Министерство науки и высшего образования Российской Федерации (Минобрнауки России)

Федеральное государственное бюджетное образовательное учреждение высшего образования «Ивановский государственный энергетический университет имени В.И. Ленина»

Кафедра программного обеспечения компьютерных систем

2750

ПЕРЦЕПТРОНЫ

Методические указания

Иваново 2021

Составитель С.Г. СИДОРОВ

Редактор А.Б. ГНАТЮК

Методические указания предназначены для подготовки к лабораторным работам и оказания помощи в самостоятельном выполнении индивидуальных заданий. Рассматриваются вопросы программной реализации перцептронов, предназначенных для распознавания как простых, так и сложных образов на базе технологии искусственных нейронных сетей. Приводятся примеры на языках программирования Pascal и C++.

Для студентов, изучающих дисциплины "Нейрокомпьютерные системы и параллельные вычисления", "Разработка интеллектуальных систем", "Системы искусственного интеллекта", "Нейрокомпьютерные системы".

Рецензент
кафедра программного обеспечения компьютерных систем ФГБОУ ВО «Ивановский государственный энергетический университет имени В.И. Ленина»

Содержание

Лабораторная работа №1
Распознавание двух образов

Цель работы:

- знакомство с принципами реализации перцептрона, как устройства для распознавания образов;
- изучение вопросов обучения и контроля обученности перцептрона;
- изучение характеристик и возможностей перцептрона;
- определение параметров, влияющих на обучаемость.

Структура перцептрона

Перцептрон – это устройство для распознавания образов, предложенное в 1957 году Френком Розенблаттом.

Перцептрон классифицируется как однослойная искусственная нейронная сеть, использующая дельта-метод обучения с учителем. Под числом слоев в данном случае понимается количество решающих слоев, на которых может производиться обучение.

В целом перцептрон включает три основных слоя: рецепторный, ассоциативный и эффекторный (рис. 1).

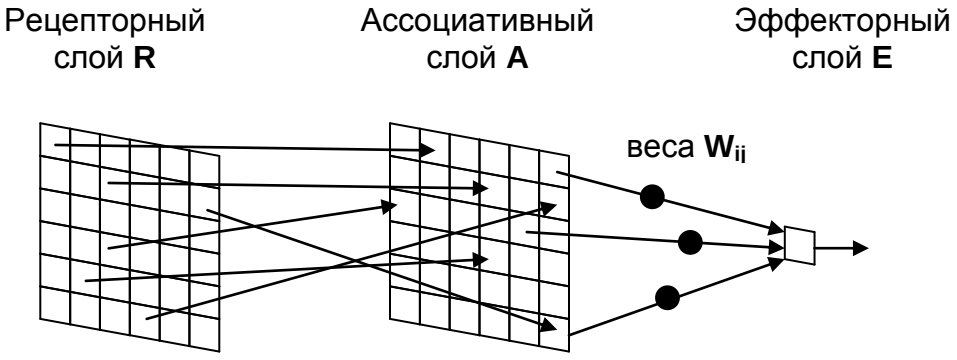

Рис. 1. Структура однослойного перцептрона

На рецепторном слое формируется изображение для распознавания. Возбужденные рецепторы, на которые падает изображение, переводятся в состояние "1". Остальные рецепторы остаются в состоянии "0".

По связям, сгенерированным случайным образом, сигналы "1" от клеток рецепторного слоя попадают на ассоциативный слой, где накапливаются, и при превышении заданного порога приводят клетки ассоциативного слоя в возбужденное состояние, т.е. "1". Невозбужденные клетки остаются в состоянии "0".

$$U^A_{вх} = \sum_{i=1}^{n} U^R_i \tag{1}$$

$$U^A_{вых} = \begin{cases} 1, \text{ если } U^A_{вх} > \Theta \\ 0, \text{ если } U^A_{вх} \le \Theta \end{cases} \tag{2}$$

где U^R_i – входные значения от рецепторов (0 или 1);

$U^A_{вх}$ – значение на входе в ассоциативной клетки;

Θ – пороговое значение срабатывания ассоциативной клетки;

$U^A_{вых}$ – значение на выходе из ассоциативной клетки.

Сигналы "1" передаются, с учетом веса связи, на эффекторный слой. На эффекторном слое формируются возбуждающие, либо тромозящие сигналы, на основе оценки которых принимается решение о распознанном образе. Обычно в качестве порога выступает нулевое значение.

$$U^E_{вх} = \sum_{j=1}^{m} W_j \cdot U^A_{вых,j} \tag{3}$$

$$U^E_{вых} = \begin{cases} 1, \text{ если } U^E_{вх} > 0 \\ 0, \text{ если } U^E_{вх} \le 0 \end{cases} \tag{4}$$

где W_j – вес j-й связи;

$U^A_{вых,i}$ – выходное значение j-й клетки ассоциативного слоя;

$U^E_{вх}$ – значение на входе в эффекторной клетке;

$U^E_{вых}$ – значение на выходе эффекторной клетки.

Чтобы перцептрон мог распознавать образы необходимо провести процедуру его обучения. Под обучением понимается подбор таких значений весов **W**, при которых отклик системы соответствует образу, подаваемому на рецепторный слой. Подбор весов, в ходе обучения,

происходит таким образом, чтобы минимизировать на каждом из шагов ошибку распознавания. Важно при этом вести обучение не вообще всех весов, а только тех из них, которые «виноваты» в формировании ошибки.

$$\begin{cases} W_j = W_j + \Delta W, \text{если } U^A_{\text{вых}} = 1 \text{ и } \text{dic} = 0 \\ W_j = W_j - \Delta W, \text{если } U^A_{\text{вых}} = 1 \text{ и } \text{dic} = 1 \end{cases} \quad (5)$$

где W_j – вес j-й связи;

ΔW – вес обучения;

$U^A_{\text{вых}}$ – значение на выходе ассоциативной клетки;

dic – тип распознаваемого образа (0-нолик, 1-крестик).

