**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ(МИИТ)**)

**Институт управления и цифровых технологий**

Кафедра «Вычислительные системы, сети и информационная безопасность»

**Отчет по Лабораторной работе №5**

**«Сравнение результатов обучения искусственного нейрона и однослойного персептрона в условиях помех»**

**По дисциплине «Нейроинформатика»**

*Направление:* 10.03.01*Информационная безопасность*

*Профиль:**Безопасность компьютерных систем*

Выполнил:   
студент группы УИБ-311

Москаленко Виталий Александрович

Проверил:

Доцент Малинский С.В.

(должность, ФИО)

**МОСКВА 2024**

**СОДЕРЖАНИЕ**

[1 ЗАДАНИЕ 4](#_Toc178724917)

[2 ОСНОВНЫЕ ПОНЯТИЯ ОБ ИСКУССТВЕННОМ НЕЙРОНЕ 7](#_Toc178724918)

[3 ОСНОВНЫЕ ПОНЯТИЯ ОБ ОДНОСЛОЙНОМ ПЕРСЕПТРОНЕ 9](#_Toc178724919)

[4 ФОРМИРОВАНИЕ ОБУЧАЮЩЕЙ ВЫБОРКИ 11](#_Toc178724920)

[4.1 Добавление помех в обучающую выборку 11](#_Toc178724921)

[5 ОБУЧЕНИЕ ИСКУССТВЕННОГО НЕЙРОНА ПРИ ЗАДАННОМ КОЛИЧЕСТВЕ ПОМЕХ 12](#_Toc178724922)

[6 ОБУЧЕНИЕ ОДНОСЛОЙНОГО ПЕРСЕПТРОНА ПРИ ЗАДАННОМ КОЛИЧЕСТВЕ ПОМЕХ 13](#_Toc178724928)

[7 РЕЗУЛЬТАТ ОБУЧЕНИЯ ИСКУССТВЕННОГО НЕЙРОНА И ОДНОСЛОЙНОГО ПЕРСЕПТРОНА 14](#_Toc178724929)

[7.1 Результат обучения без добавления ошибок 15](#_Toc178724930)

[7.2 Результат обучения с добавлением 1 ошибки 16](#_Toc178724931)

[7.3 Результат обучения с добавлением 2 ошибок 17](#_Toc178724932)

[7.4 Результат обучения с добавлением 3 ошибок 18](#_Toc178724933)

[7.5 Результат обучения с добавлением 4 ошибок 19](#_Toc178724934)

[7.6 Результат обучения с добавлением 5 ошибок 20](#_Toc178724935)

[7.7 Результат обучения с добавлением 6 ошибок 21](#_Toc178724936)

[7.8 Результат обучения с добавлением 7 ошибок 22](#_Toc178724937)

[7.9 Результат обучения с добавлением 8 ошибок 23](#_Toc178724938)

[7.10 Результат обучения с добавлением 9 ошибок 24](#_Toc178724939)

[8 ИТОГОВАЯ ТОЧНОСТЬ ОБУЧЕНИЯ ИСКУССТВЕННОГО НЕЙРОНА И ОДНОСЛОЙНОГО ПЕРСЕПТРОНА 25](#_Toc178724940)

[ЗАКЛЮЧЕНИЕ 26](#_Toc178724941)

[ПРИЛОЖЕНИЕ А 27](#_Toc178724942)

[ПРИЛОЖЕНИЕ Б 33](#_Toc178724943)

1 ЗАДАНИЕ

Необходимо сравнить результаты обучения искусственного нейрона и однослойного персептрона для распознавания цифры "4" на основе ее графического представления, состоящего из сегментов шаблона индекс с добавлением определенного количества помех в обучающую выборку.

На рисунке 1 представлен шаблон, в котором каждая цифра изображается набором отрезков, расположенных в 9 сегментах.

Например:

- Признак "1" обозначает наличие сегмента в соответствующем ячейке.

- Признак "0" обозначает её отсутствие.

Изображение выглядит как линия, диаграмма

Автоматически созданное описание

Рисунок 1 – Шаблон графического изображения цифр

В таблице 1 представлен полный набор цифр, каждая из которых соответствует своему набору признаков. Эти данные используются для формирования обучающей выборки, на которой нейрон и персептрон будут обучаться распознавать цифру "4".

