|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики, искусственного интеллекта и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**ЛАБОРАТОРНАЯ РАБОТА № 1**

***Реализация однослойного перцептрона***

**ПО КУРСУ:**

***«Теория искусственных нейронных сетей»***

Студент группы ИУ9-72Б: *Терентьева А.С.*

Преподаватель: *Каганов Ю.Т.*

*Москва, 2023 г.*

1. **Цель:**

Изучение однослойного персептрона, исследование его работы на различных функциях активации.

1. **Постановка задачи:**

1. Реализовать на языке высокого уровня однослойный персептрон и проверить его работоспособность на примере искусственных данных типа цифр от 0 до 9 и букв русского алфавита. Размер поля 5x4.

2. Исследовать работу персептрона на основе использования различных функций активации. (Линейной, сигмоиды, гиперболического тангенса, ReLu).

1. **Реализация**

from PIL import Image, ImageDraw #Подключим необходимые библиотеки.

import math

import random

import numpy as np

import matplotlib.pyplot as plt

plt.grid()

n = 3 \* 5

x = [0] \* 10

y = [[0] \* 10 for i in range(10)]

def getImage(N):

  image = Image.open(f"tests/{N}.png") #Открываем изображение.

  width = image.size[0] #Определяем ширину.

  height = image.size[1] #Определяем высоту.

  pix = image.load() #Выгружаем значения пикселей.

  if width != 3 and height != 5:

    sys.exit()

  x = [0] \* n

  for i in range(width):

    for j in range(height):

      x[j \* width + i] = 0 if pix[i, j] == (0,0,0,0) else 1

  return x

def maxErr():

  max = 0

  for i in range(10): # обучение на 10 тестах

      for j in range(10): # 10 нейронов

        z = phi(sum(x[i], w[j]))

        e = abs(y[i][j] - z)

        max = e if e > max else max

  return max

def find\_answ(res):

  num = 0

  min = 1

  for i in range(len(res)):

    if abs(1 - res[i]) < min:

      min = abs(1 - res[i])

      num = i

  return num

def phi():

  return 0

def dphi(x):

  delta = 0.001

  return (phi(x + delta) - phi(x)) / delta

def lin(x):

  return x

def dlin(x):

  return 1

def relu(x):

  return x if x > 0 else 0

def drelu(x):

  return 1 if x > 0 else 0

def sigm(x):

  return 1 / (1 + np.exp(-x))

def dsigm(x):

  return np.exp(-x) / (1 + np.exp(-x)) \*\* 2

  #return sigm(x) / ( 1 - sigm(x))

def th(x):

  return np.tanh(x)

  return (np.exp(x) - np.exp(-x)) / (np.exp(x) + np.exp(-x))

def dth(x):

  return 1 - th(x) \*\* 2

def err(y0, y):

  1 / 2 \* (y - y0)\*\*2

def dErr(x, w, y0):

  XW = sum(x, w)

  y = phi(XW)

  res = [0] \* len(x)

  for i in range(len(x)):

    res[i] = lr \* (y0 - y) \* x[i] \* dphi(XW)

  return res

def sum(x, w):

  res = 0

  for i in range(n):

    res += x[i]\*w[i]

  return res

def study2(x, y, j):

  delta = dErr(x, w[j], y)

  for i in range(n):

    w[j][i] += delta[i]

def study():

  for step in range(10001):

    if step % 100 == 0:

      #print(step)

      epohs.append(step)

      errors.append(maxErr())

    for i in range(10): # обучение на 10 тестах

      for j in range(10): # 10 нейронов

        study2(x[i], y[i][j], j)

for i in range(10):

  y[i][i] = 1

  x[i] = getImage(i)

funcs = ['lin', 'relu', 'sigm', 'th']

for func in funcs:

  phi = eval(func)

  #dphi = eval('d'+func)

