

Tensorflow 2.0

Basics and syntax

Rahul Dit | 28TH July

# What is Tensorflow?

TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.

Created by the Google Brain team, TensorFlow is an open source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning (aka neural networking) models and algorithms and makes them useful by way of a common metaphor. It uses Python to provide a convenient front-end API for building applications with the framework, while executing those applications in high-performance C++.

# How Tensorflow works?

TensorFlow allows developers to create dataflow graphs—structures that describe how data moves through a graph, or a series of processing nodes. Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor.

TensorFlow provides all of this for the programmer by way of the Python language. Python is easy to learn and work with, and provides convenient ways to express how high-level abstractions can be coupled together. Nodes and tensors in TensorFlow are Python objects, and TensorFlow applications are themselves Python applications.

The actual math operations, however, are not performed in Python. The libraries of transformations that are available through TensorFlow are written as high-performance C++ binaries. Python just directs traffic between the pieces, and provides high-level programming abstractions to hook them together.

Tensorflow architecture works in three parts:

* Preprocessing the data
* Build the model
* Train and estimate the model

# What are the benefits of Tensorflow?

The single biggest benefit TensorFlow provides for machine learning development is abstraction. Instead of dealing with the nitty-gritty details of implementing algorithms, or figuring out proper ways to hitch the output of one function to the input of another, the developer can focus on the overall logic of the application. TensorFlow takes care of the details behind the scenes.

TensorFlow offers additional conveniences for developers who need to debug and gain introspection into TensorFlow apps. The eager execution mode lets you evaluate and modify each graph operation separately and transparently, instead of constructing the entire graph as a single opaque object and evaluating it all at once. The TensorBoard visualization suite lets you inspect and profile the way graphs run by way of an interactive, web-based dashboard.

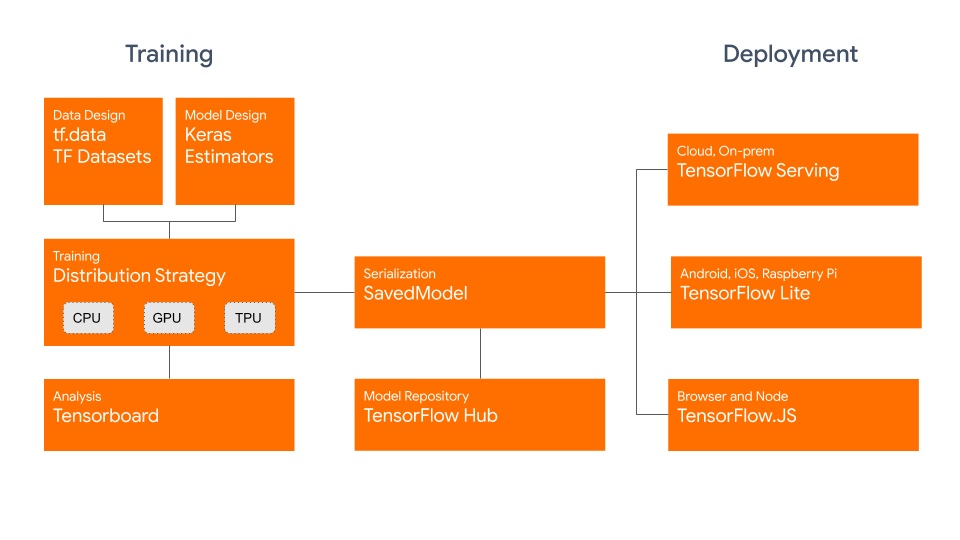
TensorFlow also gains many advantages from the backing of an A-list commercial outfit in Google. Google has not only fueled the rapid pace of development behind the project, but created many significant offerings around TensorFlow that make it easier to deploy and easier to use: the above-mentioned TPU silicon for accelerated performance in Google’s cloud; an online hub for sharing models created with the framework; in-browser and mobile-friendly incarnations of the framework; and much more.

One caveat: Some details of TensorFlow’s implementation make it hard to obtain totally deterministic model-training results for some training jobs. Sometimes a model trained on one system will vary slightly from a model trained on another, even when they are fed the exact same data. The reasons for this are slippery—e.g., how random numbers are seeded and where, or certain non-deterministic behaviors when using GPUs). That said, it is possible to work around those issues, and TensorFlow’s team is considering more controls to affect determinism in a workflow.

# Coding with Tensorflow 2.0

An excerpt from the official Tensorflow blog: -

“TensorFlow 2.0 makes development of ML applications much easier. With tight integration of Keras into TensorFlow, eager execution by default, and Pythonic function execution, TensorFlow 2.0 makes the experience of developing applications as familiar as possible for Python developers. For researchers pushing the boundaries of ML, we have invested heavily in TensorFlow’s low-level API: We now export all ops that are used internally, and we provide inheritable interfaces for crucial concepts such as variables and checkpoints. This allows you to build onto the internals of TensorFlow without having to rebuild TensorFlow.”



# Basic Syntax

**Importing Tensorflow: -**

import tensorflow as tf

**Setting version for Tensorflow: -**

%tensorflow\_version 2.x

**Declaring a constant Tensor and checking its shape: -**

tensor\_c = tf.constant([[2,3], [3,4]]) #defining a constant tensor

tensor\_c.shape

**Declaring a variable Tensor: -**

tensor\_v = tf.Variable([[2,3], [3,2]]) #defining a variable tensor

**Operation on Tensors: -**

Scalar addition and multiplication are similarly to how a normal variable is multiplied or added. All numpy operations are applicable on Tensorflow objects

Notebook Link: - https://colab.research.google.com/drive/1Err3iunxyEyZD8-T32HyoMovpeXJHfcT?usp=sharing