類神經網路 Neural Networks

作業三

學號 112522101 資工碩一

姓名 吳宥俞

目錄 Outline

目錄 Outline 2
1. 基本題+加分題(實作Hopfield) 3
1.1 GUI 功能及程式流程 3
1.2 主要 function 說明 4
1.3 實驗結果及分析 7

1. 基本題+加分題(實作Hopfield)

1.1 GUI 功能及程式流程

GUI 部分採用 Tkinter 套件呈現，並使用 PAGE 圖型編輯器完成 UI 外觀。

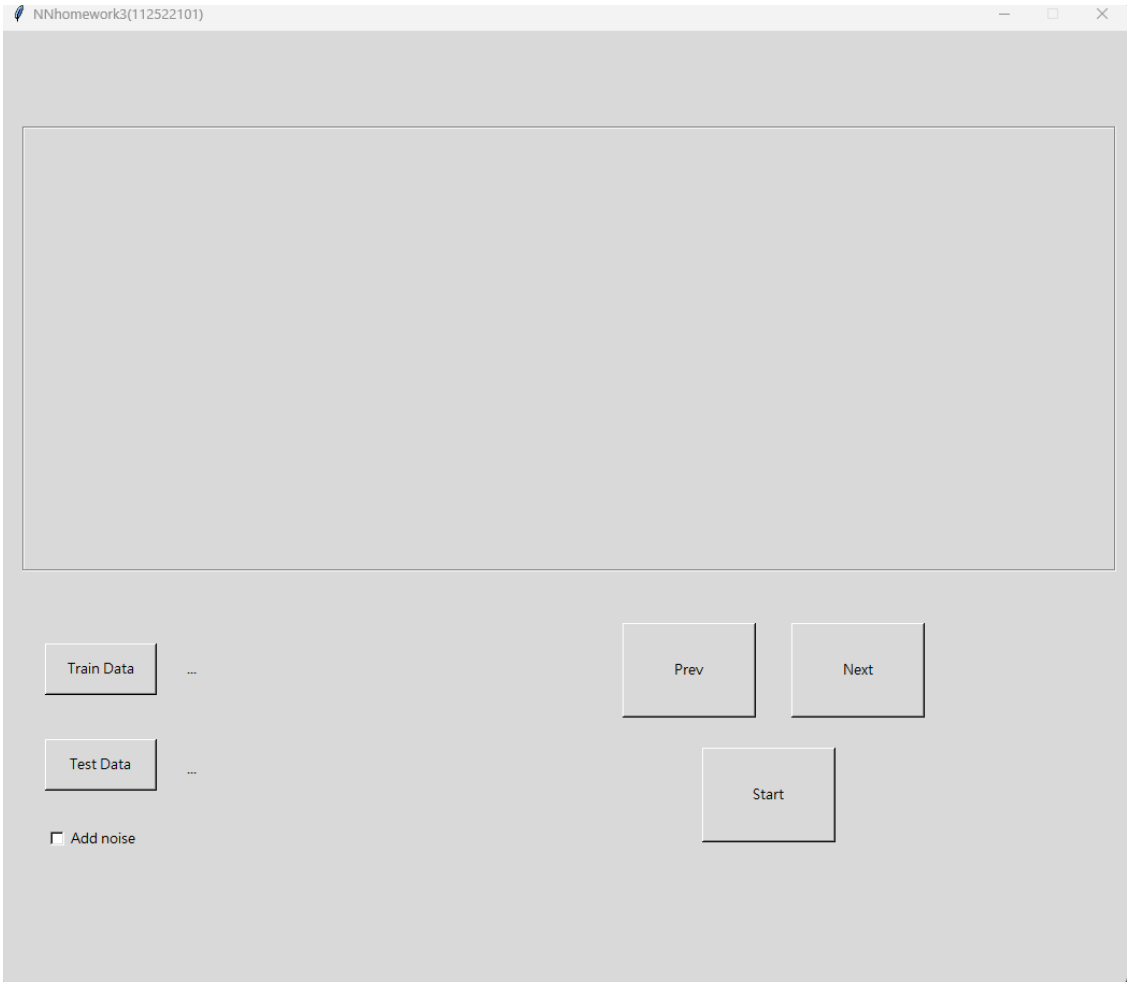


圖 1 初始外觀

使用方法:

i. 點擊 Train Data 按鈕，獲得訓練資料集路徑。

ii. 點擊 Test Data 按鈕，獲得測試資料集路徑。

iii. 按下 Start按鈕，開始訓練，若要加雜訊，選取Add noise後再點擊一次Start按鈕即可。

iv. 可以點擊Prev或是Next按鈕，來顯示前一張或下一張的結果。

核心程式流程:

i. 當按下 Start按鈕時，會去取得輸入的路徑、是否要加入雜訊值，並將參數傳進 main function 計算鍵結值以及先顯示第一張圖片的結果。

ii. 在點擊Prev或是Next按鈕時，**才會將該圖片的資訊進行回想(迭代100次)**，並將回想結果顯示出來。

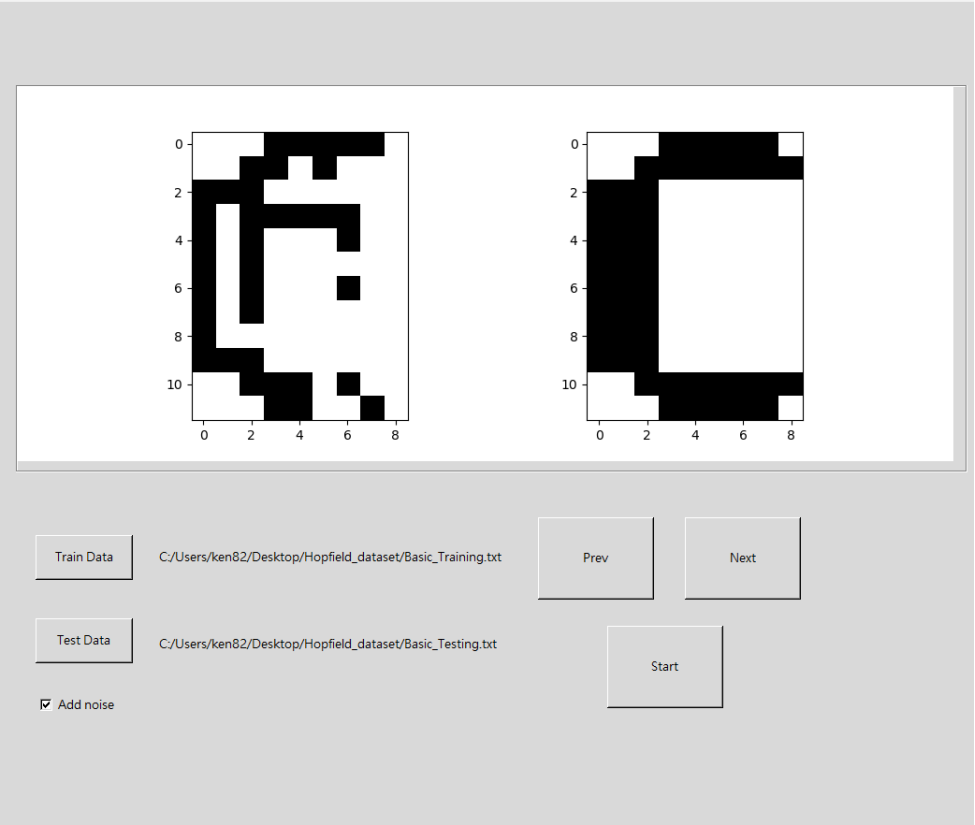


圖 2 流程完成圖(**註:左方是原圖(或加雜訊)，右方是回想結果**)