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

  print(func)

  lr = 1 if func != 'lin' and func != 'th' else 0.1

  for \_ in range(5):

    correct = True

    '''print()

    print('lr =', lr)'''

    epohs = []

    errors = []

    w = [[0.05 for i in range(n)] for j in range(10)]

    study()

    for i in range(10):

      res = [0] \* 10

      mas = [0] \* 10

      for j in range(10):

        res[j] = phi(sum(x[i], w[j]))

        mas[j] = round(res[j], 2)

      output = find\_answ(res)

      print(mas)

      print(y[i])

      print(i, '--->', output, '\n')

      if output != i:

        correct = False

    if lr == 0.01 or (lr == 1 and func == 'sigm'):

      for j in range(10):

        print(w[j])

    plt.plot(epohs, errors, label=f'{lr} {correct}')

    lr /= 10

  plt.legend()

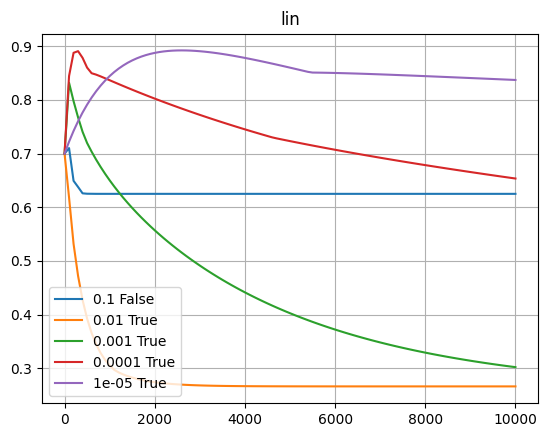
  plt.title(func)

  plt.show()

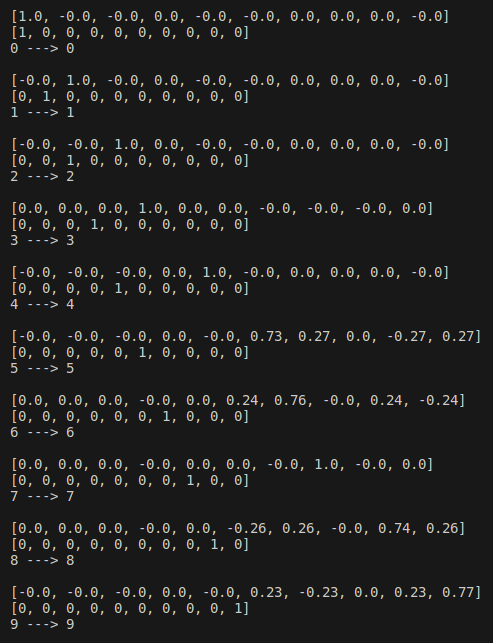
1. **Результат работы**

1. Линейная функция:

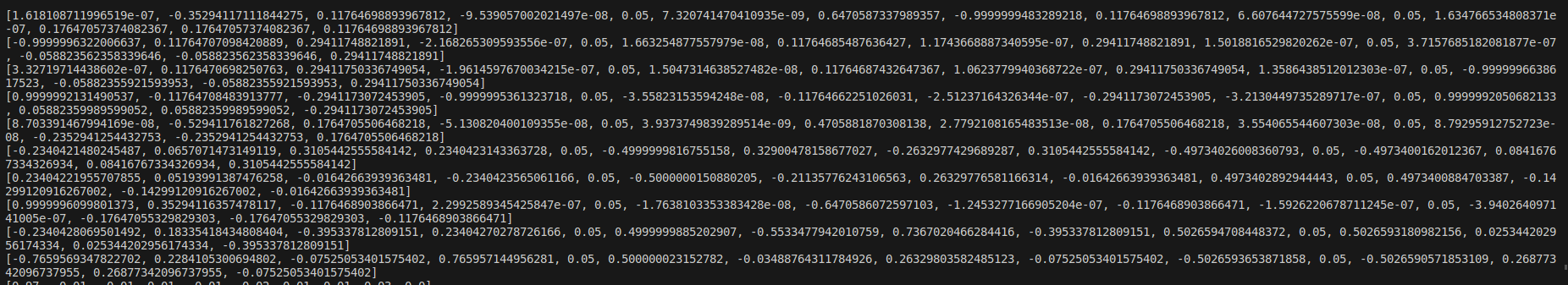
График зависимости ошибки от количества эпох:



