CHAPTER 9
Up and Running with TensorFlow

TensorFlow is a powerful open source software library for numerical computation, particularly well suited and fine-tuned for large-scale Machine Learning. Its basic principle is simple: you first define in Python a graph of computations to perform (for example, the one in Figure 9-1), and then TensorFlow takes that graph and runs it efficiently using optimized C++ code.

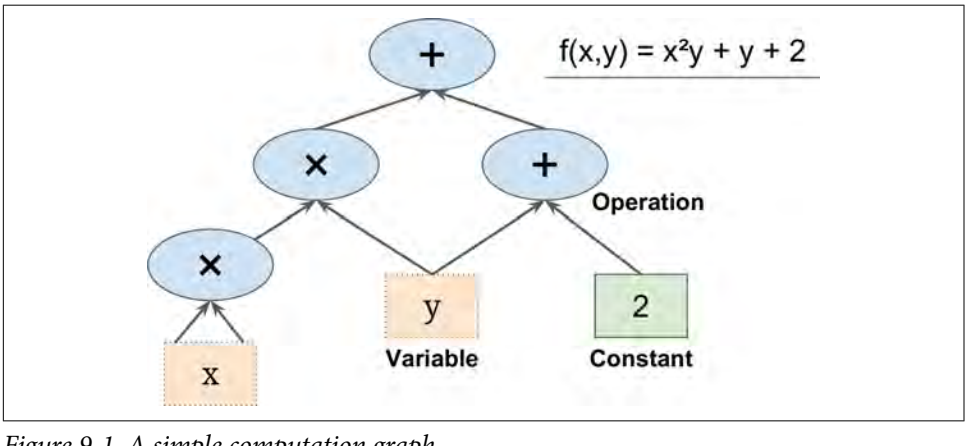

Figure 9-1. A simple computation graph

Most importantly, it is possible to break up the graph into several chunks and run them in parallel across multiple CPUs or GPUs (as shown in Figure 9-2). TensorFlow also supports distributed computing, so you can train colossal neural networks on humongous training sets in a reasonable amount of time by splitting the computations across hundreds of servers (see Chapter 12). TensorFlow can train a network with millions of parameters on a training set composed of billions of instances with millions of features each. This should come as no surprise, since TensorFlow was

developed by the Google Brain team and it powers many of Google's large-scale services, such as Google Cloud Speech, Google Photos, and Google Search.

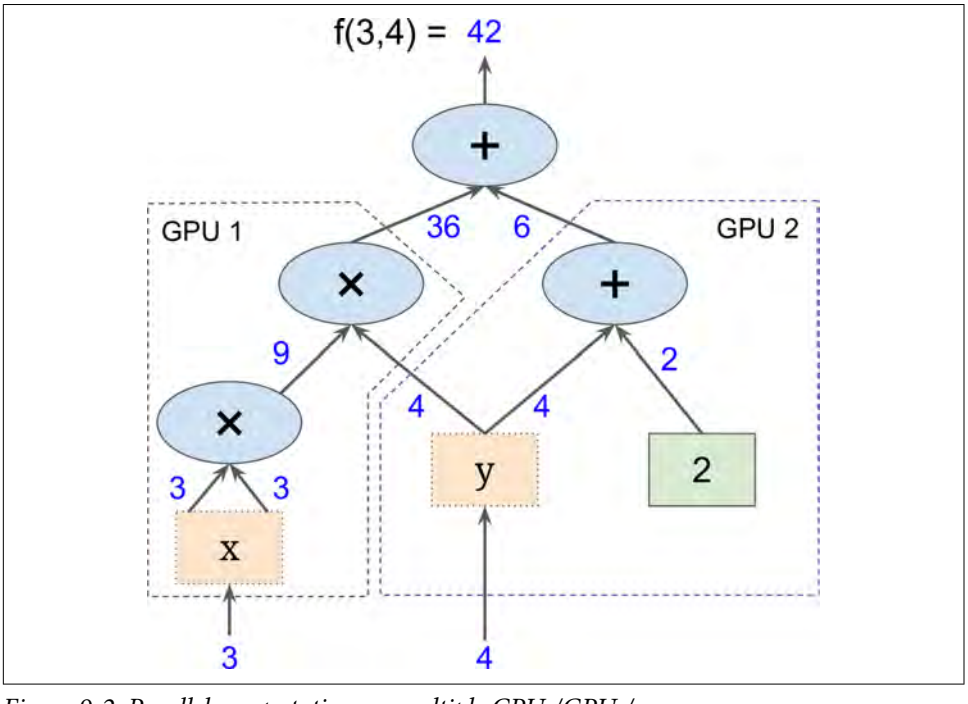

Figure 9-2. Parallel computation on multiple CPUs/GPUs/servers

When TensorFlow was open-sourced in November 2015, there were already many popular open source libraries for Deep Learning (Table 9-1 lists a few), and to be fair most of TensorFlow's features already existed in one library or another. Nevertheless, TensorFlow's clean design, scalability, flexibility,[1] and great documentation (not to mention Google's name) quickly boosted it to the top of the list. In short, TensorFlow was designed to be flexible, scalable, and production-ready, and existing frameworks arguably hit only two out of the three of these. Here are some of TensorFlow's highlights:

- It runs not only on Windows, Linux, and macOS, but also on mobile devices, including both iOS and Android.

1 TensorFlow is not limited to neural networks or even Machine Learning; you could run quantum physics simulations if you wanted.

- It provides a very simple Python API called *TF.Learn*[2] (`tensorflow.contrib.learn`), compatible with Scikit-Learn. As you will see, you can use it to train various types of neural networks in just a few lines of code. It was previously an independent project called *Scikit Flow* (or *skflow*).
- It also provides another simple API called *TF-slim* (`tensorflow.contrib.slim`) to simplify building, training, and evaluating neural networks.
- Several other high-level APIs have been built independently on top of TensorFlow, such as Keras or Pretty Tensor.
- Its main Python API offers much more flexibility (at the cost of higher complexity) to create all sorts of computations, including any neural network architecture you can think of.
- It includes highly efficient C++ implementations of many ML operations, particularly those needed to build neural networks. There is also a C++ API to define your own high-performance operations.
- It provides several advanced optimization nodes to search for the parameters that minimize a cost function. These are very easy to use since TensorFlow automatically takes care of computing the gradients of the functions you define. This is called *automatic differentiating* (or *autodiff*).
- It also comes with a great visualization tool called *TensorBoard* that allows you to browse through the computation graph, view learning curves, and more.
- Google also launched a cloud service to run TensorFlow graphs.
- Last but not least, it has a dedicated team of passionate and helpful developers, and a growing community contributing to improving it. It is one of the most popular open source projects on GitHub, and more and more great projects are being built on top of it (for examples, check out the resources page on *https://www.tensorflow.org/*, or *https://github.com/jtoy/awesome-tensorflow*). To ask technical questions, you should use *http://stackoverflow.com/* and tag your question with `"tensorflow"`. You can file bugs and feature requests through GitHub. For general discussions, join the Google group.

In this chapter, we will go through the basics of TensorFlow, from installation to creating, running, saving, and visualizing simple computational graphs. Mastering these basics is important before you build your first neural network (which we will do in the next chapter).

2 Not to be confused with the TFLearn library, which is an independent project.

Table 9-1. Open source Deep Learning libraries (not an exhaustive list)

Library	API	Platforms	Started by	Year
Caffe	Python, C++, Matlab	Linux, macOS, Windows	Y. Jia, UC Berkeley (BVLC)	2013
Deeplearning4j	Java, Scala, Clojure	Linux, macOS, Windows, Android	A. Gibson, J.Patterson	2014
H2O	Python, R	Linux, macOS, Windows	H2O.ai	2014
MXNet	Python, C++, others	Linux, macOS, Windows, iOS, Android	DMLC	2015
TensorFlow	Python, C++	Linux, macOS, Windows, iOS, Android	Google	2015
Theano	Python	Linux, macOS, iOS	University of Montreal	2010
Torch	C++, Lua	Linux, macOS, iOS, Android	R. Collobert, K. Kavukcuoglu, C. Farabet	2002

Installation

Let's get started! Assuming you installed Jupyter and Scikit-Learn by following the installation instructions in Chapter 2, you can simply use pip to install TensorFlow. If you created an isolated environment using virtualenv, you first need to activate it:

```
$ cd $ML_PATH               # Your ML working directory (e.g., $HOME/ml)
$ source env/bin/activate
```

Next, install TensorFlow:

```
$ pip3 install --upgrade tensorflow
```

For GPU support, you need to install `tensorflow-gpu` instead of `tensorflow`. See Chapter 12 for more details.

To test your installation, type the following command. It should output the version of TensorFlow you installed.

```
$ python3 -c 'import tensorflow; print(tensorflow.__version__)'
1.0.0
```

Creating Your First Graph and Running It in a Session

The following code creates the graph represented in Figure 9-1:

```
import tensorflow as tf

x = tf.Variable(3, name="x")
y = tf.Variable(4, name="y")
f = x*x*y + y + 2
```