

## **Цель и задачи практических работ**

Цель - овладение студентами современной методологии проектирования Internet-ориентированной обучающейся интеллектуальной системы.

В результате выполнения практических работ студент должен знать:

- основные принципы и методы проектирования Internet-ориентированной обучающейся интеллектуальной системы;
- алгоритмы функционирования современных обучающихся интеллектуальных систем;

В результате выполнения практических работ студент должен уметь:

- составлять входное математическое описание распределенных обучающихся интеллектуальных систем;
- составлять математические модели функционирования обучающихся интеллектуальных систем;
- оптимизировать параметры функционирования обучающихся интеллектуальных систем;
- разработать специальное программное обеспечение обучающихся интеллектуальных систем.

### **Установка PHP, Apache 1.3.x под Windows**

На сайте разработчиков: <http://http.apache.org/> выбрать оригинальный установочный пакет *Apache 1.3.x*. Для пользователей Win98 нужно обновить *Winsock* в *Winsock2*. Для начала лучше взять бинарный дистрибутив *Apache 1.3.x* без файлов разработчиков (MSI инсталлятор).

После скачивания запустить файл с расширением *.msi*, который выводит информацию: «Запускать сервер как сервис, или как самостоятельную программу?»

Если выбираем "как сервис", Apache 1.3.x будет стартовать каждый раз после запуска компьютера, его не будет видно в панели задач и он будет работать даже после изменения пользователей операционной системы. Работает это только под *WinNT* и *Win2000*.

Если выбрать запуск Apache 1.3.x как "самостоятельной программы", тогда каждый раз Apache 1.3.x придется запускать вручную, при этом будет видно консоль Apache 1.3.x и соответствующий ярлык в панели задач.

Если Вы ставите Apache 1.3.x в директорию, где уже находятся файлы предыдущего Apache 1.3.x, конфигурационные файлы не будут перезаписаны. Новые конфигурационные файлы будут созданы с двойным расширением *.default.conf*.

**Например**, если уже существует файл *conf \ httpd.conf* будет создан файл *conf \ httpd.default.conf*, а предыдущий файл будет оставлено без изменений.

**Запуск в консоли.** Откройте стартовое меню и выберите "Запустить апач в консоли".

**Запуск сервиса Apache 1.3.x.** Если Apache 1.3.x инсталлювался как сервис для всех пользователей, то его можно запустить, используя панель управления сервисами в *Win2000*. Также Вы можете в консоли набрать "*NET START APACHE*". Таким образом запустится сервер с конфигурацией по умолчанию. Можно запустить несколько сервисов Apache под разными именами и с разными конфигурациями.

**Проверка работоспособности Apache 1.3.x.** Наберите в браузере: *http: // localhost*. Если все работает хорошо, то вы должны увидеть страницу файла помощи Apache 1.3.x. Иначе нужно посмотреть файл статистики *logs \ error.log*. Одной из основных причин отказа Apache 1.3.x работать нормально то, что

80 - й порт, который по умолчанию слушает Apache 1.3.x, может быть занят другим сервисом, необходимо проверить в первую очередь.

**Инсталляционный пакет PHP.** Скачайте установочный пакет: <http://www.PHP.net/downloads.PHP>. Есть два варианта установки пакетов: пакет с файлом для автоматической установки и архив файлов.

В первом случае, для установки требуется запустить соответствующий файл, который установит *php*, создаст все необходимые файлы и настроит веб-сервер. Вся необходимая информация будет востребована у пользователя в режиме диалога. Но не все расширения устанавливаются в автоматическом режиме, и конфигурация, которая будет создана, не является оптимальной с точки зрения безопасности.

Поскольку все расширения в наличии в архивном файле, то второй вариант установки не намного сложнее. А именно: распаковываем архив на диск и копируем *PHP4ts.dll* в директорию, указанную в *PATH Windows (c: \ winnt \ system32)*. Это необходимо делать в случае, если вы планируете использовать *PHP* как *CGI*, так и в случае использования *PHP* как модуля веб-сервера.

Если планируется использовать *PHP* в форме модуля веб-сервера, скопируйте *dll* соответствующих *SAPI*, а если это будет апач, то файл *PHP4apache.dll* копируем в *c: \ winnt \ system32*.

Далее скопируйте файл *PHP.ini-dist* или *PHP.ini-optimized* в директорию *c: \ winnt*, переименовав его предварительно в *PHP.ini* и соответствующим образом отредактировав.

Измените значение переменной *extension\_dir* таким образом, чтобы она указывала на директорию, куда вы установили *PHP* (то есть на директорию, где лежат \* *.dll* файлы). Например, *c: \ PHP \ extensions*.

Сделайте соответствующие изменения в настройках Apache 1.3.x. Эти шаги являются общими для *Unix* и *Windows*. После установки *PHP* надо перезапустить Apache 1.3.x.

## Практическое занятие 1. Формирование входной обучающей матрицы

Цель: разработать и программно реализовать алгоритм формирования входной обучающей матрицы.

