Machine Learning & Android — https://stackoverflow.com/cv/daj

Aug 28, 2017 · 4 min read

How to inspect a pre-trained TensorFlow model

Let's assume somebody has given us a pre-trained TensorFlow model and asked us to embed it in an Android app.

To do this (as we saw in Using a pre-trained TensorFlow model on Android), we need to know some information about the input and output nodes in the model.

TensorBoard is an awesome tool that we can use to inspect TensorFlow models (a.k.a. "graphs"). The TensorBoard: Graph Visualization documentation is very detailed, but I found it a bit intimidating to start.

In this article, we're going to use import_pb_to_tensorboard.py to import an *existing* model into TensorBoard.

Installing TensorBoard

There are two ways to install TensorBoard:

Standalone

The quickest option is to install TensorBoard using `pip`:

```
pip install tensorboard
```

There is also a standalone GitHub project that might work for you.

In a container

TensorBoard depends on TensorFlow, and you might want to isolate all your TensorFlow binaries inside a container (as I discuss here). If you go for this approach, you need to pass an extra `-p` argument to Docker to tell it to forward HTTP traffic from the container's localhost port 6006 to the local machine.

```
docker run -p 6006:6006 -it gcr.io/tensorflow
/tensorflow:1.3.0 bash
```

Importing an existing model

Once you have TensorBoard installed, you can run it like so:

```
tensorboard --logdir=/tmp/tensorflow_logdir
```

Then simply open http://localhost:6006 in your browser to see TensorBoard. At this point you won't have anything to view in TensorBoard.

Now we can use the import_pb_to_tensorboard.py tool to import an existing model:

Get import_pb_to_tensorboard.py. I have the TensorFlow repository cloned onto my machine (the tools are in `tensorflow/python/tools`).

Get a model file. I'm using the 12.5MB mnist_model_graph.pb from this article, and I saved it to `/tmp`.

Pass the model file to import_pb_to_tensorboard.py. It doesn't matter what location you use for `log_dir`.

```
$ python import_pb_to_tensorboard.py --model_dir
```

```
/tmp/mnist_model_graph.pb --log_dir /tmp/tensorflow_logdir

<snip>

Model Imported. Visualize by running: tensorboard
--logdir=/tmp/tensorflow_logdir
```

4. Run `tensorboard --logdir=/tmp/tensorflow_logdir` to start TensorBoard.

5. Now we can view this model in our browser at http://localhost:6006 (screenshots further below).

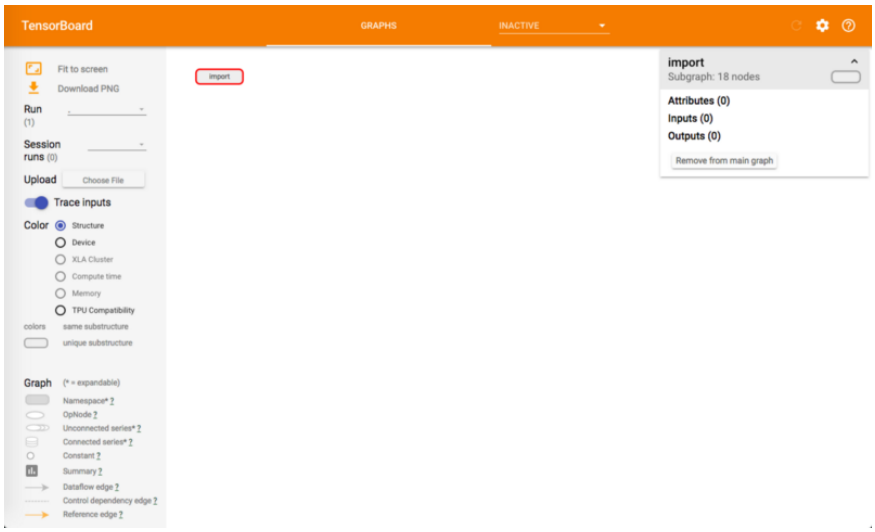
Warning: you need to be using TensorFlow r0.12 or above to use import_pb_to_tensorboard.py. If your TensorFlow installation is too old, you'll see this error. To check the version of your TensorFlow installation, run: `python -c "import tensorflow as tf; print(tf.GIT_VERSION, tf.VERSION)"`

Note: I've removed some warnings that appeared for me on Mac OS X Sierra. In case you care, here's one of them:

