Київський національний університет
імені Т.Шевченка

# Звіт

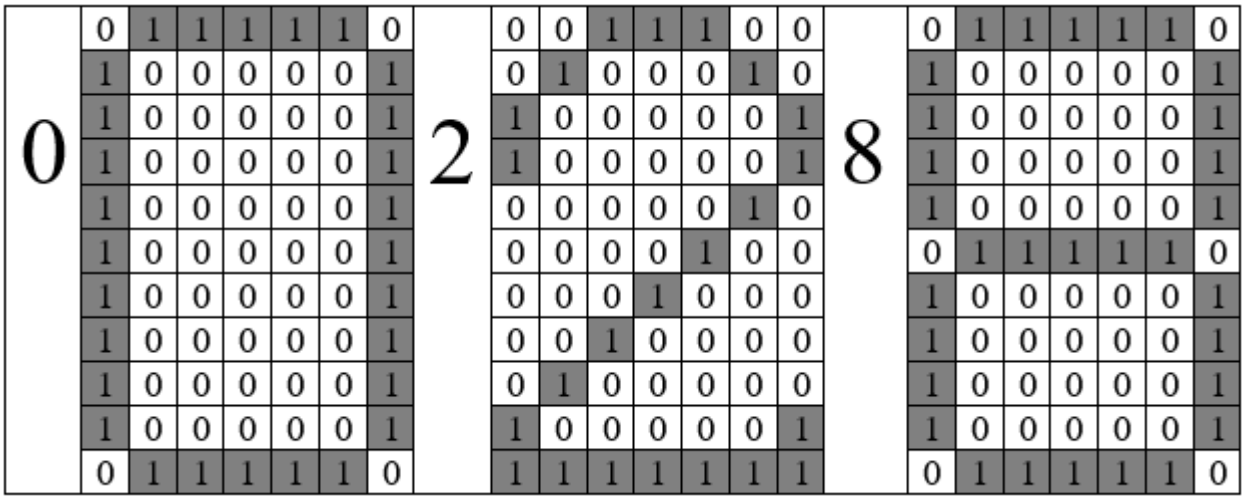
до лабораторної роботи 2
з предмету Нейронні мережі та нейрообчислення
«Мережі Хопфілда»

***Студента четвертого курсу***
***Групи ТК-41***
***Факультету комп'ютерних наук***
***та кібернетики***
***Некряча Владислава***

***Київ***
*2023*

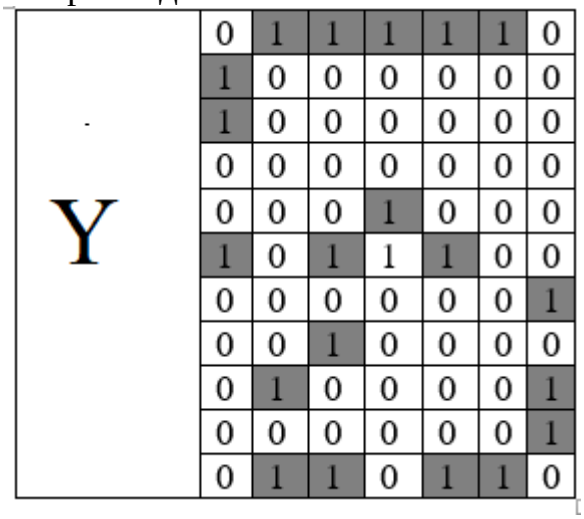
## Постановка задачі

1. Для роботи з нейронною мережею Хопфілда взяти в якості еталонів цифри від 0 до 9. Розмір цифр має бути 7*11 пікселів або 9*11 пікселів. Наприклад.




2. Навчити нейронну мереже на вибраних еталонах.
3. Створити зашумлений образ і очистити (розпізнати) його мережею Хопфілда.

Наприклад.




4. Оформити звіт.

**Алгоритм навчання мережі Хопфілда.**

Перший етап – навчання. Нехай еталонний зразок N-мірний вектор

$$\mathbf{x=[x_1,x_2,...,x_N],}$$

де $x_i$ рівні -1 або 1.

Результат навчання мережі Хопфілда - вагова матриця.

