機器學習 (Project 2)

P.S 重點再遇到的問題……QQ

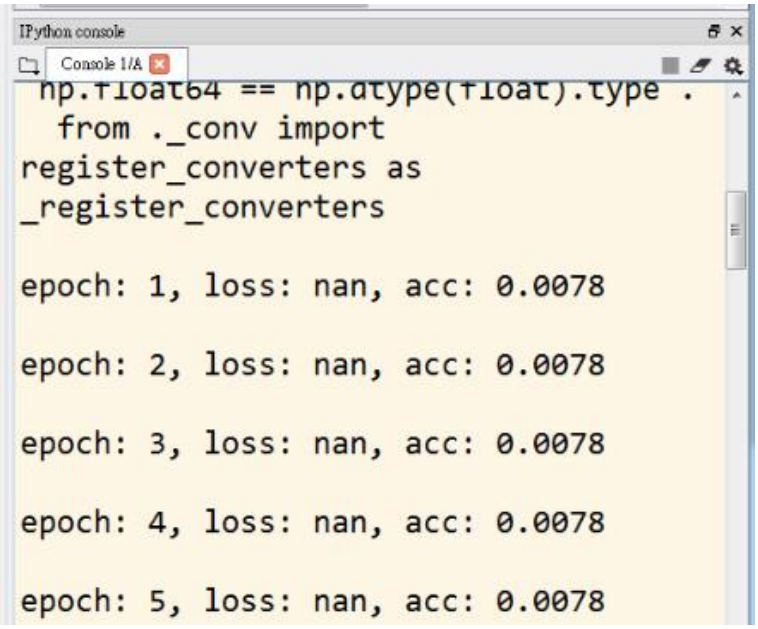
Q1: 方法描述 (Method Description)

Ans: 這次的Project是利用tensorflow來寫一個關於狗狗品種分類的Kaggle競賽。由於之前大三在修人工智慧的時候，有用過pandas等Library，並且有讀入和寫入過CSV檔，因此在Code的書寫上，並沒有遇到多大的困難。再加上上次的作業有用過cv2來對於影像做resize和imread(讀檔)，所以在讀照片上也沒有遇到困難。這次對我來說唯一新的東西應該就是要對train_y做one-hot encoding和畫圖了，原本我是用Keras內的np_utils.to_categorical(train_y)來做，但由於我的電腦GPU容量不足(會報Memory Error)，而且後來又發現PPT規定說只能用Tensorflow書寫，所以最後就用了tf.one_hot(Train_y, 120, dtype=np.float32)，來將test資料轉成float32 的one_hot型態。至於電腦GPU不足這次造成了我滿多問題的@@，包含我必須將程式下面所有的float32都要轉成float16才能成功的跑(這個BUG讓我重跑大概10次，每次約30分鐘……)，不過也是因為這樣做，所以在Load Pretrain Model的時候會造成問題，在後面遇到的困難的地方會詳細說明。至於畫圖的話其實就相對比較簡單了，就只要先宣告兩個list，然後在train的那個for迴圈中，將次數append給x_axis，再append accuracy給y_axis，和loss append給z_axis，最後再plt.plot(x_axis, y_axis, '-o')和plt.plot(x_axis, z_axis, '-o')並且plt.show()畫出即可。不過其實在畫圖時也有遇到一個小困難，一樣會在遇到的困難的地方詳細說明。

