- Reading inputs efficiently using readers, queue runners, and coordinators

Now let's use all of this to parallelize neural networks!

Parallelizing Neural Networks on a TensorFlow Cluster

In this section, first we will look at how to parallelize several neural networks by simply placing each one on a different device. Then we will look at the much trickier problem of training a single neural network across multiple devices and servers.

One Neural Network per Device

The most trivial way to train and run neural networks on a TensorFlow cluster is to take the exact same code you would use for a single device on a single machine, and specify the master server's address when creating the session. That's it—you're done! Your code will be running on the server's default device. You can change the device that will run your graph simply by putting your code's construction phase within a device block.

By running several client sessions in parallel (in different threads or different processes), connecting them to different servers, and configuring them to use different devices, you can quite easily train or run many neural networks in parallel, across all devices and all machines in your cluster (see Figure 12-11). The speedup is almost linear.[4] Training 100 neural networks across 50 servers with 2 GPUs each will not take much longer than training just 1 neural network on 1 GPU.

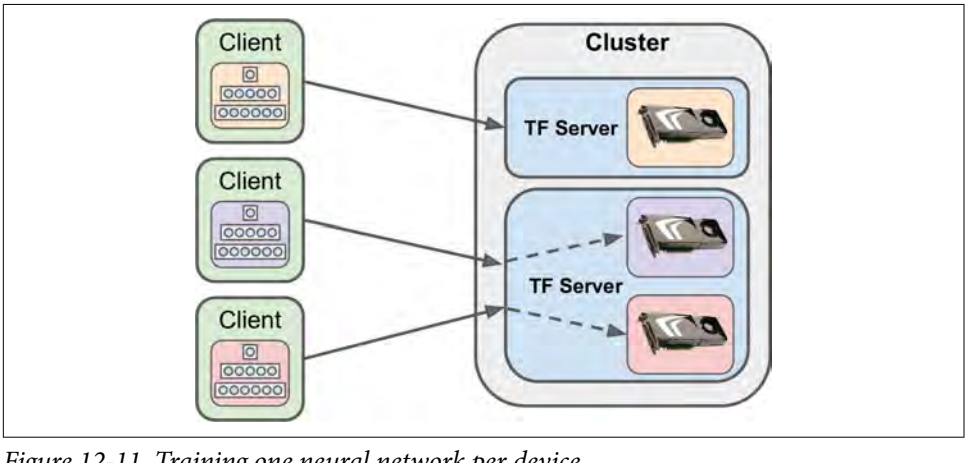

Figure 12-11. Training one neural network per device

4 Not 100% linear if you wait for all devices to finish, since the total time will be the time taken by the slowest device.