1.2 主要 function 說明

作品架構主要有 3 個檔案，3 個檔案皆相互 import:

i. UIhw3.py:UI 外觀程式碼(大部分都是 PAGE 自動生成)。

ii. UIhw3_support.py:事件反應程式碼(如按鈕點擊後會做甚麼)

iii. Hopfield.py:類神經網路程式碼(算W、回想迭代、輸出圖片)

UIhw3.py、UIhw3_support.py 重點程式碼:

```
def startbtn1(*args):
    if _debug:
        #獲取路徑、是否加入雜訊
        trainPath = str(_w1.l1.get())
        testPath = str(_w1.l2.get())
        addNoise = int(_w1.c1.get()) # 0或1
        #呼叫程式
```

(ii.)中的 Start按鈕，觸發後將參數傳入(iii.)

Hopfield.py 重點程式碼:

```
import numpy as np
import matplotlib.pyplot as plt
import random
```

使用到numpy、matplotlib、random 套件

```
def txt2ndarray(path):
    fp = open(path, 'r', encoding='utf-8')
    string = fp.read()
    fp.close()
    row_list = string.splitlines()
    #return .txt data to ndarray data
    data_list = list()
    data = list()
    for row in row_list:
        for word in row:
            if(word==" "):
                data_list.append(1)
            else:
                data_list.append(-1)
        if row == "":
            data.append(data_list)
```

txt2ndarray():將輸入的資料分好成每筆資料，並將值賦為1跟-1，以利後續處理

```
def train_hopfield_network(patterns):
    num_patterns, pattern_size = patterns.shape
    weights = np.zeros((pattern_size, pattern_size)) #初始化W
    for i in range(num_patterns):
        pattern = patterns[i, :]
        # weights += np.outer(pattern, pattern) #外積(或是內積轉置後的自己)
        weights += np.dot(pattern, pattern.T) #外積(或是內積轉置後的自己)
    np.fill_diagonal(weights, 0) #對角為0
    weights /= pattern_size #(1/P)*W

    return weights
```

Hopfield計算鍵結值方法:(因為對角線算完一定是0，我就直接設為0了)

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1p} \\ \vdots & \ddots & \vdots \\ w_{p1} & \cdots & w_{pp} \end{bmatrix} = \frac{1}{p}\sum_{k=1}^{N} \underline{x}_k \underline{x}_k^T - \frac{N}{p} I$$

```python
def update_hopfield_network(input_pattern, weights, max_iter=100):
    pattern_size = len(input_pattern)
    for _ in range(max_iter): #非同步聯想
        for i in range(pattern_size):
            input_pattern[i] = np.sign((np.dot(weights[i, :], input_pattern)))

    return input_pattern
```

回想過程，採用非同步回想(迭代100次)，**這邊theta=0，所以就不扣掉theta了**

```python
def addnoise(input_pattern): #加入雜訊(以0.25的機率將1變-1,將-1變1)
    for i in input_pattern:
        dice = random.randint(0,3)
        if dice == 0:
            input_pattern[i] *= -1

    return input_pattern
```

使用random套件來**加入雜訊**，(依老師上課的方法，以**0.25的機率將值互換(1變-1，-1變1)**)

```python
def plot_patterns(patterns): #將結果化在tkinter上
    num_patterns, pattern_size = patterns.shape
    if pattern_size == 100:
        pattern_col,pattern_row = 10,10
    else:
        pattern_col,pattern_row = 12,9
    fig1 = plt.figure(figsize=(10,4))
    plt.ion()
    ax1 = fig1.add_subplot(1,1,1)
    ax1.clear()
    for i in range(num_patterns):
        plt.subplot(1, num_patterns, i + 1)
        plt.imshow(patterns[i, :].reshape((pattern_col,pattern_row)), cmap='gray')
    canvas1 = FigureCanvasTkAgg(fig1, master=UIhw3_support._w1.Frame1)
    canvas1.get_tk_widget().place(x=0,y=0)
    canvas1.flush_events() #畫面刷新
    UIhw3_support.root.update_idletasks()
    plt.ioff()
```

