類神經網路基礎

王豐緒

銘傳大學資工系

學習目標

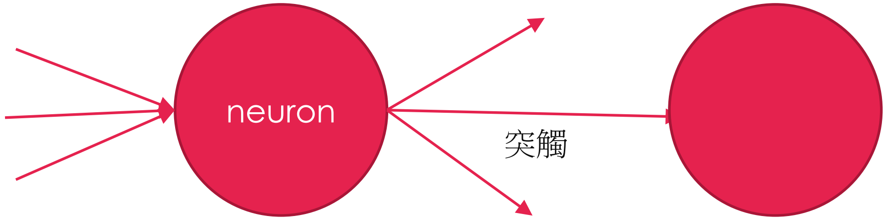
- 理解類神經元的基本結構與運作方式
- 理解何謂Perceptron類神經網路
- 理解類神經的學習方式
- 理解類神經的訓練與測試過程
- 理解矩陣運算與類神經的關聯

大綱

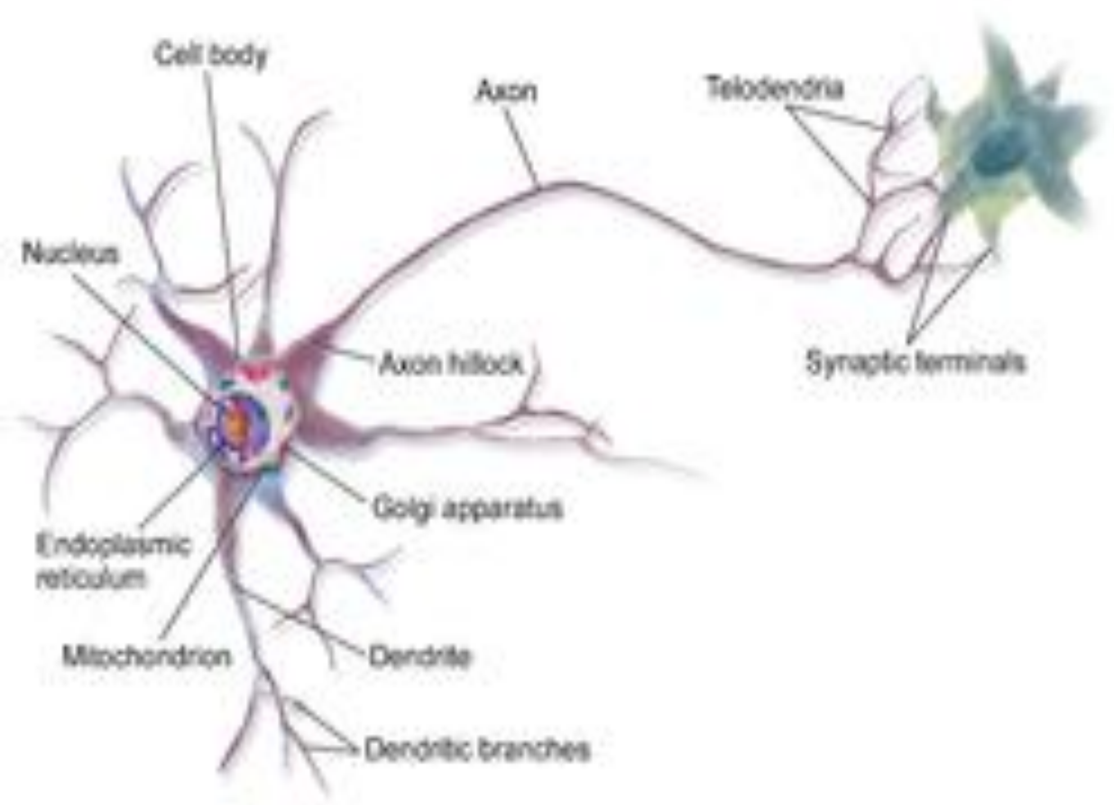
- 類神經元的結構與運作
- Perceptron類神經網路
- 學習方程式
- 學習速率的選擇
- 訓練與測試
  - 訓練階段(backward process)
  - 測試階段(Forward Process)
- 矩陣運算與類神經
- 小結

大腦神經元(NEURONS)

- 神經元
  - 大腦中的處理單元（1000億個，10^{11}）
  - 每個通過突觸與其他神經元相連（1千兆個，10^{14}）
- 具有可塑性和堅韌的操作性
  - 經由學習，修改突觸強度
  - 經由學習，建立新連接 (突觸，Synapse)



決定是否觸發



(source: Wiki)

類神經元(ARTIFICIAL NEURONS)