Таблица 1 – Цифры и их представления

|  |  |  |
| --- | --- | --- |
| Цифра | Графическое представление | Двоичное представление |
| 0 |  | 1, 1, 0, 1, 0, 1, 0, 1, 1 |
| 1 |  | 0, 0, 1, 1, 0, 0, 0, 1, 0 |
| 2 |  | 0, 1, 0, 1, 0, 0, 1, 0, 1 |
| 3 |  | 0, 1, 1, 0, 1, 0, 1, 0, 0 |
| 4 |  | 1, 0, 0, 1, 1, 0, 0, 1, 0 |

Продолжение таблицы 1:

|  |  |  |
| --- | --- | --- |
| 5 |  | 1, 1, 0, 0, 1, 0, 0, 1, 1 |
| 6 |  | 0, 0, 1, 0, 1, 1, 0, 1, 1 |
| 7 |  | 0, 1, 1, 0, 0, 1, 0, 0, 0 |
| 8 |  | 1, 1, 0, 1, 1, 1, 0, 1, 1 |
| 9 |  | 1, 1, 0, 1, 1, 0, 1, 0, 0 |

На основе данного шаблона для каждой цифры были сформированы обучающие данные, которые содержат бинарный вектор признаков (массив из 9 значений). Этот вектор однозначно описывает графическое представление каждой цифры.

2 ОСНОВНЫЕ ПОНЯТИЯ ОБ ИСКУССТВЕННОМ НЕЙРОНЕ

Искусственный нейрон — это базовый элемент нейронных сетей, основанный на принципах работы биологических нейронов мозга. В искусственных сетях нейроны обрабатывают информацию и принимают решения, основываясь на поступающих сигналах. В задаче распознавания цифр нейрон обрабатывает входные данные и классифицирует их в соответствии с установленными правилами

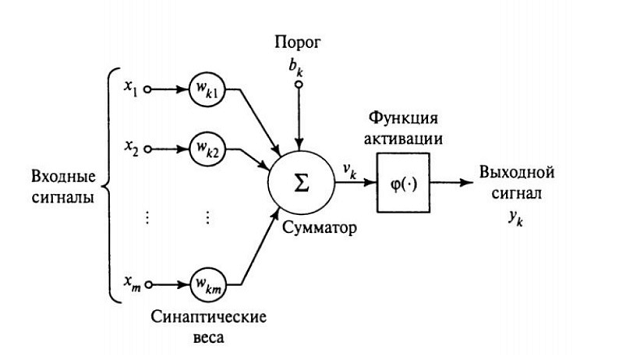


Рисунок 2 – Искусственный нейрон

Основные элементы искусственного нейрона:

1. Входные данные (признаки):

Нейрон получает набор числовых значений, которые представляют собой числовые значения (это сегменты шаблона индекс). В данной задаче каждая цифра от 0 до 9 представлена 9 признаками. Эти признаки поступают на вход нейрона.

2. Веса (коэффициенты):

Каждый входной признак умножается на свой вес. Вес определяет важность признака для принятия решения. Для распознавания цифры "4" более важным признакам присваиваются более высокие веса. Веса помогают нейрону различать ключевые и второстепенные признаки.

3. Функция активации:

После того как входные данные умножены на веса, нейрон суммирует результаты. Полученная сумма проходит через функцию активации, которая решает, активируется нейрон или нет. Простейший пример такой функции —функция с пороговым значением: если сумма превышает определённое значение, заданное ранее, нейрон активируется и выдаёт положительный результат (распознана цифра "4"), в противном случае — отрицательный результат.

4. Выход (результат):

На основе работы функции активации нейрон генерирует выходной сигнал. Этот сигнал выдается пользователю.

3 ОСНОВНЫЕ ПОНЯТИЯ ОБ ОДНОСЛОЙНОМ ПЕРСЕПТРОНЕ

Однослойный персептрон — это простейшая модель нейронной сети. В этой модели искусственные нейроны обрабатывают входные данные и классифицируют их, используя определённые правила.

