TensorBoard

In machine learning, to improve something you often need to be able to measure it. TensorBoard is a tool for providing the measurements and visualizations needed during the machine learning workflow. It enables the follwing for you-

1. Tracking and visualizing metrics such as loss and accuracy
2. Visualizing the model graph (ops and layers)
3. Viewing histograms of weights, biases, or other tensors as they change over time
4. Projecting embeddings to a lower dimensional space
5. Displaying images, text, and audio data

Upload this file to Google Colab and run the code there. Additionally, make that thrid party cookies are enabled for Google Colab, or else you won't be able to connect with TensorBoard, and Error 403 will be displayed.

Using TensorBoard in Notebooks

TensorBoard can be used directly within notebook experiences such as Colab (https://colab.research.google.com/) and Jupyter (https://jupyter.org/). This can be helpful for sharing results, integrating TensorBoard into existing workflows, and using TensorBoard without installing anything locally.

Setup

Start by installing TF 2.0 and loading the TensorBoard notebook extension:

For Jupyter users: If you've installed Jupyter and TensorBoard into the same virtualenv (like anaconda), then you should be good to go. If you're using a more complicated setup, like a global Jupyter installation and kernels for different Conda/virtualenv environments, then you must ensure that the `tensorboard` binary is on your `PATH` inside the Jupyter notebook context. One way to do this is to modify the `kernel_spec` to prepend the environment's `bin` directory to `PATH`, as described here (https://github.com/ipython/ipykernel/issues/395#issuecomment-479787997).

In [1]:
```
# Load the TensorBoard notebook extension
%load_ext tensorboard
```

Import TensorFlow, datetime, and os:

In [2]:
```
import tensorflow as tf
import datetime, os
```

Working with TensorBoard

Download the FashionMNIST (https://github.com/zalandoresearch/fashion-mnist) dataset and scale it:

In [3]:
```
fashion_mnist = tf.keras.datasets.fashion_mnist

(x_train, y_train),(x_test, y_test) = fashion_mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
```

Create a very simple model:

In [4]:
```
def create_model():
  return tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
  ])
```

Train the model using Keras and the TensorBoard callback:

In [5]:
```
def train_model():

  model = create_model()
  model.compile(optimizer='adam',
                loss='sparse_categorical_crossentropy',
                metrics=['accuracy'])

  logdir = os.path.join("logs", datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
  tensorboard_callback = tf.keras.callbacks.TensorBoard(logdir, histogram_freq=1)

  model.fit(x=x_train,
            y=y_train,
            epochs=5,
            validation_data=(x_test, y_test),
            callbacks=[tensorboard_callback])

train_model()
```

```
Epoch 1/5
1875/1875 [==============================] - 7s 3ms/step - loss: 0.4915 - accur
acy: 0.8241 - val_loss: 0.4378 - val_accuracy: 0.8378
Epoch 2/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3816 - accur
acy: 0.8601 - val_loss: 0.3899 - val_accuracy: 0.8607
Epoch 3/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3516 - accur
acy: 0.8712 - val_loss: 0.3741 - val_accuracy: 0.8658
Epoch 4/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3273 - accur
acy: 0.8790 - val_loss: 0.3549 - val_accuracy: 0.8716
Epoch 5/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3119 - accur
acy: 0.8849 - val_loss: 0.3385 - val_accuracy: 0.8779
```

Start TensorBoard within the notebook using [magics (https://ipython.readthedocs.io/en/stable/interactive/magics.html)](https://ipython.readthedocs.io/en/stable/interactive/magics.html):

In [6]:
```
%tensorboard --logdir logs
```

```
Reusing TensorBoard on port 6006 (pid 181), started 0:03:29 ago. (Use '!kill 18
1' to kill it.)
```

```
<IPython.core.display.Javascript object>
```

You can now view dashboards such as scalars, graphs, histograms, and others. Some dashboards are not available yet in Colab (such as the profile plugin).

The `%tensorboard` magic has exactly the same format as the TensorBoard command line invocation, but with a `%` -sign in front of it.

You can also start TensorBoard before training to monitor it in progress:

In [7]:
```
%tensorboard --logdir logs
```

```
Reusing TensorBoard on port 6006 (pid 181), started 0:03:29 ago. (Use '!kill 18
1' to kill it.)
```

```
<IPython.core.display.Javascript object>
```

The same TensorBoard backend is reused by issuing the same command. If a different logs directory was chosen, a new instance of TensorBoard would be opened. Ports are managed automatically.

Start training a new model and watch TensorBoard update automatically every 30 seconds or refresh it with the button on the top right:

In [8]:
```
train_model()
```

```
Epoch 1/5
1875/1875 [==============================] - 7s 3ms/step - loss: 0.4958 - accur
acy: 0.8229 - val_loss: 0.4099 - val_accuracy: 0.8541
Epoch 2/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3816 - accur
acy: 0.8609 - val_loss: 0.3833 - val_accuracy: 0.8534
Epoch 3/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3503 - accur
acy: 0.8723 - val_loss: 0.3670 - val_accuracy: 0.8668
Epoch 4/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3296 - accur
acy: 0.8788 - val_loss: 0.3504 - val_accuracy: 0.8752
Epoch 5/5
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3128 - accur
acy: 0.8849 - val_loss: 0.3513 - val_accuracy: 0.8723
```

You can use the `tensorboard.notebook` APIs for a bit more control:

In [9]:
```
from tensorboard import notebook
notebook.list() # View open TensorBoard instances
```

```
Known TensorBoard instances:
  - port 6006: logdir logs (started 0:04:00 ago; pid 181)
```

In [10]:
```
# Control TensorBoard display. If no port is provided,
# the most recently launched TensorBoard is used
notebook.display(port=6006, height=1000)
```

```
Selecting TensorBoard with logdir logs (started 0:04:00 ago; port 6006, pid 18
1).
```

```
<IPython.core.display.Javascript object>
```