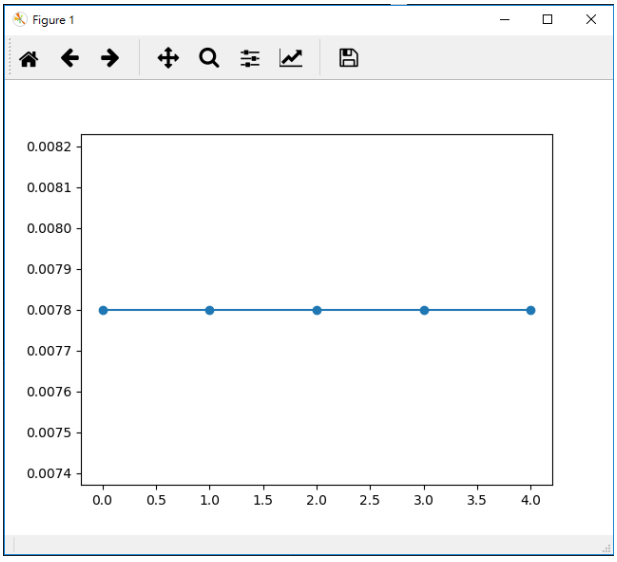
Q2: 程式結果 (Experimental Results - Accuracy)

1. **不含 Pre-train Model**，model.py 的 Fully Connected 設 1024，epoch=5，

   batch_size=16 (用自己電腦測的)



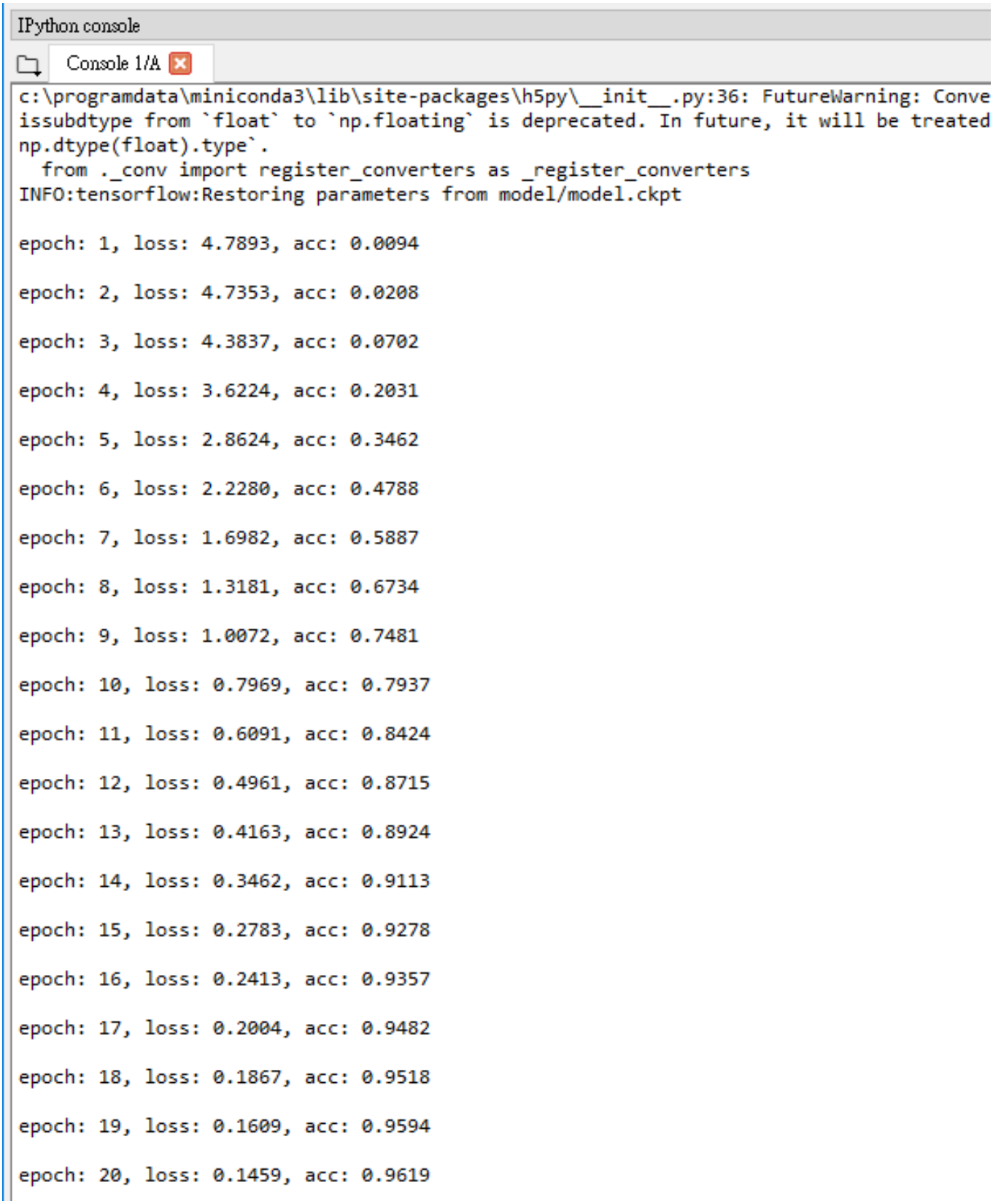
Plot 的圖: (x-axis 為 Epoch，y-axis 為 Accuracy)



