Background

To understand a neural model in a better way and to analyze how the data is flowing in the model's providing the measurements and visualizations needed during the machine learning workflow.

It enables tracking experiment metrics like

- Loss and Accuracy
- Visualizing the model graph
- Histograms
- Projecting embeddings to a lower-dimensional space

TensorBoard can be uses to monitor and document neural model-related parameters(Accuracy, lo to neural models.

TASK

Build a classification nueral model to classify handwritten digits using a Convolutional Nueral Net

Visualize the model with Tensorboard

Dataset - MNIST dataset

```
# Load the TensorBoard notebook extension

%load_ext tensorboard

import tensorflow as tf
import datetime

# Clear any logs from previous runs
!rm -rf ./logs/


# Input data — image labels — 28*28
# Model output — Classification result (0–9) — 10 output
# Build Convolutional Neural Net ( CNN ) to classify handwritten digits


# load the dataset
mnist = tf.keras.datasets.mnist

# splititng the data
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

# creating the model
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')
  ])
```

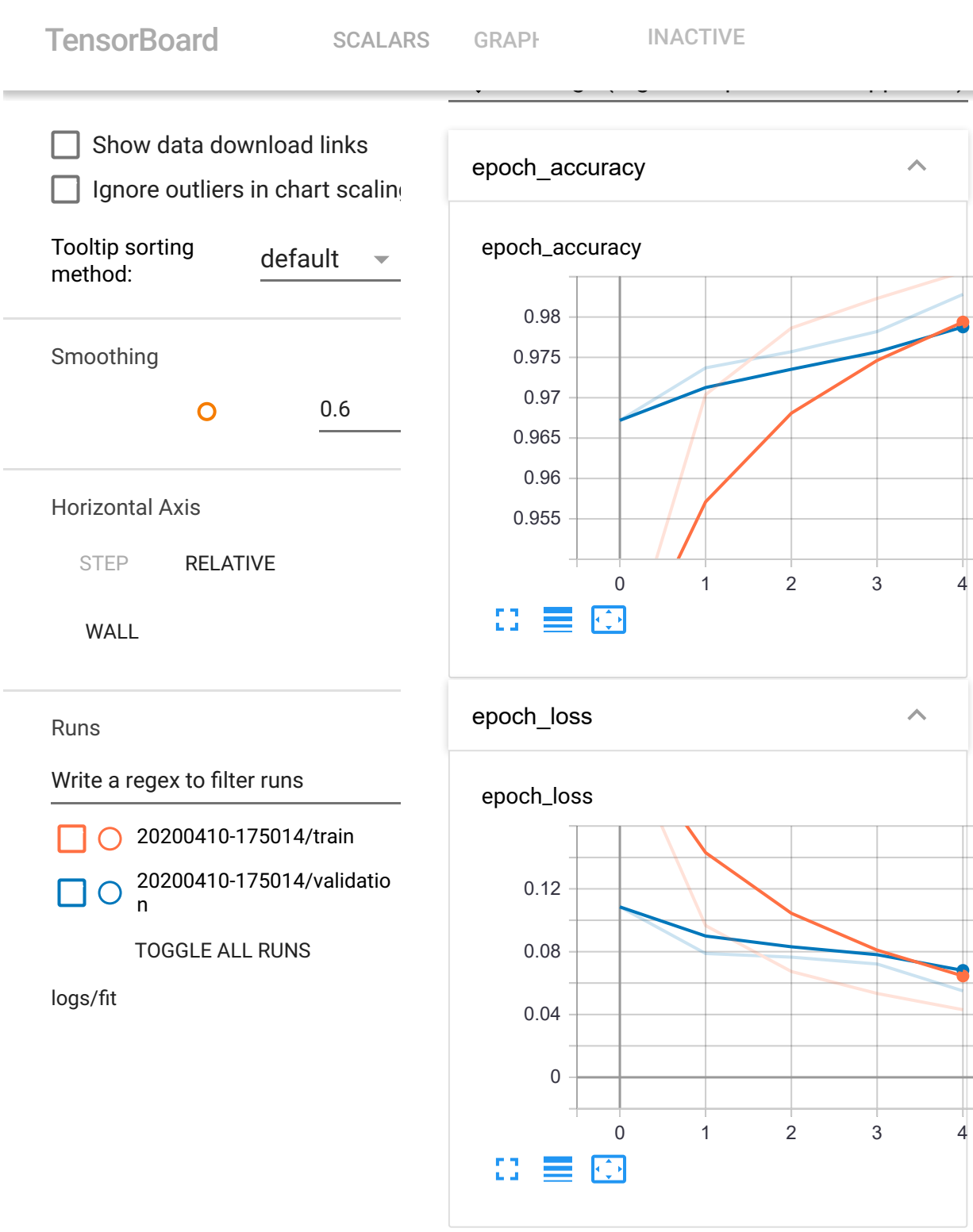
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mni
11493376/11490434 [==============================] - 0s 0us/step
```

```
# Model building & train the model with train data.

