Notes on Theory of Perceptron and Neural Networks

Evan Kim

May 19th, 2020

Contents

1 Learning Models

1.1 The Learning Problem

A learning algorithm takes random training samples and acts on these to produce a hypothesis function $h \in H$ that, provided the sample is large enough, is, with probability at least $1 - \delta$, ϵ-good for a probability distribution P. The learning algorithm can do this for each choice of ϵ and δ regardless of the distribution P.

Suppose H is a class of functions that map from a set X to $\{0, 1\}$. A learning algorithm L for H is a function

$$L : \bigcup_{m=1}^{\infty} Z^m \to H$$

with the following property that given any $\epsilon, \delta \in (0, 1)$, there is an integer $m_0(\epsilon, \delta)$ such that if $m \geq m_0(\epsilon, \delta)$, then for any probability distribution P on $Z = X \times \{0, 1\}$, if z is a training sample of length m drawn randomly according to the product probability distribution P^m, then with probability at least $1 - \delta$, L outputs the hypothesis $L(z)$ such that

$$\mathrm{er}_P(L(z)) < \mathrm{opt}_P(H) + \epsilon$$

Alternatively stated, for $m \geq m_0(\epsilon, \delta)$,

$$P^m\{\mathrm{er}_P(L(z)) < \mathrm{opt}_P(H) + \epsilon\} \geq 1 - \delta$$

1.2 Neural Network Definition

A neural network is characterized by a set Ω of states, a set X of inputs, a set Y of outputs, and a parametrized function $F : \Omega \times X \to Y$. For any $\omega \in \Omega$, the function represented by ω is $h_\omega : X \to Y$:

$$h_\omega(x) = F(\omega, x)$$

The function F describes the functionality of the network. When the network is in state ω, it computes the function h_ω

1.3 Perceptrons

A simple perceptron computes a function f from $\mathbb{R}^n$ to $\{0, 1\}$ of the form

$$f(x) = \mathrm{sgn}(w \cdot x - \theta)$$

for an input vector $x \in \mathbb{R}^n$, where $w = (w_1, \ldots, w_n) \in \mathbb{R}^n$ and $\theta \in \mathbb{R}$ are adjustable parameters called *weights*. $w \cdot x$ denotes the standard inner product with orthogonality defined as

$$\mathrm{sgn}(\alpha) = \begin{cases} 1 \text{ if } \alpha \geq 0 \\ 0 \text{ otherwise} \end{cases}$$

The *decision boundary* is the boundary between the set of points classified as 0 and 1 and is the affine subspace of $\mathbb{R}^n$ defined by the equation $w \cdot x - \theta = 0$.

1.4 Two-Layer Sigmoid Network (Perceptron with weights and inputs in $\mathbb{R}$)

The two-layer real-output sigmoid network computes a function f from $\mathbb{R}^n \to \mathbb{R}$

$$f(x) = \sum_{i=1}^{k} w_i \sigma(v_i \cdot x + v_{i,0}) + w_0$$

where $x \in \mathbb{R}^n$ is the input vector, $v_i \in \mathbb{R}^n$ and $v_{i,0}$ for $i = 0, \ldots, k$ are input weights, $w_i \in \mathbb{R}$ are output weights, and the activation function is σ, the standard sigmoid function[1] given by

$$\sigma(\alpha) = \frac{1}{1 + e^{-\alpha}}$$

Each of the mappings $x \mapsto \sigma(v_i \cdot x + v_{i,0})$ can be thought of as a smoothed version of the function computed by a simple perceptron. Thus the two-layer sigmoid network computes an affine combination of these 'squashed' affine functions. Unlike the perceptron output of $\{0, 1\}$, the output is a real number $w_i \in [0, 1]$.

2 Measuring function class complexity

Many types of neural networks can be represented as a parametrized function class[2] with an infinite parameter set. We can use the growth function and VC-dimension to measure the complexity of any finite or infinite function class.