Пример реализации перцептрона на языке С++

```
//////////////////////////////////////////////////////////////
//  Перцептрон. Распознавание 2-х образов: крестика и нолика  //
//            2020 (c) Сидоров Сергей Георгиевич              //
//                 e-mail: sgsidorov@mail.ru                  //
//////////////////////////////////////////////////////////////

#include <iostream>;
using namespace std;

const int    N      = 20;      // размерность клеток
const int    Teta   = 3;       // порог
const double DeltaW = 0.03;    // вес обучения

struct LC {                    // координаты
     int L;                    // строка
     int C;                    // столбец
};

int    R[N][N];                // слой рецепторов
int    A[N][N];                // ассоциативный слой
LC     S[N][N][N];             // связи
double W[N][N];                // веса связей

bool fTest = false;            // флаг проверки работоспособности
```

```
void InitLayers()  // инициализация слоёв
{
    for (int L = 0; L < N; L++)
        for (int C = 0; C < N; C++) {
            W[L][C] = 0;                    // очистка весов
            for (int K = 0; K < N; K++) {   // генерация связей
                S[L][C][K].L = rand() % N;
                S[L][C][K].C = rand() % N;
            }
        }
};

int Obraz(int H)  // создание образа по шагу H
{
    // чистка рецепторов
    for (int L = 0; L < N; L++)
        for (int C = 0; C < N; C++) R[L][C] = 0;
    int sd = H % 2;            // выбор образа: 0-нолик,1-крестик
    int rr = rand() % (N / 2); // выбор радиуса образа
    if (rr < 3) rr = 3;

    // выбор места расположения образа
    int Lc = rr + (rand() % int(N-2*rr));     // строка центра
    int Cc = rr + (rand() % int(N-2*rr));     // столбец центра

    // рисование выбранного образа
    switch (sd)
    {
     case 0:  // рисование нолика
          for (int i = 1; i < 150; i++) {
              int L = Lc + int(rr * cos(i));
              int C = Cc + int(rr * sin(i));
              R[L][C] = 1;
          }
          break;
     case 1:  // рисование крестика
          for (int i = -rr; i <= rr; i++) {
              int L = Lc + i;
              int C = Cc + i;
              R[L][C] = 1;
              C = Cc - i;
              R[L][C] = 1;
          }
          break;
    }

    // вывод образа на экран во время проверки работоспособности
    if (fTest) {
       if ( sd == 0 ) cout << "Рисую нолик\n";
       else cout << "Рисую крестик\n";
```

```
        for (int L = 0; L < N; L++) {
            for (int C = 0; C < N; C++)
                if (R[L][C] == 0) cout << "_"; else cout << "*";
                cout << "\n";
            }
        }

    return sd;
};

void Otobr()  // отображение образа
{
    for (int L = 0; L < N; L++)  // очистка ассоциативного слоя A
        for (int C = 0; C < N; C++) A[L][C]=0;
    for (int L = 0; L < N; L++)  // отображение в слое A (входы)
        for (int C = 0; C < N; C++)
            if (R[L][C] == 1)
               for (int K = 0; K < N; K++) {
                   int Lv = S[L][C][K].L;
                   int Cv = S[L][C][K].C;
                   A[Lv][Cv]++;
               }
    for (int L = 0; L < N; L++)  // отображение в слое A (выходы)
        for (int C = 0; C < N; C++)
            if (A[L][C] > Teta) A[L][C] = 1; else A[L][C] = 0;
};

int Reak()  // распознавание образа
{
    double E = 0;       // эффекторный слой
    for (int L = 0; L < N; L++)
        for (int C = 0; C < N; C++) E += A[L][C] * W[L][C];
    // вывод результата при проверке работоспособности
    if (fTest) {
       cout << "\n";
       if (E > 0) cout << "Я думаю что это крестик\n";
       else cout << "Я думаю что это нолик\n";
       system("pause");
    }
    if (E > 0) return 1; else return 0;
};

void Teach(int sd)  // обучение нейросети не угадавшую образ sd
{
    for (int L = 0; L < N; L++)
        for (int C = 0; C < N; C++)
            if (A[L][C] == 1) {     // обучение виноватых
               if (sd == 0) W[L][C] -= DeltaW;
               else W[L][C] += DeltaW;
            }
};
```

```
int main()
{
    setlocale(LC_ALL, "Russian");
    cout << "Обучение перцептрона распознаванию двух образов:
    крестика и нолика\n";

    int lmax = 10000;       // число шагов обучения
    int nOk  = 0;           // число удачных ответов

    InitLayers();           // инициализация слоёв
    for (int Step = 1; Step <= lmax; Step++) {
        int dic = Obraz(Step);  // новый образ: 0-нолик, 1-крестик
        Otobr();                // отображение
        int dic1 = Reak();      // опознавание
        if (dic == dic1) nOk++; else Teach(dic);   // обучение
        // вывод текущей информации на экран
        cout << "Шаг " << Step << " : Доля удачных ответов " <<
             nOk / double(Step) * 100 << " %\n";
        // тестирование за 20 шагов до конца обучения
        if (Step == (lmax - 20)) fTest = true;
    };
    cout << "Конец\n";
}
```

Пример реализации перцептрона на языке Pascal

```
(*************************************************************)
(*  Перцептрон. Распознавание 2-х образов: крестика и нолика  *)
(*            2020 (c) Сидоров Сергей Георгиевич              *)
(*                 e-mail: sgsidorov@mail.ru                  *)
(*************************************************************)
program XO;

const
      N      = 19;            (* размерность клеток *)
      Teta   = 3;             (* порог *)
      DeltaW = 0.03;          (* вес обучения *)

type
      LC = record             (* координаты *)
         L :longint;          (* строка *)
         C :longint;          (* столбец *)
      end;

var
     R :array[0..N-1,0..N-1] of byte;      (*слой рецепторов *)
     A :array[0..N-1,0..N-1] of longint;   (*ассоциативный слой*)
     S :array[0..N-1,0..N-1,0..N-1] of LC; (*связи *)
     W :array[0..N-1,0..N-1] of double;    (*веса связей *)
```

```
     fTest    :boolean;       (* флаг проверки работы *)
     lmax     :longint;       (* число шагов обучения *)
     nOk      :longint;       (* число удачных ответов *)
     Step     :longint;       (* текущий шаг обучения *)
     dic,dic1 :longint;       (* образ, распознанный образ*)

procedure InitLayers;  (* инициализация слоев *)
var  L,C,K :longint;   (* индексы *)
begin
     for L:=0 to N-1 do
         for C:=0 to N-1 do begin
             W[L][C]:=0;                (* очистка весов *)
             for K:=0 to N-1 do begin   (* генерация связей *)
                 S[L][C][K].L := random(N);
                 S[L][C][K].C := random(N);
             end;
         end;
end;

function Obraz(H:longint):longint; (*создание образа по шагу H*)
var  L,C,i :longint;   (* индексы *)
     sd    :longint;   (* код образа: 0-нолик,1-крестик *)
     rr    :longint;   (* радиус фигуры *)
     Lc    :longint;   (* строка центра *)
     Cc    :longint;   (* столбец центра *)
begin
    (* чистка рецепторов *)
    for L:=0 to N-1 do
        for C:=0 to N-1 do R[L,C]:=0;
    sd := H mod 2;          (* выбор образа: 0-нолик,1-крестик *)
    rr := random(N div 2); (* выбор радиуса образа *)
    if rr<3 then rr:=3;
    Lc := rr + random(N-2*rr); (* строка центра *)
    Cc := rr + random(N-2*rr); (* столбец центра *)
    (* рисование выбранного образа *)
    Case sd of
      0:  (* рисование нолика *)
          for i:=1 to 150 do begin
              L:=Lc + round(rr * cos(i));
              C:=Cc + round(rr * sin(i));
              R[L,C]:=1;
          end;
      1:  (* рисование крестика *)
          for i:=-rr to rr do begin
              L:=Lc+i;
              C:=Cc+i;
              R[L,C]:=1;
              C:=Cc-i;
              R[L,C]:=1;
          end;
    end;
```

```
    (* вывод образа на экран во время проверки работы *)
    if fTest then begin
       if sd = 0  then writeln('Рисую нолик')
                  else writeln('Рисую крестик');
       for L:=0 to N-1 do begin
           for C:=0 to N-1 do
               if R[L,C]=0 then write('_') else write('*');
           writeln;
       end;
    end;
    Obraz:=sd;
end;

procedure Otobr;  (* отображение образа *)
var  L,C,K,Lv,Cv :longint;  (* индексы *)
begin
     (* очистка ассоциативного слоя *)
     for L:=0 to N-1 do
         for C:=0 to N-1 do A[L,C]:=0;
     (* отображение в слое A (входы) *)
     for L:=0 to N-1 do
         for C:=0 to N-1 do
             if R[L,C]=1 then
                for K:=0 to N-1 do begin
                    Lv:=S[L,C,K].L;
                    Cv:=S[L,C,K].C;
                    inc(A[Lv,Cv]);
                end;
     (* отображение в слое A (выходы) *)
     for L:=0 to N-1 do
         for C:=0 to N-1 do
             if A[L,C]>Teta then A[L,C]:=1 else A[L,C]:=0;
end;


function Reak:longint;  (* распознавание образа *)
var  E    :double;    (* эффекторный слой *)
     L,C  :longint;   (* индексы *)
begin
     E:=0;
     for L:=0 to N-1 do
         for C:=0 to N-1 do E:=E+A[L,C]*W[L,C];
     (* вывод результата при проверке работоспособности *)
     if fTest then begin
        writeln;
        if E>0 then writeln('Я думаю, что это крестик')
               else writeln('Я думаю, что это нолик');
        readln;
     end;
     if E>0 then Reak:=1 else Reak:=0;
end;
```

```
procedure Teach(sd:longint); (*обучение сети не угадавшей образ
sd*)
var  L,C :longint;
begin
     for L:=0 to N-1 do
         for C:=0 to N-1 do
            if A[L,C]=1 then   (* обучение виноватых *)
               if sd=0 then W[L,C]:=W[L,C]-DeltaW
                       else W[L,C]:=W[L,C]+DeltaW;
end;


begin
     writeln('Обучение перцептрона распознаванию двух образов:'+
             ' крестика и нолика');

     fTest:=false;       (* флаг проверки работы *)
     lmax := 10000;      (* число шагов обучения *)
     nOk  := 0;          (* число удачных ответов *)

     InitLayers;         (* инициализация слоев *)
     for Step:=1 to lmax do begin
         dic:=Obraz(Step);  (* новый образ: 0-нолик, 1-крестик *)
         Otobr;             (* отображение *)
         dic1:=Reak;        (* опознавание *)
         if dic = dic1 then inc(nOk) else Teach(dic); (*обучение*)
         writeln('Шаг ',Step,' : Доля удачных ответов ',
                  nOk/Step*100:6:2);
         (* тестирование за 20 шагов до конца обучения *)
         if Step = (lmax - 20) then fTest:=true;
     end;
     writeln('Конец');
end.
```

Задания к лабораторной работе

На основе приведенных выше примеров реализовать и отладить программу распознавания двух образов в соответствии с заданным вариантом.