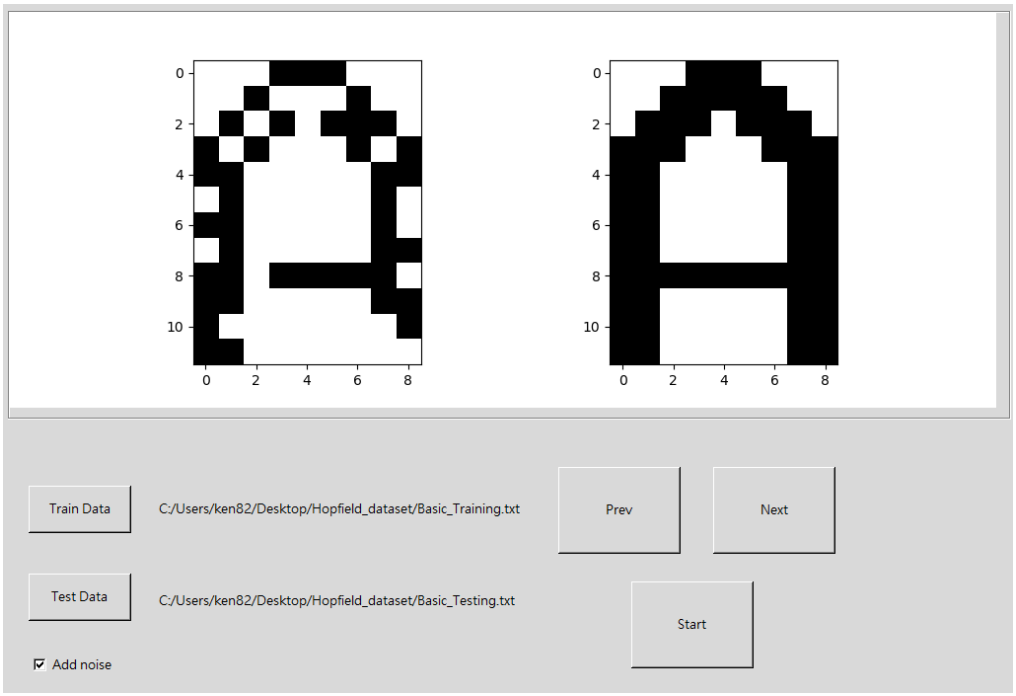
將圖顯示到Tkinter的Frame畫布上

1.3 實驗結果及分析

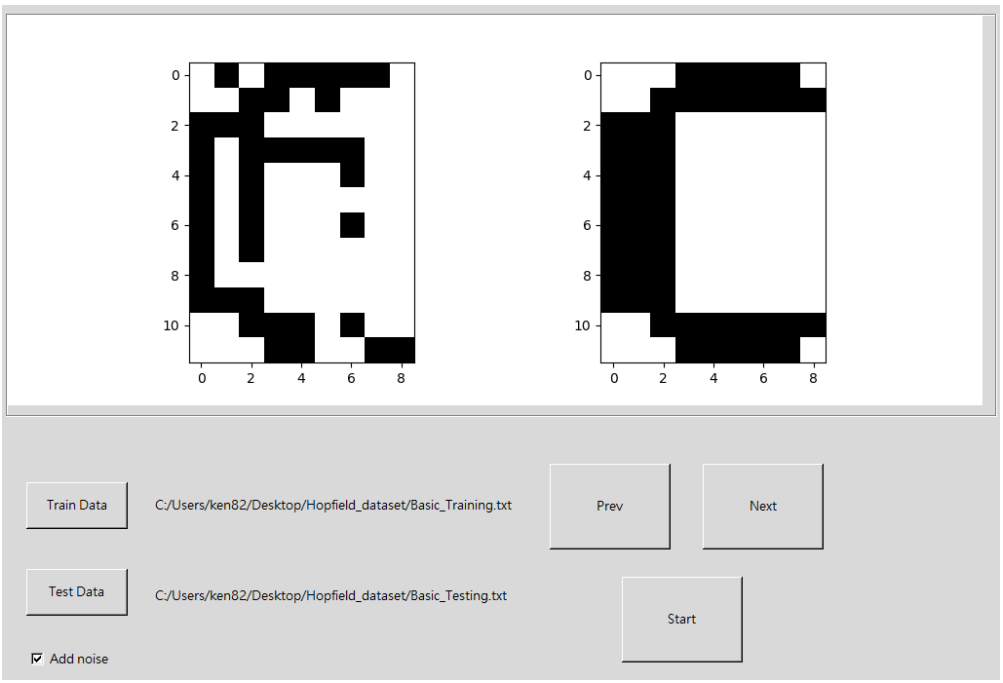
基本題的3筆資料(A、C、L)不管有沒有加入雜訊都能正確回想。

輸出的結果將針對每一筆資料進行分析:

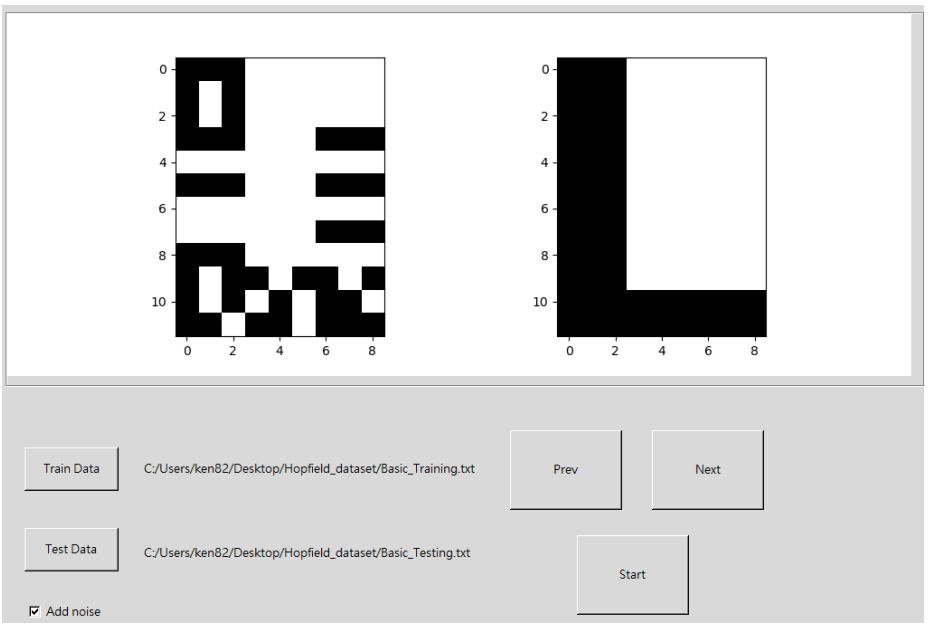
Basic_Testing.txt(皆回想成功)



