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

Lecture 19: On to Two-Layer Networks

February 24, 2020

Lecturer: Matthew Hirn

8.3.3 Two-layer neural networks

It is tempting to say that one-layer neural networks are universal approximators and therefore what else is there to say regarding the additional layer. But this ignores both the reality of the situation, which is that in practice most applications utilize several to many layers, as well as fundamental theoretical differences between one-layer neural networks and multi-layer neural networks. In this subsection we study two layer neural networks in more depth.

One important difference between a one-layer network and a two-layer network is that in one-layer networks each neuron is delocalized, whereas in two layers the composition of two neurons can lead to a localized function. Let us make this statement more precise. Let d_1 be the number of neurons in a one-layer neural network. Recall the space $\mathcal{M}_{d_1}(\sigma)$ consists of one layer neural networks with at most d_1 neurons:

$$f(x;\theta) = \sum_{k=1}^{d_1} \alpha(k)\sigma(\langle x, w_k\rangle + b(k)) = \sum_{k=1}^{d_1} \alpha(k)\sigma\left(\sum_{j=1}^{d} w_k(j)x(j) + b(k)\right) .$$

Let $x \in \mathbb{R}^d$ for $d \geq 2$ and let $f \in \mathcal{M}_{d_1}(\sigma)$, f not the function that is zero everywhere. Then, necessarily, we have

$$\forall\, 1 \leq p \leq \infty\,, \quad \int_{\mathbb{R}^d} |f(x)|^p\, dx = +\infty\,,$$

and in particular no $f \in \mathcal{M}_{d_1}(\sigma)$ has compact support.

In the two layer model the situation is different and in fact we can obtain compactly supported functions (or more precisely, functions whose support is contained in a compact set) in the second hidden layer. Recall, by a two layer neural network, we mean a function of the form:

$$\begin{aligned} f(x;\theta) &= \alpha^T\sigma(\mathbf{W}_2^T\sigma(\mathbf{W}_1^T x + b_1) + b_2) \\ &= \sum_{\ell=1}^{d_2} \alpha(\ell)\sigma\left(\sum_{k=1}^{d_1} w_{2,\ell}(k)\sigma(\langle x, w_k\rangle + b_1(k)) + b_2(\ell)\right) \\ &= \sum_{\ell=1}^{d_2} \alpha(\ell)\sigma\left(\sum_{k=1}^{d_1} w_{2,\ell}(k)\sigma\left(\sum_{j=1}^{d} w_{1,k}(j)x(j) + b_1(k)\right) + b_2(\ell)\right) . \end{aligned}$$

For example, let $\sigma_o(z)$ be the perceptron, that is

$$\sigma_o(z) = \begin{cases} 0 & z \leq 0 \\ 1 & z > 0 \end{cases}$$

Then

$$\sigma_o\left(\sum_{k=1}^{d_1} \sigma_o(\langle x, w_k\rangle - b(k)) - \left(d_1 - \frac{1}{2}\right)\right) = \begin{cases} 1 & \langle x, w_k\rangle > b(k)\,, \ \ \forall\, 1 \leq k \leq d_1 \\ 0 & \text{otherwise} \end{cases}$$

Thus with two hidden layers and the perceptron activation function we can represent the characteristic function of any convex polygonal domain; see Figure 26. If we replace the perceptron with the sigmoid function $\sigma(z)$, then we have

$$\lim_{\lambda \to +\infty} \int_{-1}^{1} |\sigma(\lambda z) - \sigma_o(z)|^2\, dz = 0\,.$$

In other words, the sigmoid function approximates the perceptron and thus the function

$$\sigma\left(\lambda \sum_{k=1}^{d_1} \sigma(\lambda(\langle x, w_k\rangle - b(k))) - \left(d_1 - \frac{1}{2}\right)\right)$$

approximates the analogous perceptron function as $\lambda \to +\infty$. It follows that the sigmoid version is a highly localized function, which gives two layer networks an advantage over one-layer networks.

Theoretically, we observed for one-layer neural networks there is an intrinsic lower bound on the rate of approximation, which depends on the number of neurons, when one considers any label function $F \in \mathbf{C}^2(\mathbb{R}^d)$. This limitation is removed in the two layer model, as the following theorem shows.