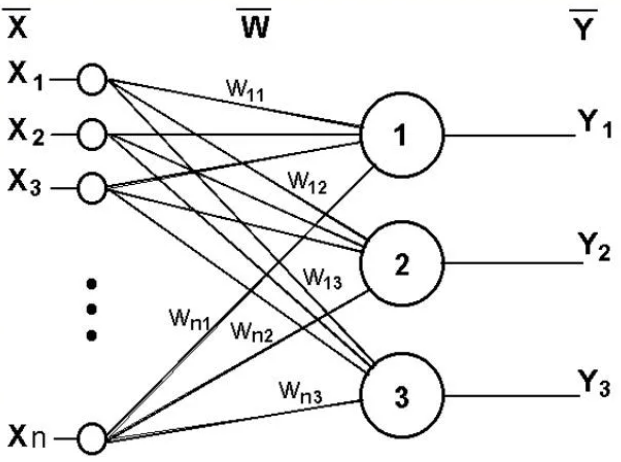


Рисунок 2 – однослойный персептрон

Основные элементы однослойного персептрона:

1. Входные данные (признаки):

Персептрон получает набор числовых значений, которые представляют собой признаки (сегменты шаблона индекс). В задаче распознавания цифр каждая цифра от 0 до 9 представлена 9 признаками. Эти признаки поступают на вход каждого нейрона персептрона.

1. Веса (коэффициенты):

Каждый входной признак умножается на соответствующий ему вес. Веса определяют важность каждого признака для процесса распознавания модели персептрона и выделяют ключевые признаки цифр.

1. Функция активации:

После умножения входных признаков на соответствующие веса, результаты суммируются. Эта сумма передаётся на вход функции активации, которая решает, будет ли персептрон активирован.

1. Выход (результат):

Персептрон генерирует выходной сигнал на основе работы функции активации. В контексте задачи распознавания цифр этот сигнал представляет собой классификацию цифр от 0 до 9.

4 ФОРМИРОВАНИЕ ОБУЧАЮЩЕЙ ВЫБОРКИ

Для успешного обучения искусственного нейрона и однослойного персептрона, способного распознавать цифры по их бинарному представлению, необходимо сформировать обучающую выборку. В нашем случае цифры представлены в виде набора признаков. Эти признаки в виде бинарных значений описывают наличие или отсутствие сегментов шаблона индекс в изображении цифры.

Используя набор данных, нейрон и персептрон смогут по входным признакам корректно определять, какая цифра перед ним. Обучающая выборка создаётся путём генерации множества примеров этих векторов и передачи их в алгоритм обучения нейрона и персептрона.

Этот процесс позволяет обучить систему эффективно распознавать цифру "4", основываясь на ее уникальных признаках.

4.1 Добавление помех в обучающую выборку

Для создания помех в обучающую выборку вводится некоторое количество ошибок. Оно определяется пользователем и варьируется от 0 до 9. Ошибка является инвертированием определенного бита бинарного вектора.

Биты для инверсии выбираются случайным образом в каждом элементе обучающей выборки. Программа следит, чтобы в одной и той же цифре в обучающей выборке ошибки не находились на одинаковых местах, то есть все представления одной и той же цифры отличаются друг от друга. Всё это симулирует ситуацию, когда данные, необходимые для обучения нейрона и персептрона, повреждены.

5 ОБУЧЕНИЕ ИСКУССТВЕННОГО НЕЙРОНА ПРИ ЗАДАННОМ КОЛИЧЕСТВЕ ПОМЕХ

После того, как обучающая выборка была сформирована и в нее было добавлено определенное количество ошибок, запускается алгоритм обучения на распознавание цифры.

Эпохой называется полное прохождение алгоритма по обучающей выборке. Для корректного распознавания программе может потребоваться одна или более эпоха.

На каждой эпохе вычисляется сумма весов для каждого элемента обучающей выборки. В зависимости от того, верно ли классифицируется элемент, то есть является ли объект цифрой "4", происходит корректировка весов:

1. Если объект является цифрой "4", но сумма меньше или равна пороговому значению, веса увеличиваются.
2. Если объект не является цифрой "4", но взвешенная сумма больше порога, веса уменьшаются.

В конце каждой эпохи выводится точность модели, то есть процент правильно классифицированных объектов.

6 ОБУЧЕНИЕ ОДНОСЛОЙНОГО ПЕРСЕПТРОНА ПРИ ЗАДАННОМ КОЛИЧЕСТВЕ ПОМЕХ

На каждой эпохе вычисляются суммы весов для каждого элемента обучающей выборки, а также находятся максимумы среди этих сумм.

