CMSE 890-002: Mathematics of Deep Learning, MSU, Spring 2020

Lecture 15: One-Layer Neural Network Approximation Theory

February 14, 2020

Lecturer: Matthew Hirn

8.1 One layer approximation theory

One of the earlier and most popular works on approximation properties of artificial neural networks is by George Cybenko [11]. The result proves that one layer neural networks with sigmoid-like activation functions can approximate any continuous function on the unit cube $\mathcal{X} = [0,1]^d$. Let us discuss the main points of this result.

In this section we consider one layer neural network regression functions. Let $\mathbf{W}$ be a $d \times m$ weight matrix, b an $m \times 1$ bias vector, and α an $m \times 1$ weight vector, which we write as:

$$\mathbf{W} = \begin{pmatrix} | & & | \\ w_1 & \cdots & w_m \\ | & & | \end{pmatrix} \quad b = \begin{pmatrix} b(1) \\ \vdots \\ b(m) \end{pmatrix} \quad \alpha = \begin{pmatrix} \alpha(1) \\ \vdots \\ \alpha(m) \end{pmatrix}.$$

Our one layer neural networks take the form:

$$f(x; \mathbf{W}, b, \alpha) = \langle \sigma(\mathbf{W}^T x + b), \alpha \rangle = \sum_{k=1}^{m} \alpha(k) \sigma(\langle x, w_k \rangle + b(k)), \tag{24}$$

where $\sigma : \mathbb{R} \to \mathbb{R}$ is a *sigmoidal function*, meaning that

$$\sigma(z) \to \begin{cases} 1 & \text{as } z \to +\infty \\ 0 & \text{as } z \to -\infty \end{cases} \tag{25}$$

Note, in particular, that the regular sigmoid function is a sigmoidal function.

We will initially consider labeling functions of the form

$$y = F(x), \quad F \in \mathbf{C}[0,1]^d,$$

where $\mathbf{C}[0,1]^d$ consists of all continuous functions $F : [0,1]^d \to \mathbb{R}$. We have the following theorem.

Theorem 8.1 (Cybenko 1989, [11]). *Let σ be any continuous sigmoidal function (25). Given an $F \in \mathbf{C}[0,1]^d$ and an $\epsilon > 0$, there is a one layer neural network $f(x; \mathbf{W}, b, \alpha)$ of the form (24) with m, $\mathbf{W} \in \mathbb{R}^{d \times m}$, $b \in \mathbb{R}^m$, and $\alpha \in \mathbb{R}^m$ depending on d, F, and ϵ, for which*

$$|f(x; \mathbf{W}, b, \alpha) - F(x)| < \epsilon, \quad \text{for all } x \in [0,1]^d.$$

We will comment on the poof of Theorem 8.1 at the end of this section. The theorem shows that any continuous label function $y = F(x)$ supported on the unit cube can be approximated by a one layer sigmoidal neural network to arbitrary accuracy. This type of result is often referred to as a "universal approximation" theorem. It can extended to perceptrons on other discontinuous sigmoid functions if we replace the $\mathbf{L}^\infty[0,1]^d$ error of Theorem 8.1 with an $\mathbf{L}^1[0,1]^d$ error. This result initially sounds very impressive, but as we will see later it in fact is not that meaningful in explaining why neural networks work so well (in fairness, though, it is one of the first such results).

First though, let us first use Theorem 8.1 to prove a result about categorical classification. Let $\mathcal{C}_1, \ldots, \mathcal{C}_M \subseteq [0,1]^d$ be a partition of the cube $[0,1]^d$, meaning that

$$\mathcal{C}_i \cap \mathcal{C}_j = \emptyset \text{ if } i \neq j \quad \text{and} \quad \bigcup_{i=1}^{M} \mathcal{C}_i = [0,1]^d \,.$$

Assign labels as:

$$y = F(x) = i \quad \text{if and only if} \quad x \in \mathcal{C}_i \,. \tag{26}$$

We refer to F in this case as a *decision function*. Note that this is a classification problem. We want to know if a neural network can implement a good decision boundary. The following theorem says it can.

Theorem 8.2 (Cybenko 1989, [11]). *Let σ be any continuous sigmoidal function (25). Let F be a decision function (26) and $\epsilon > 0$. Then, there exists a one layer neural network $f(x; \mathbf{W}, b, \alpha)$ of the form (24) with m, $\mathbf{W} \in \mathbb{R}^{d \times m}$, $b \in \mathbb{R}^m$, and $\alpha \in \mathbb{R}^m$ depending on d, F, and ϵ, and a set $\mathcal{D} \subseteq [0,1]^d$ with $\text{vol}(\mathcal{D}) \geq 1 - \epsilon$, for which*

$$|f(x; \mathbf{W}, b, \alpha) - F(x)| < \epsilon \,, \quad \text{for all } x \in \mathcal{D} \,.$$

