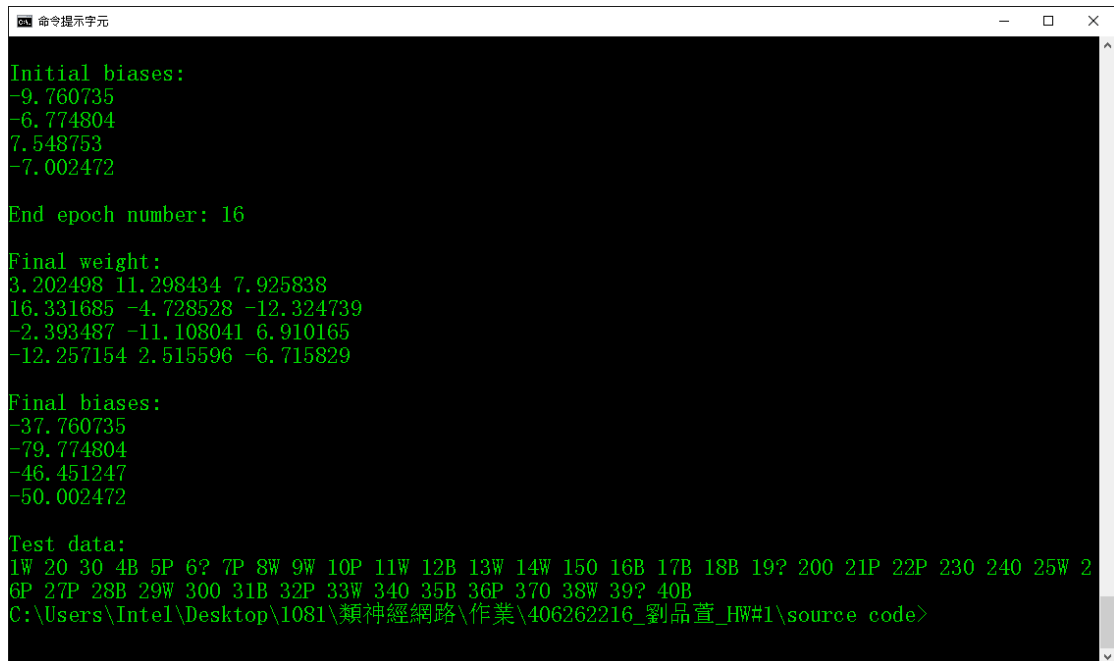


類神經網路 程式作業一 報告

1. Compare the performance of your two-neuron perceptron and four-neuron perceptron.

因為此訓練完成的 Perceptron Network，在測試 Test data 的時候，只會比對他的輸出 a 是否有符合已經設定好的種類的 target 值，有的話才會被歸類成某類，沒有的會就會是？，此現象在圖(1)的四個神經元的情況中更為明顯：



```
命令提示字元

Initial biases:
-9.760735
-6.774804
7.548753
-7.002472

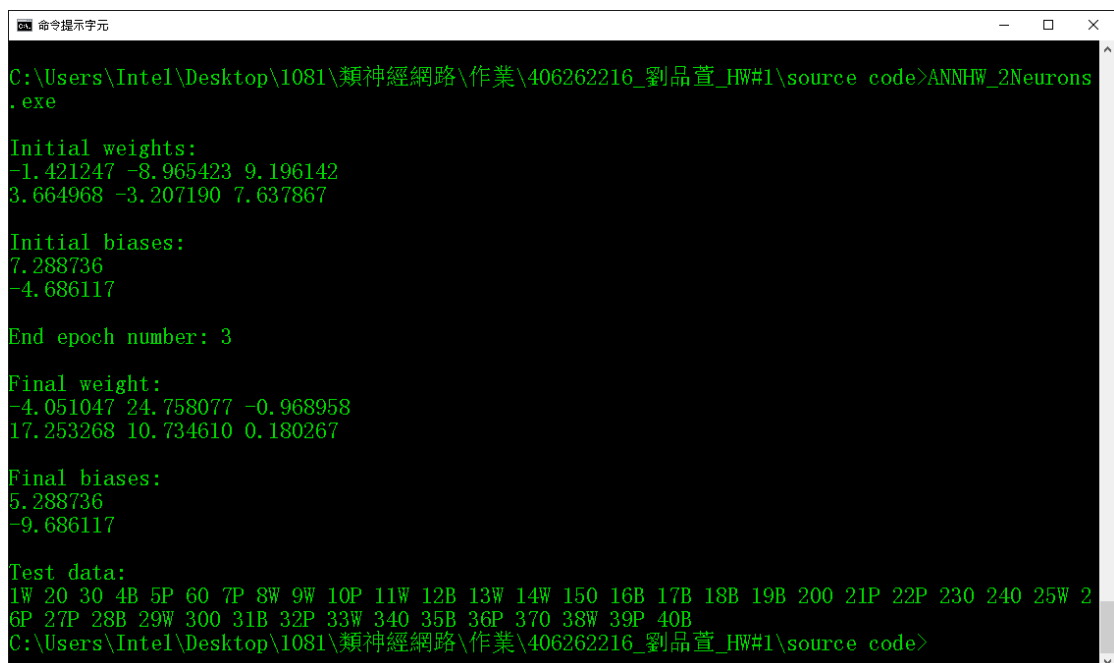
End epoch number: 16

Final weight:
3.202498 11.298434 7.925838
16.331685 -4.728528 -12.324739
-2.393487 -11.108041 6.910165
-12.257154 2.515596 -6.715829

Final biases:
-37.760735
-79.774804
-46.451247
-50.002472

Test data:
1W 20 30 4B 5P 6? 7P 8W 9W 10P 11W 12B 13W 14W 150 16B 17B 18B 19? 200 21P 22P 230 240 25W 2
6P 27P 28B 29W 300 31B 32P 33W 340 35B 36P 370 38W 39? 40B