1. $w_{ii}=0$
2. Для всіх образів корегування ваг:

$$w_{ij}=w_{ij} + x_i x_j,$$

3. $w_{ii}=0$

і нормуємо $w_{ij}:=w_{ij}/N$.

Мережа навчена!

## Розпізнавання.

Нехай шукаємо образ $Y=[y_1,y_2,...,y_N]$.

1. Цикл по *j* от *1* до *N*:
2. Покладемо $d = 0$
3. Вкладений цикл по *i* від *1* до *N*:
4. $d = d + w_{ij}y_i$
5. Кінець вкладеного циклу
6. Якщо $d > 0$, то $z_j = 1$, інакше $z_j = -1$.
7. Кінець зовнішнього циклу
8. Маємо вектор: $Z=[z_1,z_2,...,z_N]$.
   Якщо вектор *Z* є в множині еталонів, то алгоритм знайшов образ *Z*, що відповідає *Y*.
9. Покладемо Y = Z
10. Переходимо до кроку 1.

Якщо алгоритм не може знайти образ, то значить мережа Хопфілда не згадала такий образ. Це може бути у випадку, коли значення вектора *Z* не міняється в процесі навчання.

**Опис**

В якості даних для тренування – текстові файли розміром 9 на 11 символів чисел від 0 до 9:

# 0:

011111110

100000001

100000001

100000001

100000001

100000001

100000001

100000001

100000001

100000001

011111110

# 1:

000010000

000110000

001010000

010010000

000010000

000010000

000010000

000010000

000010000

000010000

111111111

# 2:

111111111

000000001

000000001

000000001

000000001

111111111

100000000

100000000

100000000

100000000

111111111

# 3:

111111111

000000001

000000001

000000001

000000001

111111111

000000001

000000001

000000001

000000001

111111111

# 4:

100000001

100000001

100000001

100000001

100000001

111111111

000000001

000000001

000000001

000000001

000000001

# 5:

111111111

100000000

100000000

100000000

100000000

111111111

000000001

000000001

000000001

000000001

111111111

# 6:

111111111

100000000

100000000

100000000

100000000

111111111

100000001

100000001

100000001

100000001

111111111

# 7:

111111111

000000001

000000001

000000001

000000001

000000001

000000001

000000001

000000001

000000001

000000001

# 8:

111111111

100000001

100000001

100000001

100000001

111111111

100000001

100000001

100000001

100000001

111111111

# 9:

111111111

100000001

100000001

100000001

100000001

111111111

000000001

000000001

000000001

000000001

111111111

Спочатку зчитуємо файли та перетворюємо їх у масив, що має значення -1 або 1: якщо цифра 0, то значення в масиві -1, інакше 1. Ініціалізуємо матрицю вагів нулями. Зчитуємо зашумлені зразки для розпізнавання.

```
def readfile(filenames):

    learning_samples = []
    Sample = []

    for filename in filenames:
        file = open(filename, 'r')
        for line in file:
            Sample.append(line.strip())
        learning_samples.append(Sample)
        Sample = []
        file.close()

    return learning_samples

def samp_to_vect(samples):

    X = []

    x_i = []

    for sample in samples:
        for row in sample:
            for char in row.strip():
                if char == '0':
                    x_i.append(-1)
                else:
                    x_i.append(1)
        X.append(x_i)
        x_i=[]
    return X


if __name__ == '__main__':
    learningfiles = []

    learnfilespath = "/home/vlad/PyCharmProjects/Uni/NN/Lab2/images_to_learn"
    srcc=[3,5]
    for i in srcc:
        learningfiles.append(learnfilespath + str(i) + ".txt")

    recognisingfiles = []

    recognisingfilespath =
"/home/vlad/PyCharmProjects/Uni/NN/Lab2/images_to_recognize"
    srcc2 = [0,1,2,3,4]
    for i in srcc2:
        recognisingfiles.append(recognisingfilespath + str(i))

    X = samp_to_vect(readfile(learningfiles))
    W = weights_initialize()
    #Z = []


    Y = samp_to_vect(readfile(recognisingfiles))
    N = len(X[0])
```

