how to build it with tensorboard

In [ ]:

```
# First we need to carefully arrange our namescope
# Then we put tf.summary.scalar or tf.summary.histogram where we need
# Then we merge the summaries by: tf.summary.merge_all()
# finally, during the sess, we sess.run the mergerd and write the output (smy) to writer:
# how? by
# writer = tf.summary.FileWriter('logs/', sess.graph)
# and
# writer.add_summary(smy, epoch)
```

In [ ]:

```
import tensorflow as tf

def variable_summaries(var):
    with tf.name_scope('summarises'):
        mean = tf.reduce_mean(var)
        stddev = tf.sqrt(tf.reduce_mean(tf.square(var - mean)))

        tf.summary.scalar('mean', mean)
        tf.summary.scalar('stddev', stddev)
        tf.summary.scalar('max', tf.reduce_max(var))
        tf.summary.scalar('min', tf.reduce_min(var))
        tf.summary.histogram('histogram',var)
```

In [ ]:

```
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)

# 98% minist
batch_size = 100
n_batch = mnist.train.num_examples // batch_size

n_dim = 784
n_out = 10

with tf.name_scope('input'):
    with tf.name_scope('pic'):
        x = tf.placeholder(tf.float32, [None, n_dim])

    with tf.name_scope('label'):
        y = tf.placeholder(tf.float32, [None, n_out])

# layer 1
with tf.name_scope('layer_1'):
    nb_n_1 = 500

    with tf.name_scope('weights'):  # inside namescope, the variable will get its appended name
        w1 = tf.Variable(tf.truncated_normal([n_dim, nb_n_1], stddev = 0.1))
        variable_summaries(w1)

    with tf.name_scope('bias'):
        b1 = tf.Variable(tf.zeros([nb_n_1]) + 0.1)
        variable_summaries(b1)

    a1 = tf.nn.tanh(tf.matmul(x, w1) + b1)
    nb_n = nb_n_1

# layer 2
with tf.name_scope('layer_2'):
    nb_n_2 = 300

    with tf.name_scope('weights'):
        w2 = tf.Variable(tf.truncated_normal([nb_n, nb_n_2], stddev = 0.1))
        variable_summaries(w2)

    with tf.name_scope('bias'):
        b2 = tf.Variable(tf.zeros([nb_n_2]) + 0.1)
        variable_summaries(b2)

    a2 = tf.nn.tanh(tf.matmul(a1, w2) + b2)
    nb_n = nb_n_2

# layer 3
with tf.name_scope('layer_3'):
    nb_n_3 = n_out

    with tf.name_scope('weights'):
        w3 = tf.Variable(tf.truncated_normal([nb_n, nb_n_3], stddev = 0.1))
        variable_summaries(w3)

    with tf.name_scope('bias'):
        b3 = tf.Variable(tf.zeros([nb_n_3]) + 0.1)
        variable_summaries(b3)

    a3 = tf.nn.softmax(tf.matmul(a2, w3) + b3)
```

```python
# model & train
with tf.name_scope('loss'):
    loss = tf.reduce_mean(tf.square(y-a3))
    # better : loss = -tf.reduce_mean(y * tf.log(a3)) * 1000.0
    tf.summary.scalar('loss', loss)

with tf.name_scope('optimizier'):
    learning_rate =  tf.placeholder(tf.float32)
    optimizer =  tf.train.AdamOptimizer(learning_rate)

with tf.name_scope('train'):
    train = optimizer.minimize(loss)


# information
with tf.name_scope('accuracy'):
    correct_predict = tf.equal(tf.argmax(a3, 1), tf.argmax(y, 1))
    # argmax is biggest location
    # this is a list of bool

    # tf.cast make True to be 1
    accuracy = tf.reduce_mean(tf.cast(correct_predict, tf.float32))
    tf.summary.scalar('accuracy', accuracy)

init   = tf.initialize_all_variables()
merged = tf.summary.merge_all()


with tf.Session() as sess:
    writer = tf.summary.FileWriter('logs/', sess.graph)
    sess.run(init)
    n_epoch = 50
    for epoch in range(n_epoch + 1):
        for batch in range(n_batch):
            batch_xs, batch_ys = mnist.train.next_batch(batch_size)

            #  sess.run(train, feed_dict={x:batch_xs, y:batch_ys, learning_rate: 0.001 * (0.98**epoc
            #  smy = sess.run(merged, feed_dict = {x:batch_xs, y:batch_ys, learning_rate: 0.001 * ((

            _,smy = sess.run([train, merged], feed_dict={x:batch_xs, y:batch_ys, learning_rate: 0.00
            writer.add_summary(smy, epoch)

        if epoch % 2 == 0:
            acc = sess.run(accuracy, feed_dict={x: mnist.test.images, y: mnist.test.labels})
            print('{}%\tIteration {} : accuracy : {}'.format(float(epoch)*100/n_epoch, epoch, acc))

tf.reset_default_graph()
```