C:\Users\Intel\Desktop\1081\類神經網路\作業\406262216_劉品萱_HW#1\source code>
```

圖(1) 4 個神經元的情況下，第 6 個、第 19 個、第 39 個 Test data 無法分類到某類別



```
命令提示字元

C:\Users\Intel\Desktop\1081\類神經網路\作業\406262216_劉品萱_HW#1\source code>ANNHW_2Neurons.exe

Initial weights:
-1.421247 -8.965423 9.196142
3.664968 -3.207190 7.637867

Initial biases:
7.288736
-4.686117

End epoch number: 3

Final weight:
-4.051047 24.758077 -0.968958
17.253268 10.734610 0.180267

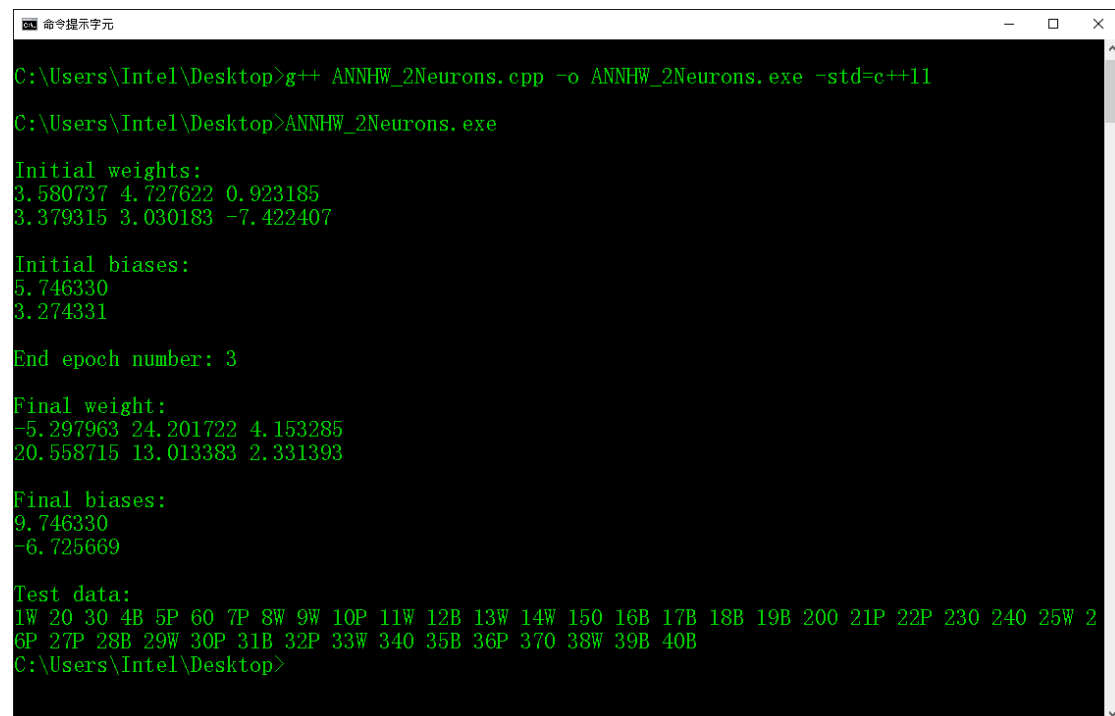
Final biases:
5.288736
-9.686117

Test data:
1W 20 30 4B 5P 60 7P 8W 9W 10P 11W 12B 13W 14W 150 16B 17B 18B 19B 200 21P 22P 230 240 25W 2
6P 27P 28B 29W 300 31B 32P 33W 340 35B 36P 370 38W 39P 40B
C:\Users\Intel\Desktop\1081\類神經網路\作業\406262216_劉品萱_HW#1\source code>
```