В зависимости от того, верно ли классифицируется элемент, происходит корректировка весов:

1. Если максимум относится к распознаваемой цифре, и он один, то веса

не изменяются и алгоритм начинает работать со следующей цифрой в выборке.

1. Если максимум не относится к распознаваемой цифре, то веса всех максимумов, кроме распознаваемой цифры, уменьшаются, а у распознаваемой – увеличиваются.

В конечном итоге это позволяет получить новую матрицу весов. В конце каждой эпохи выводится точность модели, то есть процент правильно классифицированных объектов.

7 РЕЗУЛЬТАТ ОБУЧЕНИЯ ИСКУССТВЕННОГО НЕЙРОНА И ОДНОСЛОЙНОГО ПЕРСЕПТРОНА

В результате выполнения программы были зафиксированы итерации по обучению нейрона и персептрона для распознавания цифры "4" на основе ее признаков.

Обучение происходило с добавлением в обучающую выборку определенного количества ошибок в диапазоне от 0 до 9.

7.1 Результат обучения без добавления ошибок

В данном случае обучающая выборка никак не менялась, то есть бинарные представления цифр были в изначальном виде, основанном на сегментах шаблона индекс.

Результаты данного варианта обучения представлены на рисунке 3:

Рисунок 3 – График точности обучения без добавления ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 100%, для этого ему потребовалось 2 эпохи.

Однослойный персептрон обучился распознавать цифру “4” с точностью 100%, для этого ему потребовалось 2 эпохи.

Конечный вес:

Нейрон Персептрон

16 1 0 11 2 -1 5 10 -4 -2 8 6 6 -2 -4 4 5 5

10 0 -2 4 -5 -1 4 4 -3

7.2 Результат обучения с добавлением 1 ошибки

В данном случае в обучающую выборку была добавлена 1 ошибка в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 4:

Рисунок 4 – График точности обучения с добавлением 1 ошибки

Искусственный нейрон обучился распознавать цифру “4” с точностью 100%, для этого ему потребовалось 4 эпохи.

Однослойный персептрон обучился распознавать цифру “4” с точностью 95,6%, для этого ему потребовалось 6 эпох.

Конечный вес:

Нейрон Персептрон

13 -10 -4 13 12 -7 -5 12 -9 0 10 8 4 -6 1 9 5 8

8 -2 -4 6 -1 -6 -1 4 -6

7.3 Результат обучения с добавлением 2 ошибок

В данном случае в обучающую выборку были добавлены 2 ошибки в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 5:

Рисунок 5 – График точности обучения с добавлением 2 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 92,8%, для этого ему потребовалось 19 эпох.

Однослойный персептрон обучился распознавать цифру “4” с точностью 88,8%, для этого ему потребовалось 7 эпохи.

Конечный вес:

Нейрон Персептрон

17 -12 -4 15 17 -7 -6 13 -9 1 9 4 3 -5 -1 6 4 4

7 -1 0 7 -2 -4 2 5 -2

7.4 Результат обучения с добавлением 3 ошибок

В данном случае в обучающую выборку были добавлены 3 ошибки в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 6:

Рисунок 6 – График точности обучения с добавлением 3 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 85,4%, для этого ему потребовалось 2 эпохи.

Однослойный персептрон обучился распознавать цифру “4” с точностью 81,6%, для этого ему потребовалось 5 эпох.

Конечный вес:

Нейрон Персептрон

13 -4 -1 9 9 1 0 8 0 3 7 3 4 -5 -2 5 6 2

5 1 1 6 -2 -3 3 3 0

7.5 Результат обучения с добавлением 4 ошибок

В данном случае в обучающую выборку были добавлены 4 ошибки в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 7:

Рисунок 7 – График точности обучения с добавлением 4 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 85,2%, для этого ему потребовалось 1 эпоха.

Однослойный персептрон обучился распознавать цифру “4” с точностью 81,8%, для этого ему потребовалось 3 эпохи.

Конечный вес:

Нейрон Персептрон

13 5 4 7 1 5 9 8 0 1 5 0 5 -1 -1 5 5 1

7 3 4 5 -6 -4 3 4 1

7.6 Результат обучения с добавлением 5 ошибок

В данном случае в обучающую выборку были добавлены 5 ошибок в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 8:

Рисунок 8 – График точности обучения с добавлением 5 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 86%, для этого ему потребовалось 1 эпоха.