# When training with Keras's Model.fit(), adding the TensorBoard callback will ensure logs

# Additionally, enable histogram computation every epoch with the command (histogram_freq=

# Place the logs in a timestamped subdirectory to allow easy selection of different traini


model = create_model()
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
log_dir="logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1)
model.fit(x=x_train,
          y=y_train,
          epochs=5,
          validation_data=(x_test, y_test),
          callbacks=[tensorboard_callback])
```

```
Epoch 1/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2205 - accuracy: 0
Epoch 2/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0966 - accuracy: 0
Epoch 3/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0674 - accuracy: 0
Epoch 4/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0534 - accuracy: 0
Epoch 5/5
1875/1875 [==============================] - 8s 4ms/step - loss: 0.0430 - accuracy: 0
<tensorflow.python.keras.callbacks.History at 0x7fc4dda90160>
```

Model accuracy & Loss

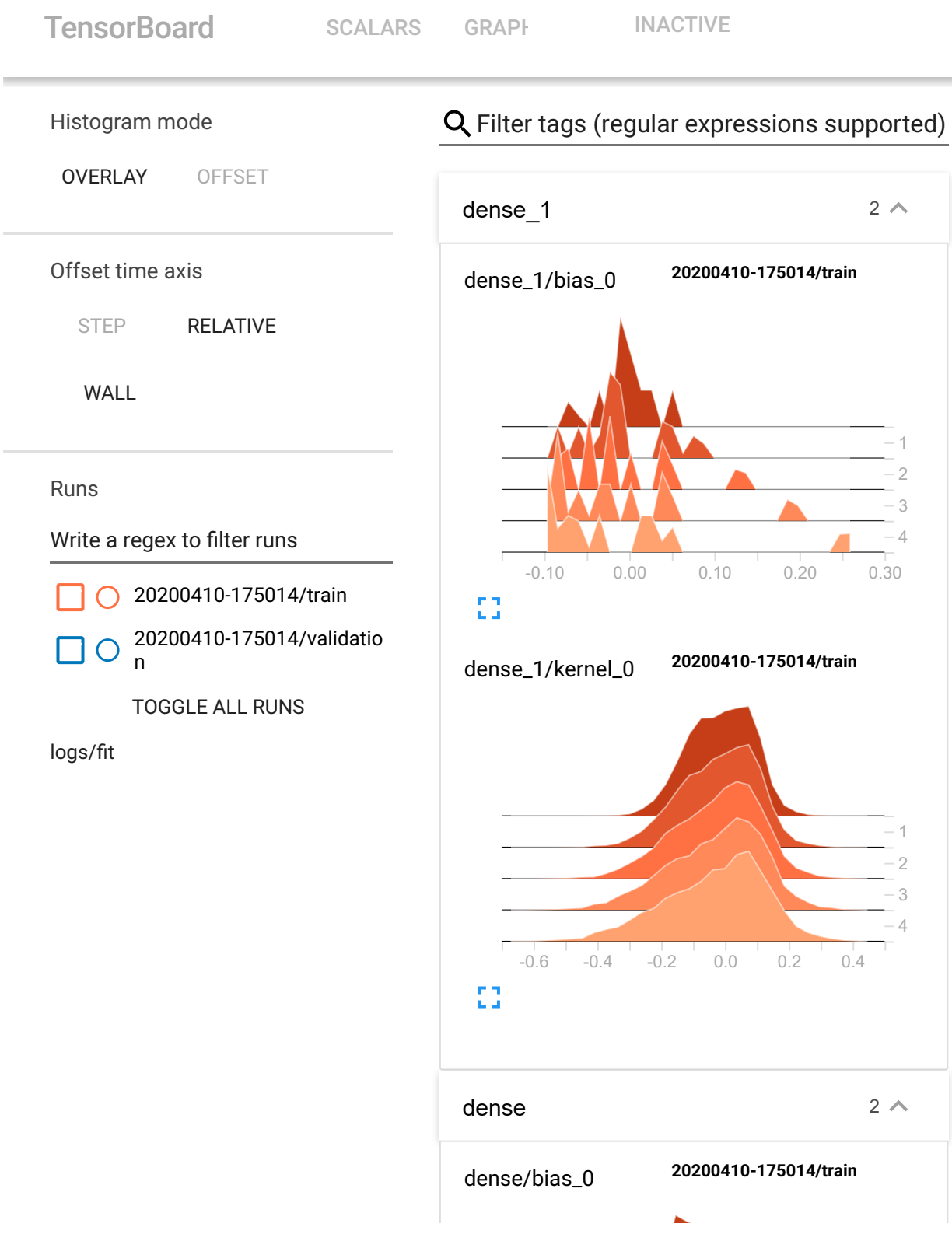
```
# Visualize the stuff what going on inside model with tensorboard