2.1 The Growth Function

Let S be a finite subset of the input space X and H be a function class. The restriction of H to the set S is denoted $H_{|S}$. We view the cardinality of $H_{|S}$[3] as a measure of the classification complexity of H with respect to the set S. The growth function H is defined as $\Pi_H : \mathbb{N} \to \mathbb{N}$

$$\Pi_H(m) = \max\{|H_{|S}| : S \subseteq X \text{ and } |S| = m\}$$

2.2 The Growth Function Applied on the Perceptron

Theorem 3.1 states that if N is the real-weight simple perceptron with $n \in \mathbb{N}$ real inputs and H is the set of functions it computes, then

$$\Pi_H(m) = 2 \sum_{k=0}^{n} \binom{m-1}{k}$$

In order to prove *Theorem 3.1*, there are three steps. The first is to show that the number of dichotomies of a set of m points is the same as the number of cells in a certain partition of the parameter space defined by the points. Then count the number of these cells when the points are in general position. Then show that general position is sufficient enough because any m-tuple points in $\mathbb{R}^n$ not in general position will have Lebesgue measure zero when regarded as a subset of $\mathbb{R}^{mn}$.

[1] **5.20.20 - What makes the sigmoid function so 'easy' to compute?**

[2] what is this?

[3] and how it compares with $2^{|S|}$

2.3 VC-Dimension

The Vapnik-Chervonenkis dimension is a combinatorial quantity that measures the complexity of predicting a binary-valued quantity with binary-valued functions such as the perceptron. The VC-dimension of H is the size of the largest shattered subset of X, assuming that H can compute all dichotomies of a set S of m points. Equivalently it is also the largest value of m for which the growth function $\Pi_H(m) = 2^m$.

Theorem 3.6[4] - For a function class H with VCdim(H) $= d$,

$$\Pi_H(m) \leq \sum_{i=0}^{d} \binom{m}{i}$$

A following corollary can be made which explicitly states that a function class of finite VC-dimension has polynomially-bounded growth function and that the log of the growth function is within a log factor of the VC-dimension.

3 General Upper Bounds on Sample Complexity

3.1 Error and sample error

Given a function $h \in H$, the error of h with respect to P is defined as

$$er_P(h) = P\{(x, y) \in Z \ : \ h(x) \neq y\}$$

This is the probability that, for (x, y) drawn randomly according to P, that h is 'wrong' in the sense that $h(x) \neq y$. $er_P(h)$ is a measure of how accurately h approximates the relationships between patterns and labels generated by P.

The sample error of h on the sample z is the proportion of labelled examples (x_i, y_i) in the training sample z on which h is 'wrong'. It is defined as

$$\hat{er}_z(h) = \frac{1}{m} \left| \{i \ : \ 1 \leq i \leq m \ \& \ h(x_i) \neq y_i\} \right|$$

The sample error is a useful quantity since it can be easily determined from the training data and provides a simple estimate for the true error $er_P(h)$.
The approximation error of the class H is

$$opt_P(H) = \inf_{h \in H} er_P(h)$$

or equivalently

$$er_P(h) < opt_P(H) + \epsilon$$

$opt_P(H)$ describes how accurately the best function in H can approximate the relationship between x and y that is determined by the probability distribution P.

[4] Bartlett, pg 39

3.2 Sample Error Minimization Algorithm

The Sample Error Minimization Algorithm (SEM) for H is defined as any function $L : \bigcup_{m=1}^{\infty} Z^m \rightarrow H$ with the property that for any $m \in \mathbb{Z}$ and training sample $z \in Z^m$, then

$$\hat{er}_z(L(z)) = \min_{h \in H} \hat{er}_z(h)$$

Finite function classes use cardinality to bound the estimation error and sample complexity of SEM algorithms. In the case when H is infinite, for H of finite VC-dimension, the estimation error and sample complexity of any SEM algorithm can be bounded in terms of the VC-dimension of H.

3.3 ∞-Function Class Bounds

The following theorem is a very general result and the bound applies to all function classes H with finite VC-dimension.