Theorem 8.13 (Maiorov and Pinkus 1999, [15])**.** *There exists and activation function $\sigma \in \mathbf{C}^\infty(\mathbb{R})$ that is sigmoidal and strictly increasing, and has the following property. For any $F \in \mathbf{C}[0,1]^d$ and $\epsilon > 0$, there exists a two-layer neural network $f(x;\theta)$ with activation function σ, with $d_1 = (2d+1)(4d+3)$ neurons in the first hidden layer, and with $d_2 = 4d+3$ neurons in the second layer, for which*

$$\sup_{x \in [0,1]^d} |F(x) - f(x;\theta)| < \epsilon\,.$$

This theorem is pretty extraordinary. Some remarks are in order.

Remark 8.14. The activation function σ is the same as the one in Theorem 8.11, and is thus, unfortunately, not practical.

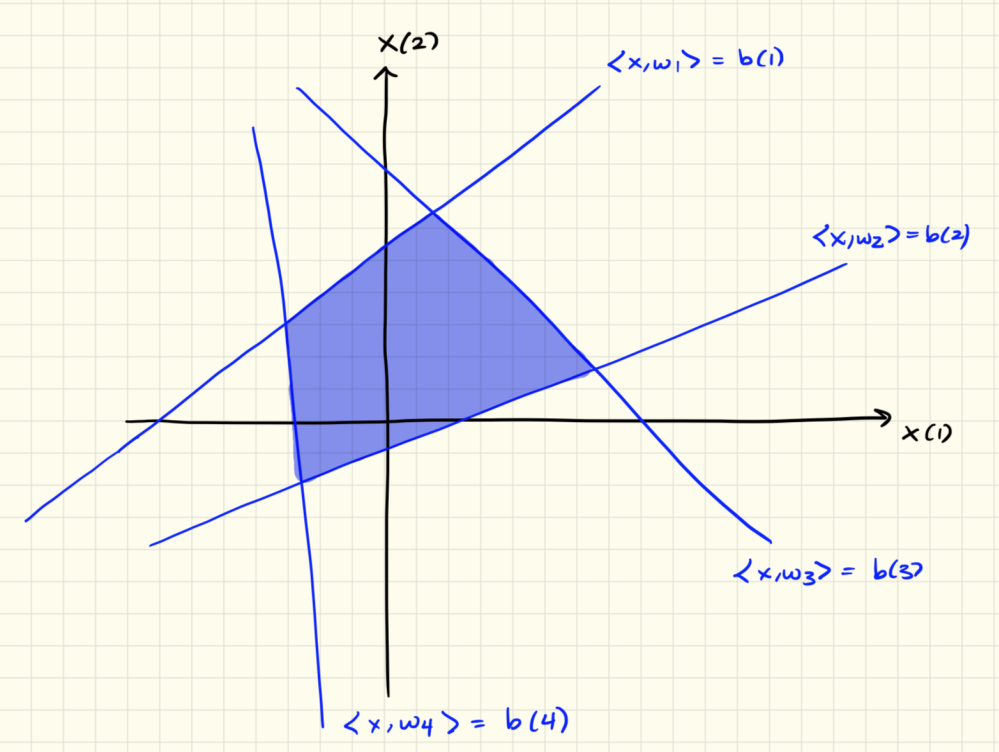

Figure 26: *A two layer perceptron network can implement the characteristic function on any convex polygonal domain.*

Remark 8.15. The number of neurons is independent of the label function F (similar to previous one-layer results) and the accuracy ϵ (very different from previous one-layer results), and grows quadratically in the dimension d. As expected, the weights of the neurons depend on F and ϵ. While it is remarkable that the number of neurons does not depend on the accuracy ϵ, it is worth examining how the magnitude of the weights depends on ϵ. It turns out the size of the weights grows very fast with epsilon, and the weights become so huge that they could not hope to be stored on a computer!

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.

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

[14] Vitaly E. Maiorov. On best approximation by ridge functions. *Journal of Approximation Theory*, 99:68–94, 1999.

[15] Vitaly E. Maiorov and Allan Pinkus. Lower bounds for approximation by mlp neural networks. *Neurocomputing*, 25:81–91, 1999.

[16] Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 907–940. PMLR, 2016.