Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №3

за 1 семестр

По дисциплине: «МиАПР»

Тема: «Нелинейные ИНС в задачах прогнозирования»

Выполнил:

Студент 2 курса

Группы ПО-4(1)

Кречко К.А

Проверил:

Крощенко А.А.

2020

**Лабораторная работа №3**

Нелинейные ИНС в задачах прогнозирования

Цель работы: Изучить обучение и функционирование нелинейной ИНС при решении задач прогнозирования.

**Вариант 2**

**Задание:**

**Код программы:**

import numpy as np

import sys

import math

def func(x):

a = 0.2

b = 0.4

c = 0.09

d = 0.4

return a \* math.cos(b \* x) + c \* math.sin(d \* x)

class Network:

def \_\_init\_\_(self, learning\_rate = 0.25):

self.weights\_0\_1 = np.random.normal(0.0, 2 \*\* -0.5, (2, 6))

self.weights\_1\_2 = np.random.normal(0.0, 1, (1, 2))

#self.bias\_0\_1 = 1

#self.bias\_1\_2 = 1

self.tanh\_mapper = np.vectorize(self.tanh)

self.learning\_rate = np.array([learning\_rate])

def tanh(self, x):

return np.tanh(x)

def predict(self, inputs):

inputs\_1 = np.dot(self.weights\_0\_1, inputs) #- self.bias\_0\_1

outputs\_1 = self.tanh\_mapper(inputs\_1)

inputs\_2 = np.dot(self.weights\_1\_2, outputs\_1) #- self.bias\_1\_2

outputs\_2 = self.tanh(inputs\_2)

return outputs\_2

def train(self, inputs, expected\_predict):

inputs\_1 = np.dot(self.weights\_0\_1, inputs) #- self.bias\_0\_1

outputs\_1 = self.tanh\_mapper(inputs\_1)

inputs\_2 = np.dot(self.weights\_1\_2, outputs\_1) #- self.bias\_1\_2

outputs\_2 = self.tanh(inputs\_2)

actual\_predict = outputs\_2[0]

error\_layer\_2 = np.array([actual\_predict - expected\_predict])

#gradient\_layer\_2 = actual\_predict \* (1 - actual\_predict)

gradient\_layer\_2 = -(self.tanh(actual\_predict) \*\* 2) - 1

weights\_delta\_layer\_2 = error\_layer\_2 \* gradient\_layer\_2

self.weights\_1\_2 -= (np.dot(weights\_delta\_layer\_2, outputs\_1.reshape(1, len(outputs\_1)))) \* self.learning\_rate

#self.bias\_1\_2 \*= self.learning\_rate \* weights\_delta\_layer\_2

error\_layer\_1 = weights\_delta\_layer\_2 \* self.weights\_1\_2

#gradient\_layer\_1 = outputs\_1 \* (1 - outputs\_1)

gradient\_layer\_1 = -(self.tanh(outputs\_1) \*\* 2) - 1

weights\_delta\_layer\_1 = error\_layer\_1 \* gradient\_layer\_1

self.weights\_0\_1 -= np.dot(inputs.reshape(len(inputs), 1), weights\_delta\_layer\_1).T \* self.learning\_rate

#self.bias\_0\_1 -= self.learning\_rate \* weights\_delta\_layer\_1

def MSE(y, Y):

return np.mean((y - Y) \*\* 2)

step = 0.1

counter = 0

train = []

for i in range(-15, 15):

combol = []

inputs = []

for j in range(6):

x = counter \* step

inputs.append(func(x))

counter += 1

combol.append(inputs)

x = counter \* step

combol.append(func(x))

combo = tuple(combol)

train.append(combo)

epochs = 1200

learning\_rate = 0.5

network = Network(learning\_rate)

losses = {'train':[], 'validation':[]}

Emin = 1e-5

epoch = 0

while True:

inputs = []

correct\_predictions = []

for input\_stat, correct\_predict in train:

network.train(np.array(input\_stat), correct\_predict)

inputs.append(np.array(input\_stat))

correct\_predictions.append(np.array(correct\_predict))

train\_loss = MSE(network.predict(np.array(inputs).T), np.array(correct\_predictions))

sys.stdout.write("\rTraining loss: {}, Epochs: {}".format(str(train\_loss)[:15], str(epoch)))

epoch += 1

if train\_loss <= Emin:

break

print("\nРЕЗУЛЬТАТЫ ОБУЧЕНИЯ:")

for input\_stat, correct\_predict in train:

print("the prediction is: {}, expected: {}, mistake: {}".format(

str(network.predict(input\_stat)),

str(correct\_predict),

str(network.predict(input\_stat) - correct\_predict)

))

predict = []

for i in range(30, 45):

combol = []

inputs = []

for j in range(6):

x = counter \* step

inputs.append(func(x))

counter += 1

combol.append(inputs)

x = counter \* step

combol.append(func(x))

combo = tuple(combol)

predict.append(combo)

print("\nРЕЗУЛЬТАТЫ ПРОГНОЗИРОВАНИЯ")

for input\_stat, correct\_predict in predict:

print("the prediction is: {}, expected: {}, mistake: {}".format(

str(network.predict(input\_stat)),

str(correct\_predict),

str(network.predict(input\_stat) - correct\_predict)