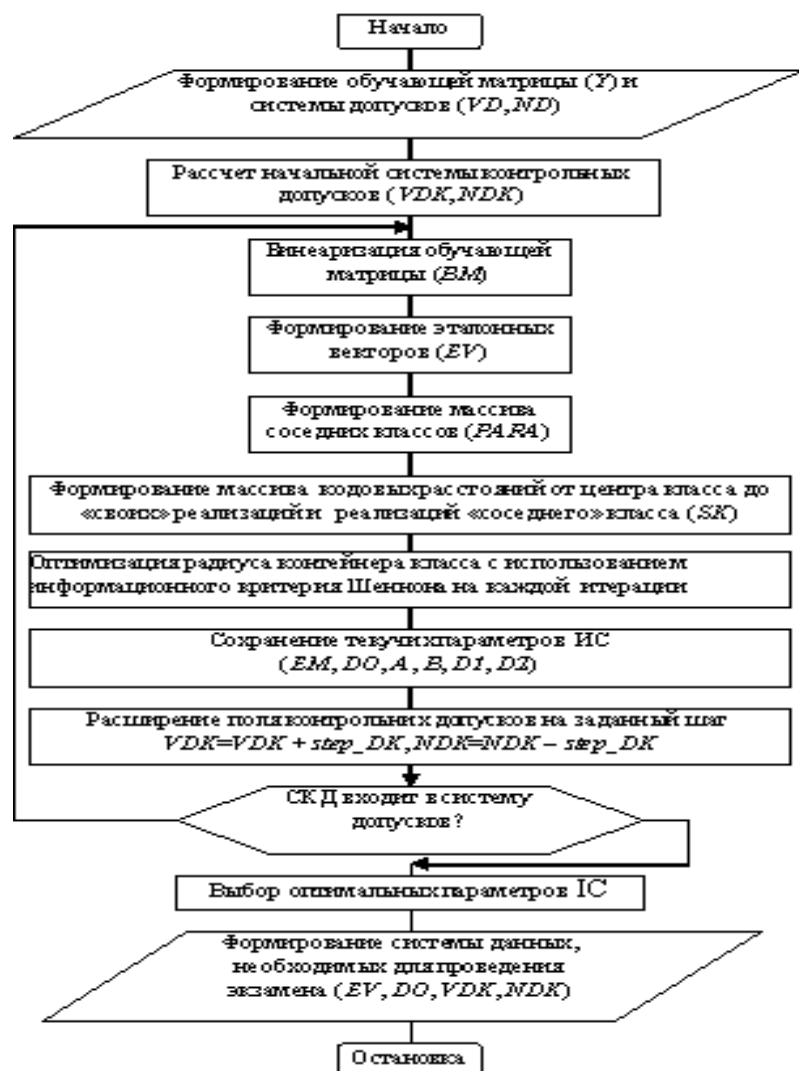


Рисунок 1. – Структурная схема базового алгоритма обучения

Назначением базового алгоритма обучения является оптимизация геометрических параметров контейнеров. Входной информацией для обучения по базовым алгоритмом является действительный, в общем случае, массив реализаций образа  $\{y_m^{(j)} \vee m = \overline{1, M}; j = \overline{1, n}\}$ ; система полей контрольных допусков  $\{\delta_{K,i}\}$  и уровень селекции  $\{\rho_m\}$ , которые по умолчанию равны 0,5 для всех классов распознавания.

Рассмотрим этапы реализации алгоритма:

1. Формирование бинарной обучающей матрицы  $\|x_{m,i}^{(j)}\|$ , элементы которой равны

$$x_{m,i}^{(j)} = \begin{cases} 1, & \text{if } y_{m,i}^{(j)} \in \delta_{K,i}, \\ 0, & \text{if } y_{m,i}^{(j)} \notin \delta_{K,i}. \end{cases}$$

2. Формирование массива эталонных двоичных векторов  $\{x_{m,i} \vee m = \overline{1, M}, i = \overline{1, N}\}$ , элементы которого определяются по правилу :

$$x_{m,i} = \begin{cases} 1, & \text{if } \frac{1}{n} \sum_{j=1}^n x_{m,i}^{(j)} > \rho_m, \\ 0, & \text{if } \text{else}, \end{cases}$$

где  $\rho_m$  – уровень селекции координат вектора  $x_m \in X_m^o$ .

3. Разбиение множества эталонных векторов на пары ближайших “сосудов”:  $\Re_m^{[2]} = \langle x_m, x_l \rangle$ , где  $x_l$  – эталонный вектор соседнего класса  $X_l^o$ , по такому алгоритму:

а) структурируется множество эталонных векторов, начиная с вектора  $x_1$  базового класса  $X_1^o$ , характеризующий наибольшую функциональную эффективность СППР;

б) строится матрица кодовых расстояний между эталонными векторами размерности  $M \times M$ ;

в) для каждой строки матрицы кодовых расстояний находится минимальный элемент, который принадлежит столбцу вектора - ближайшего к вектору, который определяет строку. При наличии нескольких одинаковых минимальных элементов выбирается из них любой, поскольку они являются равноправными;

г) формируется структурированное множество элементов парного разбиения  $\{\mathfrak{R}_m^{[2]} \mid m = \overline{1, M}\}$ , задающее план обучения.

4. Оптимизация кодового расстояния  $d_m$  происходит рекуррентной процедурой. При этом принимается  $E_m(0)=0$ .

5. Процедура заканчивается при нахождении максимума КФЭ в рабочей области его определения  $E_m^* = \max_{\{d\}} E_m$ , где

$\{d\} = \{0, 1, \dots, d < d(x_m \oplus x_l)\}$  - множество радиусов гиперсфер, центр которых определяется вершиной  $x_m \in X_m^o$ .