Вывод результатов:

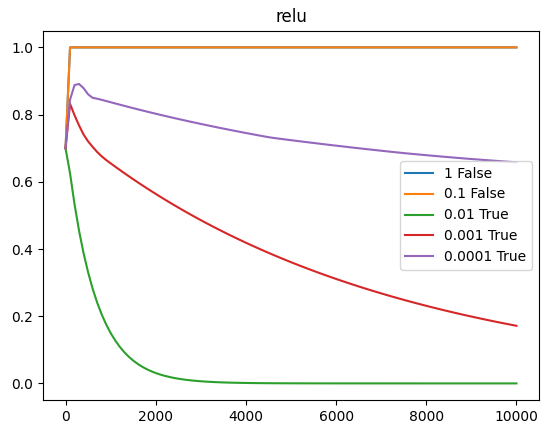


Вывод найденных весов:

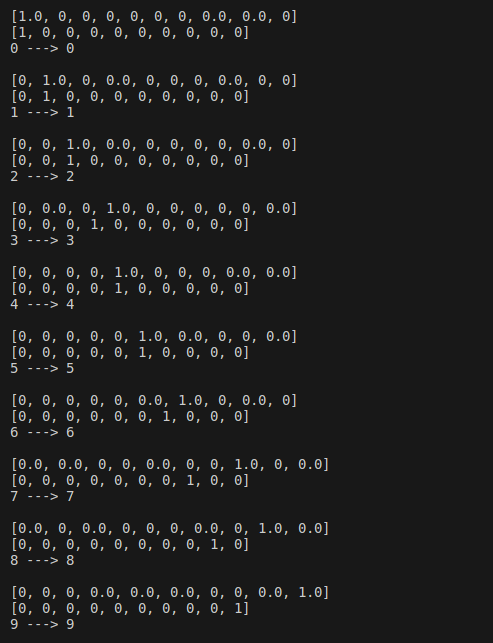


2. ReLU:

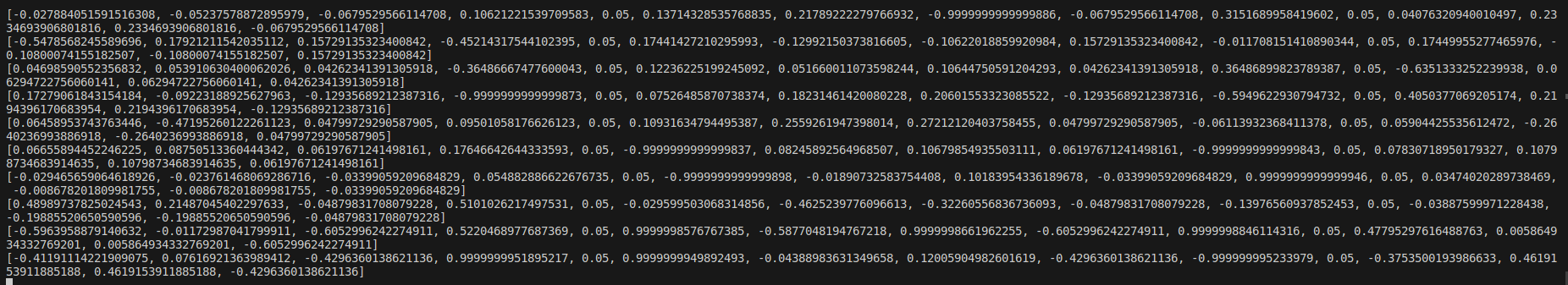
График зависимости ошибки от количества эпох:



