Module 9 Supplemental Material – TensorFlow2.0

As the materials for this course were being created, the team at TensorFlow released TensorFlow 2.0 (TF2), the second version of the package. Your knowledge of TensorFlow 1.0 (TF1) will still remain very important; nearly all TensorFlow models are still running on stable TF1 code, and much of the TensorFlow codebase remains that same. That said, there are some exciting updates in TF2 that can vastly improve the model building process and will likely become industry standard relatively quickly, so it's important that you begin to understand these changes.

Some of the major changes include:

- **API cleanup:** Many APIs like tf.gans, tf.app, tf.contrib, tf.flags are either gone or moved to separate repositories.
- **Eager Execution:** As you might recall, to build a Neural Net in TF1, we needed to define this abstract data structure called a Graph, and then to actually run the graph we needed to use an encapsulation called a Session. With eager execution, this changes. Now, TensorFlow code can be run like normal Python code. Meaning that operations are created and evaluated at once.
- **Functions, not sessions:** A session.run() call is almost like a function call: You specify the inputs and the function to be called, and you get back a set of outputs. In TensorFlow 2.0, you can decorate a Python function using tf.function() to mark it for JIT compilation so that TensorFlow runs it as a single graph. That's right, no more sessions!
- **No more globals:** TensorFlow 1.0 relied heavily on implicitly global namespaces. When you called tf.Variable(), it would be put into the default graph, and it would remain there, even if you lost track of the Python variable pointing to it. You could then recover that tf.Variable, but only if you knew the name that it had been created with (which was often a huge pain). TensorFlow 2.0 eliminates all of these mechanisms in favor of the default mechanism: Keep track of your variables! If you lose track of a tf.Variable, it gets garbage collected (essentially erased from memory). The requirement to track variables creates some extra work for the user, but with Keras objects, the burden is minimized.
- **Tight integration with Keras:** Keras is now the default and recommended high-level API for TensorFlow. The easiest way to train a model in TF 2.0 is by using the *.fit()* method on a Keras Sequential or Functional model (we cover Keras in more detail in the next module). This method is ideal for rapid prototyping for any type of model, and saves the user from having to tediously construct complicated model construction and execution pipelines.

There are many more exciting changes in TF2, and you are encouraged to take a deep dive into all of them. To get started, take a look at the following sources:

- TensorFlow 2.0 documentation: https://www.tensorflow.org/versions/r2.0/api_docs/python/tf
- The 'Effective TensorFlow 2.0 Guide', produced by the TensorFlow team: https://github.com/tensorflow/docs/blob/master/site/en/r2/guide/effective_tf2.md
- Thalles Silva's very thorough overview of the major changes in this HackerNoon article: https://hackernoon.com/everything-you-need-to-know-about-tensorflow-2-0-b0856960c074

Note: If you want to start experimenting with TF2 right away, you are encouraged to first create a separate virtual environment. You can then and install the package from the command line as follows: `pip install -q tensorflow==2.0.0-alpha0`