CHAPTER 2

Introduction to TensorFlow Primitives

This chapter will introduce you to fundamental aspects of TensorFlow. In particular, you will learn how to perform basic computation using TensorFlow. A large part of this chapter will be spent introducing the concept of tensors, and discussing how tensors are represented and manipulated within TensorFlow. This discussion will necessitate a brief overview of some of the mathematical concepts that underlie tensorial mathematics. In particular, we'll briefly review basic linear algebra and demonstrate how to perform basic linear algebraic operations with TensorFlow.

We'll follow this discussion of basic mathematics with a discussion of the differences between declarative and imperative programming styles. Unlike many programming languages, TensorFlow is largely declarative. Calling a TensorFlow operation adds a description of a computation to TensorFlow's "computation graph." In particular, TensorFlow code "describes" computations and doesn't actually perform them. In order to run TensorFlow code, users need to create `tf.Session` objects. We introduce the concept of sessions and describe how users perform computations with them in TensorFlow.

We end the chapter by discussing the notion of variables. Variables in TensorFlow hold tensors and allow for stateful computation that modifies variables to occur. We demonstrate how to create variables and update their values via TensorFlow.

Introducing Tensors

Tensors are fundamental mathematical constructs in fields such as physics and engineering. Historically, however, tensors have made fewer inroads in computer science, which has traditionally been more associated with discrete mathematics and logic. This state of affairs has started to change significantly with the advent of machine

learning and its foundation on continuous, vectorial mathematics. Modern machine learning is founded upon the manipulation and calculus of tensors.

Scalars, Vectors, and Matrices

To start, we will give some simple examples of tensors that you might be familiar with. The simplest example of a tensor is a scalar, a single constant value drawn from the real numbers (recall that the real numbers are decimal numbers of arbitrary precision, with both positive and negative numbers permitted). Mathematically, we denote the real numbers by $\mathbb{R}$. More formally, we call a scalar a rank-0 tensor.

Aside on Fields

Mathematically sophisticated readers will protest that it's entirely meaningful to define tensors based on the complex numbers, or with binary numbers. More generally, it's sufficient that the numbers come from a *field*: a mathematical collection of numbers where 0, 1, addition, multiplication, subtraction, and division are defined. Common fields include the real numbers $\mathbb{R}$, the rational numbers $\mathbb{Q}$, the complex numbers $\mathbb{C}$, and finite fields such as $\mathbb{Z}_2$. For simplicity, in much of the discussion, we will assume real valued tensors, but substituting in values from other fields is entirely reasonable.

If scalars are rank-0 tensors, what constitutes a rank-1 tensor? Formally, speaking, a rank-1 tensor is a vector; a list of real numbers. Traditionally, vectors are written as either column vectors

$$\begin{pmatrix} a \\ b \end{pmatrix}$$

or as row vectors

$$(a \;\; b)$$

Notationally, the collection of all column vectors of length 2 is denoted $\mathbb{R}^{2\times 1}$ while the set of all row vectors of length 2 is $\mathbb{R}^{1\times 2}$. More computationally, we might say that the shape of a column vector is (2, 1), while the shape of a row vector is (1, 2). If we don't wish to specify whether a vector is a row vector or column vector, we can say it comes from the set $\mathbb{R}^2$ and has shape (2). This notion of tensor shape is quite important for understanding TensorFlow computations, and we will return to it later on in this chapter.