Universal Approximation Doesn't Mean Universal Learning!

A critical subtlety exists in the universal approximation theorem. The fact that a fully connected network can represent any function doesn't mean that backpropagation can learn any function! One of the major limitations of backpropagation is that there is no guarantee the fully connected network "converges"; that is, finds the best available solution of a learning problem. This critical theoretical gap has left generations of computer scientists queasy with neural networks. Even today, many academics will prefer to work with alternative algorithms that have stronger theoretical guarantees.

Empirical research has yielded many practical tricks that allow backpropagation to find good solutions for problems. We will go into many of these tricks in significant depth in the remainder of this chapter. For the practicing data scientist, the universal approximation theorem isn't something to take too seriously. It's reassuring, but the art of deep learning lies in mastering the practical hacks that make learning work.

Why Deep Networks?

A subtlety in the universal approximation theorem is that it in fact holds true for fully connected networks with only one fully connected layer. What then is the use of "deep" learning with multiple fully connected layers? It turns out that this question is still quite controversial in academic and practical circles.

In practice, it seems that deeper networks can sometimes learn richer models on large datasets. (This is only a rule of thumb, however; every practitioner has a bevy of examples where deep fully connected networks don't do well.) This observation has led researchers to hypothesize that deeper networks can represent complex functions "more efficiently." That is, a deeper network may be able to learn more complex functions than shallower networks with the same number of neurons. For example, the ResNet architecture mentioned briefly in the first chapter, with 130 layers, seems to outperform its shallower competitors such as AlexNet. In general, for a fixed neuron budget, stacking deeper leads to better results.

A number of erroneous "proofs" for this "fact" have been given in the literature, but all of them have holes. It seems the question of depth versus width touches on profound concepts in complexity theory (which studies the minimal amount of resources required to solve given computational problems). At present day, it looks like theoretically demonstrating (or disproving) the superiority of deep networks is far outside the ability of our mathematicians.

Training Fully Connected Neural Networks

As we mentioned previously, the theory of fully connected networks falls short of practice. In this section, we will introduce you to a number of empirical observations about fully connected networks that aid practitioners. We strongly encourage you to use our code (introduced later in the chapter) to check our claims for yourself.

Learnable Representations

One way of thinking about fully connected networks is that each fully connected layer effects a transformation of the feature space in which the problem resides. The idea of transforming the representation of a problem to render it more malleable is a very old one in engineering and physics. It follows that deep learning methods are sometimes called "representation learning." (An interesting factoid is that one of the major conferences for deep learning is called the "International Conference on Learning Representations.")

Generations of analysts have used Fourier transforms, Legendre transforms, Laplace transforms, and so on in order to simplify complicated equations and functions to forms more suitable for handwritten analysis. One way of thinking about deep learning networks is that they effect a data-driven transform suited to the problem at hand.

The ability to perform problem-specific transformations can be immensely powerful. Standard transformation techniques couldn't solve problems of image or speech analysis, while deep networks are capable of solving these problems with relative ease due to the inherent flexibility of the learned representations. This flexibility comes with a price: the transformations learned by deep architectures tend to be much less general than mathematical transforms such as the Fourier transform. Nonetheless, having deep transforms in an analytic toolkit can be a powerful problem-solving tool.

There's a reasonable argument that deep learning is simply the first representation learning method that works. In the future, there may well be alternative representation learning methods that supplant deep learning methods.

Activations

We previously introduced the nonlinear function σ as the sigmoidal function. While the sigmoidal is the classical nonlinearity in fully connected networks, in recent years researchers have found that other activations, notably the rectified linear activation (commonly abbreviated ReLU or relu) $\sigma(x) = \max(x, 0)$ work better than the sigmoidal unit. This empirical observation may be due to the *vanishing gradient* problem in deep networks. For the sigmoidal function, the slope is zero for almost all values of its input. As a result, for deeper networks, the gradient would tend to zero. For the ReLU function, the slope is nonzero for a much greater part of input space, allowing non-