Провести исследование влияния на скорость и качество обучения таких параметров как:

- число шагов обучения;
- величина веса обучения;
- величина порога;
- количество связей между рецепторным слоем и ассоциативным слоем;
- количество связей между ассоциативным слоем и эффекторным слоем;
- размер рецепторного слоя;

- размер ассоциативного слоя;
- интервал изменения размеров образов;
- интервал перемещения образа по рецепторному слою.

Подготовить отчет, включающий в себя код программы, описание основных функций, результаты обучения перцептрона, графики зависимости обучаемости перцептрона от влияющих факторов, выводы по результатам исследования, ответы на контрольные вопросы.

№	Задание
1	Реализовать распознавание фигур квадрата и треугольника
2	Реализовать распознавание символов «**Е**» и «**Н**»
3	Реализовать распознавание цифр «**1**» и «**2**»
4	Реализовать распознавание фигур круга и ромба
5	Реализовать распознавание символов «**Г**» и «**И**»
6	Реализовать распознавание цифр «**3**» и «**4**»
7	Реализовать распознавание образов плюса и минуса
8	Реализовать распознавание символов «**К**» и «**П**»
9	Реализовать распознавание цифр «**5**» и «**6**»
10	Реализовать распознавание знаков «**#**» и «**=**»
11	Реализовать распознавание символов «**Т**» и «**Х**»
12	Реализовать распознавание цифр «**7**» и «**8**»
13	Реализовать распознавание фигур квадрата и ромба
14	Реализовать распознавание символов «**Ш**» и «**Г**»
15	Реализовать распознавание цифр «**9**» и «**0**»
16	Реализовать распознавание фигур круга и треугольника
17	Реализовать распознавание латинских символов «**F**» и «**Z**»
18	Реализовать распознавание цифр «**1**» и «**3**»
19	Реализовать распознавание скобок «**[**» и «**]**»
20	Реализовать распознавание латинских символов «**L**» и «**V**»
21	Реализовать распознавание символов «**&**» и «**?**»
22	Реализовать распознавание букв «**В**» и «**Ю**»
23	Реализовать распознавание образов овалов, вытянутых в вертикальном и горизонтальном направлениях
24	Реализовать распознавание образов веселого и грустного смайликов

Контрольные вопросы

1) Что такое перцептрон?
2) Сколько слоев нейронов используется в однослойном перцептроне?
3) Что такое решающий слой?
4) Перечислите названия слоев, входящих в состав перцептрона.
5) Что реализует рецепторный слой?
6) Что реализует ассоциативный слой?
7) Что реализует эффекторный слой?
8) Как устанавливаются и инициализируются связи между рецепторным слоем и ассоциативным слоем?
9) Как устанавливаются и инициализируются связи между ассоциативным слоем и эффекторным слоем?
10) По каким принципам выбирается размер рецепторного слоя?
11) По каким принципам выбирается размер ассоциативного слоя?
12) По каким принципам выбирается размер эффекторного слоя?
13) Как определить размер веса обучения?
14) Какой алгоритм обучения реализован в перцептроне?
15) Как влияет число шагов обучения на качество обученности перцептрона?
16) Как влияет вес обучения на скорость обучения перцептрона?
17) Как влияет число связей между слоями на способность перцептрона к обучению?
18) Как правильно оценить качество обученности перцептрона?
19) Что такое обучающая выборка?
20) Какие требования предъявляются к обучающей выборке?

Лабораторная работа №2
Распознавание четырех образов

Цель работы:

- знакомство с особенностями реализации перцептрона для распознавания большого числа образов;
- изучение вопросов кодирования решения и применение двоичного кода в процессе обучения перцептрона.

Особенности распознавания большого числа образов

Для распознавания с помощью перцептрона большого количества образов необходимо увеличить количество эффекторных клеток в выходном слое, и количество весов связей между ассоциативным слоем и клетками эффекторного слоя.

В данной работе рассмотрим вариант, когда для каждого из распознаваемых образов устанавливается в соответствие «своя» эффекторная клетка со своим набором весов связей (см. рис. 2).

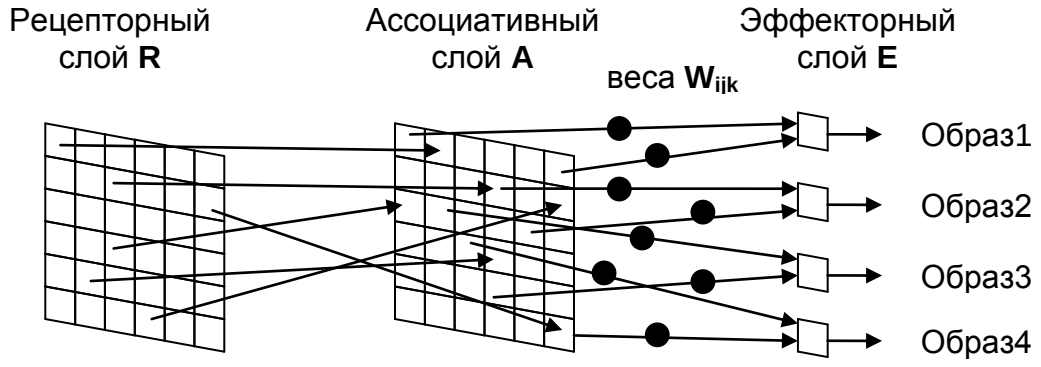

Рис. 2. Структура перцептрона для распознавания 4-х образов

Такой подход позволяет анализировать образы на схожесть между собой, находить аналоги, выявлять закономерности. Минусом такого подхода является значительный расход памяти, поэтому пользоваться таким вариантом лучше при относительно ограниченном наборе распознаваемых образов.