- McCulloch and Pitts Neurons

$$h = \sum_i W_i X_i - \theta$$

$$o = g(h) = \begin{cases} 1 & if\ h > 0 \\ 0 & if\ h \leq 0 \end{cases}$$

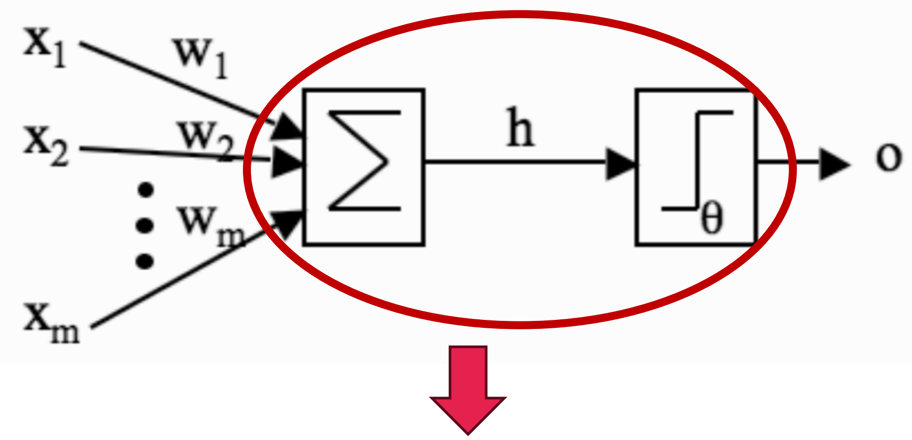
g : 激活函數 (activation function)

X: the input vector: [x_1, x_2, …, x_m, **-1**]

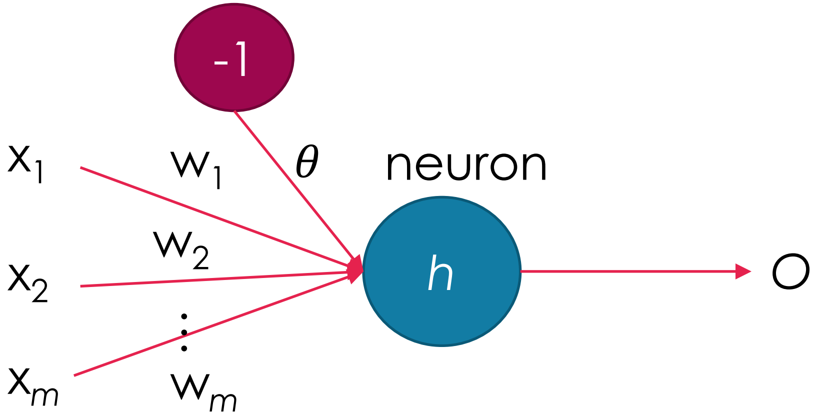
W: the weight vector: [w_1, w_2, …, w_m, θ]

$$o = g(W \odot X)$$

$\odot$: the inner product of W，X

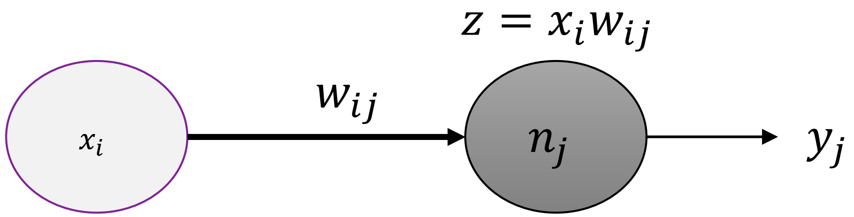


偏值(bias) 在所有输入都是零的情况下需要



PERCEPTRON類神經網路

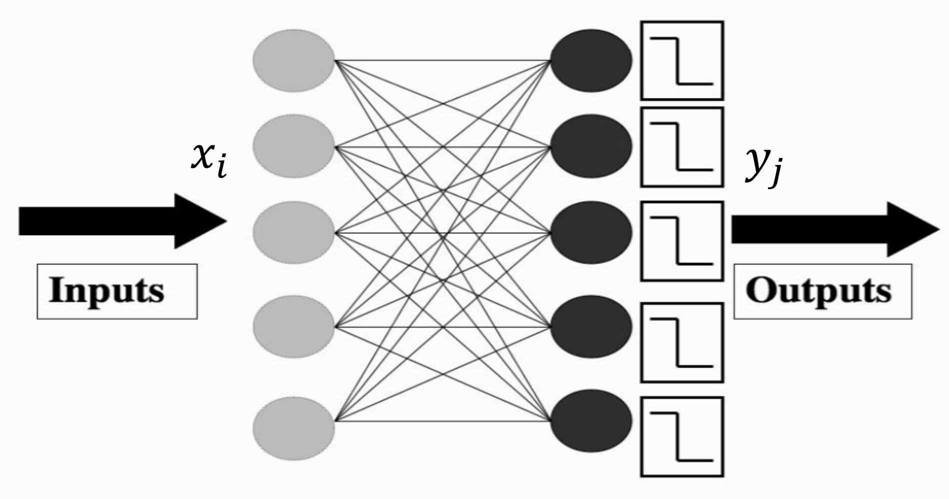
- 最早的多神經元網路
  - 啟發式學習法則



x_i: 網路輸入
y_j: 網路輸出
t_j : 真正的輸出
η : 學習速率 (learning rate)

學習方程式

$\triangle w_{ij} = -\eta(y_j - t_j) \cdot x_i$ ← WHY?