A

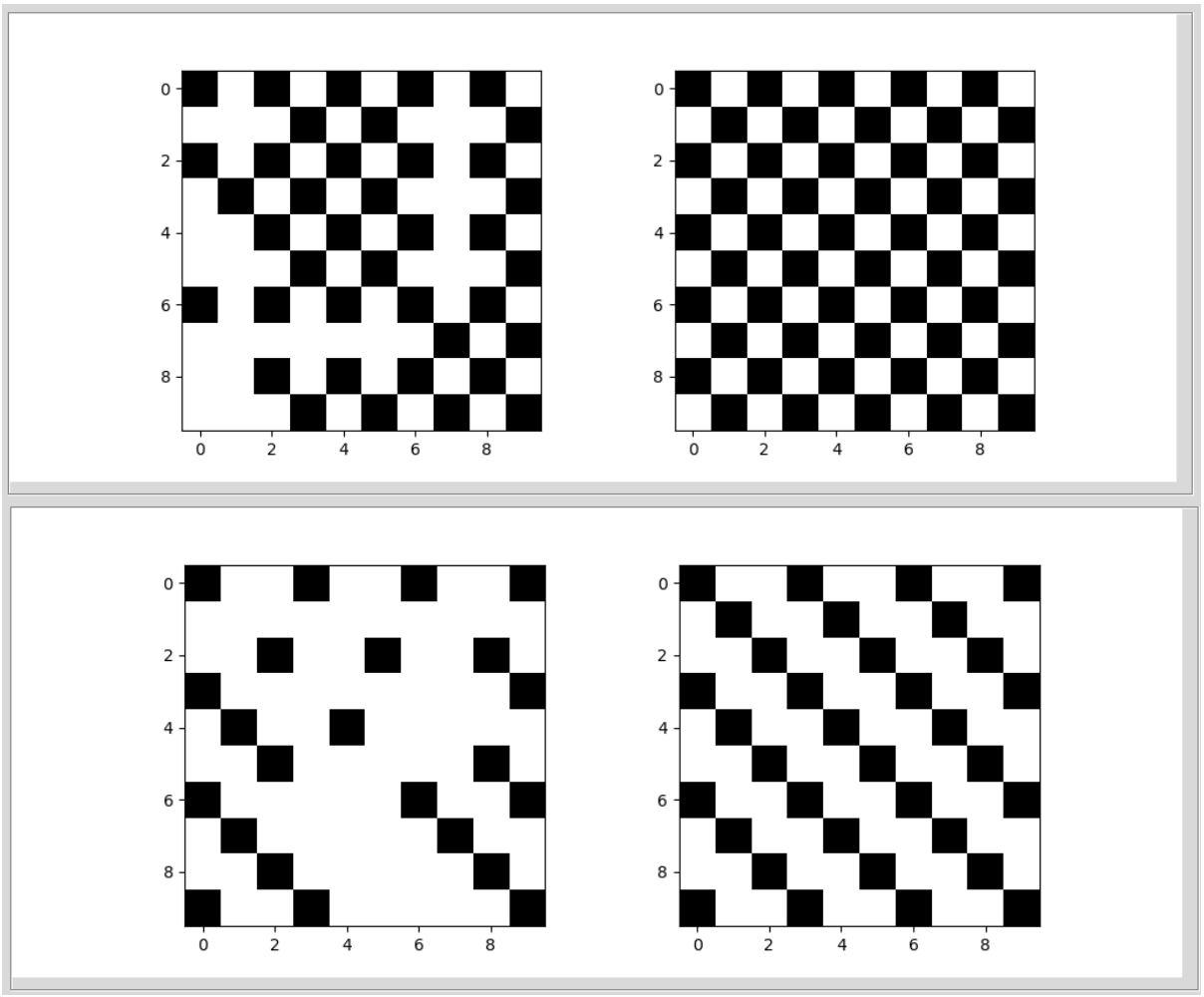


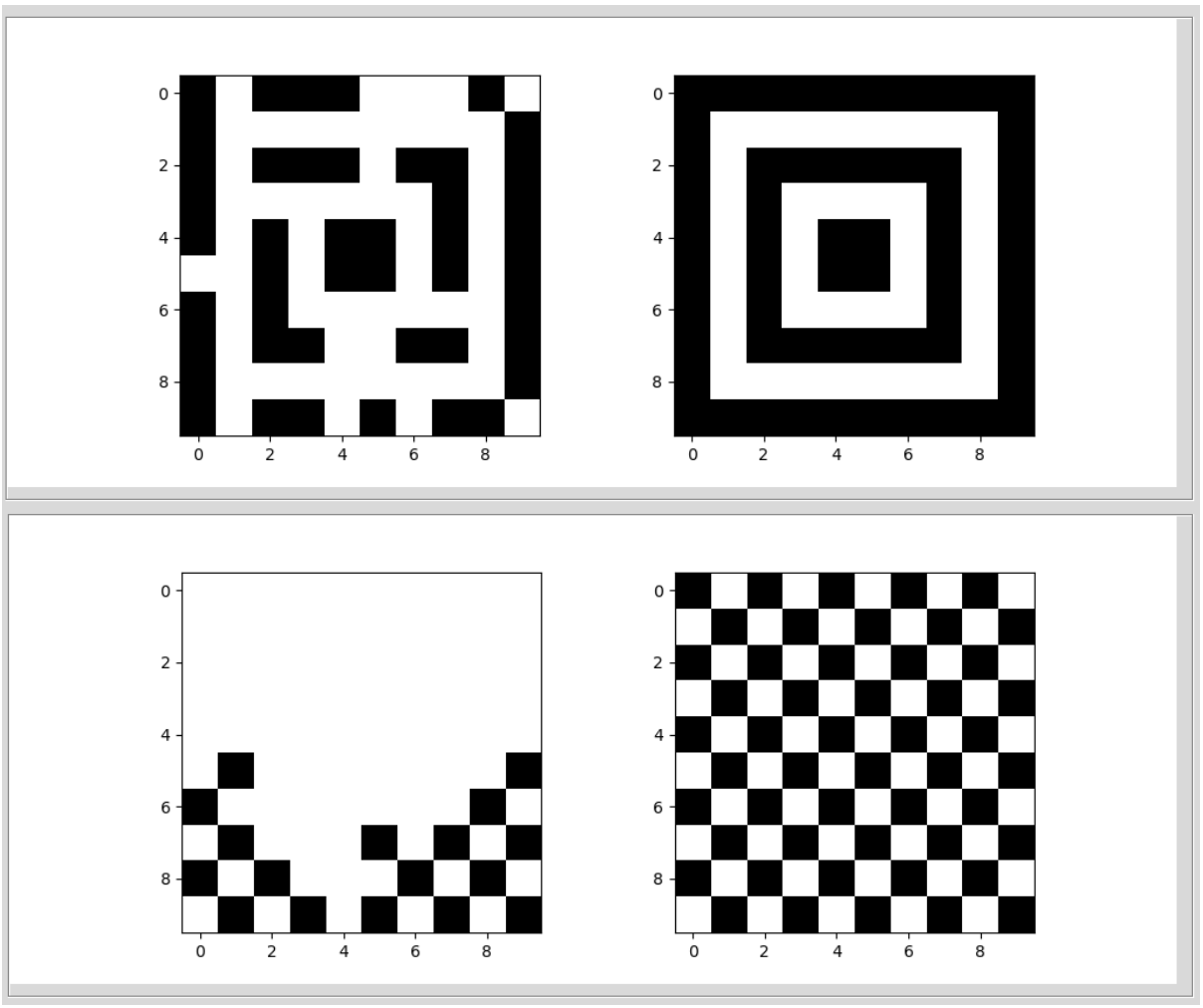
C