Пример реализации перцептрона на языке C++

```
//////////////////////////////////////////////////////////////
//              Перцептрон. Распознавание 4-х образов:      //
//                  крестика, нолика, плюса, ромба          //
//             2020 (c) Сидоров Сергей Георгиевич           //
//                  e-mail: sgsidorov@mail.ru               //
//////////////////////////////////////////////////////////////

#include <iostream>
using namespace std;

const int    N = 20;           // размерность клеток
const int    Teta = 3;         // порог
const double DeltaW = 0.03;    // вес обучения

struct LC {                    // координаты
    int L;                     // строка
    int C;                     // столбец
};

int    R[N][N];                // слой рецепторов
int    A[N][N];                // ассоциативный слой
LC     S[N][N][N];             // связи
double W[N][N][4];             // веса связей

bool fTest = false;            // флаг проверки работоспособности

void InitLayers()  // инициализация слоёв
{
    // генерация связей
    for (int L = 0; L < N; L++)
        for (int C = 0; C < N; C++) {
            for (int K = 0; K < N; K++) {
                S[L][C][K].L = rand() % N;
                S[L][C][K].C = rand() % N;
            }
        }
    // очистка весов
    for (int K = 0; K < 4; K++)
        for (int L = 0; L < N; L++)
            for (int C = 0; C < N; C++) W[L][C][K] = 0;
};
```

```
string NameDic(string dic) // имя образа: нолик,крестик,плюс,ромб
{
    string res = "не знаю";
    if (dic == "1000") res = "нолик";
    if (dic == "0100") res = "крестик";
    if (dic == "0010") res = "плюс";
    if (dic == "0001") res = "ромб";
    return res;
};

string Obraz(int H)  // создание образа по шагу H
{
    for (int L = 0; L < N; L++)   // чистка рецепторов
        for (int C = 0; C < N; C++) R[L][C] = 0;
    int sd = H % 4;  // выбор образа: 0-ноль,1-крест,2-плюс,3-ромб
    int rr = rand() % (N / 3);    // выбор радиуса образа
    if (rr < 5) rr = 5;
    string res = "0000";                          // код образа
    // выбор места расположения образа
    int Lc = rr + (rand() % int(N - 2 * rr));  // строка центра
    int Cc = rr + (rand() % int(N - 2 * rr));  // столбец центра
    // рисование выбранного образа
    switch (sd)
    {
    case 0:  // рисование нолика
      res = "1000";   // код нолика
      for (int i = 1; i < 150; i++) {
          int L = Lc + int(rr * cos(i));
          int C = Cc + int(rr * sin(i));
          R[L][C] = 1;
      }
      break;
    case 1:  // рисование крестика
      res = "0100";   // код крестика
      for (int i = -rr; i <= rr; i++) {
          int L = Lc + i;
          int C = Cc + i;
          R[L][C] = 1;
          C = Cc - i;
          R[L][C] = 1;
      }
      break;
    case 2:  // рисование плюса
      res = "0010";   // код плюса
      for (int i = -rr; i <= rr; i++) {
          int L = Lc + i;
          int C = Cc;
          R[L][C] = 1;
          L = Lc;
          C = Cc + i;
          R[L][C] = 1;
      }
```

```
      break;
    case 3:  // рисование ромба
      res = "0001";   // код ромба
      for (int i = 0; i <= rr; i++) {
          int L = Lc - i;
          int C = Cc - rr + i;
          R[L][C] = 1;
          L = Lc + i;
          C = Cc - rr + i;
          R[L][C] = 1;
          L = Lc - rr + i;
          C = Cc + i;
          R[L][C] = 1;
          L = Lc + rr - i;
          C = Cc + i;
          R[L][C] = 1;
      }
      break;
    };

    // вывод образа на экран во время проверки работы
    if (fTest) {
       cout << "Рисую " + NameDic(res) + "\n";
       for (int L = 0; L < N; L++) {
           for (int C = 0; C < N; C++)
               if (R[L][C] == 0) cout << "_"; else cout << "*";
           cout << "\n";
       }
    };
    return res;
};


void Otobr()  // отображение образа
{
    for (int L = 0; L < N; L++)  // очистка ассоциативного слоя A
        for (int C = 0; C < N; C++) A[L][C] = 0;
    for (int L = 0; L < N; L++)  // отображение в слое A (входы)
        for (int C = 0; C < N; C++)
            if (R[L][C] == 1)
               for (int K = 0; K < N; K++) {
                   int Lv = S[L][C][K].L;
                   int Cv = S[L][C][K].C;
                   A[Lv][Cv]++;
               }
    for (int L = 0; L < N; L++)  // отображение в слое A (выходы)
        for (int C = 0; C < N; C++)
            if (A[L][C] > Teta) A[L][C] = 1; else A[L][C] = 0;
};
```

```
string Reak()  // распознавание образа: 1000..0001
{
    double E[4];        // эффекторный слой
    string res = "";    // код распознавания
    for (int K = 0; K < 4; K++) {
        E[K] = 0;
        for (int L = 0; L < N; L++)
            for (int C = 0; C < N; C++)
                E[K] += A[L][C] * W[L][C][K];
        if (E[K] > 0) res += '1'; else res += '0';
    }
    // вывод результата при проверке работоспособности
    if (fTest) {
       cout << "\n" << "Я думаю что это " + NameDic(res) << "\n";
       system("pause");
    }
    return res;
};


void Teach(string sd, string sd1)  // обучение нейросети
{
    for (int K = 0; K < 4; K++) {
        if (sd[K] != sd1[K]) {
           for (int L = 0; L < N; L++) {
               for (int C = 0; C < N; C++) {
                   if (A[L][C] == 1) {   // обучение виноватых
                      if (sd[K] == '0') W[L][C][K] -= DeltaW;
                      else W[L][C][K] += DeltaW;
                   }
               }
           }
        }
    }
};
```

```
int main()
{
    setlocale(LC_ALL, "Russian");
    cout << "Обучение перцептрона распознаванию четырех образов:
            крестика, нолика, плюса, ромба\n";
    int lmax = 10000;       // число шагов обучения
    int nOk = 0;            // число удачных ответов
    InitLayers();           // инициализация слоёв
    for (int Step = 1; Step <= lmax; Step++) {
        string dic = Obraz(Step);           // новый образ
        Otobr();                            // отображение
        string dic1 = Reak();               // опознавание
        if (dic == dic1) nOk++;
        else Teach(dic, dic1);              // обучение
        // вывод текущей информации на экран
        cout << "Шаг " << Step << " : Доля удачных ответов "
             << nOk / double(Step) * 100 << " %\n";
        // тестирование за 20 шагов до конца обучения
        if (Step == (lmax - 20)) fTest = true;
    };
    cout << "Конец\n";
}
```

