First Steps with TensorFlow: Toolkit

developers.google.com/machine-learning/crash-course/first-steps-with-tensorflow/toolkit

Tensorflow is a computational framework for building machine learning models. TensorFlow provides a variety of different toolkits that allow you to construct models at your preferred level of abstraction. You can use lower-level APIs to build models by defining a series of mathematical operations. Alternatively, you can use higher-level APIs (like `tf.estimator`) to specify predefined architectures, such as linear regressors or neural networks.

The following figure shows the current hierarchy of TensorFlow toolkits:

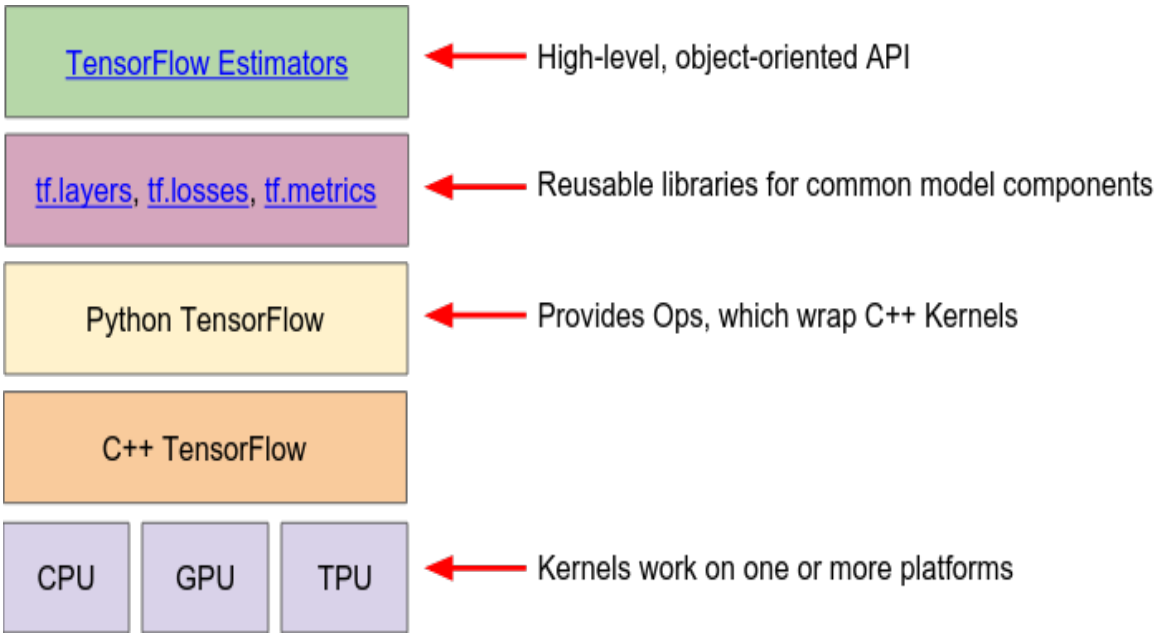

Figure 1. TensorFlow toolkit hierarchy.

The following table summarizes the purposes of the different layers:

Toolkit(s)	Description
Estimator (tf.estimator)	High-level, OOP API.
tf.layers/tf.losses/tf.metrics	Libraries for common model components.
TensorFlow	Lower-level APIs

TensorFlow consists of the following two components:

- a graph protocol buffer
- a runtime that executes the (distributed) graph

These two components are analogous to Python code and the Python interpreter. Just as the Python interpreter is implemented on multiple hardware platforms to run Python code, TensorFlow can run the graph on multiple hardware platforms, including CPU, GPU, and TPU.

Which API(s) should you use? You should use the highest level of abstraction that solves the problem. The higher levels of abstraction are easier to use, but are also (by design) less flexible. We recommend you start with the highest-level API first and get everything working. If you need additional flexibility for some special modeling concerns, move one level lower. Note that each level is built using the APIs in lower levels, so dropping down the hierarchy should be reasonably straightforward.

tf.estimator API

We'll use tf.estimator for the majority of exercises in Machine Learning Crash Course. Everything you'll do in the exercises could have been done in lower-level (raw) TensorFlow, but using tf.estimator dramatically lowers the number of lines of code.

tf.estimator is compatible with the scikit-learn API. Scikit-learn is an extremely popular open-source ML library in Python, with over 100k users, including many at Google.

Very broadly speaking, here's the pseudocode for a linear classification program implemented in tf.estimator:

```
import tensorflow as tf

# Set up a linear classifier.
classifier = tf.estimator.LinearClassifier(feature_columns)

# Train the model on some example data.
classifier.train(input_fn=train_input_fn, steps=2000)

# Use it to predict.
predictions = classifier.predict(input_fn=predict_input_fn)
```