

tensorflow中有两个关于variable的op, `tf.Variable()` 与 `tf.get_variable()` 下面介绍这两个创建变量函数的区别

先来看看这两个函数的参数列表, 就不打了, 直接截图:

```
self: Variable, initial_value=None, trainable: bool=True,
collections=None, validate_shape: bool=True, caching_device=None,
name=None, variable_def=None, dtype=None, expected_shape=None,
import_scope=None, constraint=None
```

```
4 tf.Variable()
```

```
name, shape=None, dtype=None, initializer=None, regularizer=None,
trainable: bool=True, collections=None, caching_device=None,
partitioner=None, validate_shape: bool=True, use_resource=None,
custom_getter=None, constraint=None
```

```
7 tf.get_variable()
```

首先有一个区别非常明显:

(1) `tf.Variable()` 初始化是直接传入 `initial_value`, 我们使用的时候一般是这样子初始化的:

```
a = tf.Variable(initial_value=tf.random_normal(shape=[200, 100],
stddev=0.1), trainable=True)
```

(2) `tf.get_variable()` 初始化是传入一个 `initializer`:

```
b = tf.get_variable(name = 'weights', shape=[200, 100],
dtype=tf.float32,
initializer=tf.random_normal_initializer(stddev=0.1))
```

当然, 明显的区别明显都能看出来, 重点是下面这个区别

使用tf.Variable时，如果检测到命名冲突，系统会自己处理。使用tf.get\_variable()时，系统不会处理冲突，而会报错

```
import tensorflow as tf
w_1 = tf.Variable(3,name="w_1")
w_2 = tf.Variable(1,name="w_1")
print w_1.name
print w_2.name
#输出
#w_1:0
#w_1_1:0
```

```
import tensorflow as tf
w_1 = tf.get_variable(name="w_1",initializer=1)
w_2 = tf.get_variable(name="w_1",initializer=2)
#错误信息
#ValueError: Variable w_1 already exists, disallowed. Did
#you mean to set reuse=True in VarScope?
```

所以当我们需共享变量的时候，需要使用tf.get\_variable()。在其他情况下，这两个的用法是一样的。为了方便变量管理，tensorflow 还有一个变量管理器，叫做tf.variable\_scope，举个栗子：

```
import tensorflow as tf

with tf.variable_scope("scope1"): # scopename is scope1
    w1 = tf.get_variable("w1", shape=[])
    w2 = tf.Variable(0.0, name="w2")
with tf.variable_scope("scope1", reuse=True):
    w1_p = tf.get_variable("w1", shape=[])
    w2_p = tf.Variable(1.0, name="w2")

print(w1 is w1_p, w2 is w2_p) # True False
```

---

版权声明：本文为CSDN博主「jeffery0207」的原创文章，遵循CC 4.0 by-sa版权协议，转载请附上原文出处链接及本声明。

原文链接：<https://blog.csdn.net/jeffery0207/article/details/79842611>

这就是这两个函数的区别了，在构建网络的时候用起来，你就能够更加深入的了解他们的区别。