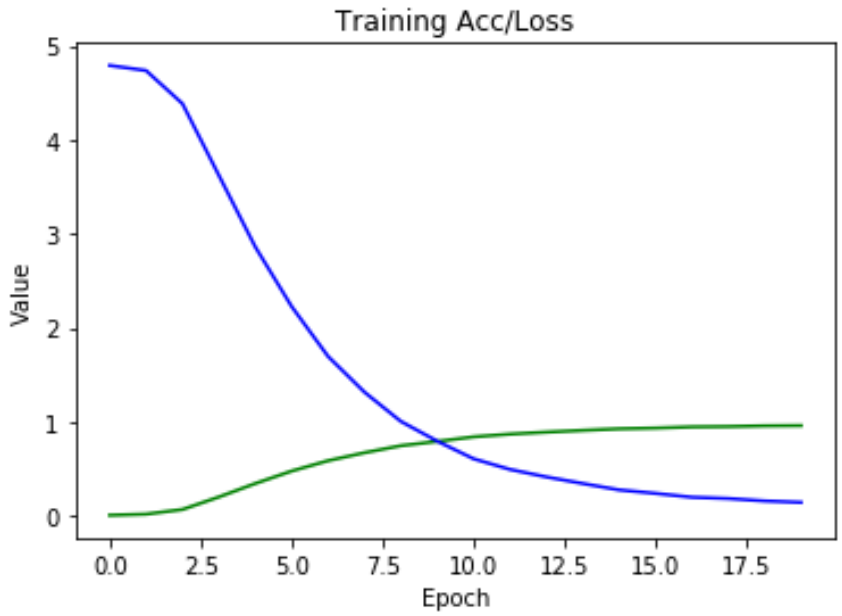
說明: 由於 Epoch 五個 Accuracy 都沒有進展 (Accuracy 也是都長一樣)，而且 loss 還大到顯示 nan。所以就沒有在繼續往下測了，估計是因為沒有 Pretrain Model 的 include 所以才會這樣，很容易掉進錯的 Local Maximum。

2. **含 Pre-train Model**，model.py 的 Fully Connected 設 4096，epoch=20，

   batch_size=16 (借別人電腦測的 QQ)