))

Результат:

Training loss: 9.9880723112419, Epochs: 193

РЕЗУЛЬТАТЫ ОБУЧЕНИЯ:

the prediction is: [0.2128805], expected: 0.21566083134884803, mistake: [-0.00278033]

the prediction is: [0.21645861], expected: 0.21895911035459031, mistake: [-0.0025005]

the prediction is: [0.20892725], expected: 0.20970576630561158, mistake: [-0.00077852]

the prediction is: [0.19034081], expected: 0.18843123836158115, mistake: [0.00190957]

the prediction is: [0.1609416], expected: 0.15635506863238505, mistake: [0.00458653]

the prediction is: [0.12150476], expected: 0.11531599308488084, mistake: [0.00618877]

the prediction is: [0.07374145], expected: 0.06766653774985257, mistake: [0.00607491]

the prediction is: [0.02052493], expected: 0.016138162401499512, mistake: [0.00438677]

the prediction is: [-0.03429855], expected: -0.03631531778004107, mistake: [0.00201677]

the prediction is: [-0.08655083], expected: -0.08668705685864561, mistake: [0.00013622]

the prediction is: [-0.13253674], expected: -0.13208954272987805, mistake: [-0.0004472]

the prediction is: [-0.16961667], expected: -0.16992012101009504, mistake: [0.00030345]

the prediction is: [-0.19630955], expected: -0.19801018972723708, mistake: [0.00170064]

the prediction is: [-0.21201294], expected: -0.2147495123693462, mistake: [0.00273657]

the prediction is: [-0.21659191], expected: -0.2191785231633661, mistake: [0.00258661]

the prediction is: [-0.21005877], expected: -0.21104333327177924, mistake: [0.00098456]

the prediction is: [-0.19246182], expected: -0.19081028472909783, mistake: [-0.00165154]

the prediction is: [-0.16401461], expected: -0.15963921782762297, mistake: [-0.00437539]

the prediction is: [-0.12542724], expected: -0.11931698437419269, mistake: [-0.00611026]

the prediction is: [-0.07831286], expected: -0.0721550181073362, mistake: [-0.00615784]

the prediction is: [-0.02544056], expected: -0.020856833953390233, mistake: [-0.00458373]

the prediction is: [0.02941034], expected: 0.03163694839911395, mistake: [-0.00222661]

the prediction is: [0.08205773], expected: 0.08231717273037709, mistake: [-0.00025944]

the prediction is: [0.12873336], expected: 0.12827864331182462, mistake: [0.00045472]

the prediction is: [0.16669124], expected: 0.16688666249216988, mistake: [-0.00019542]

the prediction is: [0.19435345], expected: 0.1959280622380926, mistake: [-0.00157461]

the prediction is: [0.21105525], expected: 0.21373807188989244, mistake: [-0.00268282]

the prediction is: [0.21663527], expected: 0.21929574955851877, mistake: [-0.00266048]

the prediction is: [0.21110065], expected: 0.2122825066497665, mistake: [-0.00118186]

the prediction is: [0.19449469], expected: 0.1931003706528747, mistake: [0.00139432]

РЕЗУЛЬТАТЫ ПРОГНОЗИРОВАНИЯ

the prediction is: [0.16700491], expected: 0.1628489392965125, mistake: [0.00415598]

the prediction is: [0.12927989], expected: 0.1232623471532765, mistake: [0.00601755]

the prediction is: [0.08283694], expected: 0.07660985802223053, mistake: [0.00622708]

the prediction is: [0.03033988], expected: 0.02556578153675297, mistake: [0.0047741]

the prediction is: [-0.024504], expected: -0.026943829095391877, mistake: [0.00243983]

the prediction is: [-0.07751556], expected: -0.07790891031330766, mistake: [0.00039335]

the prediction is: [-0.12485859], expected: -0.1244079372379214, mistake: [-0.00045066]

the prediction is: [-0.16368184], expected: -0.1637753973110943, mistake: [9.35580134e-05]

the prediction is: [-0.19230831], expected: -0.19375458827156833, mistake: [0.00144628]

the prediction is: [-0.21000745], expected: -0.21262698146889367, mistake: [0.00261953]

the prediction is: [-0.21658865], expected: -0.21931073488622205, mistake: [0.00272208]

the prediction is: [-0.21205283], expected: -0.21342270870649288, mistake: [0.00136988]

the prediction is: [-0.19643916], expected: -0.19530042843857692, mistake: [-0.00113873]

the prediction is: [-0.16991178], expected: -0.16598273658802887, mistake: [-0.00392904]

the prediction is: [-0.13306117], expected: -0.1271502419970512, mistake: [-0.00591093]

Вывод: В ходе выполнения работы спроектировал линейную ИНС с использованием адаптивного шага обучения.