AI学习笔记--Tensorflow--自动微分

本章主要学习了 TensorFlow 的自动微分处理。这是在机器学习领域十分重要的关键算法。

开始

编写代码：

```
from __future__ import absolute_import, division, print_function,
unicode_literals
import tensorflow as tf
tf.enable_eager_execution()
```

梯度

TensorFlow 提供了梯度 API，例如tf.GradientTape API,这个 API 是专门用户计算微分，并且 TensorFlow 会记录每个梯度的运算结果到 GradientTape 中的 Tape 当中，然后 Tensorflow 会依据每个记录到的 Tape 来求得积分梯度的逆。例如：

```
from __future__ import absolute_import, division, print_function,
unicode_literals
import tensorflow as tf
tf.enable_eager_execution()

x = tf.ones((2, 2))

with tf.GradientTape() as t:
  t.watch(x)
  y = tf.reduce_sum(x)
  z = tf.multiply(y, y)

# Derivative of z with respect to the original input tensor x
dz_dx = t.gradient(z, x)
for i in [0, 1]:
  for j in [0, 1]:
    assert dz_dx[i][j].numpy() == 8.0
```

而且，你也可以在 记录 tf.GradientTape context 过程中输出计算出的梯度。

```
x = tf.ones((2, 2))
with tf.GradientTape() as t:
  t.watch(x)
  y = tf.reduce_sum(x)
  z = tf.multiply(y, y)

# Use the tape to compute the derivative of z with respect to the
# intermediate value y.
dz_dy = t.gradient(z, y)
assert dz_dy.numpy() == 8.0
```

```
print("dzx")
print(dz_dx)
print("dzy")
print(dz_dy)
```

默认情况下，GradientTape所拥有的资源在调用GradientTape.gradient（）方法时立即释放。要在同一计算中计算多阶，需要创建一个 persistent gradient tape。这允许多次调用gradient（）方法。当垃圾回收磁带对象时释放资源。例如：