# 類神經網路作業一

資工三乙 406262319 黃育晧

## TwoNeuron V.S. FourNeuron

### Error

- Four Neuron
  - 由於每個神經元皆只表示是否屬於該物件，導致訓練完成後將測試資料丟進去會造成有些資料並不屬於任何一個亦或著同時屬於多種結果
  - 每次進行測試時，這些Error的數量在40個測試資料中平均下來落在 2.0~2.5 個之間
- Two Neuron
  - 藉由二神經元將所有物件區分成四個區塊，每個測試資料丟進來皆必定屬於其中的某個區塊

### 時間差異

我將二種方式皆進行1000次測試，並記錄在其進行訓練的時間

- Four Neuron
  - 平均每次測試的時間落在 0.10~0.15 sec 左右
- Two Neuron
  - 平均每次測試的時間落在 0.02 sec 左右

二個神經元的訓練速度大約是四個神經元的五到六倍時間

## Different Initial Weights/Biases

我在測試期間的 `weights/biase` 皆是用隨機變數去產生的，根據 `weights/biase` 的不同，所需的 Epoch 數亦不同，其中

- Four Neuron
  - Epoch數量的差距頗大，在測試資料讀進去後得出的結果會在Error的部分有蠻大的出入，其餘結果大致皆相同
- Two Neuron
  - Epoch數量的差距不大，在測試資料讀進去後得出的結果也大致都是相同的

## Different Learning Rate

我是利用進行多次測試來觀察Learning Rate的影響，測試1000次並在每10次後Learning Rate遞減0.01

- 觀察結果
  - 隨著Learning Rate的降低，所需的Epoch數量有所提升，尤其是在小於0.1之後更為明顯，且其出來的結果也有發現會有所不同，相比於Learning Rate設定成1時，其結果判定的一致性低了很多

## TwoComponents V.S. ThreeComponents

- 在只使用2個components的時候，由於data2中的資料有許多是無法只靠前面二個components就可分辨出來的，因此他會跑到所設定的Epoch上限才會停止
- 其最終所分析出來的結果也都是錯誤的

## README

- Two-neuron perceptron, dataset 1
  - `py perceptronLearning.py 2N2C 0`
  - output 檔: `TwoNeuron_Data1.out`
- Two-neuron perceptron, dataset 2, two components
  - `py perceptronLearning.py 2N2C`
  - output 檔: `TwoNeuron.out`
- Two-neuron perceptron, dataset 2, three components
  - `py perceptronLearning.py 2N 1000`
  - output 檔: `TwoNeuron_twoComponents.out`
  - 1000 可更改成想讓其執行的次數
- Four-neuron perceptron, dataset 1
  - `py perceptronLearning.py 4N2C 0`
  - output 檔: `FourNeuron_Data1.out`
- Four-neuron perceptron, dataset 2, two components
  - `py perceptronLearning.py 2N2C`
  - output 檔: `FourNeuron.out`
- Four-neuron perceptron, dataset 2, three components
  - `py perceptronLearning.py 2N2C 1000`
  - output 檔: `FourNeuron_twoComponents.out`
  - 1000 可更改成想讓其執行的次數
- 或是可直接執行 `run.bat` 可一次將上面的盡數跑完