Таким образом, базовый алгоритм обучения является итерационной процедурой поиска глобального максимума информационного КФЭ в рабочей области определения его функции

$$d_m^* = \arg \max_{\{d\}} E_m^* .$$

**Задание 1** Провести описание основных констант и переменных согласно табл.1.

Таблица 1– Основные константы и переменные

|   |
|---|
| <b><math>m</math></b> – количество классов ( $k = 2$ )                          |
| <b><math>N</math></b> – количество признаков <i>распознавания</i> ( $N = 100$ ) |
| <b><math>n</math></b> – количество реализаций ( $n = 100$ )                     |
| <b><math>k</math></b> – текущий класс   |

|  |
|--|
| $I, J$ – текущие признак и реализация в соответствии   |
| $Y[1..m, 1..N, 1..n]$ – входная обучающая матрица  |
| продолжение табл.1   |
| $X[1..m, 1..N, 1..n]$ – бинарная учебная матрица   |
| $VD[1..N], ND[1..N]$ –система допусков   |
| $VDK[1..N], NDK[1..N]$ –система контрольных допусков   |
| $EV[1..m, 1..N]$ – центры классов  |
| $PARA[1..m]$ –массив соседних классов  |
| $SK[1..2, 1..n]$ – массив кодовых расстояний до реализаций   |
| $E[1..m], A[1..m], B[1..m], D1[1..m], D2[1..m]$ – оптимальное значение информационного критерия и точностные характеристики соответственно |
| $DO[1..m]$ – оптимальное значение радиуса контейнера класса  |
| $step\_DK$ – шаг изменения системы контрольных допусков  |

**Задание 2.** Сформировать согласно приведенному примеру входную учебную матрицу  $Y[1..m, 1..N, 1..n]$  для распознавания стационарных по яркости изображений (текстур).

**Задание 3.** Разработать предварительный интерфейс системы. Основное внимание при этом уделить созданию диалогового режима с пользователями.



## Порядок выполнения работы

1. Записать тему и цель работы.
2. Ознакомиться с базовым алгоритмом обучения интеллектуальной системы, структурная схема которого показана на рис.1.
3. Описать основные константы и переменные согласно табл. 1.
4. В соответствии с вариантом задания скопировать два изображения из библиотеки изображений типа текстуры [1].
5. Сформировать согласно приведенному примеру входную обучающую матрицу  $X[1, \dots, m, 1, \dots N, 1, \dots n]$  для распознавания изображений (текстур).
6. Разработать интерактивный интерфейс программной системы.

## Пример формирования обучающей матрицы



```
<?php
function s4it($id,$delta,$radius)
{
    $image1=imageCreateFromJpeg("image/".$id.".jpg");
    $file1=fopen("matr/".$id.".txt",w);
    $file3=fopen("matr/sr_etal ".$id.".txt",w);
    $file5=fopen("matr/vdk ".$id.".txt",w);
    $file7=fopen("matr/ndk ".$id.".txt",w);
    $file9=fopen("matr/bin_matr ".$id.".txt",w);
    $file11=fopen("matr/etalon ".$id.".txt",w);
    for ($i=0; $i<100; $i++)
    {
        for ($j=0; $j<100; $j++){
            //---формирование обучающей матрицы и ее запись в файл---
            $rgb1=imageColorAt($image1, $i, $j);
            list($r1, $g1, $b1) = array_values(imageColorsForIndex($image1,
            $rgb1));
            $l1=round((1/3)*($r1+$g1+$b1));
            $_SESSION['obuch_matr'][$id][$j][$i]=$l1;
            fwrite($file1,"$l1");
            if ($j<99)
                fwrite($file1," ");
        }
    }
}
```

```

    }
    fwrite($file1, "\r\n");
}
}
//---вывод обучающей матрицы---
function obu4_vivod($id)
{
    for ($i=0; $i<100; $i=$i+10)
    {
        for ($j=0; $j<100; $j=$j+10)
        {
            $temp=$_SESSION['obuch_matr'][$id][$i][$j];
            echo "<font size=\"2\">$temp..\t\t</font>";
        }
        echo "<br>";
    }
}
?>

```

**Графическое отображение изображений и обучающей матрицы  
для двух классов распознавания**

|   |   |
|---|---|
| <p>Рисунки 1,2</p>   | <p>Рисунки 1,3</p>   |
| <p>Начальная матрица 1</p> <pre> 207 50 100 104 85 40 42 701 74 701 40 6 206 101 126 171 66 65 156 71 85 12 19 50 38 109 14 77 100 71 162 70 5 117 129 41 134 70 107 38 100 20 0 37 143 76 102 156 106 71 51 136 22 46 9 193 135 73 170 701 143 170 38 39 100 138 134 38 100 61 1 5 101 111 47 56 170 135 137 3 173 86 16 106 6 44 120 110 106 101 101 45 146 27 120 69 145 136 134 27 </pre> | <p>Начальная матрица 2</p> <pre> 129 127 81 140 124 105 157 110 104 106 100 170 165 104 56 146 82 152 154 61 140 31 100 100 130 132 140 107 103 55 152 145 128 102 167 177 102 129 80 100 106 65 177 110 136 146 55 100 140 172 52 69 73 100 143 131 61 130 100 70 101 130 120 104 100 156 135 25 110 120 104 96 84 85 124 126 120 100 61 72 120 170 38 140 100 42 74 61 110 50 101 107 60 173 65 54 71 107 172 56 </pre> |

### Контрольные вопросы

1. Что такое стационарное изображение? В чем заключается его отличие от нестационарного?
2. Какие различия имеют алгоритмы создания обучающей матрицы со стационарными и нестационарными изображениями?
4. Какой из компонентов РНР применяется для обработки изображений? Какие методы и свойства этих компонентов были использованы?
5. Какие компоненты РНР были использованы при создании интерфейса системы?

