**simulation 3: Hopfield nn**

10867024 廖育賢

# **objective - the problem and the purpose**

以HNN 識別阿拉伯數字0,2,4,6,8 ，並分析結果。

# **PRocrdure**

## method

### Storage phase :

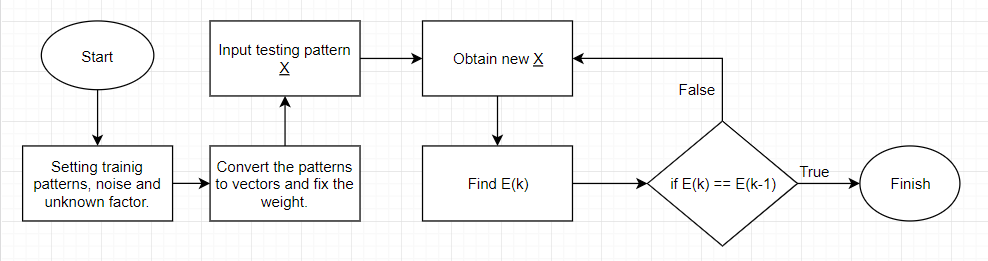
X1, X2, …, XM are known data

Xp  = [ X 1p, X 1p, …,X Np]T

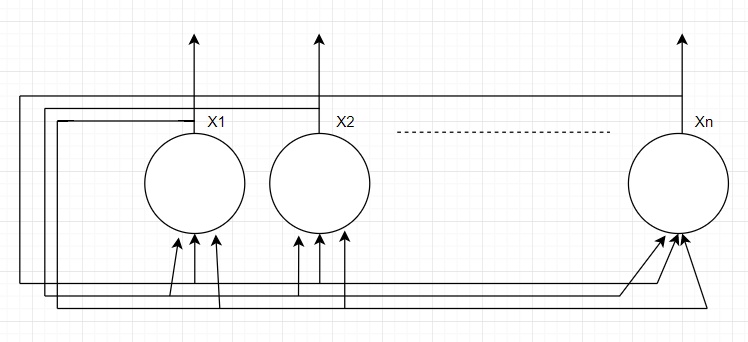
### Retrieval phase :

1. Input testing pattern X
2. Obtain new X
3. Find E
4. Repeat Step (2) until E converges

## program flow chart



## network structure



## calculation

# **simulation results**

## program codes

X0 = np.array([ [-1,-1,-1, 1, 1, 1, 1,-1,-1,-1],

[-1, 1, 1, 1, 1, 1, 1, 1, 1,-1],

[-1, 1, 1,-1,-1,-1,-1, 1, 1,-1],

[ 1, 1, 1,-1,-1,-1,-1, 1, 1, 1],

[ 1, 1, 1,-1,-1,-1,-1, 1, 1, 1],

[ 1, 1, 1,-1,-1,-1,-1, 1, 1, 1],

[ 1, 1, 1,-1,-1,-1,-1, 1, 1, 1],

[-1, 1, 1,-1,-1,-1,-1, 1, 1,-1],

[-1, 1, 1, 1, 1, 1, 1, 1, 1,-1],

[-1,-1,-1, 1, 1, 1, 1,-1,-1,-1]])

X2 = np.array([ [-1, 1, 1, 1, 1, 1, 1, 1, 1,-1],

[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

[-1,-1,-1,-1,-1,-1,-1,-1, 1, 1],

[-1,-1,-1,-1,-1,-1,-1,-1, 1, 1],

[-1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

[ 1, 1, 1, 1, 1, 1, 1, 1, 1,-1],

[ 1, 1,-1,-1,-1,-1,-1,-1,-1,-1],

[ 1, 1,-1,-1,-1,-1,-1,-1,-1,-1],

[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])