Однослойный персептрон обучился распознавать цифру “4” с точностью 79,8%, для этого ему потребовалось 5 эпох.

Конечный вес:

Нейрон Персептрон

2 13 5 5 2 9 7 6 6 5 2 3 7 -4 -3 4 4 2

3 6 1 3 -3 -2 4 5 0

7.7 Результат обучения с добавлением 6 ошибок

В данном случае в обучающую выборку были добавлены 6 ошибок в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 9:

Рисунок 9 – График точности обучения с добавлением 6 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 86,2%, для этого ему потребовалось 20 эпох.

Однослойный персептрон обучился распознавать цифру “4” с точностью 82,2%, для этого ему потребовалось 3 эпохи.

Конечный вес:

Нейрон Персептрон

-4 13 3 -6 1 9 5 -3 14 5 3 3 8 -2 -2 5 5 -1

3 5 1 2 -5 -3 3 4 3

7.8 Результат обучения с добавлением 7 ошибок

В данном случае в обучающую выборку были добавлены 7 ошибок в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 10:

Рисунок 10 – График точности обучения с добавлением 7 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 89%, для этого ему потребовалось 15 эпох.

Однослойный персептрон обучился распознавать цифру “4” с точностью 84,2%, для этого ему потребовалось 5 эпох.

Конечный вес:

Нейрон Персептрон

-5 14 8 -7 -7 10 9 -8 13 9 1 2 8 -2 -4 3 6 -2

-1 7 2 2 -5 -1 5 3 4

7.9 Результат обучения с добавлением 8 ошибок

В данном случае в обучающую выборку были добавлены 8 ошибок в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 11:

Рисунок 11 – График точности обучения с добавлением 8 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 100%, для этого ему потребовалось 3 эпохи.

Однослойный персептрон обучился распознавать цифру “4” с точностью 91%, для этого ему потребовалось 7 эпох.

Конечный вес:

Нейрон Персептрон

-4 9 9 -4 -9 9 9 -3 9 12 1 3 8 3 -3 5 11 1

-4 7 1 2 -10 -2 3 -2 1

7.10 Результат обучения с добавлением 9 ошибок

В данном случае в обучающую выборку были добавлены 9 ошибок в каждый ее элемент.

Результаты данного варианта обучения представлены на рисунке 12:

Рисунок 12 – График точности обучения с добавлением 9 ошибок

Искусственный нейрон обучился распознавать цифру “4” с точностью 100%, для этого ему потребовалось 2 эпохи.

Однослойный персептрон обучился распознавать цифру “4” с точностью 100%, для этого ему потребовалось 2 эпохи.

Конечный веса:

Нейрон Персептрон

8 11 4 3 -9 7 11 0 4 3 3 4 10 3 -7 4 9 2

5 5 0 0 -10 2 4 0 0

8 ИТОГОВАЯ ТОЧНОСТЬ ОБУЧЕНИЯ ИСКУССТВЕННОГО НЕЙРОНА И ОДНОСЛОЙНОГО ПЕРСЕПТРОНА

После обучения искусственного нейрона и однослойного персептрона на распознавание цифры "4" с добавлением определенного количества ошибок в диапазоне от 0 до 9 были получены следующие результаты.

График зависимости точности от количества ошибок представлен на рисунке 13:

Рисунок 13 – График зависимости точности от количества ошибок

ЗАКЛЮЧЕНИЕ

По завершении обучения искусственный нейрон и однослойный персептрон успешно обучились распознавать цифру "4". Распознавание происходило путём корректировки весов в процессе обучения, что продемонстрировало успешную работу алгоритма.

Также в процессе обучения в обучающую выборку добавлялось определенное количество ошибок в диапазоне от 0 до 9.

Анализируя результаты обучения, можно сделать вывод о том, что искусственный нейрон лучше обучился распознавать цифру "4", чем однослойный персептрон.

Результаты подтверждает, что искусственный нейрон и однослойный персептрон, обучаемые на принципах простых механизмов корректировки весов, могут быть обучены распознавать необходимые пользователю символы по заданным признакам.