$y_j - t_j$	目標	$z = x_i w_{ij}$
高估($y_j > t_j$)	降低訊號強度z	- ←
低估($y_j < t_j$)	增強訊號強度z	→ +

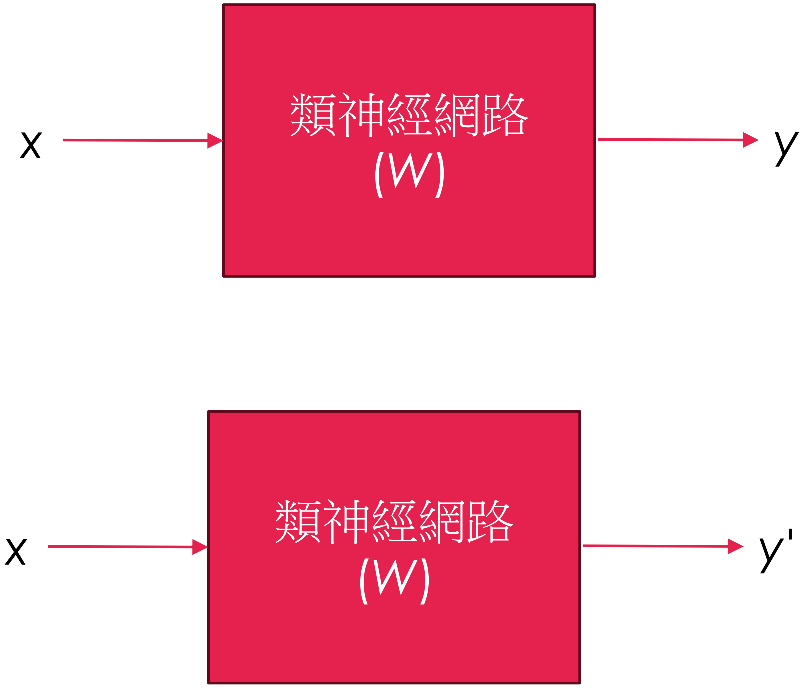
學習速率的選擇

- 較小的 learning rate
  - 學習較慢
  - 較能容忍資料雜訊和資料不一致性
- 較大的 learning rate
  - 較不穩定
- 實務上
  - 可嘗試 0.1 < η <0.4

$$\triangle w_{ij} = -\eta(y_j - t_j) \cdot x_i$$

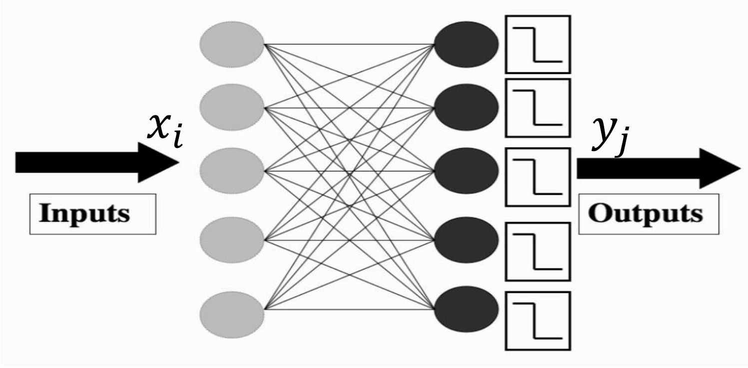
訓練(TRAIN)與測試(TEST)

- 訓練階段
  - 將訓練數據x輸入到類神經網路中並更新權重直到輸出正確答案y
- 測試階段
  - 將測試數據x輸入到類神經網路中並取得網路輸出y'



訓練階段(BACKWARD PROCESS)