圖(2) 兩個神經元的情況下，40 個 Test data 都有類別

2. Compare the performance of different initial weights/biases.

利用自定義的函數 `double fRand(double fMin, double fMax)` · 產生某隨機 M 及 b (M 及 b 的內容值控制在 $10 \sim -10$ 之間)。使得每一次的執行， M 和 b 在訓練時發生的更新變化不同，進而使得訓練 Perceptron Network 所花費的世代數不同。帶入 Test data 後產生的結果，也造成有時候會發生小幅度的差異 (差不多 1 至 2 個)。如下圖(3)、圖(4)所示：



```
C:\Users\Intel\Desktop>g++ ANNHW_2Neurons.cpp -o ANNHW_2Neurons.exe -std=c++11
C:\Users\Intel\Desktop>ANNHW_2Neurons.exe

Initial weights:
3.580737 4.727622 0.923185
3.379315 3.030183 -7.422407

Initial biases:
5.746330
3.274331

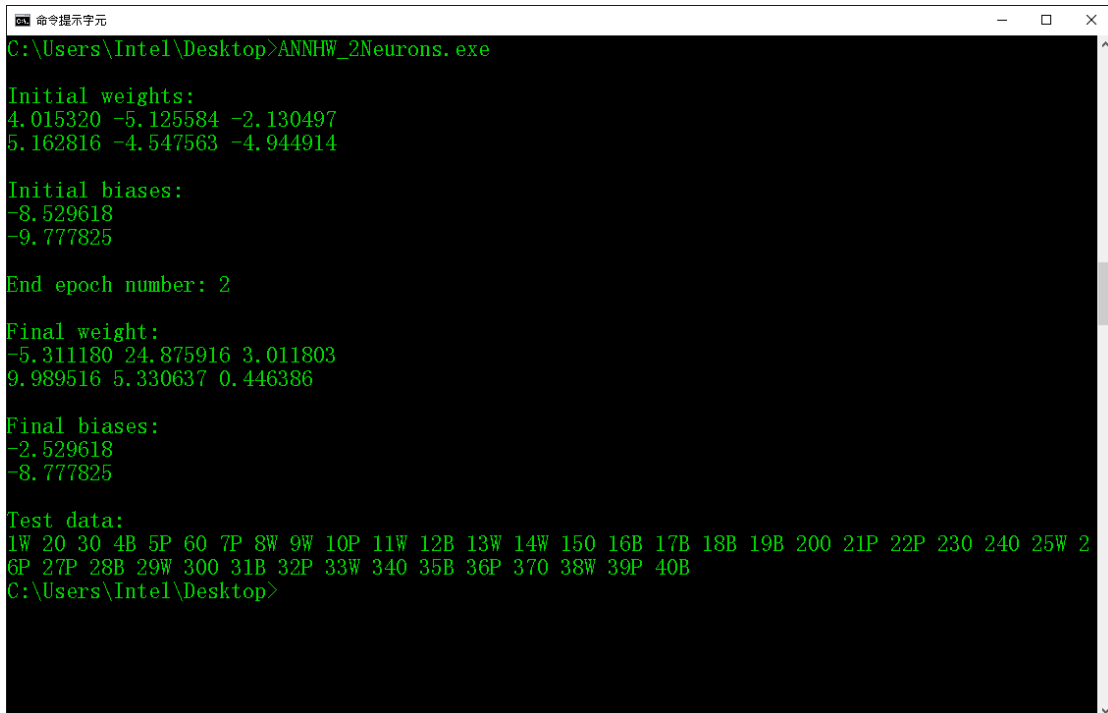
End epoch number: 3

Final weight:
-5.297963 24.201722 4.153285
20.558715 13.013383 2.331393

Final biases:
9.746330
-6.725669

Test data:
1W 20 30 4B 5P 60 7P 8W 9W 10P 11W 12B 13W 14W 150 16B 17B 18B 19B 200 21P 22P 230 240 25W 2
6P 27P 28B 29W 30P 31B 32P 33W 340 35B 36P 370 38W 39B 40B
C:\Users\Intel\Desktop>
```

