TensorFlow 基礎篇〈中〉

2017-08-01

前言

瞭解完tensorflow的特點後，我們要開始正式介紹tensorflow的內容，這回會先教大家Tensorflow會用到的基礎元件
用，幫助各位在真正開始訓練一個屬於自己的模型之前，先對tensorflow運作的原理有基礎的認識。

張量(TENSOR)

tensorflow運算中最基本的單位，通常是高維度的矩陣，舉個例子如果我們需要一個可以拿來做圖像辨識的Model
片即可用一個2維的tensor表示，圖上的每個像素(pixel)都是tensor中的element，同樣的方式也可以用三維的張量來
色的圖，第三個維度長度就是3分別代表RGB三原色在現在這個(x,y)位置的pixel的值。

張量的型態主要有三種Constant、Variable、Placeholder

如果有寫過其他程式語言的應該可以很輕易地了解Constant 和 Variable的意義，沒學過也沒關係，可以就如字面上
Constant就是不可變的常數、Variable即是變數，Placeholder則是tensorflow中比較特別的概念，我們就留到與
講。
創建Tensor的方式也很簡單

```
import tensorflow as tf
node1 = tf.constant(3) # 常數3的tensor
node2 = tf.constant(4) # 常數4的tensor
print(node1, node2)
```

我們來看看print出來的結果

```
Tensor("Const:0", shape=(), dtype=float32) Tensor("Const_1:0", shape=(), dtype=float32)
```

node1、node2 分別代表了兩個Const和Tensor，還可以注意到如果我們沒有特別指定data type，default是使用fl
始值，但為什麼我們看不到我們賦予這兩個tensor的值(3、4)呢?
這時候我們就必須執行下面這段程式碼

```
sess = tf.Session() # 創建一個tensorflow session
print(sess.run([node1, node2])) # 把node1 node2丟進session中run
```

大家現階段可能對於這段程式碼的功用不太理解，沒關係這我們在下個階段就會有詳細的解說，這邊就當作是把Ten
出吧。

運算圖(COMPUTATIONAL GRAPH)

在開始之前我們先理解一下Tensorflow主要的運作流程分為以下兩個部分

建立運算圖(Build)

其實這個部分就是所謂的建立模型(Build Model)，決定整個運算流程，這也是tensorflow和其他框架最大的不同，我
定義好Graph的長相，舉下圖為例，如果想用tensorflow算 3+4=7 那我們該怎麼做呢？首先我們要先建立一個用來
的Graph如下圖，就是把兩個Constant的tensor做一個加法的運算。


```
node3 = tf.add(node1, node2) # 把前面創建的const tensor node1、node2相加
print("node3: ", node3) # node3就是相加後的結果
print("sess.run(node3): ",sess.run(node3))
```

執行運算圖(Run)

相信看完上面那段程式碼，大家一定很好奇第三行又再度出現的 sess 是在做甚麼，我不是做完第一行以後 node3 就
的結果了嗎?
那我們就來看一下 node3 print出來的結果會是甚麼。

```
node3:  Tensor("Add:0", shape=(), dtype=float32) # node3 print結果
```

看到這邊可以發現node3其實和上面的node1、node2都一樣是一個Tensor差別只是名字被命為add也就是加法產生
那我們加法加出來的值又去哪了呢?聰明的你們應該可以想到第三行print出來的結果就會是加法的結果了。

```
sess.run(node3):  7.0 # sess.run(node3) print結果
```

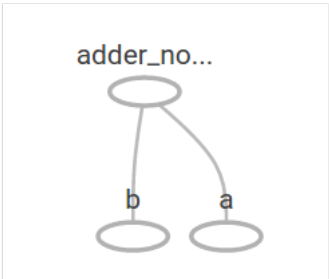
好那到底為什麼我們需要session才會有值呢?這就牽涉到了tensorflow設計的原理，顧名思義Tensorflow就是讓Ten
像是我們在第一回所看到的圖一樣，我們在前面建立Graph的過程其實只是定義好Tensor如何流動並運算的過程，但
其實並沒有被運算，所以我們需要session。

session幫助我們定義我們所需要run的Graph的input和output，在上面的例子由於我們已經先定義好constant Ten
需要給Graph任何Input我們就可以得到node3的output，但如果我們要的不是兩個常數相加呢?在一般程式設計中
寫 `c = a + b` ，a和b兩個變數又是由其他運算所得來，那在Tensorflow中會是長怎樣呢?

Placeholder

正如上面所提到，在Tensorflow中我們都是先建好Graph再決定資料的input與output，這時候我們就需要Placehol
們在還沒有資料的時候先佔個位子(正如其名)。

```
a = tf.placeholder(tf.float32)
b = tf.placeholder(tf.float32)
adder_node = a + b  # 這行等效於 adder_node = tf.add(a, b)
```



以上就是我們將加法改為兩個變數相加，用Placeholder實作的程式碼，跟前面用兩個Const Tensor相加最大的區別
邊並沒有賦予a、b任何值，而只是建立了一個讓a跟b兩個Tensor相加的Graph。

```
print(sess.run(adder_node, {a: 3, b:4.5}))
>> 7.5
```

接著我們當然又是找session幫我們執行這個Graph，可以看到在 `sess.run` 的參數中我們除了第一個參數指定了這
Output外，在第二個參數我們給了一個dictionary，這就是我們這次run的過程賦予a和b兩個Placeholder
adder_node的計算就會根據我們feed進去的資料作改變，在這邊很顯然的答案就是 `3+4.5=7.5` 。

```
print(sess.run(adder_node, {a: [1,3], b: [2, 4]}))
>> [ 3.  7.]
```

但在真正機器學習的任務中當然不可能都是做這種簡單的兩個數相加，稍微有上過課或者有點概念的讀者應該就會知
器學習中大部分的運算都是高維度的矩陣運算，這也是為什麼基本單位叫做Tensor(高維度矩陣)，我們其
placeholder任何維度任何長度的data。

這邊示範了一個餵給a跟b兩個一維的向量，那加法出來的結果就會是這兩個向量相加，那當然也不是隨便餵給Plac
data都可以的，至少在這個例子中如果餵給a跟b兩個不同維度的Data那在 `sess.run` 之後便後產生Runtime Error，
們所有要餵進Graph的Data都必須先想好之後會遇到甚麼運算，譬如矩陣相乘就必須在要相乘的維度上要有相同的長
不同的限制。

Variable

到最後大家可能會有一個疑問：那Variable在Tensorflow中的功用是甚麼呢?其實所有除了Placeholder跟Constant
都屬於Variable，也就是剛剛做加法運算的Graph中的adder_node也屬於一個Variable，Variable就是會因為Pla
Input的Data不同會隨之改變，這也是在整個Model中最重要的部分。

大家應該知道Deep Learning的Training過程就是針對輸出(Output)與事實(Ground Truth)的差別，去更新Model中
這些可以被更新的參數就是Variable，在下篇我們回提到怎麼建立一個簡單的模型，讀者看完以後應該能對於整個M
底是怎麼被Train出來的更有概念。

> 作者：台大電機所碩二 林哲賢