**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 №4**

**«Обучение 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 3](#_Toc178117551)

[2 ОСНОВНЫЕ ПОНЯТИЯ ОБ ОДНОСЛОЙНОМ ПЕРСЕПТРОНЕ 6](#_Toc178117552)

[3 ФОРМИРОВАНИЕ ОБУЧАЮЩЕЙ ВЫБОРКИ 8](#_Toc178117553)

[3.1 Добавление помех в обучающую выборку 8](#_Toc178117554)

[4 ОБУЧЕНИЕ ИСКУССТВЕННОГО НЕЙРОНА ПРИ ЗАДАННОМ КОЛИЧЕСТВЕ ПОМЕХ 9](#_Toc178117555)

[5 РЕЗУЛЬТАТ ОБУЧЕНИЯ ОДНОСЛОЙНОГО ПЕРСЕПТРОНА 11](#_Toc178117557)

[5.1 Результат обучения без добавления ошибок 11](#_Toc178117558)

[5.2 Результат обучения с добавлением 1 ошибки 13](#_Toc178117559)

[5.3 Результат обучения с добавлением 2 ошибок 14](#_Toc178117560)

[5.4 Результат обучения с добавлением 3 ошибок 16](#_Toc178117561)

[5.5 Результат обучения с добавлением 4 ошибок 17](#_Toc178117562)

[5.6 Результат обучения с добавлением 5 ошибок 19](#_Toc178117563)

[5.7 Результат обучения с добавлением 6 ошибок 20](#_Toc178117564)

[5.8 Результат обучения с добавлением 7 ошибок 22](#_Toc178117565)

[5.9 Результат обучения с добавлением 8 ошибок 23](#_Toc178117566)

[5.9 Результат обучения с добавлением 9 ошибок 25](#_Toc178117567)

[6 ИТОГОВАЯ ТОЧНОСТЬ ОБУЧЕНИЯ ИСКУССТВЕННОГО НЕЙРОНА 27](#_Toc178117568)

[ЗАКЛЮЧЕНИЕ 28](#_Toc178117569)

[ПРИЛОЖЕНИЕ 29](#_Toc178117570)

1 ЗАДАНИЕ

Необходимо разработать программу, которая реализует обучение однослойного персептрона для распознавания цифр от 0 до 9 на основе их графических представлений, состоящих 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 персептрон будет обучаться распознавать цифры.

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 ОДНОСЛОЙНОМ ПЕРСЕПТРОНЕ

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

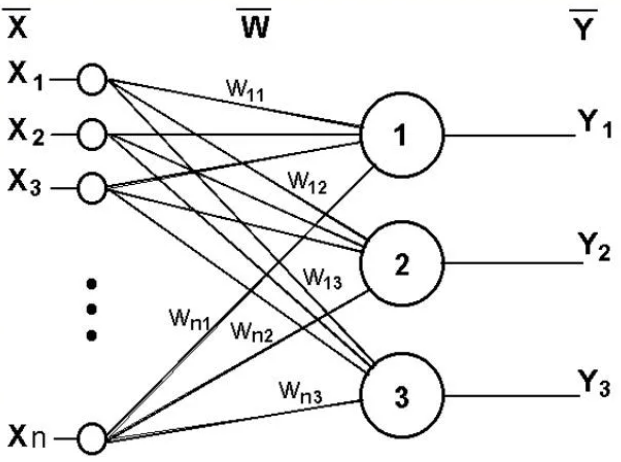


Рисунок 2 – однослойный персептрон

Основные элементы однослойного персептрона:

* 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.

* 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.

* Функция активации:

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

* 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.

3 ФОРМИРОВАНИЕ ОБУЧАЮЩЕЙ ВЫБОРКИ

Для успешного обучения 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, персептрон сможет по входным признакам корректно определять, какая цифра перед ним. Обучающая выборка создаётся путём генерации множества примеров этих векторов и передачи их в алгоритм обучения персептрона.

Этот процесс позволяет обучить систему эффективно распознавать цифры от 0 до 9, основываясь на уникальных признаках каждой цифры.

3.1 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 однослойного персептрона, повреждены.

4 ОБУЧЕНИЕ ОДНОСЛОЙНОГО ПЕРСЕПТРОНА ПРИ ЗАДАННОМ КОЛИЧЕСТВЕ ПОМЕХ

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

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

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

Начальная матрица весов выглядит следующим образом:

{0,1,4,7,2,-8,1,9,2}

{7,3,9,4,9,0,1,7,4}

{1,7,9,2,6,0,7,6,0}

{3,9,8,6,-9,6,2,0,0}

{8,7,0,3,-9,3,7,0,0}

{9,0,8,8,4,7,4,9,3}

{2,8,7,4,-1,7,9,1,1}

 {9,8,4,0,8,7,0,8,7}

 {5,7,8,5,7,8,4,0,2}

 {9,4,8,5,7,1,9,-8,5}

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.

