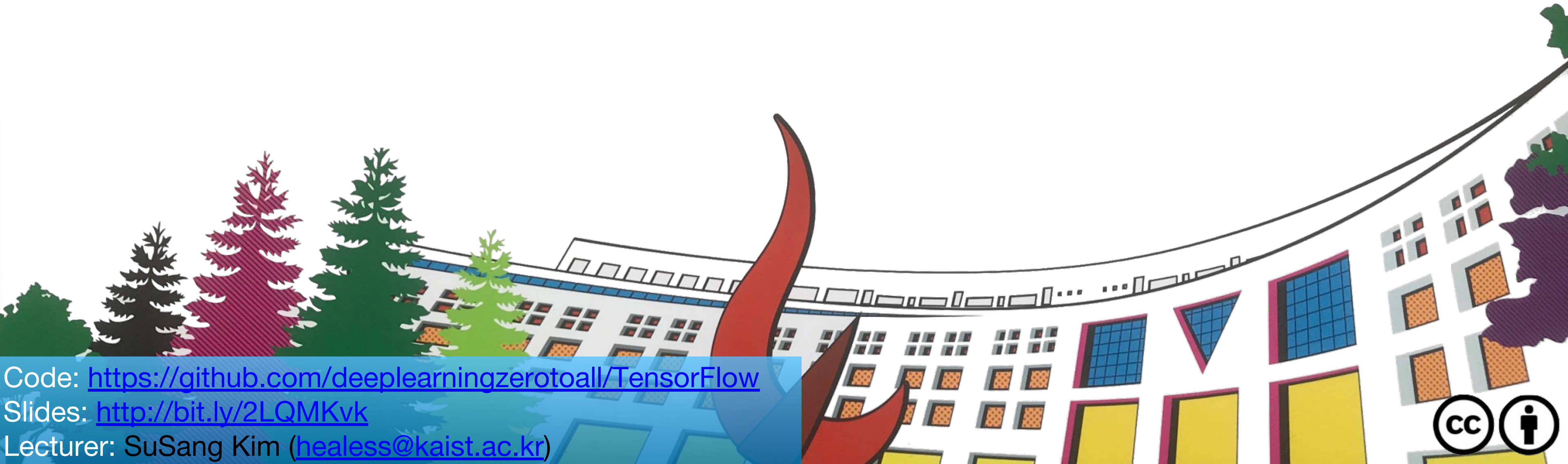
ML/DL for Everyone Season2

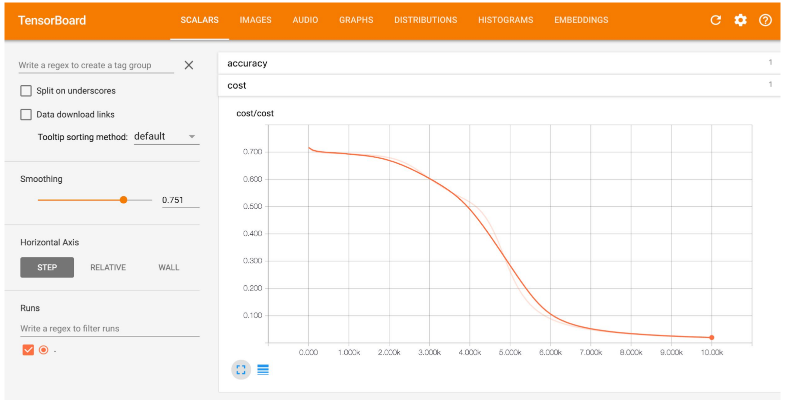
with TensorFlow

Lab 09-2 Tensorboard for XOR NN



Code: https://github.com/deeplearningzerotoall/TensorFlow
Slides: http://bit.ly/2LQMKvk
Lecturer: SuSang Kim (healess@kaist.ac.kr)

Tensorboard for XOR NN



```
$ pip install tensorboard
$ tensorboard —logdir=./logs/xor_logs
```

