal{F}_p^\delta$ depends on the parameter α.

2.5 Loss, Risk, and Errors

The *squared loss* of an estimator $\hat{f}$ at data point (x, y) is defined as $\ell(y, \hat{f}(x)) = (y - \hat{f}(x))^2$, and the *(prediction) risk* of $\hat{f}$ using any loss function ℓ (e.g. the squared loss) is the expected loss $\mathbb{E}[\ell(Y, \hat{f}(X))]$, where the expectation is taken over the joint distribution of X and Y. The *empirical risk* of an estimator $\hat{f}$ given n data points is defined as $\hat{R}(\hat{f}) = \frac{1}{n}\sum_{i=1}^n \ell(Y_i, \hat{f}(X_i))$.

In general, consider a function class $\mathcal{F}$ and an estimator $\hat{f}_{n,\delta}$ that minimizes the empirical risk, up to some *optimization error* ε_{opt}, within a sub-function class $\mathcal{F}^\delta \subseteq \mathcal{F}$ that is characterized by some $\delta > 0$. We define the *generalization error* or *excess risk* as $R(\hat{f}_{n,\delta}) - \inf_{f\in\mathcal{F}} R(f)$.

Assuming a G-Lipschitz-continuous continuous loss ℓ, such as the squared loss on a bounded domain $\mathcal{X} = B_r$, and a sub-function class $\mathcal{F}^\delta \subseteq \mathcal{F}$ with $\hat{f}_\delta = \operatorname{arginf}_{f\in\mathcal{F}^\delta} R(f)$, we will use the following decomposition of the generalization error [7]:

$$R(\hat{f}_{n,\delta}) - \inf_{f\in\mathcal{F}} R(f) \le G \sup_{x\in\mathcal{X}} |\hat{f}_\delta(x) - f(x)| + 2 \sup_{f\in\mathcal{F}^\delta} |\hat{R}(f) - R(f)| + \varepsilon_{opt} \tag{3}$$

where the first two terms are respectively called as the *approximation error* and the *estimation error*.

2.6 Minimax Theory

The *minimax risk* of a function class $\mathcal{F}$ is defined as

$$R_n = R_n(\mathcal{F}) = \inf_{\hat{f}_n} \sup_{f\in\mathcal{F}} \mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right]$$

where the infimum is over all estimators using n data points and the expectation is taken over the randomness of $\hat{f}_n$ assuming that the true function is f.

For each $\varepsilon > 0$, an *ε-packing set* $N_\varepsilon \subseteq \mathcal{F}$ is a finite set such that for any $f, g \in N_\varepsilon$ and $f \neq g$ we have $\|f - g\|_2 > \varepsilon$. The *($L^2$) metric entropy* of $\mathcal{F}$, denoted as $M(\varepsilon)$, is the logarithm of the maximum cardinality of all ε-packing sets for $\mathcal{F}$.

3 Key Results

3.1 Minimax Rate of Convergence for Neural Networks

We first present the universal approximation property of single-hidden-layer neural networks.

Theorem 3.1 (Universal Approximation; [8, 9]) *Sigmoidal neural networks are universal approximators for any Borel-measurable real-valued function on a compact subset $K \subseteq \mathbb{R}^d$. That is, given any Borel-measurable function $f : \mathbb{R}^d \to \mathbb{R}$ and any $\varepsilon > 0$, there exists a neural network $\hat{f}$ of the form* (1), *where σ is sigmoidal, such that*

$$\sup_{x\in K} |\hat{f}(x) - f(x)| \le \varepsilon.$$

The number of neurons in $\hat{f}$ can depend on both f and ε.

This version of the universal approximation can be found in Theorem 2.2 of Hornik et al. [8] for any continuous nonconstant activation function (e.g., sigmoidal and rectifier activations). Cybenko (Theorem 2, [9]) and Hornik et al. (Theorem 2.1, [8]) concurrently proved analogous results for approximating any continuous real-valued function on a compact domain.

Given that single-hidden-layer neural networks are universal approximators, one can claim that it is reasonable to assume the true regression function is a single-hidden-layer neural network. In this sense, we are further interested in evaluating its statistical performance as a function class using minimax theory. But as a first step, in [2], Barron presents an upper bound on the squared L^2 risk using a choice of function class that allows for arguments using Fourier analysis.