## Практическое занятие 2. Формирование бинарной обучающей матрицы

**Цель** —разработать и программно реализовать алгоритм формирования бинарной обучающей матрицы.

### Теоретические сведения

Рассмотрим обучающую матрицу типа «объект-свойство», которая характеризует  $m$ -ое функциональное состояние СПР - класс распознавания  $X_m^o$  :

$$||y_{m,i}^{(j)}|| = \begin{vmatrix} y_{m,1}^{(1)} & y_{m,2}^{(1)} & \dots & y_{m,1}^{(1)} & \dots & y_{m,N}^{(1)} \\ y_{m,1}^{(2)} & y_{m,2}^{(2)} & \dots & y_{m,1}^{(2)} & \dots & y_{m,N}^{(2)} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ y_{m,1}^{(j)} & y_{m,2}^{(j)} & \dots & y_{m,1}^{(j)} & \dots & y_{m,N}^{(j)} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ y_{m,1}^{(n)} & y_{m,2}^{(n)} & \dots & y_{m,1}^{(n)} & \dots & y_{m,N}^{(n)} \end{vmatrix} \quad (1)$$

Элементами этой матрицы являются экспериментальные данные, отражающие основные свойства объекта и, в общем случае являются значениями случайной величины - признаки распознавания. В матрице (1) строка является реализацией образа  $\{y_{m,i}^{(j)} \vee i = \overline{1, N}\}$ , где  $N$  - количество признаков распознавания, а столбец матрицы - случайная обучающая выборка  $\{y_{m,i}^{(j)} \vee j = \overline{1, n}\}$ , где  $n$  - объем выборки, которая формируется в процессе испытаний.

Для изображения обучающаяся матрица формируется путем сканирования рецепторного поля и получения в каждом пикселе значения яркости, которое для черно-белого графического редактора изменяется от 0 до 255 градаций яркости. Таким образом, матрица яркости является целой и рассматривается как входная матрица для системы распознавания.

**Задание1.** Сформировать бинарную обучающуюся матрицу  $X[1..m, 1..N, 1..n]$  путем преобразования обучающейся матрицы  $Y[1..m, 1..N, 1..n]$  в бинарное пространство Хемминга.

**Задание2.** Разработать модуль для графического отображения бинарной обучающейся матрицы.

### Порядок выполнения работы

1. Записать тему и цель работы.
2. Задать значение поля контрольных допусков ( $\delta = \pm 50$ ).
3. Перевести обучающуюся матрицу  $Y[1..m, 1..N, 1..n]$  в бинарное пространство Хемминга по правилу:

$$X[k, i, j] = \begin{cases} 1, & \text{если } NDK[i] \leq Y[k, i, j] \leq VDK[i], \\ 0, & \text{якщо } NDK[i] \geq Y[k, i, j] \text{ или } Y[k, i, j] \geq VDK[i], \end{cases}$$

где  $NDK[i]$ ,  $VDK[i]$  – нижний и верхний контрольные допуски  $i$ -го признака соответственно.

4. Разработать интерактивный интерфейс программной системы.

### Пример функции формирования бинарной матрицы

```
for ($i=0; $i<100; $i++){
    for ($j=0; $j<100; $j++){
        //---формирование обучающейся матрицы и запись ее в файл---
        $rgb1=imageColorAt($image1, $i, $j);
        list($r1, $g1, $b1) = array_values(imageColorsForIndex($image1,
        $rgb1));
        $l1=round((1/3)*($r1+$g1+$b1));
        $_SESSION['obuch_matr'][$i][$j][$i]=$l1;
        fwrite($file1, "$l1");
        if ($j<99)
            fwrite($file1, " ");
    }
}
```

```

        fwrite($file1, "\r\n");
    }
    //---расчет средних значений---
    for ($j=0; $j<100; $j++){
        $s1_elem=0;
        for ($i=0; $i<100; $i++){
            $s1_elem=$s1_elem+$_SESSION['obuch_matr'][$id][$i][$j];
        }
        $sr_etal1[$j]=round((1/100)*$s1_elem);
        fwrite($file3, "$sr_etal1[$j] ");
    }

    //---расчет контрольных допусков---
    $vdk_1[$j]=$sr_etal1[$j]+$delta;
    $ndk_1[$j]=$sr_etal1[$j]-$delta;
    $_SESSION['vdk'][$id][$j]=$vdk_1[$j];
    $_SESSION['ndk'][$id][$j]=$ndk_1[$j];
    fwrite($file5, "$vdk_1[$j] ");
    fwrite($file7, "$ndk_1[$j] ");
}

//---расчет бинарной матрицы и эталонного вектора---
for ($j=0; $j<100; $j++){
    $kol1_1=0;
    $kol1_0=0;
    for ($i=0; $i<100; $i++){
        $zom=$_SESSION['obuch_matr'][$id][$i][$j];
        if (($zom>=$ndk_1[$j]) && ($zom<=$vdk_1[$j])){
            $_SESSION['bin_matr'][$id][$i][$j]=1;
            $kol1_1=$kol1_1+1;
        } else {
            $_SESSION['bin_matr'][$id][$i][$j]=0;
            $kol1_0=$kol1_0+1;
        }
    }
    if (($kol1_1/100)>$radius){
        $_SESSION['etalon'][$id][$j]=1;
    } else {
        $_SESSION['etalon'][$id][$j]=0;
    }
    $str=$_SESSION['etalon'][$id][$j];
    fwrite($file11,$str);
}