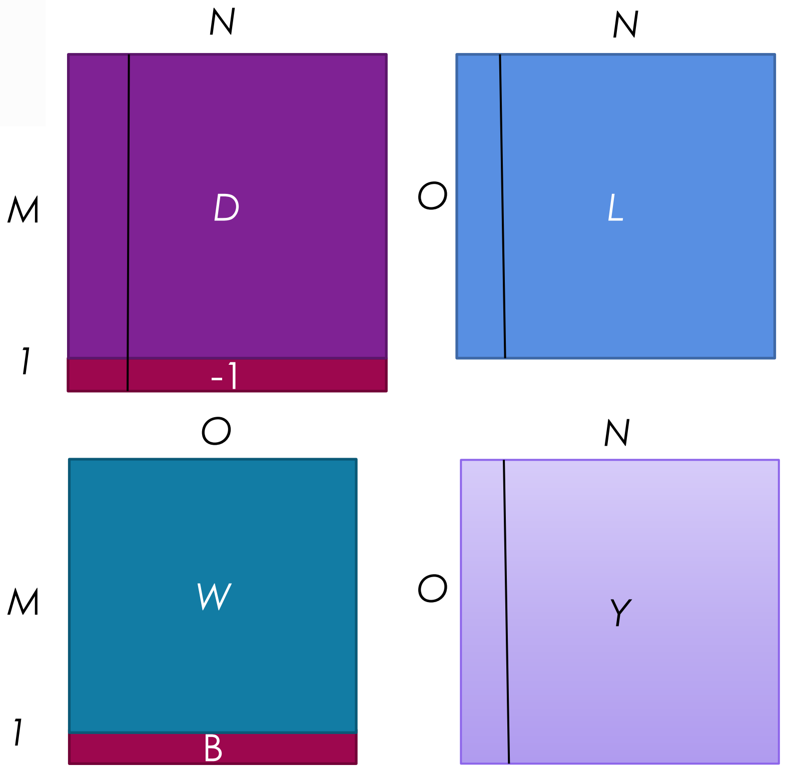
- 輸入訓練資料：$D_{(M+1)\times N}$
  - N筆 (M+1)-input 向量
    - 加入偏值向量(-1)
- 輸出訓練資料：$L_{O\times N}$
- 權重向量(Weight Matrix)
  - $W_{(M+1)\times O}$
  - 加入偏值權重(B)(bias weight): $B_{1\times O}$
- 網路輸出Y：$Y_{O\times N}$
- 總權重修正量矩陣: $\Delta W_{O\times(M+1)}$



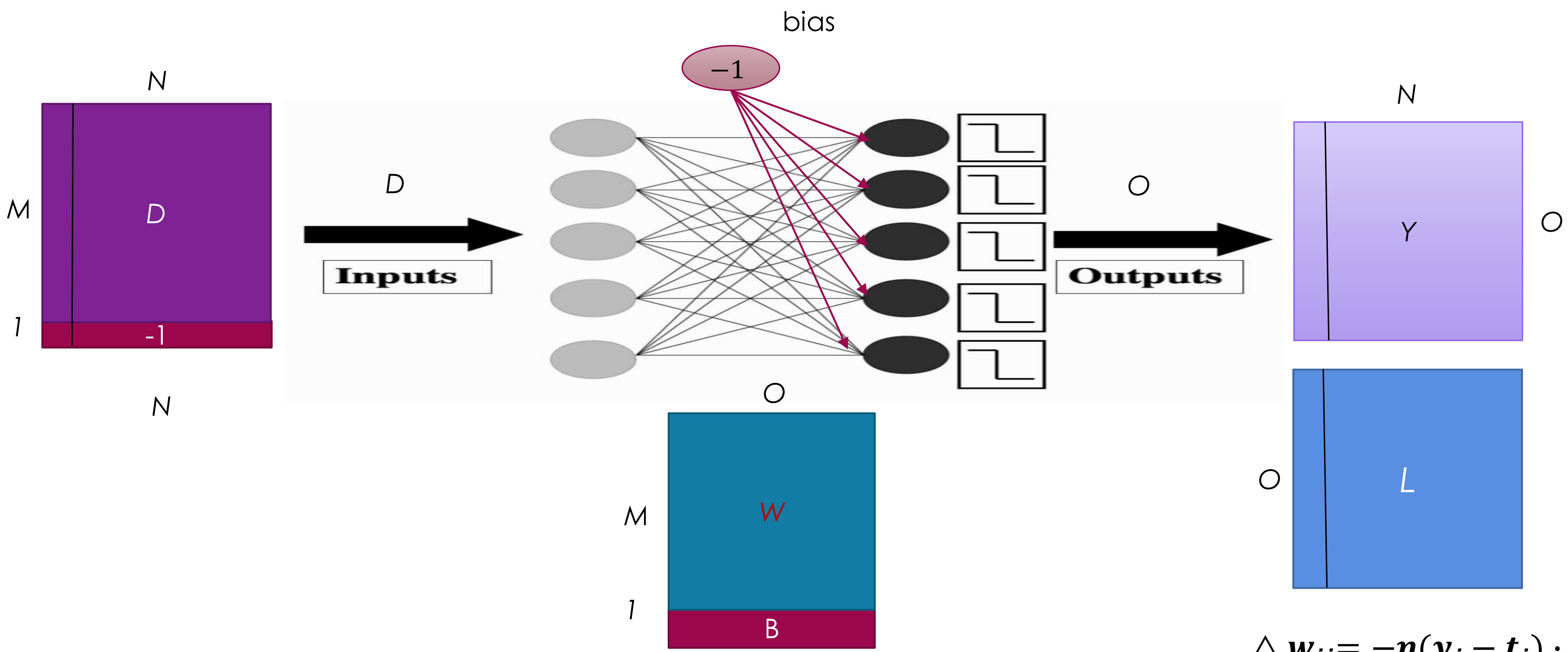
$$W^T_{O\times(M+1)}\times D_{(M+1)\times N}=H_{O\times N}$$

