TensorFlow in Action (Part 1)

Chapter 1 : TensorFlow 2

This chapter starts by contrasting TensorFlow 1's static-graph workflow with TensorFlow 2's eager execution style, showing how TF2 feels more like regular Python while still allowing graph optimization when needed. You build a simple multilayer perceptron twice—once in the older style and once in TF2—to appreciate how Keras layers, automatic differentiation, and the high-level APIs simplify model definition, training loops, and debugging in modern TensorFlow. The author then briefly explains how TensorFlow executes computations "under the hood" by constructing dataflow graphs, scheduling operations, and mapping them efficiently to hardware such as CPUs, GPUs, or TPUs.

Next, the chapter introduces the **building blocks** of TensorFlow: tf.Variable, tf.Tensor, and operations. A tf.Variable represents mutable state, typically model parameters; you see how to initialize variables from constants, NumPy arrays, or Keras initializers, control whether they are trainable, and update slices of a variable with methods like assign and slicing syntax. In contrast, tf.Tensor denotes the (usually immutable) outputs of operations, including higher-dimensional arrays that generalize scalars, vectors, and matrices, and you learn to think in terms of axes (e.g., a 3D tensor with shape 2 × 4 × 3 has row, column, and depth axes).

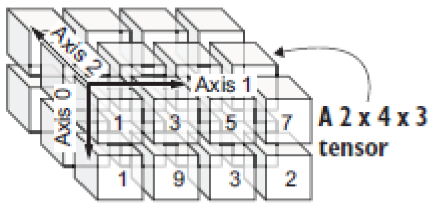

Figure 1 A 2 x4x3 tensor with the three axes. The first axis (axis 0) is the row dimension, the second axis (axis 1) is the column axis, and the final axis (axis 2) is the depth axis.

Under eager execution, these appear as EagerTensor objects whose values are computed immediately, but they are still TensorFlow tensors and can be converted to NumPy arrays when needed.

The last part focuses on **neural network–related computations** in raw TensorFlow. You implement matrix multiplication, convolution, and pooling using TensorFlow ops, reinforcing how typical deep learning layers are just combinations of these primitives. For example, the chapter walks through 2D convolutions as sliding filters over an image tensor, explains how strides and padding affect the output, and shows how pooling operations like max pooling reduce spatial resolution while keeping salient features. By the end, you have a clearer mental model of how TF2 represents parameters, data, and operations, setting the stage for higher-level Keras usage in later chapters.