```

```

        if ($j<99) fwrite($file11," ");
    }
    for ($i=0; $i<100; $i++){
        for ($j=0; $j<100; $j++){
            $znac=$_SESSION['bin_matr'][$id][$i][$j];
            fwrite($file9,"$znac");
            if ($j<99) fwrite($file9," ");
        }
        fwrite($file9,"\r\n");
    }
fclose($file1);
fclose($file3);
fclose($file5);
fclose($file7);
fclose($file9);
fclose($file11);
}
//---вывод VDK---
function dopusk_VDK_vivod($id){
    for ($i=0; $i<100; $i=$i+10){
        $temp=$_SESSION['vdk'][$id][$i];
        echo "<font size=\"2\">$temp.</font>";
    }
    echo "<br>";
}
//---вывод NDK---
function dopusk_NDK_vivod($id){
    for ($i=0; $i<100; $i=$i+10){
        $temp=$_SESSION['ndk'][$id][$i];
        echo "<font size=\"2\">$temp.</font>";
    }
    echo "<br>";
}
//---вывод бинарной матрицы--
function bin_vivod($id){
    for ($i=0; $i<100; $i=$i+10){
        for ($j=0; $j<100; $j=$j+10){
            $temp=$_SESSION['bin_matr'][$id][$i][$j];
            echo "<font size=\"2\">$temp.</font>";
        }
    }
}

```

```

        echo "<br>";
    }
}
//---изображение бинарной матрицы---
function bin_izobr($id){
    $im1=imagecreate(100,100);

    $black1 = imagecolorallocate ($im1, 0, 0, 0);
    $white1 = imagecolorallocate ($im1, 255, 255, 255);


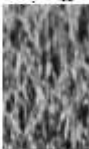


    for ($i=0; $i<100; $i++){
        for ($j=0; $j<100; $j++){
            if ($_SESSION['bin_matr'][$id][$i][$j]==1){
                imageSetPixel($im1,$i,$j,$black1);
            } else {
                imageSetPixel($im1,$i,$j,$white1);
            }
        }
    }
    imagejpeg ($im1,"image/bin_". $id." .jpg");
}

```

## Графическое отображение бинарной матрицы



## для двух классов распознавания

|  |   |
|--|---|
| Рисунок 2.1<br>   | Рисунок 2.2<br>  |
| Начальная матрица 1<br>207 50 193 164 39 69 42 211 24 211<br>40 6 296 110 126 121 16 63 136 75<br>85 13 15 59 38 139 14 22 190 31<br>162 70 5 112 139 41 114 70 187 20<br>199 39 0 32 143 76 182 156 166 21<br>51 138 33 46 9 151 123 73 128 211<br>143 178 39 39 182 138 124 28 190 61<br>1 5 221 111 47 56 178 135 133 3<br>173 86 16 196 6 44 150 118 196 111<br>19 45 146 27 138 69 145 136 154 27 | Начальная матрица 2<br>138 127 85 140 134 105 152 110 164 106<br>160 170 163 104 96 146 83 151 154 65<br>141 31 119 166 138 132 140 117 113 29<br>153 145 138 102 165 177 102 131 98 101<br>106 93 171 110 136 146 33 168 148 172<br>52 90 72 108 143 131 91 138 189 70<br>101 128 120 104 208 156 135 95 110 151<br>194 96 84 85 134 116 133 160 84 72<br>150 178 38 149 109 42 74 61 118 50<br>101 187 68 173 63 54 71 187 172 36 |
| Допуски VDK первого класса   | Допуски VDK второго класса  |
| Допуски NDK первого класса   | Допуски NDK второго класса  |
| Бинарная матрица 1<br>0 1 0 0 1 0 0 0 0 0<br>0 0 0 1 0 1 0 1 1 1<br>1 0 0 0 0 1 0 0 0 0<br>0 1 0 1 0 0 1 1 0 0<br>0 1 0 0 0 0 0 1 0<br>0 0 0 0 0 1 1 1 0<br>0 0 0 0 0 1 1 0 0 1<br>0 0 0 1 0 0 0 0 1 0<br>0 1 0 0 0 0 0 1 0 1<br>0 1 0 0 0 0 1 0 1 0   | Бинарная матрица 2<br>1 1 1 1 1 0 0 1 1 1<br>0 0 0 1 0 1 1 1 1 1<br>1 0 1 0 1 1 0 1 1 0<br>1 1 0 1 1 0 1 1 0 1<br>1 1 0 1 1 0 0 1 0<br>0 1 1 1 1 1 1 0 1<br>1 1 1 0 1 0 1 1 0<br>0 1 1 1 1 0 1 0 1<br>1 0 0 1 1 0 1 0 1 0<br>1 0 1 0 0 0 1 0 0 0  |
| Графичне відображення бінарної матриці першого класу<br>  | Графичне відображення бінарної матриці другого класу<br>   |

## Контрольные вопросы

1. Какой тип переменных используется для описания системы контрольных допусков?
2. С какой целью проводится перевод обучающейся матрицы в бинарное пространство Хемминга?

3. Какой компонент РНР использован для графического отображения бинарной матрицы?
4. Что называется контрольным полем допусков на признаки распознавания?
5. Что называется нормированным полем допусков на признаки распознавания?

### Практическое занятие 3. **Определение эталонных геометрических векторов классов распознавания**

**Цель** — научиться формировать геометрические центры классов распознавания.

**Задание 1.** Сформировать массив  $EV [1..m, 1..N]$  геометрических центров классов распознавания, где  $m$  - количество классов,  $N$  - количество признаков распознавания.

#### Теоретические сведения

Эталонный вектор  $x_m$  - это математическое ожидание реализаций класса  $X_m^0$ . Он представлен в виде детерминированного структурированного бинарного вектора  $x_m = \langle x_m, 1, \dots, x_m, i, \dots, x_m, N \rangle, \dots, m = \overline{1, M}$ , где  $M$  - количество классов распознавания,  $N$  - количество признаков,  $x_m, i$  - итая координата вектора, которая принимает единичное значение, если значение  $i$ -го признака распознавания находится в нормированном поле допусков  $\delta_{H,i}$ , и нулевое значение, если не находится.