Theorem 3.2 ([2] Theorem 3, Generalization Error Bound) *Suppose that the true regression function f has a Fourier transform with a bounded absolute mean*[1]*, that is, $f \in \Gamma^C$.*

Let $\hat{f}_n = \mathit{arginf}_{f \in \mathcal{N}_1^\delta} \frac{1}{n} \sum_{i=1}^n \left(Y_i - f(X_i)\right)^2$ be a neural network that minimizes the mean squared error subject to the constraint that $\gamma_1(f) \leq \delta$. Let k be the number of neurons in the hidden layer of $\hat{f}_n$. Then,

$$\mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right] \leq O\left(\frac{1}{k}\right) + O\left(\frac{kd\log n}{n}\right)$$

where the expectation is taken over the data and the two terms on the right hand side are approximation and estimation errors, respectively.

By choosing $k = O\left(\frac{d\log n}{n}\right)^{-1/2}$, we obtain the following upper bound:

$$\mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right] \leq O\left(\frac{d\log n}{n}\right)^{1/2} \tag{4}$$

Finally, we present the near-minimax rate of convergence in squared L^2 risk for the set of single-hidden-layer neural networks.

Theorem 3.3 ([3] Section 7.8, Minimax Rate of Convergence) *Consider $\mathcal{N}_1^\delta$, the set of all single-hidden-layer neural networks with a* sigmoidal *activation function and output weights that are bounded by δ in the L^1 variation norm. Then,*

$$\left(\frac{(\log n)^{1+1/d}}{n}\right)^{\frac{1+1/d}{2+1/d}} \preceq \inf_{\hat{f}_n} \sup_{f\in\mathcal{N}_1^\delta} \mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right] \preceq \left(\frac{\log n}{n}\right)^{\frac{1+1/d}{2+1/d}}$$

That is, up to a logarithmic factor, we obtain the following minimax rate of convergence:

$$\inf_{\hat{f}_n} \sup_{f\in\mathcal{N}_1^\delta} \mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right] \asymp \left(\frac{\log n}{n}\right)^{\frac{1+1/d}{2+1/d}} \tag{5}$$

Even for a moderately large d, the rate is approximately $n^{-1/2}$.

As a reference, the minimax rate for nonparametric estimation using the set of all Lipschitz-continuous functions is $n^{-2/(2+d)}$, which implies that as d increases one requires exponentially more data points to achieve the same error rate. This phenomenon is often called *the (statistical) curse of dimensionality*. The rate in (5) appears to suggest that single-hidden-layer neural networks do not suffer from the curse of dimensionality; however, since the output weights η are bounded by a fixed constant, we see that the set of neural networks we consider is implicitly shrinking as $d \to \infty$.

3.2 Generalization Bounds for Convex Neural Networks

In this section, we present results from [4] based on a theoretical approach using convex neural networks. We assume that the activation function is homogeneous of order α, i.e. $\sigma(x) = (x)_+^\alpha$. Note that we no longer have a minimax result but only an upper bound on the (mean) generalization error, which can be decomposed into approximation, estimation, and optimization errors as in (3).

We first show that any convex neural network of the form (2) can be approximated arbitrarily closely with a neural network of the form (1) containing finitely many neurons. While the result was first presented by Barron in [10] and is used to prove Theorem 3.2, Bach in [4] provides a constructive proof using a convex optimization procedure with conditional gradients.

[1] This function class includes linear functions, sigmoidal functions, functions with derivatives of order at least $\lfloor d/2 \rfloor + 2$, absolutely convergent Fourier series, and more. See Section IX in [10] for examples of functions that belong to this class.

Proposition 3.4 ([10, 4], Conditional Gradient Algorithm for Convex Neural Networks) *For any $f \in \mathcal{F}_1$, there exists a neural network $\hat{f} \in \mathcal{N}_1^\delta$ that has the form* (1)*, with $k = O(\gamma_1(f)^2 \varepsilon^{-2})$, and satisfies $\|\hat{f} - f\|_2^2 \leq \varepsilon^2$.*

Assuming that the loss function is convex and smooth, there exists an iterative algorithm that searches the space $\mathcal{F}_1^\delta$ and converges to a solution $\hat{f}$ at rate $O(1/t)$, where t is the number of iterations. The solution is a convex combination of t single neurons of the form $x \to \pm\delta\sigma(w^T x + b)$ for some $(w, b) \in \mathcal{V}$, and in particular $\hat{f} \in \mathcal{N}_1^\delta$.