Theorem 4.2[5] Suppose that H is a set of functions from a set X to $\{0, 1\}$ and that H has finite VC-dimension. Let L be any sample error minimization algorithm for H. Then L is a learning algorithm for H. In particular, if $m \geq d/2$, then the estimation error of L satisfies

$$\epsilon_L(m, \delta) \leq \epsilon_0(m, \delta) = \left(\frac{32}{m}(d \ln(2em/d) + \ln(4/\delta)) \right)^{1/2}$$

and its sample complexity satisfies the inequality

$$m_L(\epsilon, \delta) \leq m_0(\epsilon, \delta) = \frac{64}{\epsilon^2}\left(2d \ln(12/\epsilon) + \ln(4/\delta) \right)$$

The crux of *Theorem 4.2* towards proving learnability is to obtain a result on the uniform convergence of sample errors to true errors when H is an infinite function class stated as follows:

Theorem 4.3[6] - Suppose that H is a set of $\{0, 1\}$-valued functions defined on a set X and that P is a probability distribution on $Z = X \times \{0, 1\}$. For $0 < \epsilon < 1$ and a positive integer m, we have

$$P^m\{ |er_p(h) - \hat{er}_z(h)| \geq \epsilon \text{ for some } h \in H \ \} \leq 4\Pi_H(2m) \exp\left(-\frac{\epsilon^2 m}{8} \right)$$

Note that if $\Pi_H(2m)$ grows exponentially quickly in m, then the bound is trivial and never drops below 1. On the other hand, if $\Pi_H(2m)$ grows polynomially quickly in m, the bound goes to zero exponentially fast.

4 General Lower Bounds on Sample Complexity

The consequence of Theorem 5.2 completes other direction (other direction is in chapter 3) shows that if a class of function is learnable, then it is has a finite VC-dimension. The proof uses the

[5] Bartlett, pg 43
[6] Bartlett, pg 44

probabilistic method technique for proving the existence of objects with certain properties starting with the assumption that the underlying probability distribution P is drawn uniformly at random from a finite class of distributions $\mathcal{P}$. Then it can be shown that for any learning algorithm and distribution P, the probability of failure is at least δ if we concentrate the distribution on a shattered set and set the conditional probability to $Pr(y = 1|x) = \frac{1+c\epsilon}{2}$ for some constant c for each point x in the shattered set. To obtain a near-optimal error, the algorithm must estimate the conditional probabilities with an accuracy of $c\epsilon$ for a significant proportion of points. Lemma 5.1 shows that this means that the algorithm must give order $\frac{1}{\epsilon^2}$ examples of each point.

Lemma 5.1[7] estimates a Bernoulli random variable parameter and shows that for every function f that is a decision rule, there exists a limitation on accuracy that is dependent on the similarity of the two choices (ϵ) and the amount of data (m) and is stated as follows:

Suppose that α is a random variable uniformly distributed on $\{\alpha_-, \alpha_+\}$, where $\alpha_- = \frac{1}{2} - \frac{\epsilon}{2}$ and $\alpha_+ = \frac{1}{2} + \frac{\epsilon}{2}$ for $0 < \epsilon < 1$. Suppose that x_i for $i = 1, \ldots, m$ are i.i.d $\{0, 1\}$ Bernoulli random variables with $Pr(x_i = 1) = \alpha$. Let f be a function from $\{0, 1\}^m$ to $\{a_-a_+\}$. Then

$$Pr\left(f(x_1, \ldots, x_m) \neq \alpha\right) > \frac{1}{4}\left(1 - \sqrt{1 - \exp\left(\frac{-2\lceil m/2\rceil\epsilon^2}{1-\epsilon^2}\right)}\right)$$

Hence if the probability is no more than δ for $0 < \delta < \frac{1}{4}$, then

$$m \geq 2\left\lfloor \frac{1-\epsilon^2}{2\epsilon^2} \ln\left(\frac{1}{8\delta(1-2\delta)}\right)\right\rfloor$$

Theorem 5.2[8] Suppose that H is a class of $\{0, 1\}$ valued functions and H has a finite VC-dimensions D. For any learning algorithm L of H, the sample complexity of L, $m_L(\epsilon, \delta)$, is bounded below by

$$m_L(\epsilon, \delta) \geq \frac{d}{320\epsilon^2} \quad \forall\, 0 < \epsilon, \quad 0 < \delta < \frac{1}{4}$$

