Министерство образования Республики Беларусь

Учреждение образования

“Брестский государственный университет”

Кафедра ИИТ

Лабораторная работа №5

По дисциплине: “МиАПР”

Тема: “ Нелинейные ИНС в задачах распознавания образов”

Выполнил:

Студент 2-го курса

Группы ПО-7

Степанюк А.С.

Проверил:

Крощенко А.А.

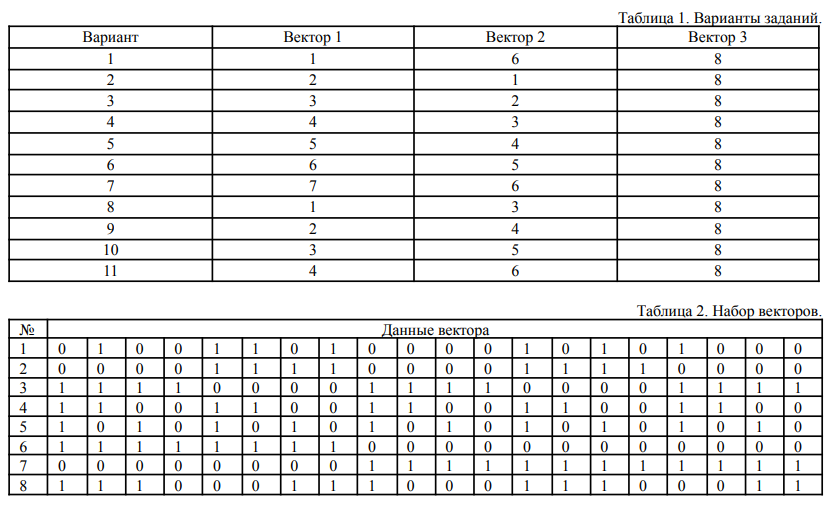
Брест, 2021

**Цель работы:** изучить обучение и функционирование нелинейной ИНС при решении задач распознавания образов.

***Задание.***

Написать на любом ЯВУ программу моделирования нелинейной ИНС для распознавания образов (рекомендуется использовать сигмоидную функцию). Количество НЭ в скрытом слое взять согласно варианту работы №9 (можно варьировать, если сеть не обучается или некорректно функционирует).

Провести исследование полученной модели. При этом на вход сети необходимо подавать искаженные образы, в которых инвертированы некоторые биты. Критерий эффективности процесса распознавания - максимальное кодовое расстояние (количество искаженных битов) между исходным и поданным образом.



**Main.py**

import numpy as np

class NeuralNet :

def \_\_init\_\_(self):

self.weights\_x = np.random.normal(-1, 1, (20, 4))

self.threshold\_h = np.random.normal(-1, 1, 4)

self.weights\_h = np.random.normal(-1, 1, (4, 3))

self.threshold\_y = np.random.normal(-1, 1, 3)

def go(self, x):

self.input = x

self.sum\_h = np.dot(self.input, self.weights\_x) - self.threshold\_h

self.hidden = 1.0 / (1.0 + np.exp(-self.sum\_h))

self.sum\_y = np.dot(self.hidden, self.weights\_h) - self.threshold\_y

self.output = self.sum\_y

return self.output

def changing(self, error\_y, alpha):

error\_h = np.dot(error\_y, self.weights\_h.transpose())

gamma\_y = alpha \* error\_y

self.weights\_h -= np.dot(self.hidden.reshape(-1, 1), gamma\_y.reshape(1, -1))

self.threshold\_y += gamma\_y

gamma\_h = alpha \* error\_h \* self.hidden \* (1.0 - self.hidden)

self.weights\_x -= np.dot(self.input.reshape(-1, 1), gamma\_h.reshape(1, -1))

self.threshold\_h += gamma\_h

def learning(self, x, e, alpha):

square\_error = 0

for i in range(len(e)):

output = self.go(x[i])

error = output - e[i]

square\_error += (error \* \*2 / 2).sum()

self.changing(error, alpha)

return square\_error

vectors = np.array(

[

[0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0],

[1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0],

[1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1],

]

)

etalons = np.array(

[

[1, 0, 0],

[0, 1, 0],

[0, 0, 1],

]

)

neural\_net = NeuralNet()

speed = 0.06

for i in range(30000):

error = neural\_net.learning(vectors, etalons, speed)

if error <= 1e-3:

break

for i in range(3):

print(f"\nВектор {i}")

output = neural\_net.go(vectors[i])

result = output.argmax() == i

print(f" 20/20: {result}")

for numb, j in enumerate(np.random.choice(20, 20, replace = False)):

vectors[i][j] = 1 - vectors[i][j]

output = neural\_net.go(vectors[i])

result = output.argmax() == i

print(f"{19 - numb: 3}/20: {result}")

**Результат выполнения программы:**

Вектор 0

20/20: True

19/20: True

18/20: True

17/20: True

16/20: True

15/20: True

14/20: True

13/20: False

12/20: False

11/20: False

10/20: False

9/20: False

8/20: False

7/20: False

6/20: False

5/20: False

4/20: False

3/20: False

2/20: False

1/20: False

0/20: False

Вектор 1

20/20: True

19/20: True

18/20: False

17/20: False

16/20: False

15/20: False

14/20: False

13/20: True

12/20: True

11/20: False

10/20: False

9/20: False

8/20: False

7/20: False

6/20: False

5/20: False

4/20: False

3/20: False

2/20: False

1/20: False

0/20: False

Вектор 2

20/20: True

19/20: True

18/20: True

17/20: True

16/20: True

15/20: True

14/20: True

13/20: True

12/20: True

11/20: True

10/20: True

9/20: True

8/20: True

7/20: True

6/20: True

5/20: True

4/20: True

3/20: True

2/20: True

1/20: True

0/20: True

**Вывод:** В ходе лабораторной работы реализовал нелинейную ИНС для распознавания образов.