$$Y_{O\times N}=g(H_{O\times N})$$

$$\Delta W=-\eta D_{(M+1)\times N}\times(Y_{O\times N}-L_{O\times N})^T$$



訓練階段(BACKWARD PROCESS)



$$\triangle w_{ij} = -\eta(y_j - t_j) \cdot x_i$$

訓練階段(BACKWARD PROCESS)

$$D=\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} \qquad L=\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \qquad W=\begin{bmatrix} 0.4 & 0.8 \\ 0.1 & 0.3 \\ 0.5 & 0.2 \end{bmatrix}$$

$$H = W^T \times D = \begin{bmatrix} 0.4 & 0.1 & 0.5 \\ 0.8 & 0.3 & 0.2 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} -0.4 & -0.1 & 0 \\ 0.1 & 0.6 & 0.9 \end{bmatrix}$$

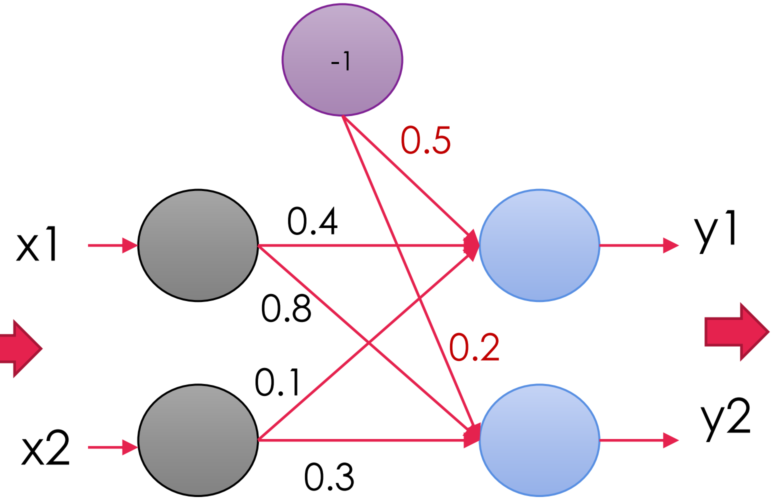
$$Y = g(H) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\Delta W = -\eta D_{(M+1)\times N} \times (Y_{O\times N} - L_{O\times N})^T$$

$$= -0.1 \cdot \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} \times \begin{bmatrix} -1 & 0 \\ 0 & 1 \\ -1 & 1 \end{bmatrix}$$