Пример реализации перцептрона на языке Pascal

```
(*************************************************************)
(*             Перцептрон. Распознавание 4-х образов:        *)
(*                 крестика, нолика, плюса, ромба            *)
(*            2020 (c) Сидоров Сергей Георгиевич             *)
(*                 e-mail: sgsidorov@mail.ru                 *)
(*************************************************************)

Program XOPR4;

const
      N      = 20;             (* размерность клеток  *)
      Teta   = 3;              (* порог               *)
      DeltaW = 0.03;           (* вес обучения        *)

type
      LC = record              (* координаты          *)
           L :word;            (* строка              *)
           C :word;            (* столбец             *)
      end;

var
     R:array[0..N-1,0..N-1] of byte;     (* слой рецепторов    *)
     A:array[0..N-1,0..N-1] of word;     (* ассоциативный слой *)
     S:array[0..N-1,0..N-1,0..N-1] of LC;   (* связи        *)
     W:array[0..N-1,0..N-1,0..3] of double; (* веса связей  *)
```

```
     fTest :boolean;           (* флаг проверки работы  *)
     lmax  :longint;           (* число шагов обучения  *)
     nOk   :longint;           (* число удачных ответов *)
     Step  :longint;           (* текущий шаг           *)
     dic   :string;            (* рисуемый образ        *)
     dic1  :string;            (* распознанный образ    *)

procedure InitLayers; (* инициализация слоёв *)
var  L,C,K :longint;  (* индексы *)
begin
     (* генерация связей *)
     for L:=0 to N-1 do
         for C:=0 to N-1 do
             for K:=0 to N-1 do begin
                 S[L,C,K].L := random(N);
                 S[L,C,K].C := random(N);
             end;
     (* очистка весов *)
     for K:=0 to 3 do
         for L:=0 to N-1 do
             for C:=0 to N-1 do W[L,C,K]:=0;
end;


function NameDic(dic:string):string; (* имя образа *)
begin
     NameDic:='не знаю';
     if dic='1000' then NameDic:='нолик';
     if dic='0100' then NameDic:='крестик';
     if dic='0010' then NameDic:='плюс';
     if dic='0001' then NameDic:='ромб';
end;


function Obraz(H:longint):string;  (* создание образа по шагу H *)
var  L,C,i :longint;  (* индексы *)
     sd    :longint;  (* образ: 0-ноль,1-крест,2-плюс,3-ромб *)
     rr    :longint;  (* радиус образа  *)
     Lc    :longint;  (* строка центра  *)
     Cc    :longint;  (* столбец центра *)
     res   :string;   (* результат      *)
begin
     for L:=0 to N-1 do   (* чистка рецепторов *)
         for C:=0 to N-1 do R[L,C]:=0;
     sd:=H mod 4;  (*выбор образа: 0-ноль,1-крест,2-плюс,3-ромб*)
     rr:=random(N div 3);   (* выбор радиуса образа *)
     if rr<5 then rr:=5;
     Obraz:='0000';         (* код образа *)
     (* выбор места расположения образа *)
     Lc:=rr + random(N - 2 * rr);   (* строка центра  *)
     Cc:=rr + random(N - 2 * rr);   (* столбец центра *)
```

```
(* рисование выбранного образа *)
Case sd of
  0: begin   (* рисование нолика *)
       res:='1000';   (* код нолика *)
       for i:=1 to 150 do begin
           L:=Lc + round(rr * cos(i));
           C:=Cc + round(rr * sin(i));
           R[L,C] := 1;
       end;
     end;
  1: begin  (* рисование крестика *)
       res:='0100';   (* код крестика *)
       for i:=-rr to rr do begin
           L:=Lc + i;
           C:=Cc + i;
           R[L,C]:=1;
           C:=Cc - i;
           R[L,C]:=1;
       end;
     end;
  2: begin  (* рисование плюса *)
       res:='0010';   (* код плюса *)
       for i:=-rr to rr do begin
           L:=Lc + i;
           C:=Cc;
           R[L,C]:=1;
           L:=Lc;
           C:=Cc + i;
           R[L,C]:=1;
       end;
     end;
  3: begin  (* рисование ромба *)
       res:='0001';   (* код ромба *)
       for i:=0 to rr do begin
           L:=Lc - i;
           C:=Cc - rr + i;
           R[L,C]:=1;
           L:=Lc + i;
           C:=Cc - rr + i;
           R[L,C]:=1;
           L:=Lc - rr + i;
           C:=Cc + i;
           R[L,C]:=1;
           L:=Lc + rr - i;
           C:=Cc + i;
           R[L,C]:=1;
       end;
     end;
end;
```

```
    (* вывод образа на экран во время проверки работы *)
    if fTest then begin
       writeln('Рисую ' + NameDic(res));
       for L:=0 to N-1 do begin
           for C:=0 to N-1 do
               if R[L,C]=0 then write('_') else write ('*');
           writeln;
       end;
    end;
    Obraz:=res;
end;

procedure Otobr;  (* отображение образа *)
var  L,C,K,Lv,Cv :longint;  (* индексы *)
begin
     for L:=0 to N-1 do   (* очистка ассоциативного слоя A *)
         for C:=0 to N-1 do A[L,C]:=0;
     for L:=0 to N-1 do   (* отображение в слое A (входы) *)
         for C:=0 to N-1 do
            if R[L,C]=1 then
               for K:=0 to N-1 do begin
                   Lv:=S[L,C,K].L;
                   Cv:=S[L,C,K].C;
                   inc(A[Lv,Cv]);
               end;
     for L:=0 to N-1 do   (* отображение в слое A (выходы) *)
         for C:=0 to N-1 do
             if A[L,C]>Teta then A[L,C]:=1 else A[L,C]:=0;
end;

function Reak:string;  (* распознавание образа: 1000..0001 *)
var  E     :array[0..3] of double;  (* эффекторный слой *)
     L,C,K :longint;                (* индексы *)
     res   :string;                 (* результат *)
begin
     res:='';   (* код распознавания *)
     for K:=0 to 3 do begin
         E[K]:=0;
         for L:=0 to N-1 do
            for C:=0 to N-1 do
                E[K]:=E[K] + A[L,C] * W[L,C,K];
         if E[K]>0 then res:=res+'1' else res:=res+'0';
     end;
     (* вывод результата при проверке работоспособности *)
     if fTest then begin
        writeln('Я думаю что это ' + NameDic(res));
        readln;
     end;
     Reak:=res;
end;
```

```
procedure Teach(sd,sd1:string);  (* обучение нейросети *)
var  L,C,K :longint;  (* индексы *)
begin
     for K:=0 to 3 do begin
         if sd[K+1]<>sd1[K+1] then begin
            for L:=0 to N-1 do begin
                for C:=0 to N-1 do begin
                    if A[L,C]=1 then begin (*обучение виноватых*)
                       if sd[K+1]='0'
                          then W[L,C,K]:=W[L,C,K]-DeltaW
                          else W[L,C,K]:=W[L,C,K]+DeltaW;
                    end;
                end;
            end;
         end;
     end;
end;


begin
     fTest :=false;            (* флаг проверки работы *)
     writeln('Обучение перцептрона распознаванию четырех'+
             ' образов: крестика, нолика, плюса, ромба');
     lmax := 10000;         (* число шагов обучения  *)
     nOk  := 0;             (* число удачных ответов *)
     InitLayers;            (* инициализация слоёв   *)
     for Step:=1 to lmax do begin
         dic:=Obraz(Step);                  (* новый образ *)
         Otobr;                             (* отображение *)
         dic1:=Reak;                        (* опознавание *)
         if dic=dic1 then inc(nOk)
                     else Teach(dic,dic1);  (* обучение    *)
         (* вывод текущей информации на экран *)
         writeln('Шаг ', Step, ' : Доля удачных ответов ',
                  nOk/Step*100:6:2);
         (* тестирование за 20 шагов до конца обучения *)
         if Step=(lmax - 20) then fTest:=true;
     end;
     writeln('Конец');
end.
```

Задания к лабораторной работе

На основе приведенных выше примеров реализовать и отладить программу распознавания нескольких образов в соответствии с заданным вариантом.

Подготовить отчет, включающий в себя код программы, описание основных функций, результаты обучения перцептрона, выводы по результатам исследования, ответы на контрольные вопросы.

№	Задание
1	Реализовать распознавание цифр "**1**", "**2**", "**3**", "**4**", "**5**", "**6**"
2	Реализовать распознавание символов "**А**", "**Б**", "**Г**", "**Е**"
3	Реализовать распознавание цифр "**2**", "**3**", "**4**", "**5**", "**6**", "**7**"
4	Реализовать распознавание символов "**Б**", "**Г**", "**Е**", "**И**"
5	Реализовать распознавание цифр "**3**", "**4**", "**5**", "**6**", "**7**", "**8**"
6	Реализовать распознавание символов "**Г**", "**Е**", "**И**", "**К**"
7	Реализовать распознавание цифр "**4**", "**5**", "**6**", "**7**", "**8**", "**9**"
8	Реализовать распознавание символов "**Е**", "**И**", "**К**", "**Л**"
9	Реализовать распознавание цифр "**5**", "**6**", "**7**", "**8**", "**9**", "**0**"
10	Реализовать распознавание символов "**И**", "**К**", "**Л**", "**Н**"
11	Реализовать распознавание цифр "**6**", "**7**", "**8**", "**9**", "**0**", "**1**"
12	Реализовать распознавание символов "**К**", "**Л**", "**Н**", "**О**"
13	Реализовать распознавание цифр "**7**", "**8**", "**9**", "**0**", "**1**", "**2**"
14	Реализовать распознавание символов "**Л**", "**Н**", "**О**", "**П**"
15	Реализовать распознавание цифр "**8**", "**9**", "**0**", "**1**", "**2**", "**3**"
16	Реализовать распознавание символов "**Н**", "**О**", "**П**", "**С**"
17	Реализовать распознавание цифр "**9**", "**0**", "**1**", "**2**", "**3**", "**4**"
18	Реализовать распознавание символов "**О**", "**П**", "**С**", "**Т**"
19	Реализовать распознавание цифр "**0**", "**1**", "**2**", "**3**", "**4**", "**5**"
20	Реализовать распознавание символов "**П**", "**С**", "**Т**", "**Х**"
21	Реализовать распознавание знаков "#", "**=**", "**!**","**^**"
22	Реализовать распознавание символов "**С**", "**Т**", "**Х**", "**Ч**"
23	Реализовать распознавание скобок "**<**", "**>**", "**[**","**]**"
24	Реализовать распознавание символов "**Т**", "**Х**", "**Ч**", "**Ш**"

Контрольные вопросы

1) Сколько эффекторных клеток необходимо для распознавания нескольких образов?
2) Как меняется число слоев перцептрона при увеличении числа распознаваемых образов?
3) Как изменяются связи при увеличении числа распознаваемых образов?
4) По каким принципам кодируется распознаваемый образ?
5) Как определить схожесть распознаваемых образов?
6) Какие требования предъявляются к обучающей выборке?
7) Какой алгоритм используется для обучения перцептрона?
8) Какие значения влияют на качество обучения перцептрона?
9) На что влияет увеличение числа распознаваемых образов?
10) Можно ли совмещать веса перцептронов, обученных для распознавания различных образов?
11) Назовите признаки отсутствия обучения перцептрона.
12) Какие веса связей подлежат обучению?
13) От чего зависит размер веса обучения?
14) Как влияет число шагов обучения на качество обученности перцептрона?
15) Как влияет размер образа на способность перцептрона к обучению?
16) Как правильно оценить качество обученности перцептрона?