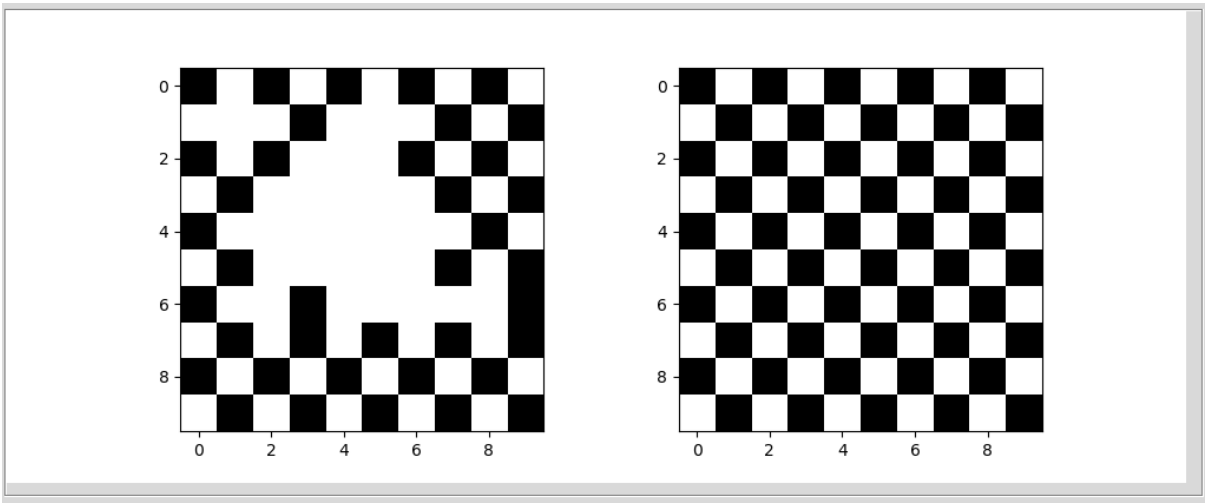
L

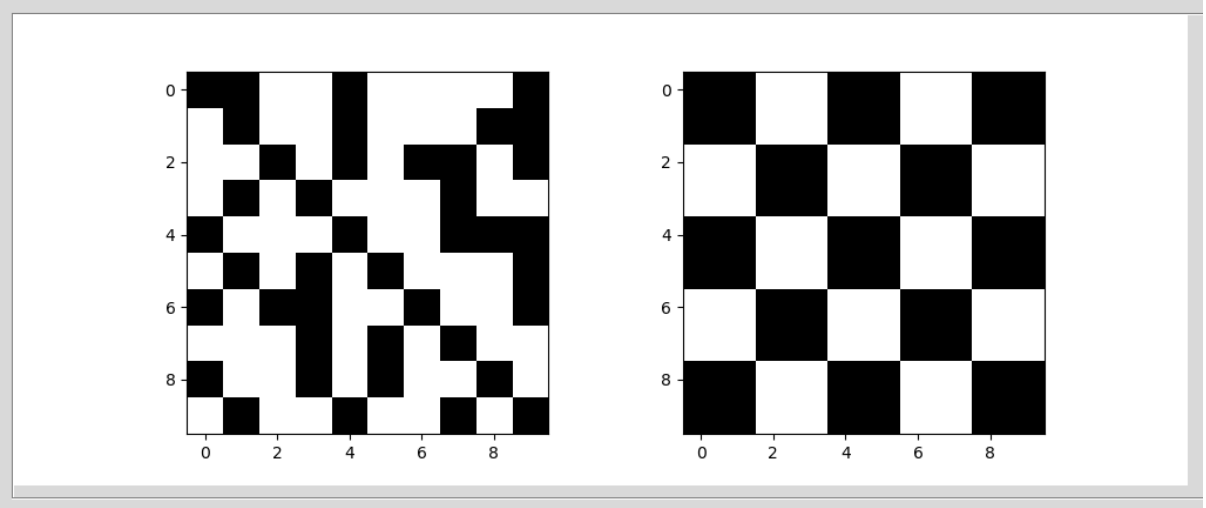
Bonus_Testing.txt(回想成功數:10/15)