Элементы  $EV[I, K]$  — двоичного эталонного вектора  $K$  - го класса  $X_1^K$  вычисляются по правилу:

$$EV[I,1] = \begin{cases} 1 & \text{if } \frac{1}{IMAX} \sum_{I=1}^{IMAX} X[J, I, 1] > 0,5; \\ 0 & \text{if else.} \end{cases}$$

При обосновании гипотезы компактности (четкой, или нечеткой) реализаций образа геометрическим центром класса  $X_m^0$  является бинарный эталонный вектор  $x_m$ .

### Порядок выполнения работы

1. Записать тему и цель работы.
2. Сформировать массив  $EV$  [1..m, 1..N] геометрических центров классов распознавания. При этом значение каждого элемента массива  $EV$  [k, i] рассчитывается по правилу:

$$EV[k, i] = \begin{cases} 1, & \text{если } \frac{1}{n} \sum_{j=1}^n X[k, i, j] > 0,5, \\ 0, & \text{если } \frac{1}{n} \sum_{j=1}^n X[k, i, j] \leq 0,5. \end{cases}$$

3. Разработать интерактивный интерфейс программной системы.

### Пример формирования эталонных векторов классов распознавания

```
//---расчет бинарной матрицы и эталонного вектора
for ($j=0; $j<100; $j++){
    $kol1_1=0;
    $kol1_0=0;
    for ($i=0; $i<100; $i++){
        $zom=$_SESSION['obuch_matr'][$i][$j];
        if (($zom>=$ndk_1[$j]) && ($zom<=$vdk_1[$j])){
            $_SESSION['bin_matr'][$i][$j]=1;
            $kol1_1=$kol1_1+1;
        } else {
            $_SESSION['bin_matr'][$i][$j]=0;
            $kol1_0=$kol1_0+1;
        }
    }
}
```

```

        }
    }
    if (($kol1_1/100)>$radius)
    {
        $_SESSION['etalon'][$id][$j]=1;
    } else {
        $_SESSION['etalon'][$id][$j]=0;
    }
    $str=$_SESSION['etalon'][$id][$j];
    fwrite($file11,"$str");
    if ($j<99)
        fwrite($file11," ");
}
for ($i=0; $i<100; $i++)
{
    for ($j=0; $j<100; $j++)
    {
        $znac=$_SESSION['bin_matr'][$id][$i][$j];
        fwrite($file9,"$znac");
        if ($j<99)
            fwrite($file9," ");
        }
        fwrite($file9,"\r\n");
    }
}

//---вывод эталонного вектора--
function etal_vivod($id)
{
    for ($i=0; $i<100; $i=$i+10)
    {
        $temp=$_SESSION['etalon'][$id][$i];
        echo "<font size=\"2\">$temp..</font>";
    }
    echo "<br>";
}

//---изображение эталонного вектора ---
function et_izobr ($id)
{
    $im1=imagecreate(100,10);







```

```

$black1 = imagecolorallocate ($im1, 0, 0, 0);
$white1 = imagecolorallocate ($im1, 255, 255, 255);
for ($j=0; $j<10; $j++)
{
    for ($i=0; $i<100; $i++)
    {
        if ($_SESSION['etalon'][$id][$i]==1){
            imageSetPixel($im1,$i,$j,$black1);
        } else {
            imageSetPixel($im1,$i,$j,$white1);
        }
    }
}
imagejpeg ($im1,"image/etalon_". $id.".jpg");
}

```

### Графическое отображение эталонных векторов классов распознавания

|  |   |
|--|---|
| Рисунок 11<br>  | Рисунок 12<br>   |
| Начальна матриця 11<br>207.50.198.164.89.69.42.211.24.211.<br>40.6.216.110.126.121.16.63.136.75.<br>83.13.13.39.38.139.14.22.190.31.<br>163.70.5.112.159.41.114.70.187.20.<br>199.38.0.32.143.76.182.156.166.21.<br>51.138.33.46.9.152.123.73.129.211.<br>143.178.39.39.182.130.124.28.190.61.<br>1.5.221.111.47.36.178.135.133.3.<br>173.86.16.196.6.44.150.118.196.111.<br>19.45.146.27.138.69.145.136.134.27. | Начальна матриця 12<br>139.127.83.140.134.185.152.110.164.106.<br>160.170.163.104.96.146.83.151.154.63.<br>141.31.119.166.138.152.140.117.113.23.<br>153.145.138.182.165.177.102.131.94.101.<br>106.93.171.110.136.146.33.168.148.172.<br>52.90.72.108.143.131.91.138.189.70.<br>101.138.129.184.200.156.135.95.110.151.<br>194.96.84.85.134.116.133.161.84.72.<br>138.178.38.148.189.42.74.61.118.50.<br>101.187.68.173.63.54.71.187.172.56. |
| Допуски VDK першого класу  | Допуски VDK другого класу   |
| Допуски NDK першого класу  | Допуски NDK другого класу   |
| Бінарна матриця 1<br>0.1.0.0.1.0.0.0.0.<br>0.0.0.1.0.1.0.1.1.1.<br>1.0.0.0.0.1.0.0.0.0.<br>0.1.0.1.0.0.1.1.0.0.<br>0.1.0.0.0.0.0.1.0.<br>0.0.0.0.0.1.1.1.0.<br>0.0.0.0.1.1.0.0.1.<br>0.0.0.1.0.0.0.0.1.0.<br>0.1.0.0.0.0.1.0.1.<br>0.1.0.0.0.1.0.1.0.  | Бінарна матриця 2<br>1.1.1.1.0.0.1.1.1.<br>0.0.0.1.0.1.1.1.1.<br>1.0.1.0.1.0.1.1.0.<br>1.1.0.1.0.1.1.0.1.<br>1.1.0.1.1.0.0.1.0.<br>0.1.1.1.1.1.1.0.1.<br>1.1.1.1.0.1.0.1.0.<br>0.1.1.1.1.0.1.0.1.<br>1.0.0.1.1.0.1.0.<br>1.0.1.0.0.1.0.0.0.   |
| Графічне відображення бінарної матриці першого класу<br>   | Графічне відображення бінарної матриці другого класу<br>  |
| Еталонний вектор 1<br>0.1.0.0.0.0.0.0.0.   | Еталонний вектор 2<br>1.1.0.1.1.1.1.1.0.  |
| Графічне відображення масиву геометричних центрів першого класу<br>   | Графічне відображення масиву геометричних центрів другого класу<br>  |