As a consequence, define the classifier $\widetilde{f}(x; \mathbf{W}, b, \alpha)$ as:

$$\widetilde{f}(x; \mathbf{W}, b, \alpha) = \text{Round}[f(x; \mathbf{W}, b, \alpha)] \,,$$

where $\text{Round}[z]$ is the closest integer to $z \in \mathbb{R}$. If $\epsilon < 1/2$, then

$$\widetilde{f}(x; \mathbf{W}, b, \alpha) = F(x) \,, \quad \text{for all } x \in \mathcal{D} \,.$$

Proof sketch. Use Lusin's theorem combined with Theorem 8.1 to prove the result for f. The result for $\widetilde{f}$ follows immediately. □

Because $f(x; \mathbf{W}, b, \alpha)$ is always a continuous function and a decision function $F(x)$ is necessarily not, there will always be some points classified incorrectly. On the other hand, the result says the volume of the number of points classified incorrectly can be made arbitrarily small. While this theorem does not say anything about the geometry of the set $\mathcal{D}$, one can refine the analysis further to conclude that the one layer neural network can learn a "natural"

approximation of $F(x)$ where any point sufficiently far away from a boundary defined by $F(x)$ is classified correctly.

The key to Cybenko's results in [11] is the notion of a *discriminatory function*. We give the definition here without explaining everything since it relies on notions from graduate level analysis. Cybenko says that a function $\sigma : \mathbb{R} \to \mathbb{R}$ is discriminatory if for any finite, signed Borel measure μ on $[0,1]^d$,

$$\text{for all } w \in \mathbb{R}^d\,,\ b \in \mathbb{R}\,,\ \int_{[0,1]^d} \sigma(\langle x, w\rangle + b)\, d\mu(x) = 0 \quad \Longrightarrow \quad \mu \equiv 0\,.$$

Cybenko proves that sigmoidal functions are discriminatory and then uses this to obtain his results. An example such a measure is $d\mu(x) = p_X(x)\, dx$, where $p_X : [0,1]^d \to \mathbb{R}$ is a probability density function on $\mathcal{X} = [0,1]^d$, but there are other more exotic finite, signed Borel measures.

Cybenko's results were generalized by Hornik in [12]. Among his results, he proves the following.

Theorem 8.3 (Hornik 1991, [12][13])**.** *Let σ be any non-constant, bounded activation function (e.g., sigmoidal, but others are okay too). Let p_X be a probability density function on $[0,1]^d$ and let $F \in \mathbf{L}^2([0,1]^d, p_X)$, which means that*

$$\mathbb{E}_X[|F(X)|^2] = \int_{[0,1]^d} |F(x)|^2 p_X(x)\, dx < \infty\,.$$

Then for each $\epsilon > 0$ there exists a one layer neural network $f(x; \mathbf{W}, b, \alpha)$ of the form (24) with m, $\mathbf{W} \in \mathbb{R}^{d\times m}$, $b \in \mathbb{R}^m$, and $\alpha \in \mathbb{R}^m$ depending on d, F, ϵ, and p_X, such that

$$\mathbb{E}_X[(F(X) - f(X; \mathbf{W}, b, \alpha))^2] = \int_{[0,1]^d} (F(x) - f(x; \mathbf{W}, b, \alpha))^2 p_X(x)\, dx < \epsilon\,.$$

Most of the proofs contained in [11] and [12] are not constructive and they give no insight into the relationship between m, the number of neurons, and ϵ, the desired accuracy. They also give no relationship between the magnitude of the weights $\mathbf{W}$ and biases b and ε. We are interested in this type of analysis because each weight and bias needs to be learned from training data, and thus the number of such weights is a proxy for the amount of training data we will need. The magnitude of these values is also important, as computers cannot store infinitely large values. We will see that these relationships are not favorable, at least by the analysis of this section.

[13] This result is Theorem 1 in [12], which says "unbounded" but should say "bounded." Thanks to Gautam Sreekumar asking about this result in class!

References

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations*, 2015.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, 2009.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012.

[4] Larry Greenemeier. AI versus AI: Self-taught AlphaGo Zero vanquishes its predecessor. *Scientific American*, October 18, 2017.

[5] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre G.R. Day, Clint Richardson, Charles K. Fisher, and David J. Schwab. A high-bias, low-variance introduction to Machine Learning for physicists. arXiv:1803.08823, 2018.

[6] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer-Verlag New York, 2nd edition, 2009.

[7] J. Hartlap, P. Simon, and P. Schneider. Why your model parameter confidences might be too optimistic - unbiased estimation of the inverse covariance matrix. *Astronomy and Astrophysics*, 464(1):399–404, 2007.

[8] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning. The MIT Press, 2002.

[9] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[10] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations*, San Diego, CA, USA, 2015.

[11] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2:303–314, 1989.

[12] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4:251–257, 1991.