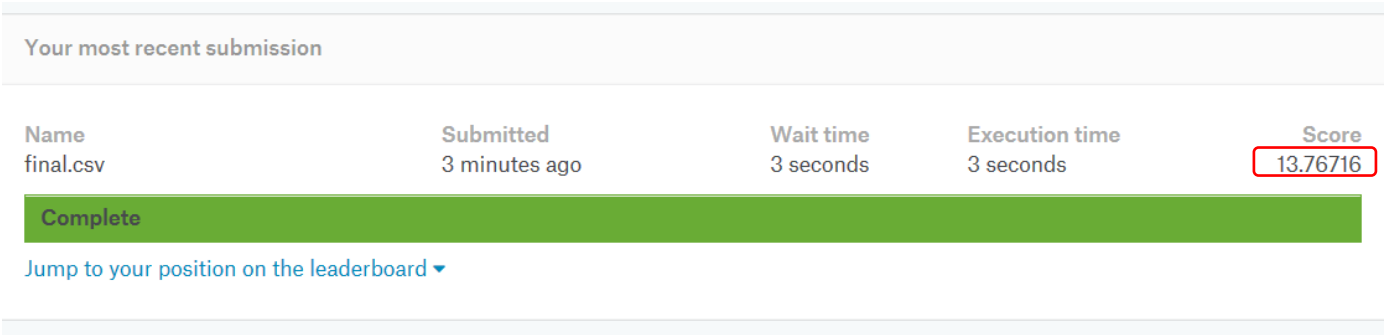


Plot 的圖: (x-axis 為 Epoch，y-axis (綠線)為 Accuracy，z-axis (藍線)為 Loss)




```
In [2]:
```

3. **My Score:** 13.76716

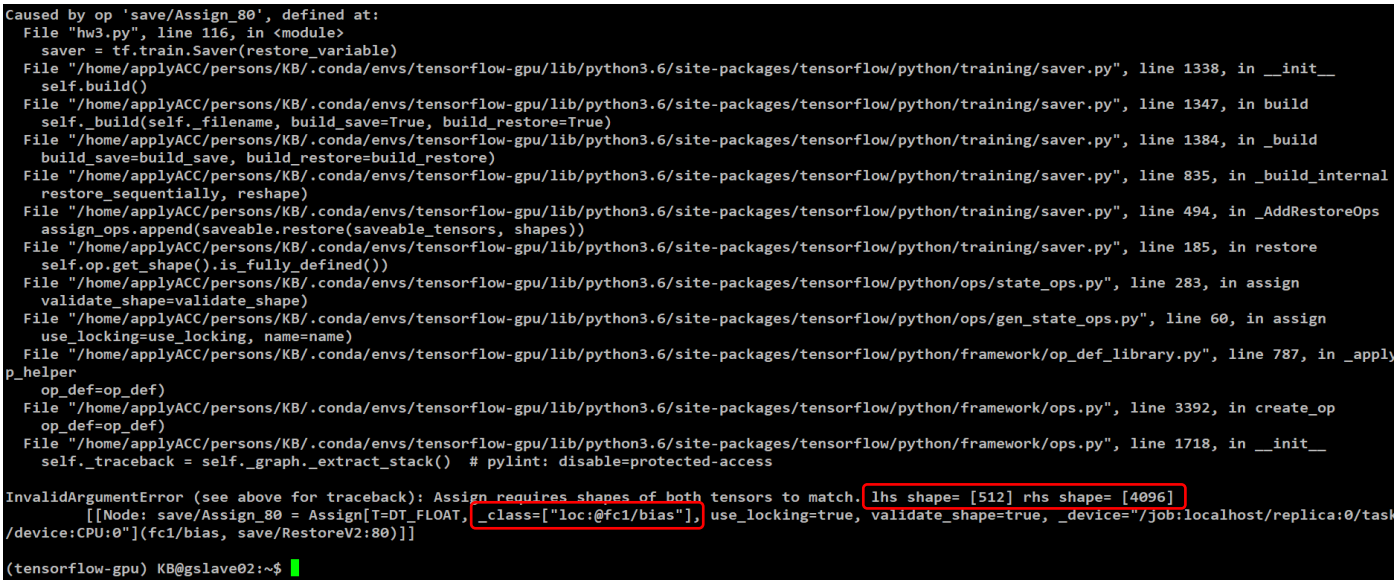


Kaggle 上的成績: 約為第 1257 名

1253	new	**heyy**		12.59116	5	3mo
1254	▾ 90	**slimbumzie**		12.92990	1	4mo
1255	▾ 90	**Inspelliam Chen**		13.10965	5	5mo
1256	▾ 90	**mangoli**		13.21279	4	4mo
1257	▾ 90	**AdarshHonawad**		13.93958	2	5mo
1258	▾ 90	**SebastianHuber**		14.38992	4	4mo
1259	▾ 90	**xuexia**		14.75662	1	5mo

