You can see how the new trainable variables and the dropout operation are represented here. Everything looks to be in the right place. Let's end now by looking at the loss curve over time (Figure 4-12).

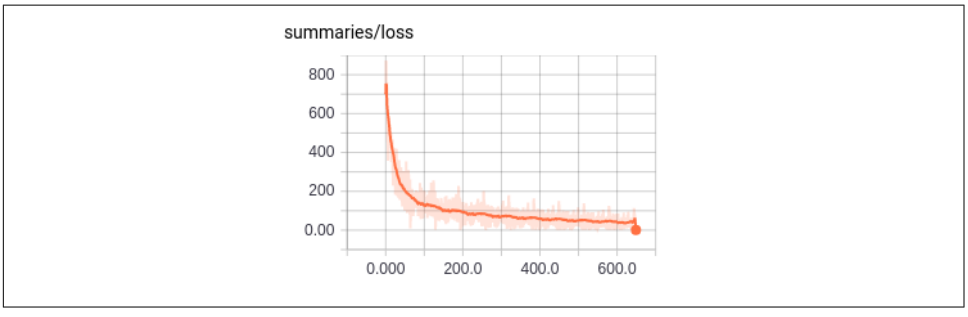

Figure 4-12. Visualizing the loss curve for a fully connected network.

The loss curve trends down as we saw in the previous section. But, let's zoom in to see what this loss looks like up close (Figure 4-13).

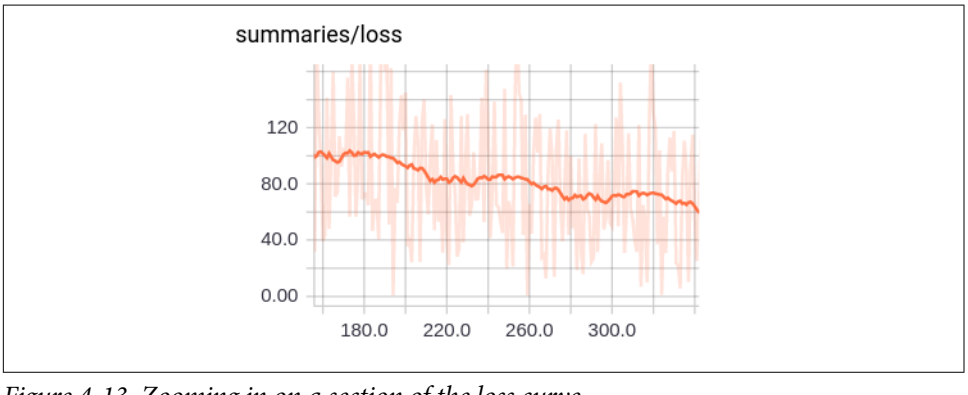

Figure 4-13. Zooming in on a section of the loss curve.

Note that loss looks much bumpier! This is one of the prices of using minibatch training. We no longer have the beautiful, smooth loss curves that we saw in the previous sections.

Review

In this chapter, we've introduced you to fully connected deep networks. We delved into the mathematical theory of these networks, and explored the concept of "universal approximation," which partially explains the learning power of fully connected networks. We ended with a case study, where you trained a deep fully connected architecture on the Tox21 dataset.

In this chapter, we haven't yet shown you how to tune the fully connected network to achieve good predictive performance. In Chapter 5, we will discuss "hyperparameter optimization," the process of tuning network parameters, and have you tune the parameters of the Tox21 network introduced in this chapter.

CHAPTER 5

Hyperparameter Optimization

Training a deep model and training a good deep model are very different things. While it's easy enough to copy-paste some TensorFlow code from the internet to get a first prototype running, it's much harder to transform that prototype into a high-quality model. The process of taking a prototype to a high-quality model involves many steps. We'll explore one of these steps, hyperparameter optimization, in the rest of this chapter.

To first approximation, hyperparameter optimization is the process of tweaking all parameters of a model not learned by gradient descent. These quantities are called "hyperparameters." Consider fully connected networks from the previous chapter. While the weights of fully connected networks can be learned from data, the other settings of the network can't. These hyperparameters include the number of hidden layers, the number of neurons per hidden layer, the learning rate, and more. How can you systematically find good values for these quantities? Hyperparameter optimization methods provide our answer to this question.

Recall that we mentioned previously that model performance is tracked on a held-out "validation" set. Hyperparameter optimization methods systematically try multiple choices for hyperparameters on the validation set. The best-performing set of hyperparameter values is then evaluated on a second held-out "test" set to gauge the true model performance. Different hyperparameter optimization methods differ in the algorithm they use to propose new hyperparameter settings. These algorithms range from the obvious to quite sophisticated. We will only cover some of the simpler methods in these chapters, since the more sophisticated hyperparameter optimization techniques tend to require very large amounts of computational power.

As a case study, we will tune the Tox21 toxicity fully connected network introduced in Chapter 4 to achieve good performance. We strongly encourage you (as always) to