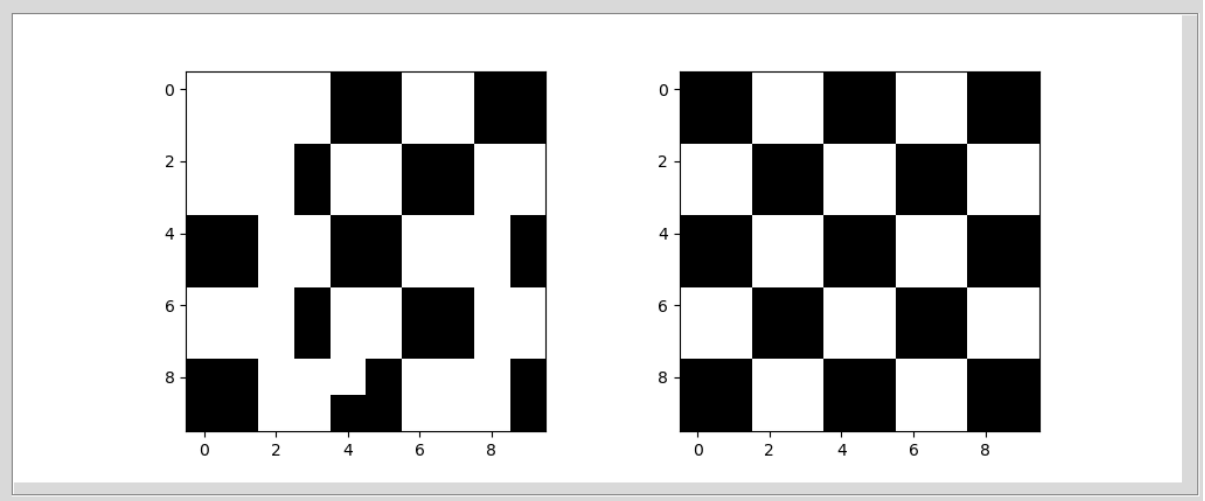


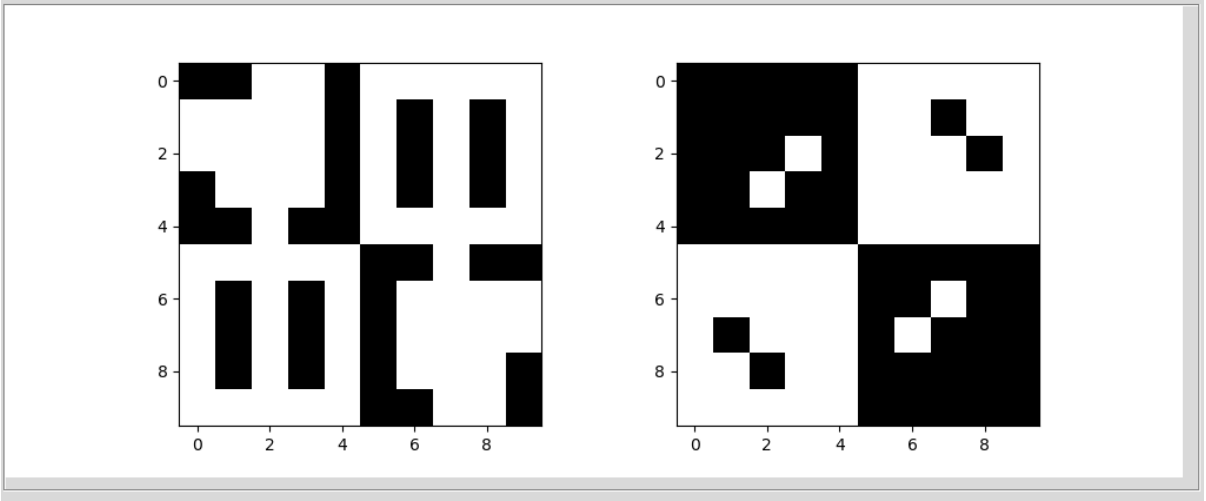
（上圖回想錯誤）



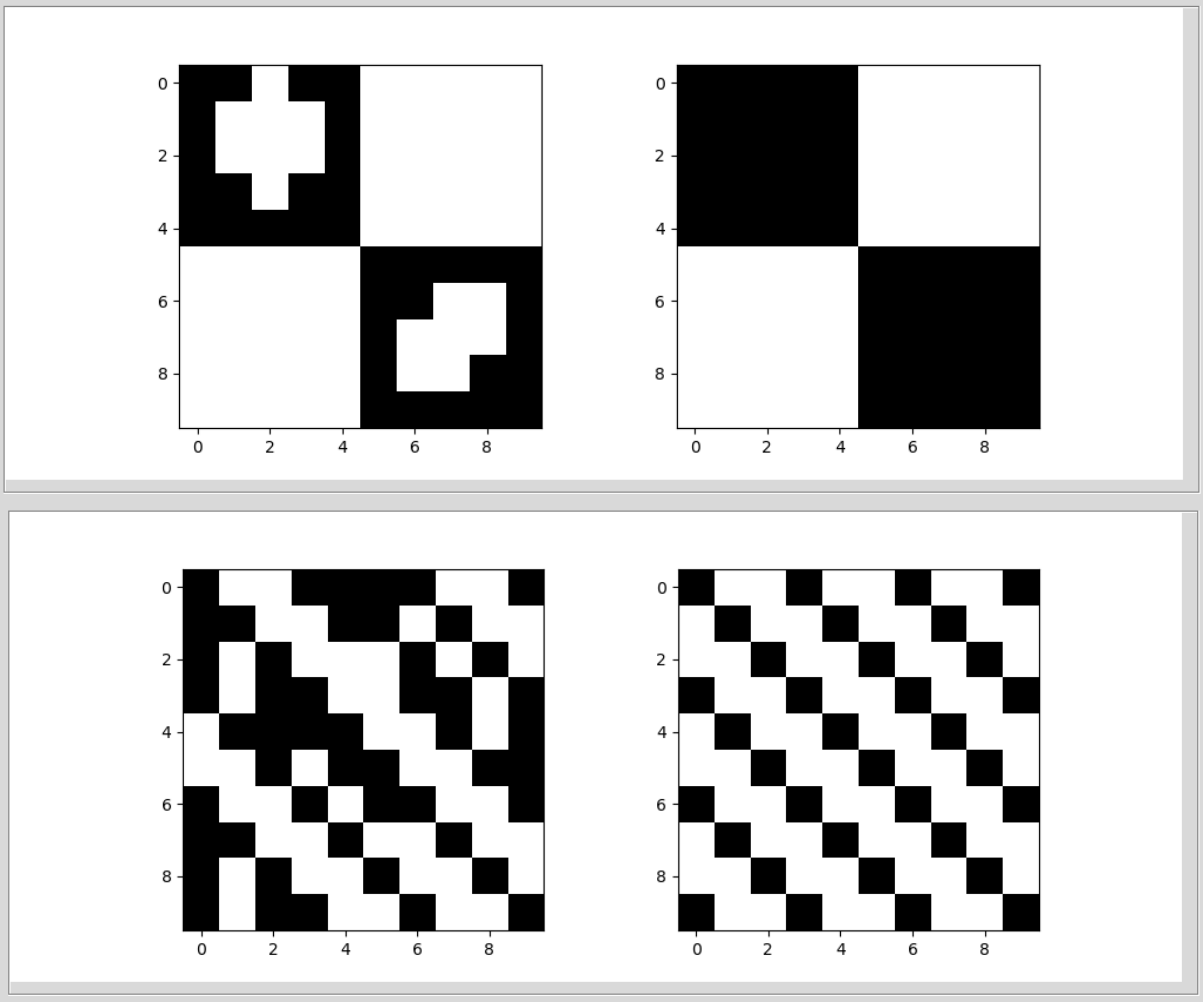


（上圖回想錯誤）

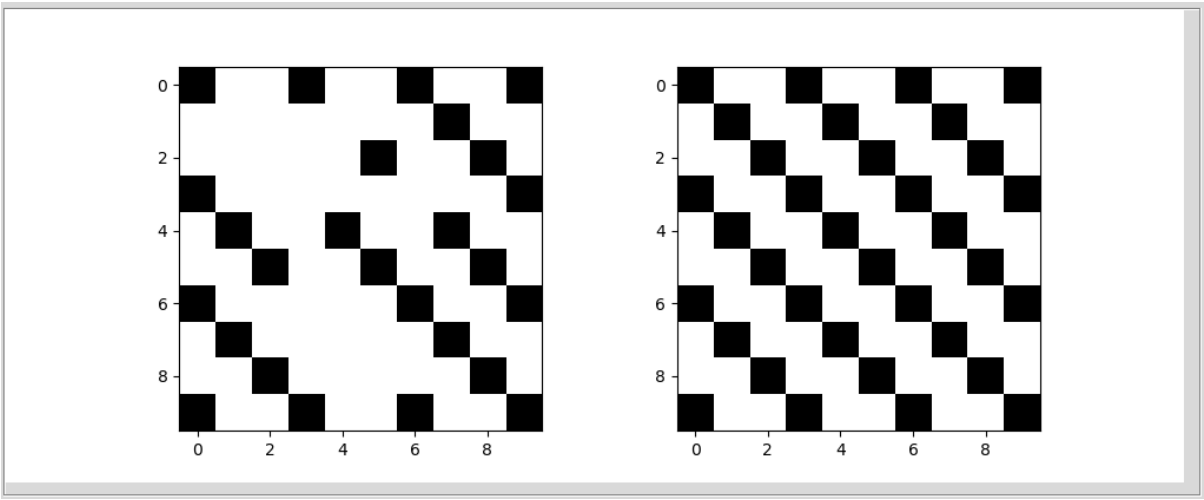


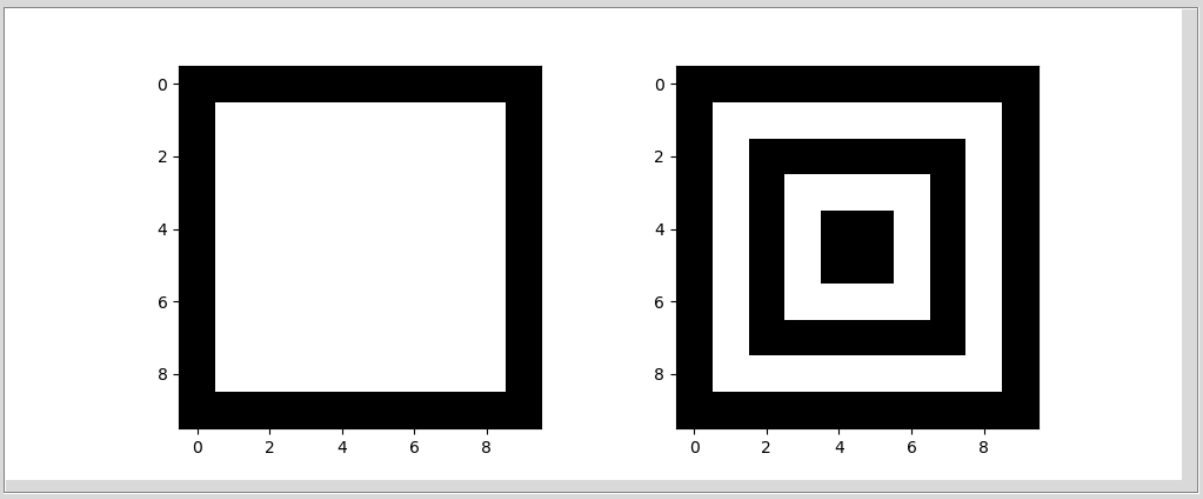


（上圖回想錯誤）

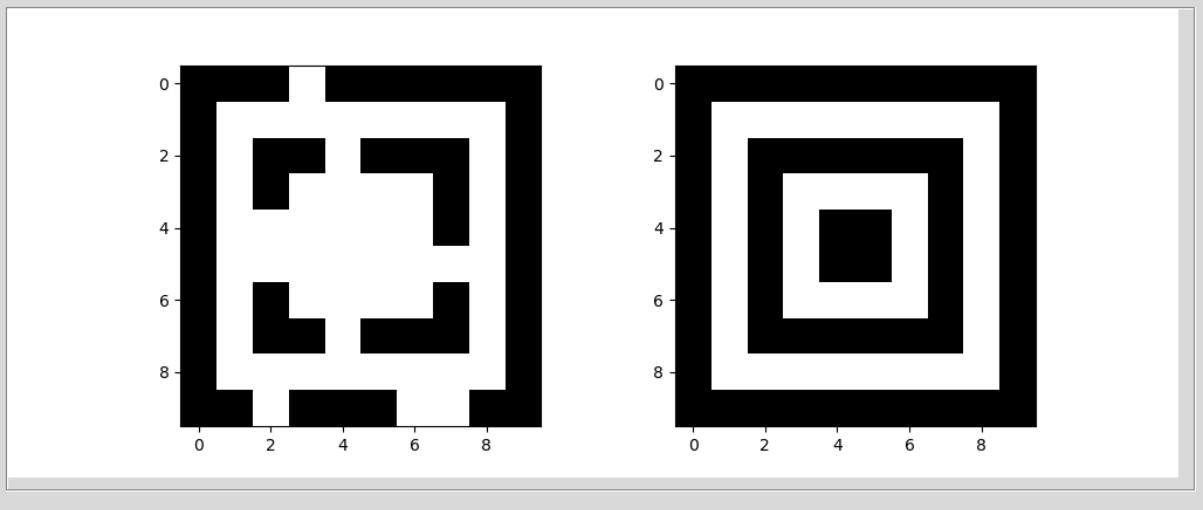


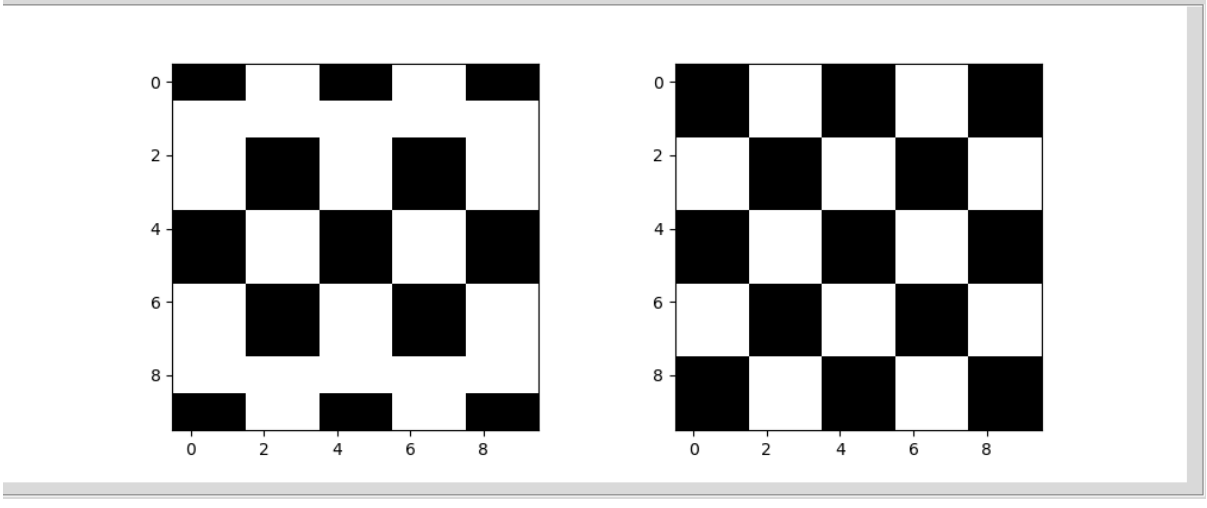
（上圖回想錯誤）

