**常量：**tf.constant(tensor,name=””)

**变量：**

创建：tf.Variable(tensor,name=””)

或tf.Variable(其他变量.initialized\_value(),name=””)

初始化：tf.global\_variables\_initializer() 运行计算之前

**启动:**

sees = tf.Seesion()

sees.run(op)

**评估模型：**

tf.argmax(y,label) label可以是0,1 如tf.argmax(y,1)

给出某个tensor对象在某一维上的其数据最大值所在的索引值，由于标签向量是由0,1组成，因此最大值1所在的索引位置就是类别标签，比如tf.argmax(y,1)返回的是模型对于任一输入x预测到的标签值，而 tf.argmax(y\_,1) 代表正确的标签

tf.equal 用来检测我们的预测是否真实标签匹配(索引位置一样表示匹配)

correct\_prediction = tf.equal(tf.argmax(y,1), tf.argmax(y\_,1))

这行代码会给我们一组布尔值。为了确定正确预测项的比例，我们可以把布尔值转换成浮点数，然后取平均值。例如，[True, False, True, True] 会变成 [1,0,1,1] ，取平均值后得到 0.75.

accuracy = tf.reduce\_mean(tf.cast(correct\_prediction, "float"))

最后，我们计算所学习到的模型在测试数据集上面的正确率。

print (sess.run(accuracy, feed\_dict={x: mnist.test.images, y\_: mnist.test.labels}))

**保存变量：**

用tf.train.Saver()创建一个Saver来管理模型中的所有变量。

saver = tf.train.Saver() # 可以通过传入一个dict来选择保存哪些变量saver = tf.train.Saver({"my\_v2": v2})

save\_path = saver.save(sess, "/tmp/model.ckpt")

**关闭：**

sees.close()

**恢复变量**

用同一个Saver对象来恢复变量。注意，当你从文件中恢复变量时，不需要事先对它们做初始化。

saver = tf.train.Saver()

saver.restore(sess, "/tmp/model.ckpt")

常用函数：

**求和：**

tf.reduce\_sum(tensor)

**矩阵乘法**：

tf.matmul(a,b)

**对数：**

tf.log(tensor)

**softmax函数**:

tf.softmax(tensor)

**占位符**

x = tf.placeholder("float", [None, 10])

x不是一个特定的值，而是一个占位符placeholder, 我们在TensorFlow运行计算时输入这个值。我们希望能够输入任意数量的MNIST图像，每一张图展平成10维的向量。我们用2维的浮点数张量来表示这些图，这个张量的形状是[None，10 ]。（这里的None表示此张量的第一个维度可以是任何长度的。）

**梯度下降法**

train\_step =tf.train.GradientDescentOptimizer(0.01).minimize(cross\_entropy)

在这里，我们要求TensorFlow用梯度下降算法（gradient descent algorithm）以0.01的学习速率最小化交叉熵。梯度下降算法（gradient descent algorithm）是一个简单的学习过程，TensorFlow只需将每个变量一点点地往使成本不断降低的方向移动。当然TensorFlow也提供了[其他许多优化算法](http://www.tensorfly.cn/tfdoc/api_docs/python/train.html#optimizers)：只要简单地调整一行代码就可以使用其他的算法。

**区分一下：**

启动模型与训练模型

1. 先在Seesion中启动模型

sess = tf.Session()

sess.run(init)

1. 再开始训练

sess.run(train\_step, feed\_dict={x: batch\_xs, y\_: batch\_ys})

注意sees已经在1启动了模型，但还没开始训练！！！只是存了各个op，然后才执行训练（训练中包含了梯度下降法（train\_step已经在上面定义了）与输入输出值（feed\_dict用来替换上面定义的占位符））

cross\_entropy是一个最终的op,此op含有初始的输入值（该输入值用占位符定义）

train\_step = tf.train.GradientDescentOptimizer(0.01).minimize(cross\_entropy)

sess.run(train\_step, feed\_dict={x: batch\_xs, y\_: batch\_ys} 此句的意思是：用feed\_dict替代占位符来运行trian\_step