Лабораторная работа №3
Двоичное распознавание четырех образов

Цель работы:

- знакомство с особенностями реализации перцептрона для распознавания большого числа образов;
- изучение вопросов компактного кодирования решения и применение двоичного кода в процессе обучения перцептрона.

Двоичное распознавание большого числа образов

Для распознавания с помощью перцептрона большого количества образов необходимо увеличить количество эффекторных клеток в выходном слое и количество весов связей между ассоциативным слоем и клетками эффекторного слоя. Однако в таком случае расход памяти становится прямопропорционален количеству распознаваемых образов, что допустимо только при ограниченном количестве образов.

В данной работе рассмотрим вариант, когда для снижения вычислительной нагрузки используется двоичное кодирование распознаваемых образов. В этом случае число эффекторных клеток определяется двоичным логарифмом от числа распознаваемых образов, а комбинация состояний клеток эффекторного слоя кодирует в двоичном коде решение (см. рис. 3).

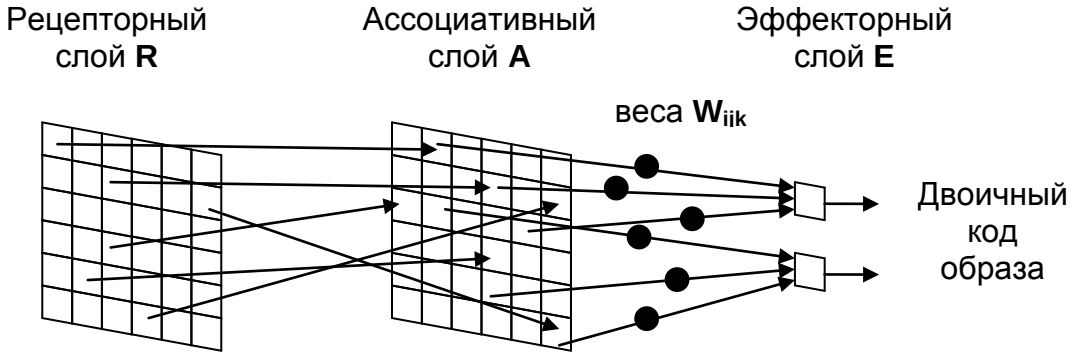

Рис. 3. Структура двоичного перцептрона для распознавания 4-х образов

Двоичное кодирование позволяет компактно представлять веса связей и снижает вычислительную нагрузку в процессе обучения и работы перцептрона. Например, для распознавания 32 образов,

каждому из которых соответствует одна эффекторная клетка, требуется при заданных в программе параметрах 32x20x20=12800 весов связей. При использовании двоичного кодирования число весов связей снижается до 5x20x20=2000 элементов.

Пример реализации перцептрона на языке C++

```
//////////////////////////////////////////////////////////////
//       Перцептрон. Двоичное распознавание 4-х образов:    //
//                  крестика, нолика, плюса, ромба          //
//             2020 (c) Сидоров Сергей Георгиевич           //
//                  e-mail: sgsidorov@mail.ru               //
//////////////////////////////////////////////////////////////

#include <iostream>
using namespace std;

const int    N = 20;           // размерность клеток
const int    Teta = 3;         // порог
const double DeltaW = 0.03;    // вес обучения

struct LC {                    // координаты
    int L;                     // строка
    int C;                     // столбец
};

int    R[N][N];                // слой рецепторов
int    A[N][N];                // ассоциативный слой
LC     S[N][N][N];             // связи
double W[N][N][2];             // веса связей

bool fTest = false;            // флаг проверки работоспособности


void InitLayers()  // инициализация слоёв
{
    // генерация связей
    for (int L = 0; L < N; L++)
        for (int C = 0; C < N; C++) {
            for (int K = 0; K < N; K++) {
                S[L][C][K].L = rand() % N;
                S[L][C][K].C = rand() % N;
            }
        }
    // очистка весов
    for (int K = 0; K < 2; K++)
        for (int L = 0; L < N; L++)
            for (int C = 0; C < N; C++) W[L][C][K] = 0;
};
```

```
string NameDic(string dic)  // название образа
{
    string res = "не знаю";
    if (dic == "00") res = "нолик";
    if (dic == "01") res = "крестик";
    if (dic == "10") res = "плюс";
    if (dic == "11") res = "ромб";
    return res;
};


string Obraz(int H)  // создание образа по шагу H
{
    for (int L = 0; L < N; L++)      // чистка рецепторов
        for (int C = 0; C < N; C++) R[L][C] = 0;
    int sd = H % 4;    // выбор образа: ноль,крест,плюс,ромб
    int rr = rand() % (N / 3);       // выбор радиуса образа
    if (rr < 5) rr = 5;
    string res = "00";               // код образа
    // выбор места расположения образа
    int Lc = rr + (rand() % int(N - 2 * rr));   // строка центра
    int Cc = rr + (rand() % int(N - 2 * rr));   // столбец центра

    // рисование выбранного образа
    switch (sd)
    {
    case 0:  // рисование нолика
      res = "00";   // код нолика
      for (int i = 1; i < 150; i++) {
          int L = Lc + int(rr * cos(i));
          int C = Cc + int(rr * sin(i));
          R[L][C] = 1;
      }
      break;
    case 1:  // рисование крестика
      res = "01";   // код крестика
      for (int i = -rr; i <= rr; i++) {
          int L = Lc + i;
          int C = Cc + i;
          R[L][C] = 1;
          C = Cc - i;
          R[L][C] = 1;
      }
      break;
```

```
    case 2:  // рисование плюса
      res = "10";   // код плюса
      for (int i = -rr; i <= rr; i++) {
          int L = Lc + i;
          int C = Cc;
          R[L][C] = 1;
          L = Lc;
          C = Cc + i;
          R[L][C] = 1;
      }
      break;
    case 3:  // рисование ромба
      res = "11";   // код ромба
      for (int i = 0; i <= rr; i++) {
          int L = Lc - i;
          int C = Cc - rr + i;
          R[L][C] = 1;
          L = Lc + i;
          C = Cc - rr + i;
          R[L][C] = 1;
          L = Lc - rr + i;
          C = Cc + i;
          R[L][C] = 1;
          L = Lc + rr - i;
          C = Cc + i;
          R[L][C] = 1;
      }
      break;
    };

    // вывод образа на экран во время проверки работоспособности
    if (fTest) {
       cout << "Рисую " + NameDic(res) + "\n";
       for (int L = 0; L < N; L++) {
           for (int C = 0; C < N; C++)
               if (R[L][C] == 0) cout << "_"; else cout << "*";
               cout << "\n";
       }
    };

    return res;
};
```

```
void Otobr()  // отображение образа
{
    for (int L = 0; L < N; L++)  // очистка ассоциативного слоя A
        for (int C = 0; C < N; C++) A[L][C] = 0;
    for (int L = 0; L < N; L++)  // отображение в слое A (входы)
        for (int C = 0; C < N; C++)
            if (R[L][C] == 1)
               for (int K = 0; K < N; K++) {
                   int Lv = S[L][C][K].L;
                   int Cv = S[L][C][K].C;
                   A[Lv][Cv]++;
               }
    for (int L = 0; L < N; L++)  // отображение в слое A (выходы)
        for (int C = 0; C < N; C++)
            if (A[L][C] > Teta) A[L][C] = 1; else A[L][C] = 0;
};

string Reak()  // распознавание образа: 00..11
{
    double E[2];        // эффекторный слой
    string res = "";    // код распознавания
    for (int K = 0; K < 2; K++) {
        E[K] = 0;
        for (int L = 0; L < N; L++)
            for (int C = 0; C < N; C++)
                E[K] += A[L][C] * W[L][C][K];
        if (E[K] > 0) res += '1'; else res += '0';
    }
    // вывод результата при проверке работоспособности
    if (fTest) {
       cout << "\n" << "Я думаю что это " + NameDic(res) << "\n";
       system("pause");
    }
    return res;
};

void Teach(string sd, string sd1)  // обучение нейросети
{
    for (int K = 0; K < 2; K++) {
        if (sd[K] != sd1[K]) {
           for (int L = 0; L < N; L++) {
               for (int C = 0; C < N; C++) {
                   if (A[L][C] == 1) {   // обучение виноватых
                      if (sd[K] == '0') W[L][C][K] -= DeltaW;
                      else W[L][C][K] += DeltaW;
                   }
               }
           }
        }
    }
};
```

```
int main()
{
    setlocale(LC_ALL, "Russian");
    cout << "Обучение перцептрона распознаванию четырех образов:
             крестика, нолика, плюса, ромба\n";

    int lmax = 10000;       // число шагов обучения
    int nOk = 0;            // число удачных ответов

    InitLayers();           // инициализация слоёв
    for (int Step = 1; Step <= lmax; Step++) {
        string dic = Obraz(Step);          // новый образ
        Otobr();                           // отображение
        string dic1 = Reak();              // опознавание
        if (dic == dic1) nOk++;
        else Teach(dic, dic1);             // обучение
        // вывод текущей информации на экран
        cout << "Шаг " << Step << " : Доля удачных ответов "
             << nOk / double(Step) * 100 << " %\n";
        // тестирование за 20 шагов до конца обучения
        if (Step == (lmax - 20)) fTest = true;
    };
    cout << "Конец\n";
}
```