Proposition 3.4 allows us to only consider neural networks in $\mathcal{N}_1^\delta$ with finitely many neurons, even though the theoretical results hold for convex neural networks in $\mathcal{F}_1^\delta$ more generally. It also allows us to assume that the optimization error can be made arbitrarily close to zero, and for the rest of the analysis we assume that the generalization error can be bounded by the sum of approximation and estimation errors.[2]

We now give a bound on the approximation error.

Proposition 3.5 ([4] Proposition 6, Approximation Error Bound) *Let f be any L-Lipschitz-continuous function and let $\delta \geq C(d, \alpha)$ where $C(d, \alpha)$ is some constant that only depends on d and α. Then, there exists a neural network $\hat{f}_\delta \in \mathcal{F}_2^\delta \subseteq \mathcal{F}_1^\delta$ such that*

$$\sup_{x \in \mathcal{X}} |\hat{f}_\delta(x) - f(x)| \leq C(d, \alpha) \left(\frac{\delta}{L}\right)^{-1/(\alpha + (d-1)/2)} \log\left(\frac{\delta}{L}\right) \tag{6}$$

This bound gives an interesting corollary in the case where the number of neurons is fixed.

Corollary 3.6 ([4] Section 4.7, Approximation Error Bounds using Finitely Many Neurons) *Let $\alpha = 1$ and k be the number of neurons. Then, any f be any L-Lipschitz-continuous function can be approximated by a neural network $\hat{f} \in \mathcal{N}_1^\delta$ with k neurons with a uniform error*

$$Lk^{-1/d} \log k$$

where $\delta = Ln^{\frac{d+1}{2d}}$.

Bach notes that the approximation error of $O(k^{-1/d})$ for rectified linear units with finite number of neurons was previously known in the literature as the best bound under the L^2 loss and various activation functions [11, 12, 13].

Next, we give a uniform bound on the (mean) estimation error.

Proposition 3.7 ([4] Proposition 7, Estimation Error Bound) *Let R be a risk function defined using a loss function ℓ that is G-Lipschitz-continuous in its second argument. Then, for $\alpha \geq 1$ and for any $\delta > 0$,*

$$\mathbb{E}\left[\sup_{f \in \mathcal{N}_1^\delta} |\hat{R}(f) - R(f)|\right] \leq 4C(d, \alpha)\frac{G\delta}{\sqrt{n}} \tag{7}$$

where $C(d, \alpha) = \alpha\sqrt{2\log(d+1)}$.

Now we present the main result, which provides a set of bounds on the mean generalization error using Propositions 3.5 and 3.7.

Theorem 3.8 ([4] Section 5, Generalization Error Bounds) *Suppose that the true regression function has the form*

$$f(x) = \sum_{j=1}^{k} f_j(w_j^T x) \tag{8}$$

[2] The result assumes that the true risk function is known. In [4], Bach notes that there is a "representer theorem" for the finite sample case that allows the loss function to only depend on the n data points, in which case at most n neurons are used. He also notes that this is very different from the usual RKHS representer theorem, however, because the n neurons are not known in advance.

where $w_j \in \mathcal{V} = B_l$ and f_j is bounded as well as 1-Lipschitz-continuous for each $j = 1, \dots, k$. Let $\mathcal{F}_k$ be the set of all such functions.[3]

Suppose that the weight matrix $W = [w_1, \dots, w_k] \in \mathbb{R}^{d \times k}$ has at most $s \ll d$ nonzero elements. In particular, assume that $\|w_j\|_1 \leq \sqrt{s}l$. Then, for $\alpha \geq 1$,

$$\mathbb{E}\left[R(\hat{f}_{n,\delta}) - \inf_{f \in \mathcal{F}_k} R(f)\right] \leq O\left(\frac{ks^{1/2}\left(\log d\right)^{1/(\alpha+1)}}{n^{1/(2\alpha+2)}}\log n\right) \tag{9}$$

where $\hat{f}_{n,\delta}$ = arginf$_{f \in \mathcal{F}_1^\delta}$ $\hat{R}(f)$ with $\delta = \left(\frac{n}{\log d}\right)^{\alpha/(2\alpha+2)}$ and the expectation is taken over the randomness of the data $(X_1, Y_1), \dots, (X_n, Y_n)$.