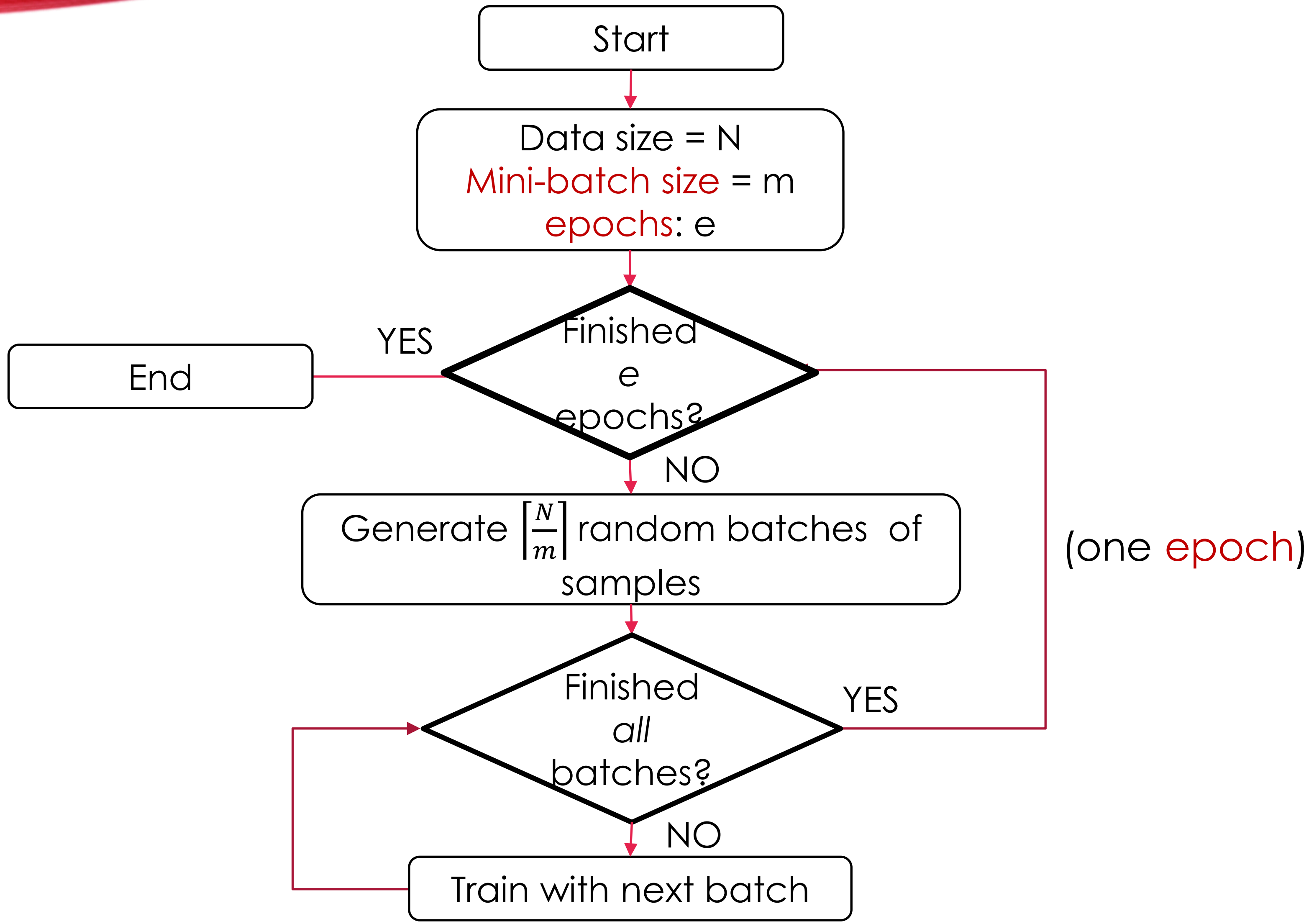
$$= -0.1 \cdot \begin{bmatrix} -1 & 2 \\ -2 & 1 \\ 2 & -2 \end{bmatrix} = \begin{bmatrix} 0.1 & -0.2 \\ 0.2 & -0.1 \\ -0.2 & 0.2 \end{bmatrix}$$

$$\begin{matrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{matrix}$$



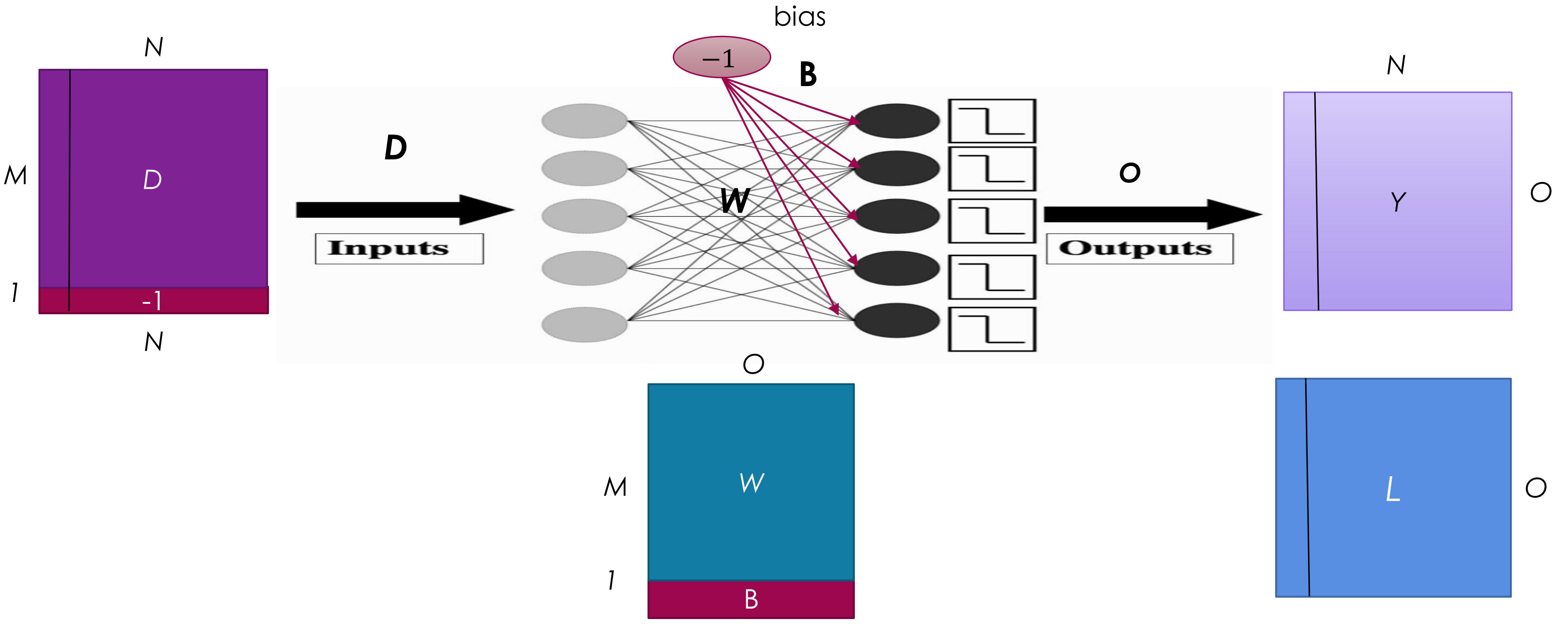
$$\begin{matrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$$