Пример реализации перцептрона на языке Pascal

```
(*************************************************************)
(*             Перцептрон. Распознавание 4-х образов:        *)
(*                 крестика, нолика, плюса, ромба            *)
(*             2020 (c) Сидоров Сергей Георгиевич            *)
(*                 e-mail: sgsidorov@mail.ru                 *)
(*************************************************************)

Program XOPR2;

const
      N      = 20;             (* размерность клеток  *)
      Teta   = 3;              (* порог               *)
      DeltaW = 0.03;           (* вес обучения        *)

type
      LC = record              (* координаты          *)
           L :word;            (* строка              *)
           C :word;            (* столбец             *)
      end;
```

```
var
    R:array[0..N-1,0..N-1] of byte;     (* слой рецепторов  *)
    A:array[0..N-1,0..N-1] of word;     (* ассоциативный слой *)
    S:array[0..N-1,0..N-1,0..N-1] of LC;   (* связи        *)
    W:array[0..N-1,0..N-1,0..1] of double; (* веса связей  *)

    fTest :boolean;           (* флаг проверки работы  *)
    lmax  :longint;           (* число шагов обучения  *)
    nOk   :longint;           (* число удачных ответов *)
    Step  :longint;           (* текущий шаг           *)
    dic   :string;            (* рисуемый образ        *)
    dic1  :string;            (* распознанный образ    *)

procedure InitLayers; (* инициализация слоёв *)
var  L,C,K :longint;  (* индексы *)
begin
     (* генерация связей *)
     for L:=0 to N-1 do
         for C:=0 to N-1 do
             for K:=0 to N-1 do begin
                 S[L,C,K].L := random(N);
                 S[L,C,K].C := random(N);
             end;
     (* очистка весов *)
     for K:=0 to 1 do
         for L:=0 to N-1 do
             for C:=0 to N-1 do W[L,C,K]:=0;
end;


function NameDic(dic:string):string; (* имя образа *)
begin
     NameDic:='не знаю';
     if dic='00' then NameDic:='нолик';
     if dic='01' then NameDic:='крестик';
     if dic='10' then NameDic:='плюс';
     if dic='11' then NameDic:='ромб';
end;
```

```
function Obraz(H:longint):string;  (* создание образа по шагу H *)
var  L,C,i :longint;  (* индексы *)
     sd    :longint;  (* образ: 0-ноль,1-крест,2-плюс,3-ромб *)
     rr    :longint;  (* радиус образа  *)
     Lc    :longint;  (* строка центра  *)
     Cc    :longint;  (* столбец центра *)
         res   :string;   (* resultat *)
begin
     for L:=0 to N-1 do   (* чистка рецепторов *)
         for C:=0 to N-1 do R[L,C]:=0;
     sd:=H mod 4;  (*выбор образа: 0-ноль,1-крест,2-плюс,3-ромб*)
     rr:=random(N div 3);  (* выбор радиуса образа *)
     if rr<5 then rr:=5;
     Obraz:='00';          (* код образа *)
     (* выбор места расположения образа *)
     Lc:=rr + random(N - 2 * rr);   (* строка центра  *)
     Cc:=rr + random(N - 2 * rr);   (* столбец центра *)

    (* рисование выбранного образа *)
    Case sd of
      0: begin   (* рисование нолика *)
            res:='00';   (* код нолика *)
            for i:=1 to 150 do begin
                L:=Lc + round(rr * cos(i));
                C:=Cc + round(rr * sin(i));
                R[L,C] := 1;
            end;
         end;
      1: begin  (* рисование крестика *)
            res:='01';   (* код крестика *)
            for i:=-rr to rr do begin
                L:=Lc + i;
                C:=Cc + i;
                R[L,C]:=1;
                C:=Cc - i;
                R[L,C]:=1;
            end;
         end;
      2: begin  (* рисование плюса *)
           res:='10';   (* код плюса *)
           for i:=-rr to rr do begin
               L:=Lc + i;
               C:=Cc;
               R[L,C]:=1;
               L:=Lc;
               C:=Cc + i;
               R[L,C]:=1;
           end;
         end;
```

```
      3: begin  (* рисование ромба *)
           res:='11';   (* код ромба *)
           for i:=0 to rr do begin
               L:=Lc - i;
               C:=Cc - rr + i;
               R[L,C]:=1;
               L:=Lc + i;
               C:=Cc - rr + i;
               R[L,C]:=1;
               L:=Lc - rr + i;
               C:=Cc + i;
               R[L,C]:=1;
               L:=Lc + rr - i;
               C:=Cc + i;
               R[L,C]:=1;
           end;
         end;
    end;

    (* вывод образа на экран во время проверки работы *)
    if fTest then begin
       writeln('Рисую ' + NameDic(res));
       for L:=0 to N-1 do begin
           for C:=0 to N-1 do
               if R[L,C]=0 then write('_') else write ('*');
           writeln;
       end;
    end;
    Obraz:=res;
end;


procedure Otobr;  (* отображение образа *)
var  L,C,K,Lv,Cv :longint;  (* индексы *)
begin
     for L:=0 to N-1 do   (* очистка ассоциативного слоя A *)
         for C:=0 to N-1 do A[L,C]:=0;
     for L:=0 to N-1 do   (* отображение в слое A (входы) *)
         for C:=0 to N-1 do
            if R[L,C]=1 then
               for K:=0 to N-1 do begin
                   Lv:=S[L,C,K].L;
                   Cv:=S[L,C,K].C;
                   inc(A[Lv,Cv]);
               end;
     for L:=0 to N-1 do   (* отображение в слое A (выходы) *)
         for C:=0 to N-1 do
             if A[L,C]>Teta then A[L,C]:=1 else A[L,C]:=0;
end;
```

```
function Reak:string;  (* распознавание образа: 00..11 *)
var  E     :array[0..1] of double;  (* эффекторный слой *)
     L,C,K :longint;                (* индексы *)
     res   :string;                 (* результат *)
begin
     res:='';   (* код распознавания *)
     for K:=0 to 1 do begin
         E[K]:=0;
         for L:=0 to N-1 do
            for C:=0 to N-1 do
                E[K]:=E[K] + A[L,C] * W[L,C,K];
         if E[K]>0 then res:=res+'1' else res:=res+'0';
     end;
     (* вывод результата при проверке работоспособности *)
     if fTest then begin
        writeln('Я думаю что это ' + NameDic(res));
        readln;
     end;
     Reak:=res;
end;


procedure Teach(sd,sd1:string);  (* обучение нейросети *)
var  L,C,K :longint;  (* индексы *)
begin
     for K:=0 to 1 do begin
         if sd[K+1]<>sd1[K+1] then begin
            for L:=0 to N-1 do begin
                for C:=0 to N-1 do begin
                    if A[L,C]=1 then begin (*обучение виноватых*)
                       if sd[K+1]='0'
                          then W[L,C,K]:=W[L,C,K]-DeltaW
                          else W[L,C,K]:=W[L,C,K]+DeltaW;
                    end;
                end;
            end;
         end;
     end;
end;
```

```
begin
     fTest :=false;            (* флаг проверки работы *)
     writeln('Обучение перцептрона распознаванию образов: '+
             'крестика, нолика, плюса, ромба');
     lmax := 10000;          (* число шагов обучения  *)
     nOk  := 0;              (* число удачных ответов *)
     InitLayers;             (* инициализация слоёв   *)
     for Step:=1 to lmax do begin
         dic:=Obraz(Step);                    (* новый образ *)
         Otobr;                               (* отображение *)
         dic1:=Reak;                          (* опознавание *)
         if dic=dic1 then inc(nOk)
                     else Teach(dic,dic1);  (* обучение    *)
         (* вывод текущей информации на экран *)
         writeln('Шаг ', Step, ' : Доля удачных ответов ',
                  nOk/Step*100:6:2);
         (* тестирование за 20 шагов до конца обучения *)
         if Step=(lmax - 20) then fTest:=true;
     end;
     writeln('Конец');
end.
```