Using the squared loss function and assuming $\alpha = 1$, in which case the estimation is done using rectified linear units, we obtain

$$\mathbb{E}\left[\|\hat{f}_{n,\delta} - f\|_2^2\right] \leq O\left(\frac{k\left(s\log d\right)^{1/2}}{n^{1/4}}\log n\right) \tag{10}$$

There are several remarks to be made about Theorem 3.8.

1. The exponent of n is independent of d assuming the form (8). When there is only a Lipschitz-continuity assumption on f, the rate becomes

   $$O\left(\frac{C(d)s^{1/2}}{n^{1/(d+3)}}\log n\right)$$

   where $C(d)$ is some constant that only depends on d.

   This does not necessarily imply, however, that the assumption is too strict. It includes all single-hidden-layer neural networks with at most k neurons, which we showed are universal approximators (Theorem 3.1) with seemingly appealing minimax rates (Theorem 3.3) as $k \to \infty$. Of course, as $k \to \infty$ the risk bound (9) becomes meaningless – see next item for a better way to understand the bound in a way similar to Theorems 3.2 and 3.3.
2. A regression model of the form (8) is used in *projection pursuit regression*. The scaling by k is due to the fact that we approximate each f_j and add up the k error terms. We can remove k in the error bound by assuming a more specific form $f(x) = \sum_{j=1}^k \eta_j f_j(w_j^T x)$ such that $\|\eta\|_1 \leq c$ for some constant $c > 0$, and this will give a bound more similar to Theorem 3.2. But it should also be noted that a fixed constant on the output weights implies that the function space becomes relatively smaller as $k \to \infty$ or as $d \to \infty$.
3. The assumption that the weight matrix W is s-sparse by an L^1-penalty is crucial. The assumption suggests that there is a low-dimensional subspace that can be selected using convex neural networks. In [4], Bach claims that this means the set of convex neural networks with an L^1-penalty on the weights are capable of doing *high-dimensional nonlinear variable selection*. The rate of convergence seems to resemble that of the lasso [14], which is a linear variable selection method that gives the rate $O\left(\frac{s\log d}{n}\right)$.

   Without the sparsity assumption, the mean generalization bound is only

   $$O\left(\frac{kd^{1/2}}{n^{1/(2\alpha+2)}}\log n\right)$$

   which gives a slower rate of convergence than in Theorem 3.2.
4. Other than the sparsity and boundedness assumptions, there is no other assumption made – in particular, the result still holds with correlated inputs [4]. This is different from the assumptions made in Theorem 3.3.
5. It appears that the use of rectifiers ($\alpha = 1$) gives the slower rate $n^{-1/4}$, compared to Theorems 3.2 and 3.3, which give the rate $n^{-1/2}$. However, the use of rectifiers allow for a theoretical connection to RKHS theory [4] that leads to Theorem 3.8.

[3] Note that this set includes single-hidden-layer neural networks of the form (1) having k neurons, by using an augmented input $x \leftarrow (x, 1)$.

6. The bound (10) using $\alpha = 1$ is gives the best rate among all choices of $\alpha \geq 1$ in (9). Intuitively, the role of α in the analysis is similar to that of the smoothness parameter in Sobolev or Besov spaces for nonparametric regression.
7. The loss function can be any Lipschitz-continuous function, such as the hinge loss, the logistic loss, and the squared loss on a bounded domain.

4 Proof Outline

4.1 Minimax Rate of Convergence for Neural Networks

We will skip the proof of the rather classical result in Theorem 3.1. We give a proof outline of Theorem 3.3, which can be viewed as a minimax version of Theorem 3.2.

We start with a general result in minimax theory involving metric entropy. Recall our regression setting $Y_i = f(X_i) + \varepsilon_i, i = 1, \dots, n$, where f is the regression function. Suppose that $X_i \overset{iid}{\sim} P$ for some fixed distribution P and the errors $\varepsilon_i \overset{iid}{\sim} \mathrm{Normal}(0, \sigma^2)$ are independent of the X_i's. Note that the definition of the squared loss of an estimator $\hat{f}_n$ to f involves P: $\|\hat{f}_n - f\|_2^2 = \int_{\mathcal{X}} (\hat{f}_n(x) - f(x))^2 dP$.