圖(3) 花費 3 個世代訓練完畢



```
命令提示字元
C:\Users\Intel\Desktop>ANNHW_2Neurons.exe

Initial weights:
4.015320 -5.125584 -2.130497
5.162816 -4.547563 -4.944914

Initial biases:
-8.529618
-9.777825

End epoch number: 2

Final weight:
-5.311180 24.875916 3.011803
9.989516 5.330637 0.446386

Final biases:
-2.529618
-8.777825

Test data:
1W 20 30 4B 5P 60 7P 8W 9W 10P 11W 12B 13W 14W 150 16B 17B 18B 19B 200 21P 22P 230 240 25W 2
6P 27P 28B 29W 300 31B 32P 33W 340 35B 36P 370 38W 39P 40B
C:\Users\Intel\Desktop>
```

圖(4) 花費 2 個世代訓練完畢

3. Compare the performance of different learning rates.

我針對四個神經元的 Perceptron Network，在 cpp 檔內 define 了一個名為 learningRate 的變數，並加上迴圈，使得在某一個特定 learning rate 的情況下執行了 10000 次的訓練（每一次所使用的 W 及 b 都為亂數產生）。

其 learning rate 包括了 1(LearningRate1In4Neurons.txt)、

0.75(LearningRate1In4Neurons.txt)、0.5(LearningRate1In4Neurons.txt)、

0.25(LearningRate1In4Neurons.txt)、0.01(LearningRate1In4Neurons.txt)、

0.001(LearningRate1In4Neurons.txt)六種。

而在訓練完這六種不同 learning rate 10000 次的情況下，產生的差異如下：

在 learning rates 為 1 的狀況時：平均 epoch 數為 8

在 learning rates 為 0.75 的狀況時：平均 epoch 數為 10

在 learning rates 為 0.5 的狀況時：平均 epoch 數為 11

在 learning rates 為 0.25 的狀況時：平均 epoch 數為 10

在 learning rates 為 0.01 的狀況時：平均 epoch 數為 29

在 learning rates 為 0.001 的狀況時：平均 epoch 數為 298

從以上數據分析可得知，當 rate 從 1 在逐一情況降低至 0.25 的時候，平均花費的 epoch 有往上升的趨勢，但是仍有一些小浮動。而在分析的結果中，可以看出最明顯的差異是在 learning rates 設置為 0.01 與 0.001 時 epoch

數大幅度增加。因此可知 learning rate 與 epoch 數有一定關係，通常當 learning rate 降低的時候，epoch 數會增加。

(上述分析的數據都以檔案寫出的方式寫入各自 txt 檔案中)