Вывод результатов:

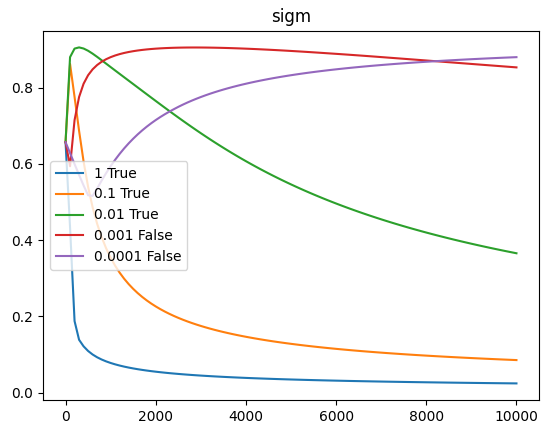


Вывод найденных весов:

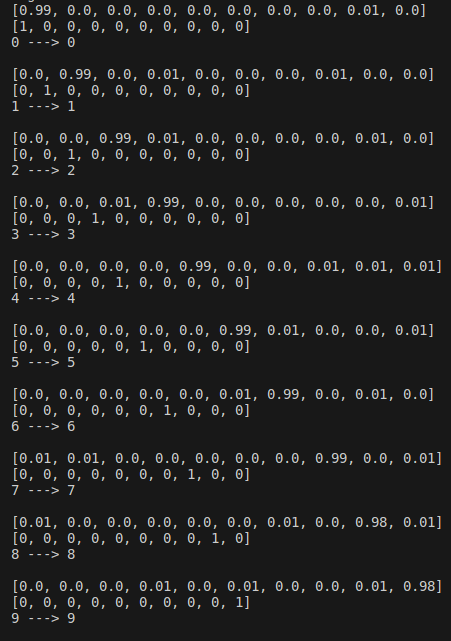


3. Сигмоида:

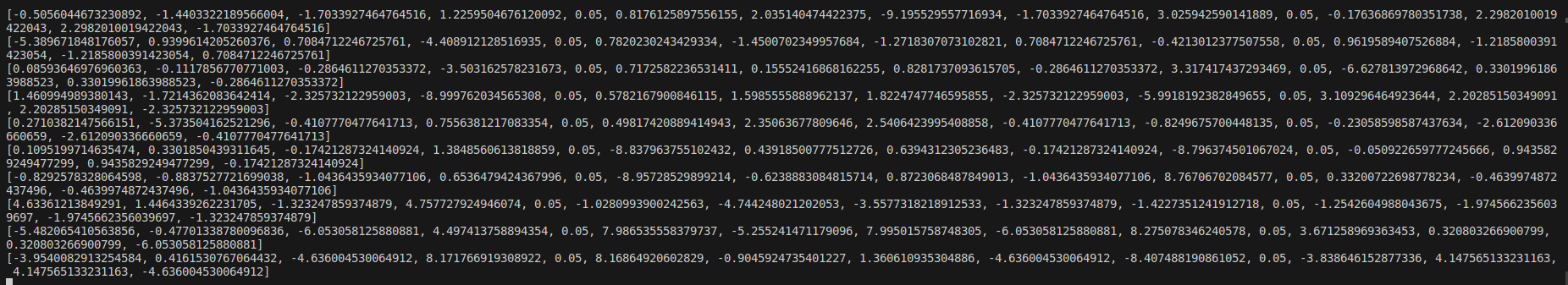
График зависимости ошибки от количества эпох:



Вывод результатов:

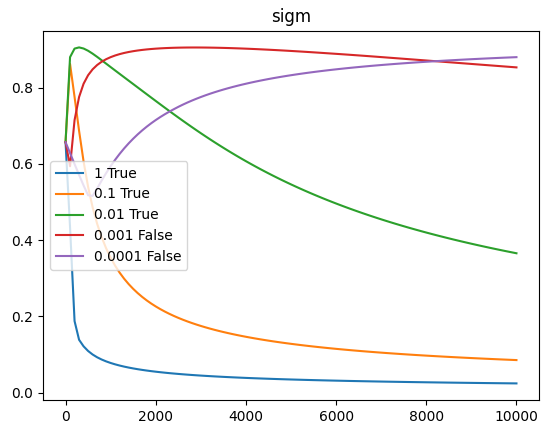


Вывод найденных весов:

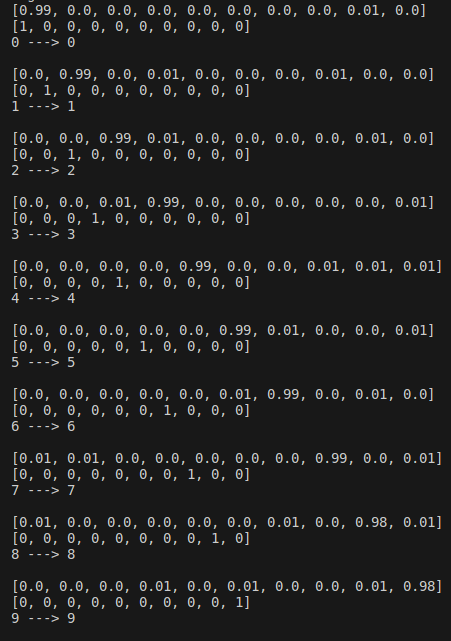


3. Гиперболический тангенс:

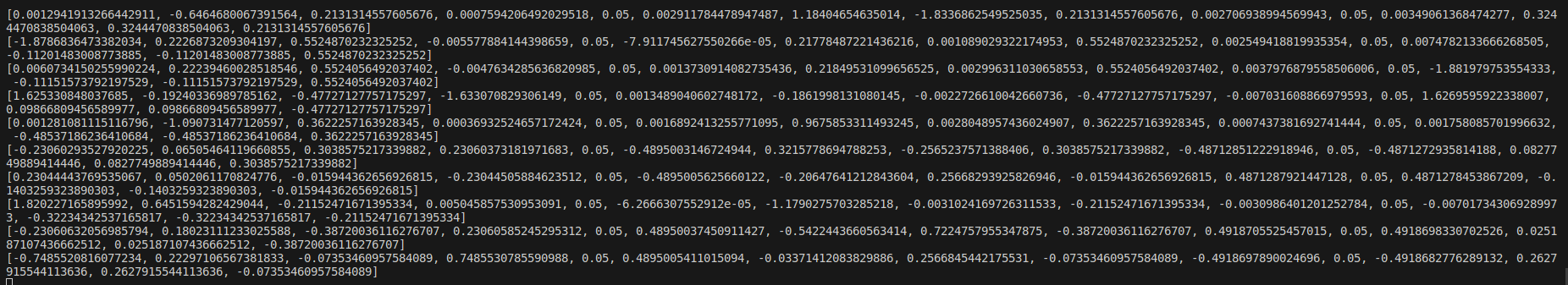
График зависимости ошибки от количества эпох:



Вывод результатов:



Вывод найденных весов:



1. **Выводы**

В ходе выполнения лабораторной работы был изучен принцип работы однослойного персептрона, была написана его реализация на языке программирования Python.

В ходе эксперимента по исследованию работы программы на основе различных функций активаций: линейной, сигмоиды, гиперболического тангенса, ReLu, были сделаны следующие выводы:

1. Самыми точными функциями оказались Relu (при скорости обучения 0.01) и сигмоида (при скорости обучения 1).
2. Быстрее всего персептрон обучался при функции Relu.
3. Линейная функция и гиперболический тангенс также оказались эффективными, с незначительными отклонениями, не влияющими на результа.
4. Линейная функция и функция ReLu выдают верный результат при скорости обучения, большей 0,1, а сигмоида – при скорости, меньшей 0,01.