Задания к лабораторной работе

На основе приведенных выше примеров реализовать и отладить программу распознавания нескольких образов в соответствии с заданным вариантом и применением двоичнго кодирования образов.

Подготовить отчет, включающий в себя: код программы, описание основных функций, результаты обучения перцептрона, выводы по результатам исследования, ответы на контрольные вопросы.

№	Задание
1	Реализовать распознавание цифр "**1**", "**2**", "**3**", "**4**", "**5**", "**6**"
2	Реализовать распознавание символов "**А**", "**Б**", "**Г**", "**Е**"
3	Реализовать распознавание цифр "**2**", "**3**", "**4**", "**5**", "**6**", "**7**"
4	Реализовать распознавание символов "**Б**", "**Г**", "**Е**", "**И**"
5	Реализовать распознавание цифр "**3**", "**4**", "**5**", "**6**", "**7**", "**8**"
6	Реализовать распознавание символов "**Г**", "**Е**", "**И**", "**К**"
7	Реализовать распознавание цифр "**4**", "**5**", "**6**", "**7**", "**8**", "**9**"
8	Реализовать распознавание символов "**Е**", "**И**", "**К**", "**Л**"
9	Реализовать распознавание цифр "**5**", "**6**", "**7**", "**8**", "**9**", "**0**"
10	Реализовать распознавание символов "**И**", "**К**", "**Л**", "**Н**"
11	Реализовать распознавание цифр "**6**", "**7**", "**8**", "**9**", "**0**", "**1**"
12	Реализовать распознавание символов "**К**", "**Л**", "**Н**", "**О**"
13	Реализовать распознавание цифр "**7**", "**8**", "**9**", "**0**", "**1**", "**2**"
14	Реализовать распознавание символов "**Л**", "**Н**", "**О**", "**П**"
15	Реализовать распознавание цифр "**8**", "**9**", "**0**", "**1**", "**2**", "**3**"

№	Задание
16	Реализовать распознавание символов **"Н"**, **"О"**, **"П"**, **"С"**
17	Реализовать распознавание цифр **"9"**, **"0"**, **"1"**, **"2"**, **"3"**, **"4"**
18	Реализовать распознавание символов **"О"**, **"П"**, **"С"**, **"Т"**
19	Реализовать распознавание цифр **"0"**, **"1"**, **"2"**, **"3"**, **"4"**, **"5"**
20	Реализовать распознавание символов **"П"**, **"С"**, **"Т"**, **"Х"**
21	Реализовать распознавание знаков **"#"**, **"="**, **"!"**,**"^"**
22	Реализовать распознавание символов **"С"**, **"Т"**, **"Х"**, **"Ч"**
23	Реализовать распознавание скобок **"<"**, **">"**, **"["**,**"]"**
24	Реализовать распознавание символов **"Т"**, **"Х"**, **"Ч"**, **"Ш"**

Контрольные вопросы

1) Сколько эффекторных клеток необходимо для распознавания нескольких образов при двоичном кодировании образов?
2) Как меняется число слоев перцептрона при увеличении числа распознаваемых образов?
3) Как изменяется число связей при увеличении числа распознаваемых образов с использованием двоичного кодирования?
4) По каким принципам кодируется распознаваемый образ?
5) Можно ли определить схожесть распознаваемых образов?
6) Какие требования предъявляются к обучающей выборке?
7) Какой алгоритм используется для обучения перцептрона?
8) Какие значения влияют на качество обучения перцептрона?
9) На что влияет увеличение числа распознаваемых образов?
10) Можно ли совмещать веса перцептронов, обученных для распознавания различных образов при двоичном кодировании?
11) Назовите признаки отсутствия обучения перцептрона.
12) Какие веса связей подлежат обучению?
13) От чего зависит размер веса обучения?
14) Как влияет число шагов обучения на качество обученности перцептрона?
15) Как влияет размер образа на способность перцептрона к обучению?
16) Как правильно оценить качество обученности перцептрона?

Список рекомендуемой литературы

1. **Уоссермен, Филипп.** Нейрокомпьютерная техника : Теория и практика / Ф. Уоссермен; Перевод с англ. Ю. А. Зуева, В. А. Точенова; Под ред. А. И. Галушкина. – М. : Мир, 1992. – 236,[1] с. : ил.; 22 см.; ISBN 5-03-002115-9 : Б. ц.
2. Нейрокомпьютеры. Устройство, работа, моделирование на ПК: методические указания / С.Г. Сидоров [и др.] ; Министерство образования Российской Федерации, Ивановский государственный энергетический университет, Лаборатория быстрых алгоритмов Российской академии наук, Московский государственный университет коммерции, Ивановский филиал, Ивановская государственная текстильная академия ; ред. Н. А. Коробов. – Иваново: Б.и., 2002. – 24 с: ил.
3. **Розенблатт, Фрэнк.** Принципы нейродинамики [Текст] : Перцептроны и теория механизмов мозга / Перевод с англ. В. Я. Алтаева [и др.] ; Под ред. [и с предисл.] д-ра физ.-мат. наук С. М. Осовца. - Москва : Мир, 1965. – 480 с. : черт.; 22 см.
4. **Хайкин, Саймон.** Нейронные сети : полный курс / Саймон Хайкин ; [пер. с англ. Н. Н. Куссуль, А. Ю. Шелестова]. – Изд. 2-е, испр. – Москва [и др.] : Вильямс, 2008. – 1103 с. : ил.; 24 см.; ISBN 978-5-8459-0890-2 (В пер.)
5. **Ростовцев, В. С.** Искусственные нейронные сети : учебник для вузов / В. С. Ростовцев. – 2-е изд., стер. – Санкт-Петербург : Лань, 2021. – 216 с. – ISBN 978-5-8114-7462-2. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/160142 (дата обращения: 08.11.2021). – Режим доступа: для авториз. пользователей.
6. **Данилов, В. В.** Нейронные сети : учебное пособие / В. В. Данилов. – Донецк : ДонНУ, 2020. – 158 с. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/179953 (дата обращения: 08.11.2021). – Режим доступа: для авториз. пользователей.
7. **Филиппов, Ф. В.** Моделирование нейронных сетей глубокого обучения : учебное пособие / Ф. В. Филиппов. – Санкт-Петербург : СПбГУТ им. М.А. Бонч-Бруевича, 2019. – 79 с. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/180053 (дата обращения: 08.11.2021). – Режим доступа: для авториз. пользователей.

ПЕРЦЕПТРОНЫ
Методические указания

Составитель СИДОРОВ Сергей Георгиевич

Редактор Т.В. Соловьева

Подписано в печать 17.11.2021 Формат 60х84 1/16
Печать плоская. Усл.печ.л. 2,32. Тираж 100 экз. Заказ № 53.

ФГБОУ ВО «Ивановский государственный энергетический университет имени В.И. Ленина»
Отпечатано в УИУНЛ ИГЭУ
153003 г.Иваново, ул.Рабфаковская, 34