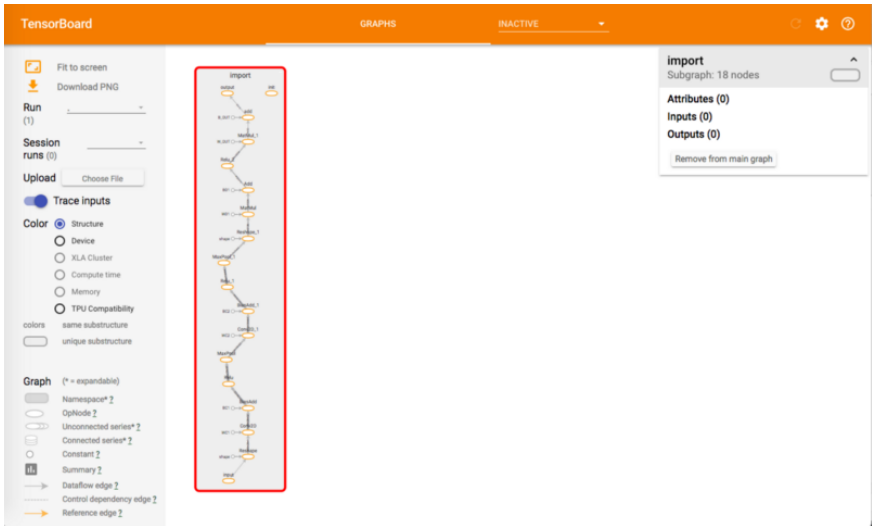
```
2017-08-27 18:11:31.334823: W tensorflow/core/platform
/cpu_feature_guard.cc:45] The TensorFlow library wasn't
compiled to use SSE4.2 instructions, but these are available
on your machine and could speed up CPU computations.
```

Viewing the model in TensorBoard

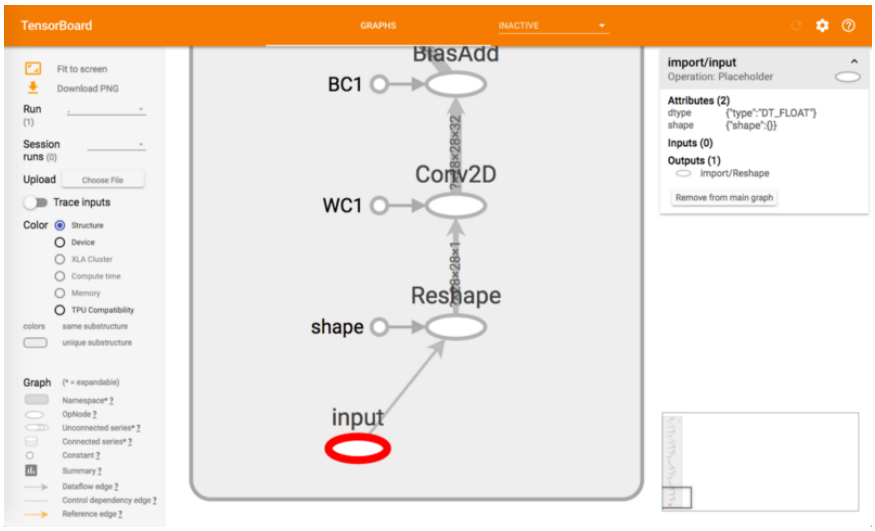
When you first open the model in TensorBoard, you'll just see one node called "import". At the top right you'll see "Subgraph: 18 nodes", which is the hint that there is more to see.



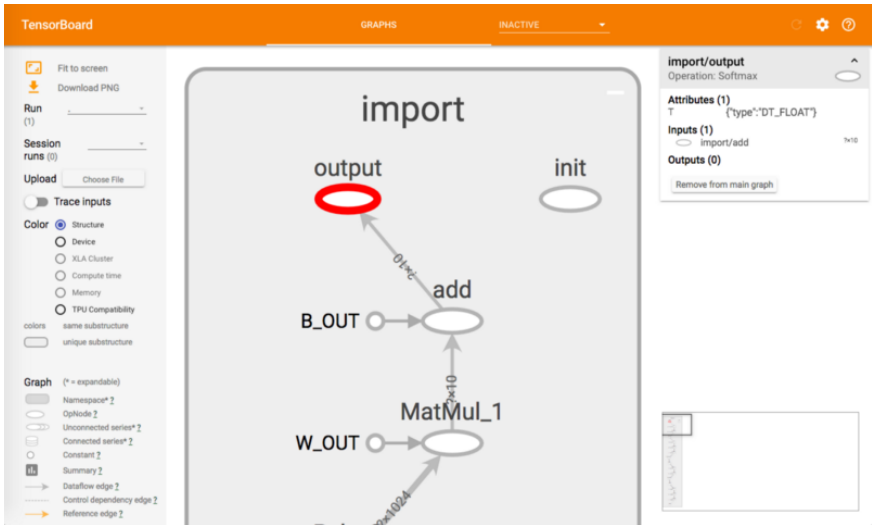
You can double click on the “import” node and it will expand to show you the full graph. You’ll need to zoom out and scroll around to get it to fit nicely, but you should be able to see the full graph on one screen:



You can also zoom in and see the input node at the bottom of the screen:



…and the output node at the top:



The information window at the top right provides some extra information about the node. In this case, the output is a float value (`DT_FLOAT`), which was output by using a `Softmax` function.

You can click on any node that you like. In this case, the Conv2D node is quite interesting. You can see `?x28x28x1` listed in a couple of places here—which makes sense because this particular model processes 28x28 pixel images.