Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра комп’ютерного моделювання процесів і систем

ЗВІТ

з лабораторної роботи №4

“ МОДЕЛЮВАННЯ РОБОТИ АСОЦІАТИВНОЇ ПАМ'ЯТІ (МЕРЕЖІ ХОПФІЛДА)”

з курсу

«Нейронні мережі та методи проектування»

Виконав: студент групи ІКМ-М222к Черкас Ю.В.

Перевірив: професор, д.т.н. Успенський В.Б.

Харків 2023р

Постановка задачі

Спроектувати мережу Хопфілда або Хемінга за допомогою будь-якого пакету для випадку розмірність вектора 100, кількість образів 10, навести декілька прикладів визначення найближчого еталонного образу для різних спотворених образів. Запротоколювати роботу.

Виконання

Проведемо моделювання роботи мережі Хопфілда за допомогою мови Python та бібліотек NumPy і NeuroLab.

1. Генерація матриці еталонних образів. Розмірність вектору 100, кількість образів – 10.

```
import numpy as np

# створення матриці еталонних образів
num_patterns = 10
pattern_size = 100
target = np.random.randint(0, 2, (num_patterns, pattern_size))
target[target == 0] = -1
```

2. Створення та навчання мережі Хопфілда.

```
# створення та навчання мережі Хопфілда на еталонних образах
import neurolab as nl
hopfield_net = nl.net.newhop(target)
```

3. Перевірка роботи мережі на неспотворених векторах.

```
print('Розпізнавання неспотворених векторів')
output_hopfield = hopfield_net.sim(target)
for i in range(len(target)):
    print(f'Вектор {i} розпізнаний:', (output_hopfield[i] == target[i]).all())
```

```
Розпізнавання неспотворених векторів
Вектор 0 розпізнаний: True
Вектор 1 розпізнаний: True
Вектор 2 розпізнаний: True
```

```
Вектор 3 розпізнаний: True
Вектор 4 розпізнаний: True
Вектор 5 розпізнаний: True
Вектор 6 розпізнаний: True
Вектор 7 розпізнаний: True
Вектор 8 розпізнаний: True
Вектор 9 розпізнаний: True
```

4. Моделювання роботи мережі Хопфілда при розпізнавання спотворених образів.

```
def model_with_error(n_errors):
    # створення спотвореного образу
    distorted_target = target.copy()
    distorted_target[:,:n_errors] *= -1  # спотворення перших n_errors
елеменів образу

    # розпізнавання спотворених образів
    output_hopfield = hopfield_net.sim(distorted_target)

    print(f'Спотворені {n_errors} елементів')
    for i in range(len(distorted_target)):
        print(f'Вектор {i} розпізнаний:', (output_hopfield[i] ==
target[i]).all())

model_with_error(10)
model_with_error(20)
model_with_error(30)
model_with_error(40)
model_with_error(50)
```

Проведемо моделювання для різної кількості спотворених елементів та запротоколюємо результат роботи.

```
Спотворені 10 елементів
Вектор 0 розпізнаний: True
Вектор 1 розпізнаний: True
Вектор 2 розпізнаний: True
Вектор 3 розпізнаний: True
Вектор 4 розпізнаний: True
Вектор 5 розпізнаний: True
Вектор 6 розпізнаний: True
Вектор 7 розпізнаний: True
Вектор 8 розпізнаний: True
Вектор 9 розпізнаний: True
```

```
Спотворені 20 елементів
Вектор 0 розпізнаний: True
Вектор 1 розпізнаний: True
Вектор 2 розпізнаний: True
Вектор 3 розпізнаний: True
Вектор 4 розпізнаний: True
Вектор 5 розпізнаний: True
Вектор 6 розпізнаний: True
Вектор 7 розпізнаний: True
Вектор 8 розпізнаний: True
Вектор 9 розпізнаний: False

Спотворені 30 елементів
Вектор 0 розпізнаний: False
Вектор 1 розпізнаний: True
Вектор 2 розпізнаний: True
Вектор 3 розпізнаний: True
Вектор 4 розпізнаний: True
Вектор 5 розпізнаний: False
Вектор 6 розпізнаний: False
Вектор 7 розпізнаний: True
Вектор 8 розпізнаний: True
Вектор 9 розпізнаний: False

Спотворені 40 елементів
Вектор 0 розпізнаний: False
Вектор 1 розпізнаний: False
Вектор 2 розпізнаний: False
Вектор 3 розпізнаний: False
Вектор 4 розпізнаний: False
Вектор 5 розпізнаний: False
Вектор 6 розпізнаний: False
Вектор 7 розпізнаний: True
Вектор 8 розпізнаний: False
Вектор 9 розпізнаний: False

Спотворені 50 елементів
Вектор 0 розпізнаний: False
Вектор 1 розпізнаний: False
Вектор 2 розпізнаний: False
Вектор 3 розпізнаний: False
Вектор 4 розпізнаний: False
Вектор 5 розпізнаний: False
Вектор 6 розпізнаний: False
Вектор 7 розпізнаний: False
Вектор 8 розпізнаний: False
Вектор 9 розпізнаний: False
```

Висновки

На даній лабораторній роботі ми провели моделювання роботи асоціативної пам’яті на прикладі мережі Хопфілда. За основу була взята мова Python та готова бібліотека для роботи з нейронними мережами NeuroLab. Це дозволило нам згенерувати мережу Хопфілда для 10 векторів розмірністю 100.

Для даних параметрів моделювання, мережа Хопфілда гарно себе проявляє та здатна повністю відновити вхідний сигнал, якщо кількість спотворених елементів становить 10%. При кількості спотворених елементів рівних 20%, мережа не здатна розпізнати 1 вектор з 10. При спотворенні вхідних векторів в 30%, мережа помиляється в 4 випадках з 10. Якщо 50% всіх елементів початкових векторів являються спотвореними, то мережа не спроможна розпізнати жодного образу. Звісно ж отримані результати залежать від випадкових чинників (відстані Хеммінга між випадково згенерованими образами, характеру помилки) та можуть незначно відрізнятися при повторних моделюваннях. Проте сам характер результатів буде відповідати отриманим в даній лабораторній роботі.