Потім тренуємо мережу за допомогою набору масивів для тренування:

```
def learning(W, X):

    X_i_len = len(X[0])
    W_i_j = 0

    for k in range(0, len(srcc)):
        for i in range(0, 99):
            for j in range(0, 99):
                if i!=j:
                    W_i_j = W[i][j] + X[k][i]*X[k][j]
                    W[i][j] = W_i_j
                    W_i_j = 0

    for i in range(0, 99):
        for j in range(0, 99):
            W[i][j] /= X_i_len

    return W

W = learning(W, X)
```

Перетворюємо зчитаний зашумлений файл до 100 разів або доки він не стане ідентичним до одного з тренувальних.

```
while(counter!=100):
    Z = []
    for j in range(0, 99):
        d = 0
        for i in range(0, 99):
            d += W[i][j]*RecF[i]
        if d > 0:
            Z.append(1)
        else:
            Z.append(-1)
    # search for z in samples X
    flag = find_image(Z,X)
    if flag!=-1:
        print("Шуканий образ: ",flag)
        break;
    else:
        RecF = Z
    #if counter==50:
    #    print()
    counter+=1
```

## Приклад роботи:

Розмір тренувального масиву 4 цифри – 0,1,5,7

На вхід подаємо зашумлений образ з файлу 1.txt, наша програма успішно його знаходить.

**Вхід:**

**000010000**

**000110000**

**001010000**

**010010000**

**000010000**

**000001000**

**000001000**

**000001000**

**000010000**

**000010000**

**111111111**

**Вихід:**

**Шуканий образ:  2 (тобто номер 2 у списку тренувальних еталонів, а саме 1.txt)**

**000010000**

**000110000**

**001010000**

**010010000**

**000010000**

**000010000**

**000010000**

**000010000**

**000010000**

**000010000**

**111111111**

**Якщо на вхід подати зашумлену п’ятірку з файлу src0.txt, то за даного тренувального набору програма не знайде її еталон, хоча якщо змінити тренувальні набори і форму цифр у еталонах налаштувати краще, то розпізнавання можливе.**

**Вхід:**

**111111111**

**100000000**

**111000000**

**111100000**

**100000000**

**111111011**

**000000001**

**000000001**

**000000101**

**000001111**

**111111111**

**Вихід:**

**111111111**

**100000001**

**100000001**

**100000001**

**100000001**

**100000001**

**000000001**

**000000001**

**000000001**

**000000001**

**011111111**

**Якщо зменшити тренувальний набір до 2 еталонів, то і зашумлена трійка і зашумлена одиниця буде розпізнаватися**

**Вхід:**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**000000001**

**000000000**

**000000010**

**000000110**

**111111111**

**Вихід:**

**Шуканий образ:  2**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**Ще приклад:**

**Тренувальний набір складається з 5 та 3. На вхід подаємо зашумлену 3 та 5, система добре розпізнає їх.**

**Вхід:**

**111111111**

**100000000**

**110000000**

**100000000**

**110000000**

**111111011**

**000000001**

**000000111**

**000000001**

**000001111**

**111111111**

**Вихід:**

**Шуканий образ:  2**

**111111111**

**100000000**

**100000000**

**100000000**

**100000000**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**Вхід:**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**000000001**

**000000000**

**000000010**

**000000110**

**111111111**

**Вихід:**

**Шуканий образ:  1**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**000000001**

**000000001**

**000000001**

**000000001**

**111111111**

**Якщо за такого тренувального масиву подати на вхід цілковитий шум, то система не буде розпізнавати взагалі.**

**Вхід:**

**010101010**

**110000110**

**010101010**

**010101010**

**010101010**

**111000101**

**101010010**

**101010010**

**101001010**

**101001010**

**101001010**

**Вихід:**

**000000000**

**111111110**

**111111110**

**111111110**

**111111110**

**000000000**

**111111110**

**111111110**

**111111110**

**111111110**

**000000000**

**Якщо збільшувати розмір навчального датасету до 10 цифр, то система почне гірше працювати за наявних зразків. Якщо зразки краще настроїти то система буде краще їх розпізнавати.**

Отже, мережа Хопфілда для зображень 9 на 11 добре працює при кількості навчальних зображень 2 або 3, з більшою кількістю наявних навчальних зоюражень починає працювати гірше. Для подальшого покращення результатів роботи можна вдосконалити зображення у навчальному наборі, або збільшити кількість нейронів для збільшення розміру «пам’яті» мережі.