感知器的更新訓練演算法



測試階段(FORWARD PROCESS)

給定一組包含 *N* 個 *M* 輸入向量的輸入數據（*D*）和標籤數據（*L*），權重矩陣（*W*），計算網絡的輸出數據（*Y*）



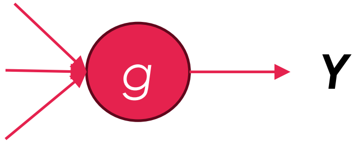
測試階段(FORWARD PROCESS)

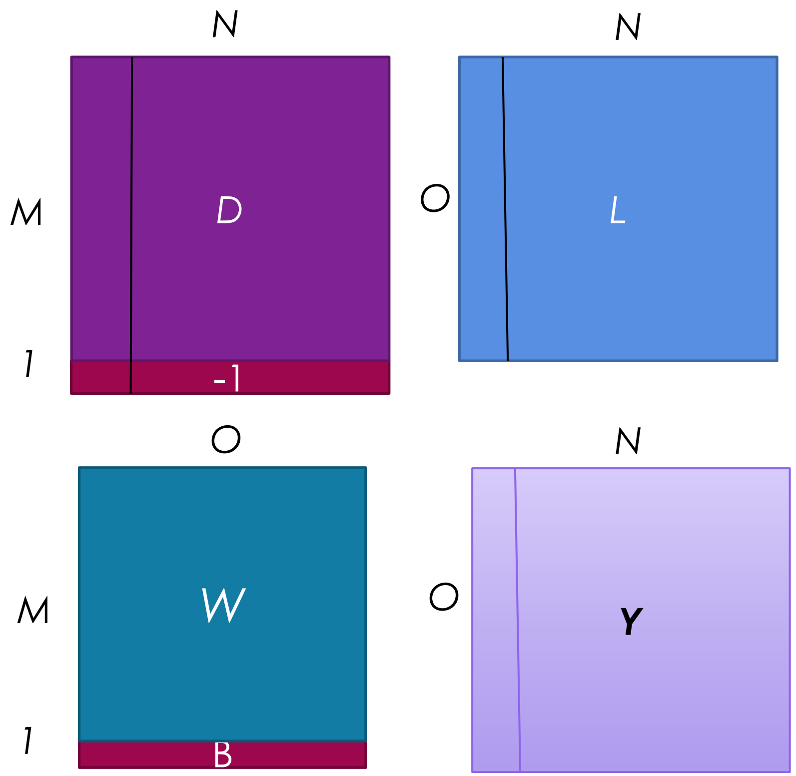
- 輸入測試資料：$D_{(M+1)\times N}$
  - *N*筆 (M+*1*)-input 向量
    - 加入偏值向量(-1)
- 輸出測試資料：$L_{O\times N}$
- 權重向量(Weight Matrix)
  - $W_{(M+1)\times O}$
  - 加入偏值權重(B)(bias weight): $B_{1\times O}$
- 網路輸出Y：$Y_{O\times N}$
- 方差矩陣(Total Squared Error Matrix): E_N

$$W^T_{O\times(M+1)} \times D_{(M+1)\times N} = Z_{O\times N}$$

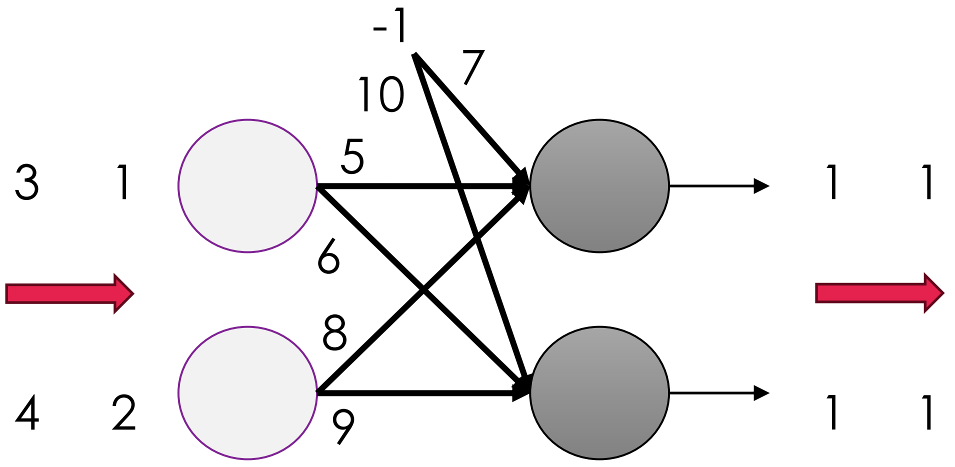
$$Y_{O\times N} = g(Z_{O\times N})$$

$$E_N = SUM^2_{col}(Y_{O\times N} - L_{O\times N})$$





測試階段(FORWARD PROCESS)