Theorem 4.1 ([3] Theorem 6, Le Cam Equation for Nonparametric Regression) *Let $\mathcal{F}$ be the set of regression functions such that*

(i) $\liminf_{\varepsilon \to 0} M(a\varepsilon)/M(\varepsilon) > 1$ *for some* $a \in (0, 1)$.

(ii) $\sup_{f \in \mathcal{F}} \|f\|_\infty < \infty$.

Then, the solution ε_n to the Le Cam equation

$$M(\varepsilon_n) = n\varepsilon_n^2 \tag{11}$$

satisfies

$$\inf_{\hat{f}_n} \sup_{f \in \mathcal{F}} \mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right] \asymp \varepsilon_n^2$$

Therefore, we can obtain the minimax rate of convergence by solving for ε_n in the Le Cam equation (11). We note that the standard statement of this result actually involves estimating from a class of density functions, which is further assumed to be convex and bounded away from zero. Theorem 4.1 is an extension of the standard version to regression using the Hellinger distance.

Next, we compute the metric entropy of the set of single-hidden-layer neural networks with a *sigmoidal* activation function.

Lemma 4.2 ([15] Proof of Theorem 4, Metric Entropy of the Sigmoidal Neural Network Class) *For each $\delta > 0$, the metric entropy of $\mathcal{N}_1^\delta$ with a sigmoidal activation function is given by*

$$\varepsilon^{-\frac{1}{1/2+1/d}} \preceq M(\varepsilon) \preceq \varepsilon^{-\frac{1}{1/2+1/2d}} \log(1/\varepsilon) \tag{12}$$

Now we give a proof sketch of Theorem 3.3 using Theorem 4.1 and Lemma 4.2.

Proof of Theorem 3.3 [3]: We will apply to Theorem 4.1 the bounds we get from Lemma 4.2. First, from Lemma 4.2, we can show that $\mathcal{N}_1^\delta$ satisfies condition (i). Intuitively, up to a logarithmic factor, we get

$$\frac{M(a\varepsilon)}{M(\varepsilon)} \asymp a^{-\frac{1}{1/2+1/d}} > 1$$

for any $a \in (0, 1)$. Condition (ii) is also satisfied because, for every $f \in \mathcal{N}_1^\delta$ of the form (1), we assume that $\gamma_1(f) = \|c\|_1 \leq \delta$ and that σ is bounded.

Then, applying the upper and lower bounds from (12) to (11) and solving for ε_n, we get

$$\left(\frac{(\log n)^{1+1/d}}{n}\right)^{\frac{1+1/d}{2+1/d}} \preceq \inf_{\hat{f}_n} \sup_{f \in \mathcal{N}_1^\delta} \mathbb{E}\left[\|\hat{f}_n - f\|_2^2\right] \asymp \varepsilon_n^2 \preceq \left(\frac{\log n}{n}\right)^{\frac{1+1/d}{2+1/d}}$$

■

Note that, while Theorem 3.3 is a more general result in the minimax sense, it also makes the assumption that the model errors are i.i.d. Normal. This is different from the assumption from Theorem 3.2 that the error is instead independent and bounded.

4.2 Generalization Bounds for Convex Neural Networks

Due to the amount of results presented, we will only give proof outlines to select propositions/theorems. All proofs can be found in [4] and its references.

A constructive proof of Proposition 3.4 is given in [4] via a conditional gradient algorithm that employs the Frank-Wolfe algorithm to add a new neuron of the form $x \to \pm\delta\sigma(w^T x + b)$ into a convex combination.

The proof of Proposition 3.5 makes use of Fourier transforms and spherical harmonics to get the result when the domain is $\mathbb{S}^{d-1}$ and then extends to $\mathbb{R}^d$ by noting that the functions in $\mathcal{N}_2^\delta$ are (L/r)-Lipschitz-continuous. Proposition 3.7 is a standard result [16] that can be proved using Radamacher complexities.