## Контрольные вопросы

1. Что называется эталонным вектором класса распознавания и как он формируется?

2. Какой компонент РНР был использован для корректного отображения геометрических центров классов обучения?

#### **Практическое занятие 4. Формирование массива кодовых расстояний между центрами классов распознавания**

**Цель** - разработать и программно реализовать алгоритм формирования массива кодовых расстояний между центрами классов распознавания.

##### **Теоретические сведения**

Матрица кодовых расстояний используется для разбиения множества эталонных векторов на пары ближайших соседей:  $\mathfrak{R}_m^{[2]} = \langle x_m, x_l \rangle$ , где  $x_l$  - эталонный вектор соседнего класса  $X_l^o$ . Данное разбиение осуществляется по следующему алгоритму:

а) структурируется множество эталонных векторов, начиная с вектора  $x_1$  базового класса  $X_1^o$ , характеризующий наибольшую функциональную эффективность СПР;

б) строится матрица кодовых расстояний между эталонными векторами размерности  $M \times N$ ;

в) для каждой строки матрицы кодовых расстояний находится минимальный элемент, который принадлежит столбцу вектора, ближайшего к вектору, который определяет строку. При наличии нескольких одинаковых минимальных элементов выбирается из них любой, поскольку они являются равноправными;

г) формируется структурированное множество элементов парного разбиения  $\{\mathfrak{R}_m^{[2]} \mid m = \overline{1, M}\}$ , задающее план обучения.

**Задание1.** Сформировать массив  $SK[1..2,1..N]$  ( $N$  – количество признаков распознавания) кодовых расстояний от геометрических центров контейнеров классов до их реализаций.

**Задание2.** Сформировать массив  $SK\_PARA[1..2,1..N]$  аналогичный предыдущему массиву  $SK$ , но за текущий класс взять ближайший соседний класс.

**Задание3.** Разработать модуль для отображения распределения реализаций между текущим классом и его ближайшим соседом.

### **Порядок выполнения работы**

1. Сформировать массив  $SK [1..2,1..N]$  ( $N$  - количество признаков распознавания) кодовых расстояний от геометрических центров контейнеров классов до их реализаций. Причем массив  $SK[1,1..N]$  должен содержать кодовые расстояния между геометрическим центром текущего класса и реализациями этого класса; массив  $SK[2,1..N]$  должен содержать кодовые расстояния между геометрическим центром текущего класса и реализациями ближайшего соседнего класса.

2. Сформировать массив  $SK\_PARA[1..2,1..N]$  аналогичный предыдущему массиву  $SK$ , но за текущий класс взять ближайший соседний класс.

3. Разработать модуль для отображения распределения реализаций между текущим классом и его ближайшим соседом. Для определения положения каждой реализации использовать значение массивов  $SK$  и  $SK\_PARA$ . Центры классов разместить на горизонтальной прямой.

### **Пример функции для отображения распределения реализаций между текущим классом и его ближайшим соседом**

```
//---таблица"соседей"  
function sosedtabl($nom1, $nom2)  
{  
    $d=0;
```



```

        if ($nom1!=$nom2)
            for($i=0; $i<100;$i++)
            {
                if
(($_SESSION['etalon'][$nom1][$i])!=($_SESSION['etalon'][$nom2][$i]))
                    $d++;
            }
        else
            $d=-1;
        $_SESSION['ssd'][$nom1][$nom2]=$d;
    }

```

///---определение расстояния между «соседями»---

```

function sosed ($kol)
{
    for ($i=0; $i<$kol; $i++)
    {
        if ($_SESSION['ssd'][$i][0]>0)
        {
            $mins=$_SESSION['ssd'][$i][0];
            $inds=0;
        }
        else
        {
            $mins=$_SESSION['ssd'][$i][1];
            $inds=1;
        }
        for ($j=0; $j<$kol; $j++)
        if
(($_SESSION['ssd'][$i][$j]>0)&&($_SESSION['ssd'][$i][$j]<$mins))
        {
            $mins=$_SESSION['ssd'][$i][$j];
            $inds=$j;
        }
        $_SESSION['sfinal'][$i]=$inds;
    }
}

```

///---рассчет массива SK\_PARA---

```

function skr($nom)

```

```

{
$nom_sosed=$_SESSION['sfinal'][$nom];
    for ($i=0; $i<100; $i++)

        {
            $d1=0;
            for($j=0; $j<100;$j++)
            {
                if
                (($_SESSION['etalon'][$nom][$j])!=($_SESSION['bin_matr'][$nom][$j][$i]))
                    $d1++;
            }
            $_SESSION['kr'][$nom][0][$i]=$d1;
        }
        for ($i=0; $i<100; $i++)
        {
            $d2=0;
            for($j=0; $j<100;$j++)
            {
                if
                (($_SESSION['etalon'][$nom][$j])!=($_SESSION['bin_matr'][$nom_sosed][$j][$i]))
                    $d2++;
            }
            $_SESSION['kr'][$nom][1][$i]=$d2;
        }
$file1=fopen("matr/kr".$nom.".txt","w");
    for($i=0;$i<100;$i++)
    {
        $kr1=$_SESSION['kr'][$nom][0][$i];
        $kr2=$_SESSION['kr'][$nom][1][$i];
        fwrite($file1,"$kr1 \t $kr2 \r\n");
    }
fclose($file1);