$$D = \begin{bmatrix} 1 & 3 \\ 2 & 4 \\ -1 & -1 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$$

$$W = \begin{bmatrix} 5 & 6 \\ 8 & 9 \\ 7 & 10 \end{bmatrix}$$

$$Z = W^T \times D = \begin{bmatrix} 5 & 8 & 7 \\ 6 & 9 & 10 \end{bmatrix} \times \begin{bmatrix} 1 & 3 \\ 2 & 4 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 14 & 32 \\ 14 & 44 \end{bmatrix}$$

$$Y = g(Z) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$E_N = SUM_{col}^2(Y - L) = SUM_{col}^2\left(\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}\right) = SUM_{col}^2\left(\begin{bmatrix} 0 & -2 \\ -1 & -3 \end{bmatrix}\right) = \begin{bmatrix} 1 & 13 \end{bmatrix}$$

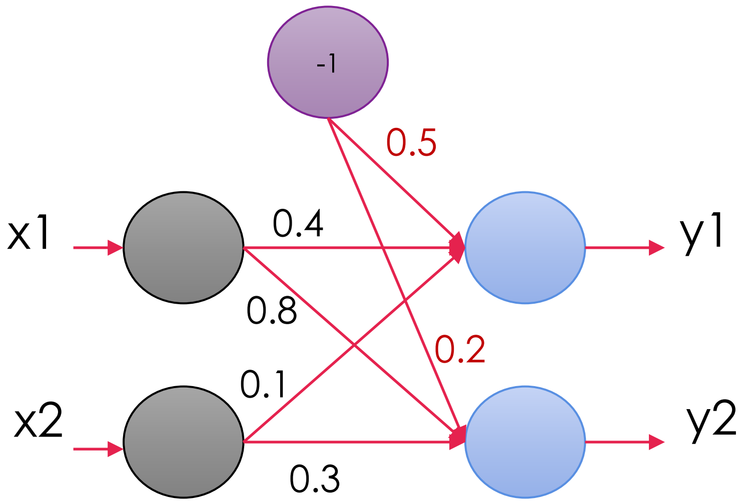
測試階段(FORWARD PROCESS)

$$D=\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} \qquad L=\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \qquad W=\begin{bmatrix} 0.4 & 0.8 \\ 0.1 & 0.3 \\ 0.5 & 0.2 \end{bmatrix}$$

$$Z = W^T \times D = \begin{bmatrix} 0.4 & 0.1 & 0.5 \\ 0.8 & 0.3 & 0.2 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ -1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} -0.4 & -0.1 & 0 \\ 0.1 & 0.6 & 0.9 \end{bmatrix}$$

$$Y = g(H) = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$E = SUM_{col}^{2}\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}\right) = SUM_{col}^{2}\left(\begin{bmatrix} -1 & 0 & -1 \\ 0 & 1 & 1 \end{bmatrix}\right) = [1\ \ 1\ \ 2]$$



矩陣運算與類神經

- 為什麼要以矩陣視角看待神經網路操作？
  - 延伸至對 TensorFlow 的觀點（何謂張量(Tensor)？）
  - 對於在晶片上（例如 GPU）進行高速計算很有用
- 在訓練階段，權重更新矩陣是由所有 N 筆訓練數據與原始網路權重計算收集而來
  - 如果 N 很大會發生什麼？
  - 如果選擇每個單一訓練數據來更新權重呢？

See how matrix multiplication can be reduced to $O(N)$ from $O(N^3)$ with parallel computation (https://www.sciencedirect.com/science/article/pii/0167819189900574)

小結

- 單一神經元的模型（McCulloch和Pitts 神經元）
  - 神經元的運作方式
- 感知器
  - 如何透過導出學習規則來訓練神經網路
- 測試感知器，看神經網絡如何進行訓練和操作
  - 初始化權重
  - 訓練
  - 測試
- 學習將神經網路操作視為矩陣操作
- 學習典型的訓練過程
  - 小批次大小
  - 執行周期的次數

參考文獻

- [1] Machine Learning: An Algorithmic Perspective, by Stephen Marsland, published by CRC Press (2014).

Q&A