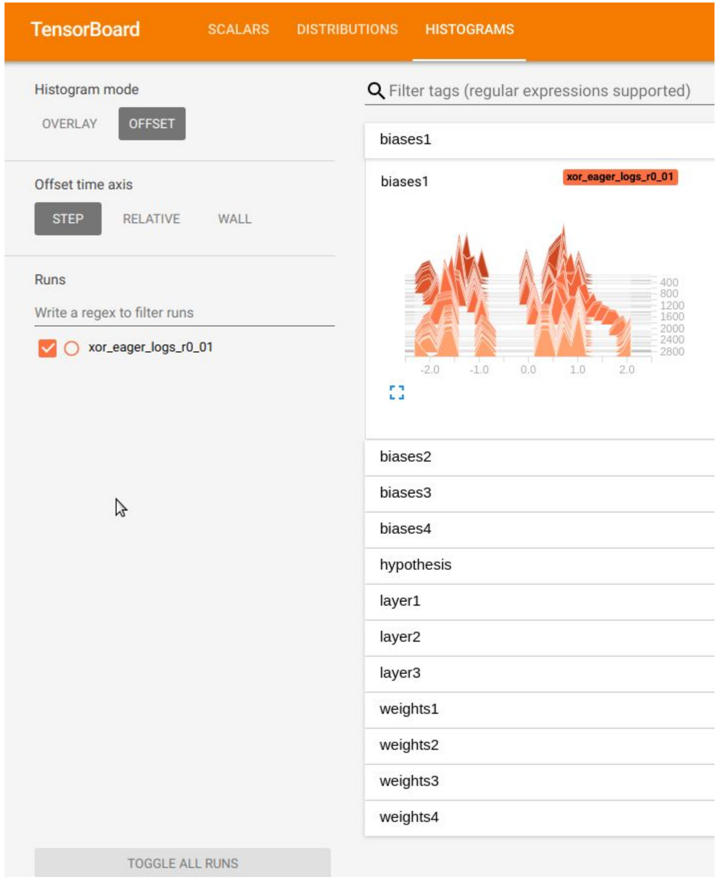
- You can navigate to http://127.0.0.1:6006

```
[Eager Execution]
writer = tf.contrib.summary.FileWriter("./logs/xor_logs")
with tf.contrib.summary.record_summaries_every_n_global_steps(1):
     tf.contrib.summary.scalar('loss', cost)
[Keras]
tb_hist = tf.keras.callbacks.TensorBoard(log_dir="./logs/xor_logs", histogram_freq=0,
write_graph=True, write_images=True)
model.fit(x_data, y_data, epochs=5000, callbacks=[tb_hist])
```

http://hunkim.github.io/ml/ (Tensorboard)

Code(Eager)

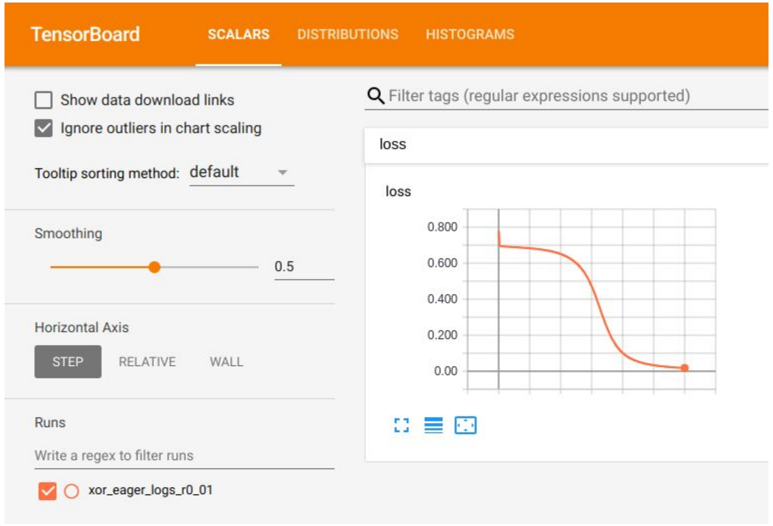
```
import tensorflow as tf
import tensorflow.contrib.eager as tfe
tf.enable_eager_execution()

x_data = [[0, 0], [0, 1], [1, 0], [1, 1]]
y_data = [[0], [1], [1], [0]]

dataset = tf.data.Dataset.from_tensor_slices((x_data, y_data)).batch(len(x_data))
def preprocess_data(features, labels):
  features = tf.cast(features, tf.float32)
  labels = tf.cast(labels, tf.float32)
  return features, labels

W1 = tf.Variable(tf.random_normal([2, 10]), name='weight1')
b1 = tf.Variable(tf.random_normal([10]), name='bias1')
                        ....
W4 = tf.Variable(tf.random_normal([10, 1]), name='weight4')
b4 = tf.Variable(tf.random_normal([1]), name='bias4')

def neural_net(features):
  layer1 = tf.sigmoid(tf.matmul(features, W1) + b1)
  layer2 = tf.sigmoid(tf.matmul(layer1, W2) + b2)
  layer3 = tf.sigmoid(tf.matmul(layer2, W3) + b3)
  hypothesis = tf.sigmoid(tf.matmul(layer3, W4) + b4)

  with tf.contrib.summary.record_summaries_every_n_global_steps(1):
     tf.contrib.summary.histogram("weights1", W1)
     tf.contrib.summary.histogram("biases1", b1)
     tf.contrib.summary.histogram("layer1", layer1)
                        ....
     tf.contrib.summary.histogram("weights3", W3)
     tf.contrib.summary.histogram("biases3", b3)
     tf.contrib.summary.histogram("layer3", layer3)
     tf.contrib.summary.histogram("weights4", W4)
     tf.contrib.summary.histogram("biases4", b4)
     tf.contrib.summary.histogram("hypothesis", hypothesis)
  return hypothesis
```



https://github.com/deeplearningzerotoall/TensorFlow/blob/master/lab-09-4-XOR-tensorboard-eager.ipynb

Code(Eager)

