МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
САМАРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
Институт автоматики и информационныхтехнологий
Кафедра «Информатика и Вычислительная техника»

ОТЧЕТ

о выполнении лабораторной работы №5

по дисциплине Компьютерные средства искусственного интеллекта

на тему Нейросеть своими руками

Преподаватель				А.А. Тюгашев
	(должность)	(подпись)	(дата)	(инициалы, фамилия)
				К.В. Портнов
	(должность)	(подпись)	(дата)	(инициалы, фамилия)
Студенты	4-ИАИТ-119			Е.А. Щаев
	(группа)	(подпись)	(дата)	(инициалы, фамилия)

Самара 2024 г.

Цель работы – Разработать нейросеть не использую библиотеки.

Программный код на языке Python:

```
import numpy as np

# Сигмоида и её производная
def sigmoid(x):
    return 1 / (1 + np.exp(-x))

def sigmoid_derivative(x):
    return x * (1 - x)


# Входные данные: [ночная температура, уровень влажности]
X = np.array([[10, 0.8],   # Ночью 10°C, влажность 80%
              [15, 0.6],   # Ночью 15°C, влажность 60%
              [20, 0.4],   # Ночью 20°C, влажность 40%
              [25, 0.2]])  # Ночью 25°C, влажность 20%

# Выходные данные: дневная температура
y = np.array([[15],   # 15°C
              [20],   # 20°C
              [25],   # 25°C
              [30]])  # 30°C

# Нормализация данных (от 0 до 1)
X = X / np.max(X, axis=0)
y = y / 30

# Размеры слоев
input_layer_size = 2
hidden_layer_size = 4
output_layer_size = 1

# Скорость обучения и количество эпох
learning_rate = 0.1
epochs = 10000

# Инициализация весов
np.random.seed(42)
weights_input_hidden = np.random.rand(input_layer_size, hidden_layer_size)
weights_hidden_output = np.random.rand(hidden_layer_size, output_layer_size)

# Обучение сети
for epoch in range(epochs):
    # Прямой проход
    hidden_layer_input = np.dot(X, weights_input_hidden)
    hidden_layer_output = sigmoid(hidden_layer_input)
    output_layer_input = np.dot(hidden_layer_output, weights_hidden_output)
```

```
    predicted_output = sigmoid(output_layer_input)

    # Ошибка
    error = y - predicted_output

    # Обратное распространение
    d_predicted_output = error * sigmoid_derivative(predicted_output)
    weights_hidden_output += np.dot(hidden_layer_output.T, d_predicted_output) *
learning_rate

    hidden_layer_error = np.dot(d_predicted_output, weights_hidden_output.T)
    d_hidden_layer_output = hidden_layer_error *
sigmoid_derivative(hidden_layer_output)
    weights_input_hidden += np.dot(X.T, d_hidden_layer_output) * learning_rate

# Де-нормализация предсказаний
predicted_output = predicted_output * 30

# Результаты
print("Обучение завершено.")
print("Прогнозируемые дневные температуры:")
for i, pred in enumerate(predicted_output):
    print(f"Input (ночь={X[i][0] * 30}°C, влажность={X[i][1] * 100}%): "
          f"Predicted дневная температура = {pred[0]:.2f}°C")


# Функция тестирования сети
def test_temperature_model(test_input):
    test_input = test_input / np.max(X, axis=0)  # Нормализация
    hidden_layer_input = np.dot(test_input, weights_input_hidden)
    hidden_layer_output = sigmoid(hidden_layer_input)
    output_layer_input = np.dot(hidden_layer_output, weights_hidden_output)
    predicted_output = sigmoid(output_layer_input)
    return predicted_output * 30  # Де-нормализация
print(f"Weights input hidden = {weights_input_hidden}\nWeight hidden output =
{weights_hidden_output}")



# Тестирование
new_data = np.array([[18, 0.5], [22, 0.3]])  # Новые данные
predictions = test_temperature_model(new_data)
for i, pred in enumerate(predictions):
    print(f"Test Input (ночь={new_data[i][0]}°C, влажность={new_data[i][1] *
100}%): "
          f"Predicted дневная температура = {pred[0]:.2f}°C")
```

Пример работы программы:

```
Обучение завершено.
Прогнозируемые дневные температуры:
Input (ночь=10.0°C, влажность=80.0%): Predicted дневная температура = 15.14°C, Заданные значения y = 15.0°C
Input (ночь=15.0°C, влажность=60.0%): Predicted дневная температура = 19.65°C, Заданные значения y = 20.0°C
Input (ночь=20.0°C, влажность=40.0%): Predicted дневная температура = 25.60°C, Заданные значения y = 25.0°C
Input (ночь=25.0°C, влажность=20.0%): Predicted дневная температура = 29.03°C, Заданные значения y = 30.0°C
```

Рисунок 1 – Проверка нейросети на входных данных

```
# Тестирование
new_data = np.array([[18, 1], [22, 0.3], [5, 0.2], [12, 0.3], [22, 0.7]])  # Новые данные
predictions = test_temperature_model(new_data)
for i, pred in enumerate(predictions):
    print(f"Test Input (ночь = {new_data[i][0]}°C, влажность = {new_data[i][1] * 100:.0f}%): "
          f"Predicted дневная температура = {pred[0]:.2f}°C")
```

```
22.0
Test Input (ночь = 18.0°C, влажность = 100%): Predicted дневная температура = 14.78°C
Test Input (ночь = 22.0°C, влажность = 30%): Predicted дневная температура = 27.77°C
Test Input (ночь = 5.0°C, влажность = 20%): Predicted дневная температура = 24.52°C
Test Input (ночь = 12.0°C, влажность = 30%): Predicted дневная температура = 25.38°C
Test Input (ночь = 22.0°C, влажность = 70%): Predicted дневная температура = 22.14°C
```

Рисунок 2 – Тестирование

Вывод: нейросеть принимает на вход определенные значения, обрабатывает их через скрытые слои с помощью весов и смещений, применяет функции активации и затем выдает выходной результат.