

CHAPTER 11

Training Deep Neural Nets

In [Chapter 10](#) we introduced artificial neural networks and trained our first deep neural network. But it was a very shallow DNN, with only two hidden layers. What if you need to tackle a very complex problem, such as detecting hundreds of types of objects in high-resolution images? You may need to train a much deeper DNN, perhaps with (say) 10 layers, each containing hundreds of neurons, connected by hundreds of thousands of connections. This would not be a walk in the park:

- First, you would be faced with the tricky *vanishing gradients* problem (or the related *exploding gradients* problem) that affects deep neural networks and makes lower layers very hard to train.
- Second, with such a large network, training would be extremely slow.
- Third, a model with millions of parameters would severely risk overfitting the training set.

In this chapter, we will go through each of these problems in turn and present techniques to solve them. We will start by explaining the vanishing gradients problem and exploring some of the most popular solutions to this problem. Next we will look at various optimizers that can speed up training large models tremendously compared to plain Gradient Descent. Finally, we will go through a few popular regularization techniques for large neural networks.

With these tools, you will be able to train very deep nets: welcome to Deep Learning!

Vanishing/Exploding Gradients Problems

As we discussed in [Chapter 10](#), the backpropagation algorithm works by going from the output layer to the input layer, propagating the error gradient on the way. Once the algorithm has computed the gradient of the cost function with regards to each