5 РЕЗУЛЬТАТ ОБУЧЕНИЯ ОДНОСЛОЙНОГО ПЕРСЕПТРОНА

В результате выполнения программы были зафиксированы итерации по обучению персептрона для распознавания цифр от 0 до 9 на основе их признаков.

Основной целью обучения было создание новой матрицы весов, в которой при подсчете сумм весов максимум этих сумм должен был быть только у распознаваемой цифры.

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

5.1 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 100%, для этого ему потребовалось 2 эпохи.

Конечная матрица весов:

3 4 1 8 -2 -2 0 9 5

3 2 11 4 4 -1 1 5 2

1 7 5 6 2 0 7 6 4

3 10 10 5 -2 1 8 0 -1

12 3 0 7 -2 0 6 5 -3

8 2 6 0 4 0 4 6 3

1 2 7 2 2 7 7 4 3

4 9 8 -2 3 9 0 1 0

5 6 5 6 6 8 3 -1 2

10 2 6 5 6 1 7 -8 4

5.2 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 94%, для этого ему потребовалось 10 эпохи.

Конечная матрица весов:

7 9 1 8 -10 9 -1 4 2

2 1 11 8 -2 -1 1 10 1

-3 8 5 8 -5 -2 9 -4 7

1 7 12 -2 6 0 10 0 -2

12 -7 1 9 7 -4 -2 4 -7

8 7 4 -10 5 -5 2 6 7

-1 -2 12 -1 4 7 5 4 7

4 10 11 -2 -3 11 0 -1 -2

2 2 4 7 3 6 3 3 5

10 4 2 8 6 0 8 -5 -5

5.3 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 54,6%, для этого ему потребовалось 15 эпох.

Конечная матрица весов:

-6 -11 -7 -4 -16 -7 -7 -10 -7

-15 -17 5 -1 -17 -12 -7 -1 -16

-14 -5 -9 1 -21 -9 4 -16 -2

-16 -4 6 -15 -8 -11 6 -17 -19

-2 -19 -6 0 -8 -12 -7 -5 -14

-3 -7 -8 -14 -10 -9 -6 -7 -3

-12 -20 1 -14 -10 -3 -2 -9 -6

-10 -7 3 -15 -20 1 -2 -18 -12

-9 -11 -9 -5 -12 -5 -8 -10 -6

-6 -6 -3 -4 -11 -11 2 -16 -17

5.4 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 15,6%, для этого ему потребовалось 11 эпох.

Конечная матрица весов:

-36 -43 -34 -40 -54 -34 -34 -48 -41

-38 -46 -28 -39 -53 -37 -33 -46 -42

-36 -40 -33 -39 -53 -35 -31 -50 -39

-37 -42 -31 -41 -52 -35 -30 -51 -42

-34 -45 -34 -38 -49 -36 -36 -48 -42

-34 -43 -33 -41 -51 -37 -34 -48 -38

-37 -44 -30 -43 -52 -32 -35 -48 -39

-40 -40 -30 -45 -55 -33 -33 -50 -44

-35 -43 -33 -41 -50 -33 -35 -50 -44

-38 -41 -33 -41 -52 -34 -34 -50 -42

5.5 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 10,4%, для этого ему потребовалось 3 эпохи.

Конечная матрица весов:

-13 -13 -12 -15 -18 -14 -16 -16 -18

-16 -13 -9 -16 -15 -15 -14 -14 -17

-13 -12 -11 -13 -18 -14 -14 -13 -19

-13 -13 -11 -17 -18 -14 -13 -15 -18

-11 -14 -10 -14 -19 -15 -15 -15 -16

-15 -12 -13 -17 -18 -13 -15 -14 -16

-14 -14 -10 -15 -17 -13 -15 -16 -16

-15 -13 -10 -15 -18 -13 -14 -15 -18

-14 -13 -12 -14 -17 -14 -16 -16 -16

-12 -12 -11 -16 -16 -15 -16 -16 -17

5.6 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 9,4%, для этого ему потребовалось 8 эпох.

Конечная матрица весов:

-47 -40 -45 -51 -45 -53 -54 -49 -41

-45 -41 -46 -50 -44 -53 -51 -49 -42

-44 -42 -45 -47 -43 -54 -53 -46 -43

-44 -42 -46 -50 -46 -54 -55 -47 -43

-47 -38 -44 -50 -45 -52 -54 -47 -44

-44 -41 -45 -50 -44 -54 -53 -48 -44

-45 -40 -45 -47 -44 -53 -52 -49 -43

-44 -40 -46 -48 -44 -56 -54 -49 -42

-45 -40 -46 -51 -44 -53 -54 -49 -43

-46 -41 -44 -49 -47 -55 -56 -47 -43

5.7 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 18,4%, для этого ему потребовалось 9 эпох.