Propositions 3.5 and 3.7 can give generalization bounds for convex neural networks under various settings which can be found in [4]. Here we presented one of the settings.

Proof of Theorem 3.8 [4]: First, we more specifically assume that f_j is bounded by $lr\sqrt{s}$ and is 1-Lipschitz-continuous. (Recall that we assumed earlier $\mathcal{X} = \{x \in \mathbb{R}^d : \|x\|_2 \leq r\}$ and $\mathcal{V} = \{v = (w, b) \in \mathbb{R}^d \times \mathbb{R} : \|v\|_1 \leq l\}$.) Then, by Proposition 3.5, we can approximate each f_j by $\hat{f}_j \in \mathcal{F}_1^{\delta'}$, where $\delta' = \delta l r\sqrt{s}$. This gives the error $C(\alpha) l r\sqrt{s}\delta^{-1/\alpha}\log\delta$. For k functions, we get a total approximation error of $kC(\alpha) l r\sqrt{s}\delta^{-1/\alpha}\log\delta$, now assuming $\delta' = k\delta l r\sqrt{s}$.

By Proposition 3.7 with $\alpha \geq 1$, we obtain the mean estimation error of $\frac{kGlr\delta\sqrt{s\log d}}{n}$. (Recall that we assume the loss function is G-Lipschitz-continuous.) By choosing $\delta = \left(\frac{n}{\log d}\right)^{\alpha/(2\alpha+2)}$, we can balance the two error terms and obtain the desired bound on the mean generalization error. ■

5 Conclusion and Discussion

We presented two sets of statistical analyses that attempts to explain whether single-hidden-layer neural networks can break the statistical curse of dimensionality.

The first set of analyses shows that neural networks with single hidden layers and a sigmoidal activation function are universal approximators with a minimax rate of convergence that scales roughly by $n^{-1/2}$. This seems to imply that single-hidden-layer neural networks do not require exponentially larger sample sizes to achieve the same error, although at the same time the function space is implicitly decreasing as $d \to \infty$.

The second set of analyses employs the framework of convex neural networks and uses the rectified linear unit (ReLU) and L^1-penalty on input weights (in addition to output weights) to show that the mean generalization error with any Lipschitz-continuous loss function is bounded roughly by $\frac{(s\log d)^{1/2}}{n^{1/4}}$, assuming there is an underlying nonlinear low-dimensional structure. This rate suggests that single-hidden-layer neural networks with ReLU and L^1-penalty may be capable of doing nonlinear variable selection in a high-dimensional setting.

These results may provide some insights into why neural networks seem to be useful in finding meaningful representations in high-dimensional settings, while a more rigorous analysis of neural networks in high-dimensional settings remains to be seen. In particular, none of these results are believed to be applicable to deep neural networks [4], which are the ones that earned deep learning huge successes in vision, speech, and many other application domains.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[2] Andrew R Barron. Approximation and estimation bounds for artificial neural networks. *Machine Learning*, 14(1):115–133, 1994.

[3] Yuhong Yang and Andrew Barron. Information-theoretic determination of minimax rates of convergence. *Annals of Statistics*, pages 1564–1599, 1999.

[4] Francis Bach. Breaking the Curse of Dimensionality with Convex Neural Networks. Research report, INRIA Paris, December 2014.

[5] Yoshua Bengio, Nicolas L Roux, Pascal Vincent, Olivier Delalleau, and Patrice Marcotte. Convex neural networks. In *Advances in neural information processing systems*, pages 123–130, 2005.

[6] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[7] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[8] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[9] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[10] Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *Information Theory, IEEE Transactions on*, 39(3):930–945, 1993.

[11] Pencho P Petrushev. Approximation by ridge functions and neural networks. *SIAM Journal on Mathematical Analysis*, 30(1):155–189, 1998.

[12] Allan Pinkus. Approximation theory of the mlp model in neural networks. *Acta Numerica*, 8:143–195, 1999.

[13] Ronald A DeVore, Ralph Howard, and Charles Micchelli. Optimal nonlinear approximation. *Manuscripta mathematica*, 63(4):469–478, 1989.

[14] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[15] Y Makovoz. Random approximants and neural networks. *Journal of Approximation Theory*, 85(1):98–109, 1996.

[16] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *The Journal of Machine Learning Research*, 3:463–482, 2003.