X4 = np.array([ [-1,-1,-1,-1, 1, 1,-1, 1, 1,-1],

[-1,-1,-1, 1, 1,-1,-1, 1, 1,-1],

[-1,-1, 1, 1,-1,-1,-1, 1, 1,-1],

[-1, 1, 1,-1,-1,-1,-1, 1, 1,-1],

[ 1, 1,-1,-1,-1,-1,-1, 1, 1,-1],

[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

[-1,-1,-1,-1,-1,-1,-1, 1, 1,-1],

[-1,-1,-1,-1,-1,-1,-1, 1, 1,-1],

[-1,-1,-1,-1,-1,-1,-1, 1, 1,-1]])

X6 = np.array([ [-1,-1, 1, 1, 1, 1, 1,-1,-1,-1],

[-1, 1, 1, 1, 1, 1, 1, 1,-1,-1],

[-1, 1, 1,-1,-1,-1,-1,-1,-1,-1],

[-1, 1, 1, 1, 1, 1, 1,-1,-1,-1],

[-1, 1, 1, 1, 1, 1, 1, 1,-1,-1],

[-1, 1, 1,-1,-1,-1, 1, 1,-1,-1],

[-1, 1, 1,-1,-1,-1, 1, 1,-1,-1],

[-1, 1, 1,-1,-1,-1, 1, 1,-1,-1],

[-1, 1, 1, 1, 1, 1, 1, 1,-1,-1],

[-1,-1, 1, 1, 1, 1, 1,-1,-1,-1]])

X8 = np.array([ [-1,-1,-1, 1, 1, 1, 1,-1,-1,-1],

[-1,-1, 1, 1, 1, 1, 1, 1,-1,-1],

[-1, 1, 1,-1,-1,-1,-1, 1, 1,-1],

[-1, 1, 1,-1,-1,-1,-1, 1, 1,-1],

[-1,-1, 1, 1, 1, 1, 1, 1,-1,-1],

[-1, 1, 1, 1, 1, 1, 1, 1, 1,-1],

[ 1, 1,-1,-1,-1,-1,-1,-1, 1, 1],

[ 1, 1,-1,-1,-1,-1,-1,-1, 1, 1],

[-1, 1, 1, 1, 1, 1, 1, 1, 1,-1],

[-1,-1, 1, 1, 1, 1, 1, 1,-1,-1]])

label = ['0','2','4','6','8']

ith = 4 # label[ith]

noise = 0.2

unknown = 0.1

#np.random.seed(150)

X\_test = add\_unknown(add\_noise(X8\_2,noise),unknown).reshape(-1,1)

# Storage Phase

# flatten the pattern

X0 = X0.reshape(1,X0.size)

X2 = X2.reshape(1,X2.size)

X4 = X4.reshape(1,X4.size)

X6 = X6.reshape(1,X6.size)

X8 = X8.reshape(1,X8.size)

X = np.array([X0,X2,X4,X6,X8])

W\_fixed = np.zeros([X0.shape[1],X0.shape[1]])

for p in range(X.shape[0]):

W\_fixed = W\_fixed + np.dot(X[p,:,:].reshape(-1,1),X[p,:,:].reshape(1,-1))

for i in range(X.shape[2]):

for j in range(X.shape[2]):

if i==j:

W\_fixed[i,j]=0

# W is fixed now

# Let's retrieve the pattern X with some noise

E = energy(X\_test,W\_fixed)

E\_cycle = [E]

print('-------------------------------------')

print('E(0) : ',E)

print('X(0) : ',X\_test[:,0])

X\_new = np.zeros([10,10])

X\_new = X\_test

# obtain new X

for cycle in range(100):

X\_new = f(W\_fixed,X\_new)

E = energy(X\_new,W\_fixed)