Конечная матрица весов:

-30 -27 -35 -31 -33 -42 -41 -31 -37

-27 -23 -37 -31 -34 -41 -41 -32 -36

-24 -26 -37 -33 -32 -37 -44 -29 -40

-28 -28 -39 -29 -35 -40 -45 -31 -35

-28 -26 -37 -34 -36 -39 -43 -31 -34

-28 -28 -33 -31 -36 -38 -40 -33 -38

-26 -24 -37 -29 -37 -40 -43 -30 -40

-26 -27 -40 -30 -31 -43 -44 -32 -39

-26 -26 -36 -32 -36 -42 -42 -31 -37

-28 -27 -37 -34 -36 -39 -44 -31 -38

5.8 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 51,8%, для этого ему потребовалось 5 эпох.

Конечная матрица весов:

-5 0 2 -2 3 -6 3 -8 -6

4 8 -6 -3 0 -2 -2 -5 -2

3 0 0 -4 -2 0 -9 2 -8

4 2 -8 7 -10 -2 -10 5 3

-4 9 0 -3 -5 -1 -1 -5 2

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

4 11 -4 8 -9 -10 -1 -2 -7

2 2 -6 6 0 -10 -5 0 -1

-4 -3 8 -4 -2 -6 3 -2 -8

-5 0 0 -3 -6 2 -13 5 0

5.9 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 94,4%, для этого ему потребовалось 11 эпох.

Конечная матрица весов:

-1 0 9 1 8 -4 7 1 -2

11 9 -3 -1 7 3 2 -1 5

9 5 8 2 6 3 -1 8 -3

10 4 -1 9 -5 8 -4 5 4

-1 11 5 1 -5 6 2 -3 8

3 2 5 14 -1 7 1 1 -3

10 10 0 8 -3 -2 5 1 -3

7 3 2 5 7 -4 5 7 5

2 3 12 0 1 -6 9 3 -2

0 1 9 2 -2 5 -5 7 8

5.9 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

Персептрон обучился распознавать цифры от 0 до 9 с точностью 100%, для этого ему потребовалось 2 эпохи.

Конечная матрица весов:

-1 1 8 6 8 -8 4 8 1

7 7 5 2 8 1 2 2 5

5 7 7 2 8 2 3 8 -2

7 8 5 9 -9 7 0 3 2

5 8 4 0 -9 7 8 0 4

6 0 7 8 3 5 5 3 -3

5 9 4 6 -1 4 9 1 -1

6 5 1 2 7 -1 1 7 5

5 6 9 4 4 6 5 0 1

6 0 10 3 4 5 2 -2 9

6 ИТОГОВАЯ ТОЧНОСТЬ ОБУЧЕНИЯ ОДНОСЛОЙНОГО ПЕРСЕПТРОНА

После обучения однослойного персептрона на распознавание цифр от 0 до 9 с добавлением определенного количества ошибок в диапазоне от 0 до 9 были получены все значения точности для каждой цифры.

График зависимости точности от количества ошибок представлен на рисунке 13:

Рисунок 13 – График зависимости точности от количества ошибок

ЗАКЛЮЧЕНИЕ

По завершении обучения однослойный персептрон успешно обучился распознавать цифры от 0 до 9. 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.

Этот результат подтверждает, что однослойный персептрон, обучаемый на принципах простых механизмов корректировки весов, может быть обучен распознавать необходимые пользователю символы по заданным признакам.

ПРИЛОЖЕНИЕ

#include <iostream>

#include <algorithm>

#include <ctime>

#include <cstring>

using namespace std;

int main()

{

    int maxeras=20;

    int kolich=0;

    int sums[10]={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[10][10]=

    {

        {0,1,4,7,2,-8,1,9,2,0},

        {7,3,9,4,9,0,1,7,4,1},

        {1,7,9,2,6,0,7,6,0,2},

        {3,9,8,6,-9,6,2,0,0,3},

        {8,7,0,3,-9,3,7,0,0,4},

        {9,0,8,8,4,7,4,9,3,5},

        {2,8,7,4,-1,7,9,1,1,6},

        {9,8,4,0,8,7,0,8,7,7},

        {5,7,8,5,7,8,4,0,2,8},

        {9,4,8,5,7,1,9,-8,5,9}

    };

    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

    };

    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];

        }

    }

    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<10;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+10);

            int kolvomax=0;

            for (int i=0;i<10;i++)

            {

                if (sums[i]==max)

                {

                    kolvomax++;

                }

            }

            if ((sums[viborka[i][9]]==max)&&(kolvomax==1))

            {

                sovpad++;

            }

            else

            {

                for (int x=0;x<10;x++)

                {

                    if ((sums[x]==max)&&(x!=viborka[i][9]))

                    {

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

                        {

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

                        }

                    }

                }

                for (int j=0;j<9;j++)

                {

                    Omegi[viborka[i][9]][j]+=viborka[i][j];

                }

            }

        }

        cout << endl;

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

        for (int i=0;i<10;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;

}