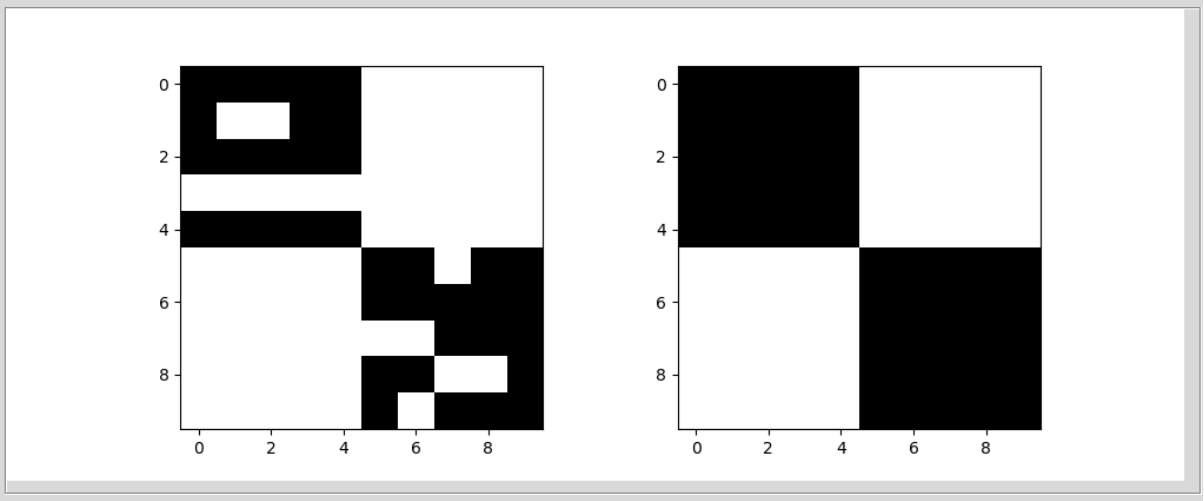



0
2
4
6
8
0
2
4
6
8
0
2
4
6
8
0
2
4
6
8

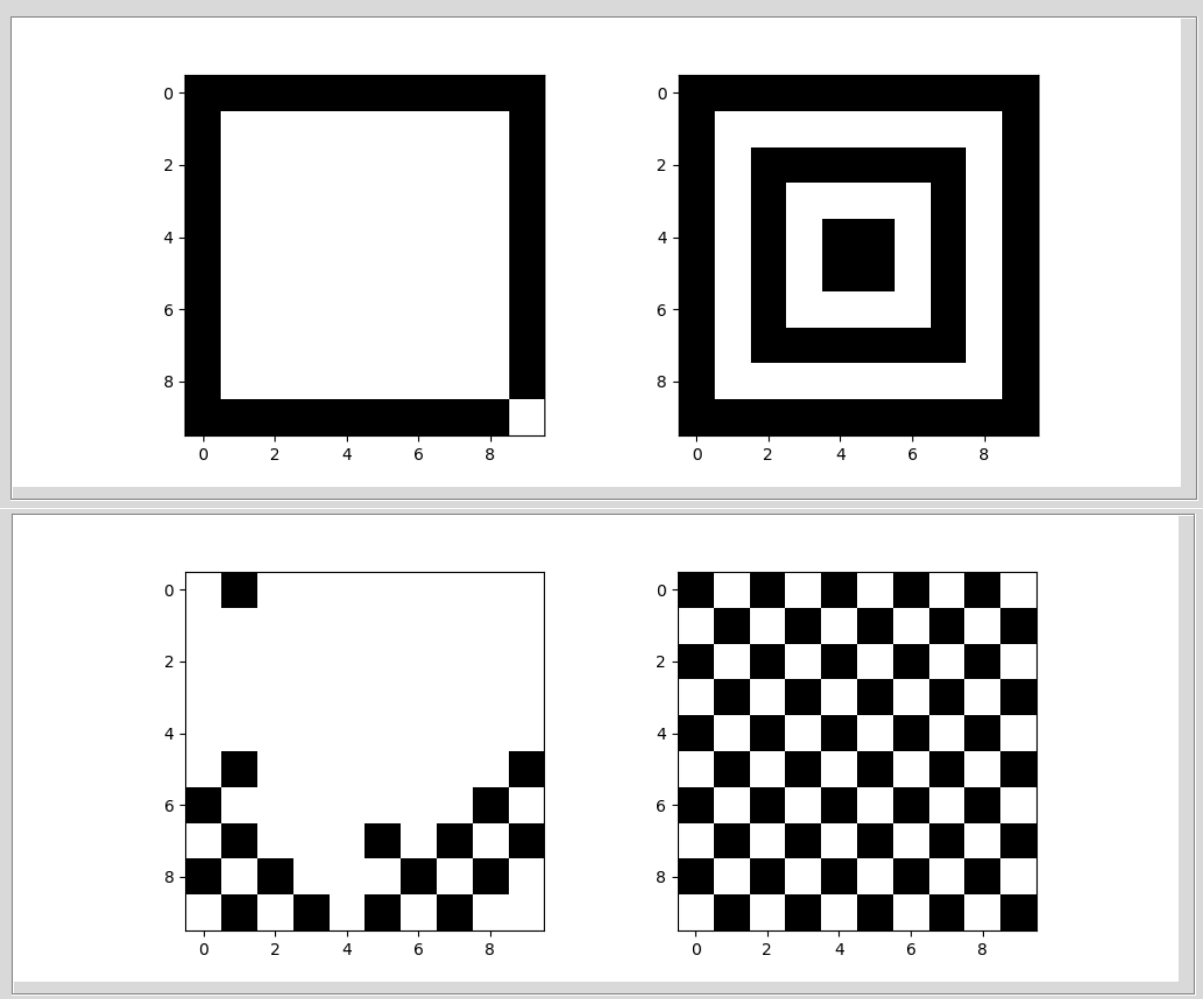
（上圖回想錯誤）

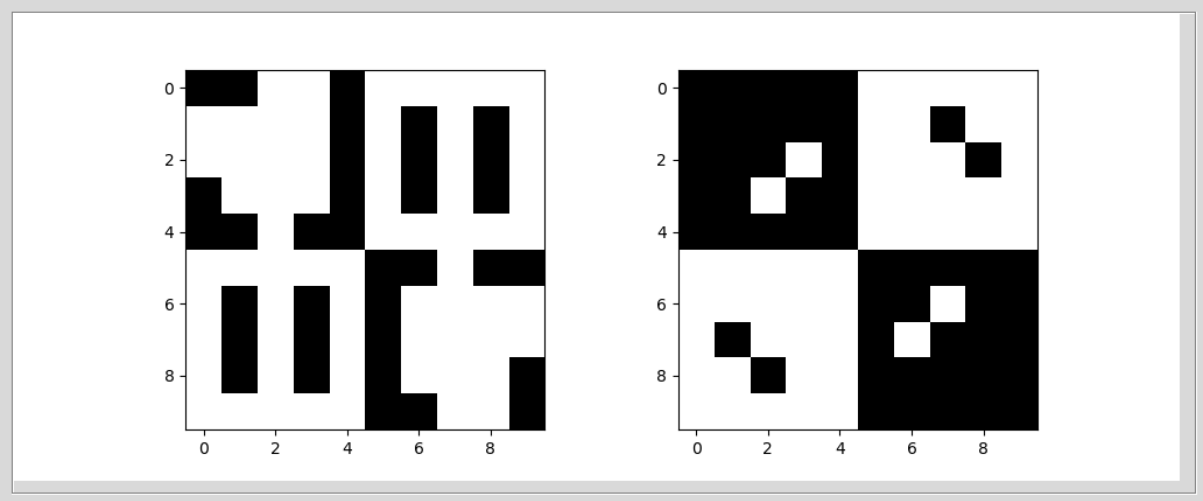

0
2
4
6
8
0
2
4
6
8
0
2
4
6
8
0
2
4
6
8


0
2
4
6
8
0
2
4
6
8
0
2
4
6
8
0
2
4
6
8



此資料集15個資料中，沒加入雜訊的情況下，**有10個正確回想**。
加入雜訊後，有12個正確回想，3個回想錯誤，錯誤結果如下:





（上3張圖皆為回想錯誤）

心得結論：

在這次的實作Hopfield練習中，基本題每個都長得很不一樣，所以**有沒有加入雜訊對於回想結果的差異不大**，都能夠**正確回想。而加分題的資料中**，有些資料長得非常相似，**所以透過加入雜訊有機會可以讓結果變更好，但是對於某些資料而言(ex.上半空的，只有下半有交叉的點那張)，不容易回想正確，或許增加資料量可以改善回想結果。**

實作Hopfield的時候，設定合適的**迭代次數**在實作上會遇到一些問題，因為資料的長相會影響迭代收斂的快慢，所以直接設定一個值不是一個好方法，要使用條件收斂的方式確保結束後的結果是穩定收斂的會比較好。

PS:原(.exe)檔案破百 MB，所以有先壓縮了，故打開需要等一陣子。