E\_cycle.append(E[0,0])

print('-------------------------------------')

print('E({}) : {}'.format(cycle+1,E[0,0]))

print('X({}) : {}'.format(cycle+1,X\_new[:,0]))

if E\_cycle[cycle+1] == E\_cycle[cycle] :

print('-------------------------------------')

print('cycle :',cycle+1)

print('E converges : {}'.format(E[0,0]))

print('X final : {}'.format(X\_new[:,0]))

break

## graphs

### add noise:

|  |
| --- |
|  |

FIGURE 1 - pattern zero with noise

|  |
| --- |
|  |

FIGURE 2 - pattern two with noise

|  |
| --- |
|  |

FIGURE 3 - pattern four with noise

|  |
| --- |
|  |

FIGURE 4 - *pattern six with noise*

|  |
| --- |
|  |

FIGURE 5 - pattern eight with noise

### add UNKNOWN:

|  |
| --- |
|  |

FIGURE 6 - pattern zero with unknown

|  |
| --- |
|  |

FIGURE 7 - pattern two with unknown

|  |
| --- |
|  |

FIGURE 8 - pattern four with unknown

|  |
| --- |
|  |

FIGURE 9 - pattern six with unknown

|  |
| --- |
|  |

FIGURE 10 - pattern eight with unknown

### add noise and unknown:

|  |
| --- |
|  |

FIGURE 11 - pattern zero with noise and unknown

|  |
| --- |
|  |

FIGURE 12 - pattern two with noise and unknown

|  |
| --- |
|  |

FIGURE 13 - pattern four with noise and unknown

|  |
| --- |
|  |

FIGURE 14 - pattern six with noise and unknown

|  |
| --- |
|  |

FIGURE 15 - pattern eight with noise and unknown

## tables

### pattern and energy:

|  |  |
| --- | --- |
| Pattern | E |
| 0 | -7456 |
| 2 | -5944 |
| 4 | -5400 |
| 6 | -6648 |
| 8 | -7998 |

TABLE 1

### pattern and noise:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pattern | zero | two | four | six | eight |
| noise | 10 | 10 | 10 | 10 | 10 |
| noise | 30 | 30 | 30 | 40 | 40 |
| noise | 50 | 50 | 80 | 70 | 60 |
| noise | 70 | 70 | 100 | 90 | 90 |
| Energy | -7456 | -5944 or -5992 | -5400 | -6648 | -7944 or -7998 or -7928 |

TABLE 2

### PATTERN AND UNKNOWN:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pattern | zero | two | four | six | eight |
| unknown | 10 | 10 | 10 | 10 | 10 |
| unknown | 30 | 40 | 30 | 30 | 50 |
| unknown | 50 | 60 | 50 | 40 | 70 |
| unknown | X | X | X | X | 90 |
| Energy | -7456 | -5944 | -5400 | -6648 | -7998 or -7928 |

TABLE 3

### PATTERN AND NOISE+UNKNOWN:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pattern | zero | two | four | six | eight |
| (noise,unknown) | 10,10 | 10,20 | 20,20 | 20,20 | 20,20 |
| (noise,unknown) | 10,30 | 30,30 | 40,30 | 30,30 | 80,50 |
| (noise,unknown) | 10,50 | 50,30 | 50,30 | 60,60 | 99,90 |
| (noise,unknown) | X | 80,40 | 60,30 | 80,60 | X |
| Energy | -7456 or -7992 | -5944 or -5992 | -5400 | -6648 | -7998 or -7992 |

TABLE 4

# **conclusion - analysis and comparison**

由FIGURE 1~15 可觀察到Hopfield 神經網路有很強抵抗干擾的能力，即便欲辨識之圖片有50%以上的雜訊或未知，仍有很高的機會正確辨識數字，由TABLE 4 ，數字 ”8” 在noise 99% 與 unknown 90% 之情況下，仍能將之正確辨識，但由於條件嚴苛，相較於其他情況來說正確率較低。

我們可以觀察到，成功辨識數字的能量都會群聚在一起，故我們將欲辨識圖片能量與正確結果之能量 (由TABLE 1) 比對後，便能得知此數字最有可能為何。