ПРИЛОЖЕНИЕ А

Lab5N.cpp

#include <iostream>

#include <algorithm>

#include <ctime>

#include <cstring>

using namespace std;

int main()

{

    int Q=35;

    int Omega[9]={8,7,0,3,-9,3,7,0,0};

    int OmegaFirst[9]={8,7,0,3,-9,3,7,0,0};

    int ves[500][9];

    int mycifra[9]={1,0,0,1,1,0,0,1,0}; //Моя цифра 4 - по заданию

    int key;

    int maxeras=20;

    int sumCheck=0;

    int cifri[10][10]=

    {

        {1,1,0,1,0,1,0,1,1,0}, //0

        {0,0,1,1,0,0,0,1,0,1}, //1

        {0,1,0,1,0,0,1,0,1,2}, //2

        {0,1,1,0,1,0,1,0,0,3}, //3

        {1,0,0,1,1,0,0,1,0,4}, //4

        {1,1,0,0,1,0,0,1,1,5}, //5

        {0,0,1,0,1,1,0,1,1,6}, //6

        {0,1,1,0,0,1,0,0,0,7}, //7

        {1,1,0,1,1,1,0,1,1,8}, //8

        {1,1,0,1,1,0,1,0,0,9}  //9

    };

    int randommassiv[500]=

    {

        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

        1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,

        2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,

        3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,

        4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,

        5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,

        6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,

        7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,

        8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,

        9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9

    };

    int viborka[500][10];

    int kolvo=0;

    random\_shuffle(randommassiv, randommassiv+500);

    for (int i=0;i<500;i++)

    {

        for (int j=0;j<10;j++)

        {

            viborka[i][j]=cifri[randommassiv[i]][j];

        }

    }

    cout << "Author - Moskalenko Vitaly Alexandrovich UIB-311" << endl;

    int KolvoOshibki;

    cout << "Enter the number of errors: ";

    cin >> KolvoOshibki;

    for (int i=0;i<500;i++)

    {

        int OshibkiAdd=0;

        bool IzmeneniyBit[9]={false};

        while (OshibkiAdd<KolvoOshibki)

        {

            int SluchBit=rand()%9;

            if (!IzmeneniyBit[SluchBit])

            {

                viborka[i][SluchBit]=!viborka[i][SluchBit];

                IzmeneniyBit[SluchBit]=true;

                OshibkiAdd++;

            }

        }

    }

    cout << "Training sample: " << endl;

    for (int i=0;i<500;i++)

    {

        cout << i+1 << " | ";

        for (int j=0;j<10;j++)

        {

            cout << viborka[i][j] << “ “;

        }

        cout << "|";

        cout << endl;

    }

    while (kolvo<maxeras)

    {

        kolvo++;

        int IzmeneniyVes=0;

        int Sovpad=0;

        cout << "Epoha " << kolvo << " :";

        cout << endl;

        cout << "Omega1: ";

        for (int i=0;i<9;i++)

        {

            cout << Omega[i] << “ “;

        }

        cout << endl;

        for (int i=0;i<500;i++)

        {

            int sum=0;

            for (int j=0;j<9;j++)

            {

                ves[i][j]=Omega[j]\*viborka[i][j];

                sum+=ves[i][j];

            }

            if ((viborka[i][9]==4&&sum>Q)||(viborka[i][9]!=4&&sum<=Q))

            {

                Sovpad++;

            }

            if ((viborka[i][9]==4)&&(sum<=Q))

            {

                for (int x=0;x<9;x++)

                {

                    Omega[x]+=viborka[i][x];

                }

                IzmeneniyVes++;

            }

            if ((viborka[i][9]!=4) && (sum > Q))

            {

                for (int x = 0; x < 9; x++)

                {

                    Omega[x] -= viborka[i][x];

                }

                IzmeneniyVes++;

            }

        }

        cout << "Omega2: ";

        for (int i=0;i<9;i++)

        {

            cout << Omega[i] << “ “;

        }

        cout << endl;

        cout << "Number of changes: " << IzmeneniyVes << “ “;

        cout << endl;

        float tochnost=static\_cast<float>(Sovpad)/500\*100;

        cout << "Accuracy: " << tochnost << "%" << endl;

        cout << endl;

    }

    cout << "Final Results: " << endl;

    cout << "First Omega: ";

    for (int i=0;i<9;i++)

    {

        cout << OmegaFirst[i] << “ “;

    }

    cout << endl;

    cout << "Last Omega: ";

    for (int i=0;i<9;i++)