Q3: 遇到的困難 (Discussion of Difficulty or Problem Encountered):

1. 最大也是最嚴重的問題，**我的GPU資源不夠ＱＱＱＱ！！！！**不管是用我自己的1050Ti 2G跑或是用工學院伺服器的跑，只要include pretrain的進來，我的電腦會顯示Memory Error，而工學院伺服器會出現這個錯誤訊息。



因此在跑我的Model的時候，除了model.py的Fully Connected最大的Node數只能設在1024之外，pretrain還不能include進來@@，**導致Accuracy奇差無比(0.0078)**。後來經由助教的幫助和我同學張世亞(亞哥)友情幫忙用實驗室的電腦001跑我的程式，才成功的跑出了0.962的Accuracy。這部分的Bug是最難De(因為要try & error)，也是最耗費時間的一環(因為常常會跑到一半掛掉，又不知道原因是什麼和該怎麼DEBUG@@，最後靠著助教和同學的幫忙才完美的解決，十分感謝他們！！)。

2. **try & error的問題。**其實一直到最後我才想到，用jupyter notebook並且用np.savez來存已經pretrain好的變數來寫，會比用spyder一直restart kernel跑來的快的多的多，因為光是讀檔就要耗費掉將近30分鐘了，常常都是等了個30分鐘左右，才能等到自己的BUG

然後再修改。之後如果還有這種作業，應該會採取用jupyter notebook並且用np.savez來寫。

3. **畫圖**。關於畫圖其實我原本是接收with下面的變數(如下圖)

```
with tf.name_scope('LossLayer'):
    loss = tf.losses.softmax_cross_entropy(onehot_labels=y, logits=logits)
with tf.name_scope('Optimizer'):
    train_op = tf.train.AdamOptimizer(LEARNING_RATE).minimize(loss)
with tf.name_scope('Accuracy'):
    accuracy = tf.reduce_mean(tf.cast(tf.equal(tf.argmax(y, axis=1), tf.argmax(logits, axis=1)), tf.float32))
```

但是後來產生Value Error後，聽助教說是要從上方的程式回傳變數來append才會是對的(如下圖，紅色框框) ，才成功的plot()出正確的圖形。

```
def train_eval(sess, x_data, y_label, batch_size, train_phase, is_eval, epoch=None):
    n_sample = x_data.shape[0]
    n_batch = int((n_sample+batch_size-1) / batch_size)
    tmp_loss, tmp_acc = 0, 0
    for batch in range(n_batch):
        start = batch * batch_size
        end = min(n_sample, start + batch_size)
        _, batch_loss, batch_acc = sess.run([train_op, loss, accuracy], feed_dict={x: x_data[start:end],
        tmp_loss += batch_loss * (end - start)
        tmp_acc += batch_acc * (end - start)
    tmp_loss /= n_sample
    tmp_acc /= n_sample

    if train_phase:
        print('\nepoch: {0}, loss: {1:.4f}, acc: {2:.4f}'.format(epoch+1, tmp_loss, tmp_acc))

    return tmp_acc, tmp_loss
```

4. **float16變float32**。這是我遇到最大的困難之一，最後也是靠助教合力才找出問題的@@(感謝助教~~)。原本因為我GPU記憶體不足，要用float16才能跑沒有pretrain weight的model，因此我就把所有float32的地方改成了float16。導致我最後在pretrain model時，node數總是會差兩倍。最後在助教們發現並且修改過來之後，程式才能夠成功的正常運行，真的真的非常感謝助教們的幫忙QQ，謝謝！