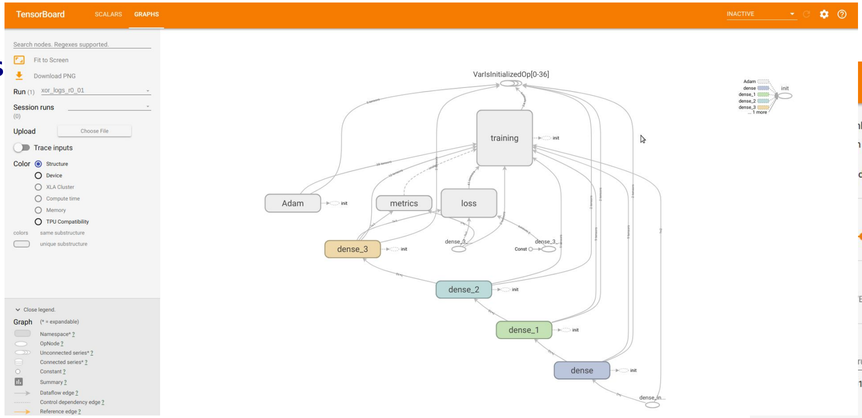
```
def loss_fn(hypothesis, labels):
  cost = -tf.reduce_mean(labels * tf.log(hypothesis) + (1 - labels) * tf.log(1 - hypothesis))
  with tf.contrib.summary.record_summaries_every_n_global_steps(1):
     tf.contrib.summary.scalar('loss', cost)
  return cost

optimizer = tf.train.GradientDescentOptimizer(learning_rate=0.1)

def accuracy_fn(hypothesis, labels):
  predicted = tf.cast(hypothesis > 0.5, dtype=tf.float32)
  accuracy = tf.reduce_mean(tf.cast(tf.equal(predicted, labels), dtype=tf.float32))
  return accuracy

def grad(hypothesis, features, labels):
  with tf.GradientTape() as tape:
     loss_value = loss_fn(neural_net(features),labels)
  return tape.gradient(loss_value, [W1, W2, W3, W4, b1, b2, b3, b4])

EPOCHS = 3000
log_path = "./logs/xor_eager_logs_r0_01"
writer = tf.contrib.summary.create_file_writer(log_path)
global_step=tf.train.get_or_create_global_step()  # global step variable
writer.set_as_default()

for step in range(EPOCHS):
  global_step.assign_add(1)
  for features, labels  in tfe.Iterator(dataset):
     features, labels = preprocess_data(features, labels)
     grads = grad(neural_net(features), features, labels)
     optimizer.apply_gradients(grads_and_vars=zip(grads,[W1, W2, W3, W4, b1, b2, b3, b4]))
     if step % 50 == 0:
        loss_value = loss_fn(neural_net(features),labels)
        print("Iter: {}, Loss: {:.4f}".format(step, loss_value))
x_data, y_data = preprocess_data(x_data, y_data)
test_acc = accuracy_fn(neural_net(x_data),y_data)
print("Testset Accuracy: {:.4f}".format(test_acc))
```

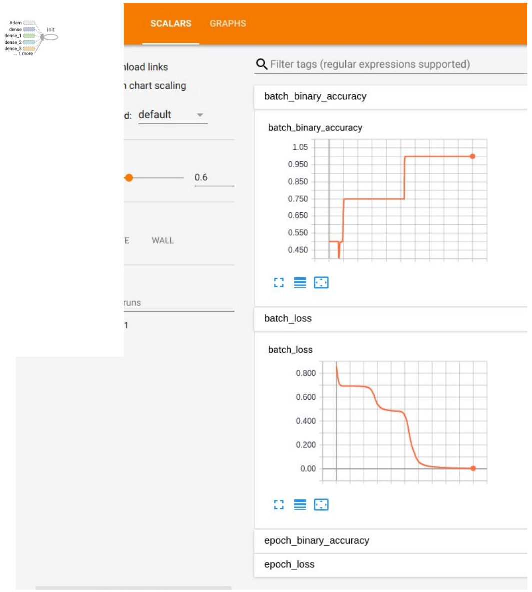


https://github.com/deeplearningzerotoall/TensorFlow/blob/master/lab-09-4-XOR-tensorboard-eager.ipynb

Code(Keras)

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import tensorflow as tf
import tensorflow.contrib.eager as

x_data = np.array([[0, 0],
         [0, 1],
         [1, 0],
         [1, 1]])
y_data = np.array([[0],
         [1],
         [1],
         [0]])

model = tf.keras.models.Sequential([
 tf.keras.layers.Dense(10,  input_dim=2, activation=tf.nn.sigmoid),
 tf.keras.layers.Dense(10, activation=tf.nn.sigmoid),
 tf.keras.layers.Dense(10, activation=tf.nn.sigmoid),
 tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)
])

model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['binary_accuracy'])
tb_hist = tf.keras.callbacks.TensorBoard(log_dir="./logs/xor_logs_r0_01", histogram_freq=0, write_graph=True,
write_images=True)
model.fit(x_data, y_data, epochs=5000, callbacks=[tb_hist])

model.predict_classes(x_data)
```





https://github.com/deeplearningzerotoall/TensorFlow/blob/master/lab-09-4-XOR-tensorboard-keras.ipynb