    {

        cout << Omega[i] << “ “;

    }

    cout << endl;

    cout << "Number of eras - " << kolvo;

    cout << endl;

    return 0;

}

ПРИЛОЖЕНИЕ Б

Lab5P.cpp

#include <iostream>

#include <algorithm>

#include <ctime>

#include <cstring>

using namespace std;

int main()

{

    int maxeras=20;

    int kolich=0;

    int sums[2]={0};

    int cifri[10][10]=

    {

        {1,1,0,1,0,1,0,1,1,0}, //0

        {0,0,1,1,0,0,0,1,0,1}, //1

        {0,1,0,1,0,0,1,0,1,2}, //2

        {0,1,1,0,1,0,1,0,0,3}, //3

        {1,0,0,1,1,0,0,1,0,4}, //4

        {1,1,0,0,1,0,0,1,1,5}, //5

        {0,0,1,0,1,1,0,1,1,6}, //6

        {0,1,1,0,0,1,0,0,0,7}, //7

        {1,1,0,1,1,1,0,1,1,8}, //8

        {1,1,0,1,1,0,1,0,0,9}  //9

    };

    int Omegi[2][10] =

    {

        {0,1,4,7,2,-8,1,9,2,0},

        {8,7,0,3,-9,3,7,0,0,4}

    };

    int randommassiv[500]=

    {

        0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

        1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,

        2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,

        3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,

        4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,

        5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,

        6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,

        7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,

        8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,

        9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9

    };

    random\_shuffle(randommassiv, randommassiv + 500);

    int viborka[500][10];

    for (int i=0;i<500;i++)

    {

        for (int j=0;j<10;j++)

        {

            viborka[i][j]=cifri[randommassiv[i]][j];

        }

    }

    cout << "Author - Moskalenko Vitaly Alexandrovich UIB-311" << endl;

    int KolvoOshibki;

    cout << "Enter the number of errors: ";

    cin >> KolvoOshibki;

    for (int i=0;i<500;i++)

    {

        int OshibkiAdd=0;

        bool IzmeneniyBit[9]={false};

        while (OshibkiAdd<KolvoOshibki)

        {

            int SluchBit=rand()%9;

            if (!IzmeneniyBit[SluchBit])

            {

                viborka[i][SluchBit]=!viborka[i][SluchBit];

                IzmeneniyBit[SluchBit]=true;

                OshibkiAdd++;

            }

        }

    }

    cout << "Training sample: " << endl;

    for (int i=0;i<500;i++)

    {

        cout << i+1 << " | ";

        for (int j=0;j<10;j++)

        {

            cout << viborka[i][j] << “ “;

        }

        cout << "|";

        cout << endl;

    }

    while (kolich<maxeras)

    {

        kolich++;

        int sovpad=0;

        for (int i=0;i<500;i++)

        {

            for (int j=0;j<2;j++)

            {

                sums[j] = 0;

                for (int x=0;x<9;x++)

                {

                    sums[j]+=Omegi[j][x]\*viborka[i][x];

                }

            }

            int max=\*max\_element(sums, sums+2);

            int kolvomax=0;

            for (int i=0;i<2;i++)

            {

                if (sums[i]==max)

                {

                    kolvomax++;

                }

            }

            int PravKlass;

            if (viborka[i][9]==4)

            {

                PravKlass=1;

            }

            else

            {

                PravKlass=0;

            }

            if ((sums[PravKlass]==max)&&(kolvomax==1))

            {

                sovpad++;

            }

            else

            {

                for (int x=0;x<2;x++)

                {

                    if ((sums[x]==max)&&(x!=PravKlass))

                    {

                        for (int j=0;j<9;j++)

                        {

                            Omegi[x][j]-=viborka[i][j];

                        }

                    }

                }

                for (int j=0;j<9;j++)

                {

                    Omegi[PravKlass][j]+=viborka[i][j];

                }

            }

        }

        cout << endl;

        cout <<  "Era: " << kolich << endl;

        for (int i=0;i<2;i++)

        {

            for (int j=0;j<9;j++)

            {

                cout << Omegi[i][j] << “ “;

            }

            cout << endl;

        }

        double tochnost=(double)sovpad/500\*100;

        cout << "Accuracy: "  << tochnost << "%" << endl;

    }

    cout << endl;

    return 0;

}