%tensorboard --logdir logs/fit
```



Histogram view of bias & kernel in a neural network in subsequent epochs

```
%tensorboard --logdir logs/fit
```

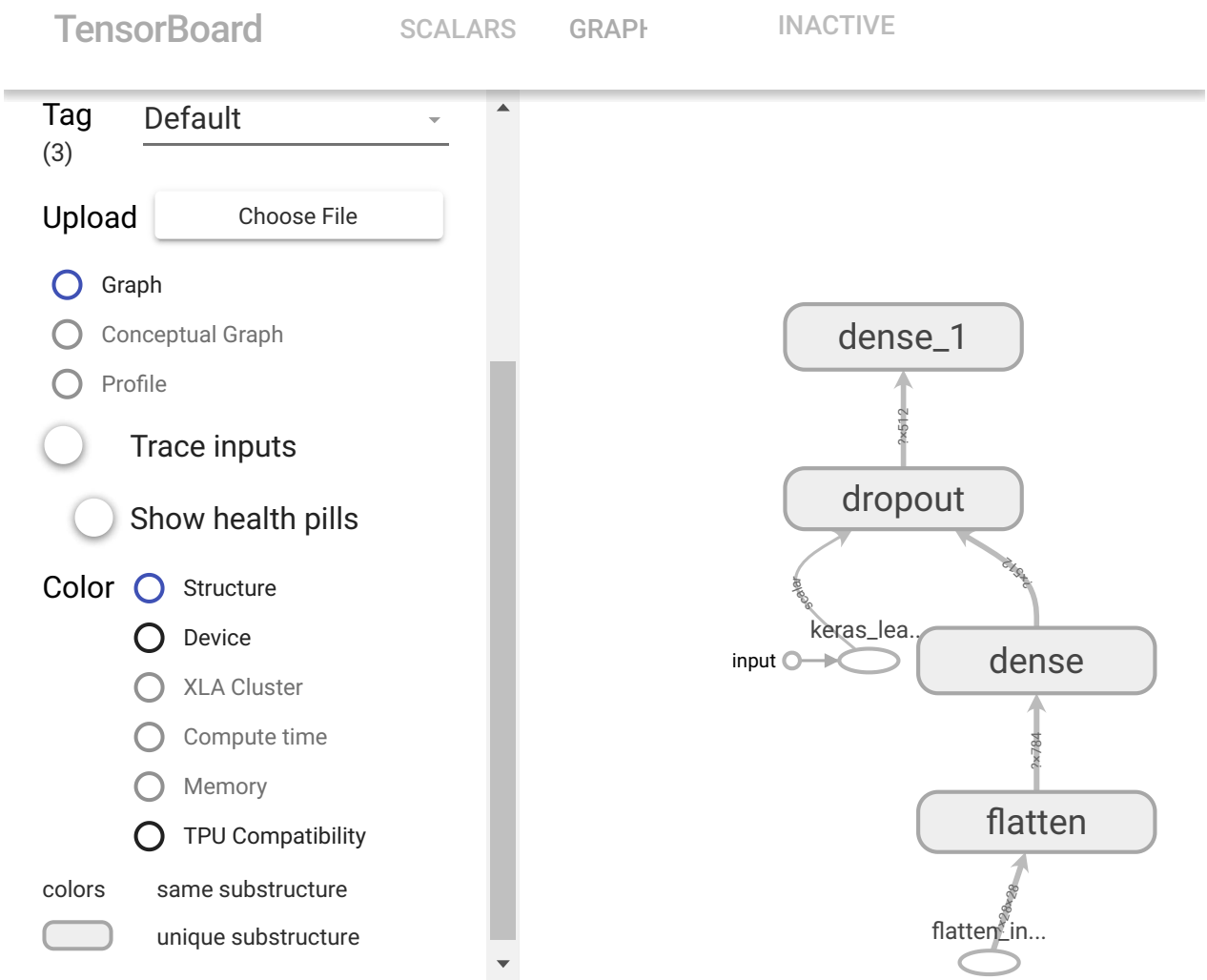
```
Reusing TensorBoard on port 6006 (pid 251), started 0:02:40 ago. (Use '!kill 251' to
```



Neural model in graph form, It helps to debug the model. It tells us which component of the model analysis of each neural model (Layers) can be done with the help of graph view

```
%tensorboard --logdir logs/fit
```

```
Reusing TensorBoard on port 6006 (pid 251), started 0:04:18 ago. (Use '!kill 251' to
```



Close legend.

Graph (* = expandable)

- Namespace* ?
- OpNode ?
- Unconnected series* ?
- Connected series* ?
- Constant ?
- Summary ?
- Dataflow edge ?
- Control dependency edge ?
- Reference edge ?