Furthermore, if H contains at least two functions, then the bound becomes

$$m_L(\epsilon, \delta) \geq 2\left\lfloor \frac{1-\epsilon^2}{2\epsilon^2} \ln\left(\frac{1}{8\delta(1-2\delta)}\right)\right\rfloor$$

The proof starts with the assumptions that there exists a shattered set of size d because H has finite VC-dimension and defines $\mathcal{P}$ as a class of distributions of P that have the following properties given random variables x_i for $i = 1, \ldots, d$:

- P assigns 0 probability to all sets not intersecting $S \times \{0, 1\}$
- for $1 \leq i \leq d$ and parameter $0 < \alpha < 1$, either

$$P(x_i, 1) = \frac{1+\alpha}{2d} \;\&\; P(x_i, 0) = \frac{1-\alpha}{2d}$$

[7] pg 59, Bartlett
[8] Bartlett, pg 62

or

$$P(x_i, 1) = \frac{1-\alpha}{2d} \;\&\; P(x_i, 0) = \frac{1+\alpha}{2d}$$

The following observation can be made that given a distribution $P \in \mathcal{P}$, the optimal error $opt_P(H)$ is achieved by any function $h^* \in H$ for which $h^*(x_i) = 1$ if and only if $P(x_i, 1) = \frac{1+\alpha}{2d}$ and we get

$$opt_P(H) = er_P(h^*) = P\{h^*(x) \neq y\} = \sum_{n=1}^{d} \frac{1-\alpha}{2d} = \frac{1}{2} - \frac{\epsilon}{2}$$

Similarly, the true error for any $h \in H$ is the sum of the possible values of the distributions

$$er_P(h) = \sum_{i=1}^{d} \left(\frac{1+\alpha}{2d} - \frac{1-\alpha}{2d} \right) = er_P(h^*) + \frac{\alpha}{d} \sum_{i=1}^{d} x_i$$

For any sample $z \in Z^m$, consider the number of occurrences $(x_i, 1)$ or $(x_i, 0)$ that occur in z and denote the number as $N(z) = \Big(N_1(z), \ldots, N_d(z) \Big)$. Then for any $h = L(z)$,

$$\begin{aligned} E\left(\frac{1}{d} \sum_{i=1}^{d} 1_{h(x_i) \neq h^*(x_i)}(x_i) \right) &= \frac{1}{d} \sum_{i=1}^{d} E(1_{h(x_i) \neq h^*(x_i)}(x_i)) \\ &= \frac{1}{d} \sum_{N} \sum_{i=1}^{d} Pr(h(x_i) \neq h^*(x_i) | N(z) = N) \cdot Pr(N(z) = N) \end{aligned}$$

where $N = (N_1, \ldots, N_d)$ ranges over the set of d-tuples of positive integers and $\sum_{i=1}^{d} N_i = m$. Applying Lemma 5.1,

$$\begin{aligned} &Pr(h(x_i) \neq h^*(x_i) | N(z) = N) \\ &= Pr(h(x_i) \neq h^*(x_i) | N_i(z) = N_i) > \frac{1}{4} \left(1 - \sqrt{1 - \exp\left(- \frac{(N_i + 1)\epsilon^2}{1 - \epsilon^2} \right)} \right) \end{aligned}$$

Since this is a convex function of N_i, we can apply Jensens inequality to get

$$E\left(\frac{1}{d} \sum_{i=1}^{d} 1_{h(x_i) \neq h^*(x_i)}(x_i) \right) > \frac{1}{4} \left(1 - \sqrt{1 - \exp\left(- \frac{(N_i + 1)\epsilon^2}{1 - \epsilon^2} \right)} \right)$$

Then the final part for the first inequality of the theorem comes from using the fact that any $[0, 1]$-valued random variable Z satisfies

$$Pr(Z > \lambda) \geq \frac{\mathbf{E}Z - \lambda}{1 - \lambda} > \mathbf{E}Z - \lambda$$

The proof for the second inequality is simpler and shows first that learning to an accuracy of ϵ is equivalent to guessing which distribution generated the labelled example. Then choose a probability distribution P uniformly at random and apply Lemma 5.1.