```

**Графическое отображение распределения реализаций между текущим классом и его ближайшим соседом**

відстань між  $ev1$  і  $ev2$ --> (перша пара) 72  
 відстань між  $ev1$  і  $ev3$ --> (друга пара) 71  
 відстань між  $ev2$  і  $ev3$ --> (третя пара) 43  
 пара з найменшою відстанню між центрами реалізації 3

SK & SK\_PARA для 3 пари

масив SK-->

49 51 49 43 37 34 44 42 46 46 43 33 31 29 32 32 28 23 32 39 41 45 45 41 36 32 40 49 45 34 32 32 33 37 35 35 39 42 43 48 ...  
 51 53 48 51 47 50 40 40 46 46 51 45 41 49 57 54 48 52 53 52 51 45 46 39 41 48 48 46 42 43 48 48 43 41 40 38 36 34 28 40 ...

масив SK\_PARA-->

41 59 54 53 53 42 34 38 42 40 45 43 45 61 65 62 60 60 59 60 55 51 48 45 47 46 50 50 42 41 34 36 43 39 40 46 48 42 32 30 ...  
 53 55 51 55 45 44 46 48 50 54 47 53 49 47 48 44 46 47 50 55 53 59 63 53 52 56 56 57 53 42 36 38 33 35 33 39 45 56 57 56 ...

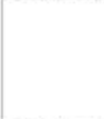





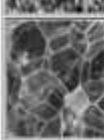
SK\_1 & SK\_PARA\_1 для першої пари

масив SK\_1-->

29 36 36 45 38 37 29 30 29 32 31 23 22 22 38 40 31 27 27 33 42 41 45 41 38 33 34 34 32 32 38 35 36 32 29 30 30 25 24 28 ...  
 46 48 50 54 64 71 67 67 67 63 66 62 58 60 61 59 59 62 55 50 52 46 54 62 65 63 63 56 56 65 73 71 72 66 70 68 58 51 48 45 ...

масив SK\_PARA\_1-->

49 51 49 43 37 34 44 42 46 46 43 33 31 29 32 32 28 23 32 39 41 45 45 41 36 32 40 49 45 34 32 32 33 37 35 35 39 42 43 48 ...  
 29 36 36 45 38 37 29 30 29 32 31 23 22 22 38 40 31 27 27 33 42 41 45 41 38 33 34 34 32 32 38 35 36 32 29 30 30 25 24 28 ...

|   |   |   |   |
|---|---|---|---|
|    |  |  |  |
|    | -   | 72  | 71  |
|   | 72  | -   | 43  |
|  | 71  | 43  | -   |

## Контрольные вопросы

1. Что такое кодовое расстояние? Как строится матрица кодовых расстояний?
2. По какому алгоритму определяется ближайший «соседний» класс распознавания?

3. Какой компонент РНР был использован для отображения распределения реализаций?

## **Практическое занятие 5. Вычисление информационного критерия функциональной эффективности обучения интеллектуальной системы**

**Цель** - разработать и программно реализовать алгоритм вычисления информационного критерия функциональной эффективности обучения системы распознавания.

### **Теоретические сведения**

Информационный подход базируется на использовании для оценки функциональной эффективности информационного критерия, который, например, по Шеннону имеет такой нормированный вид:

$$E = \frac{H_0 - H(\gamma)}{H_0}, \quad (2)$$

где  $H_0$  – априорное (безусловное) энтропия:

$$H_0 = - \sum_{l=1}^M p(\gamma_l) \log_2 p(\gamma_l), \quad (3)$$

$H(\gamma)$  – апостериорная условная энтропия, характеризующая остаточную неопределенность после принятия решений:

$$H(\gamma) = - \sum_{l=1}^M p(\gamma_l) \sum_{m=1}^M p(\mu_m / \gamma_l) \log_2 p(\mu_m / \gamma_l), \quad (4)$$

где  $p(\gamma_l)$  – априорная вероятность принятия гипотезы  $\gamma_l$ ;  $p(\mu_m / \gamma_l)$  – апостериорная вероятность появления события  $\mu_m$  при условии принятия гипотезы  $\gamma_l$ ;  $M$  – число альтернативных гипотез.

На практике при оценке функциональной эффективности обучающейся системы управления, могут иметь место такие предположения:

- решение двухальтернативное ( $M = 2$ );
- поскольку, обучающаяся система управления слабо формализованным процессом функционирует в условиях неопределенности, то по принципу Бернулли-Лапласа оправдано принятие равновероятных гипотез:  $p(\gamma_1) = p(\gamma_2) = 0,5$

Тогда критерий (2) с учетом выражений (3) и (4) принимает частный вид:

$$E = 1 + \frac{1}{2} \sum_{l=1}^2 \sum_{m=1}^2 p(\mu_m / \gamma_l) \log_2 p(\mu_m / \gamma_l). \quad (5)$$

При двухальтернативном решении ( $M = 2$ ) основной считаем гипотезу  $\gamma_1$  о нахождении значения признака распознавания в поле допусков  $\delta$  и как альтернативную ей - гипотезу  $\gamma_2$ . При этом имеют место четыре возможных результата оценки измерения признаков:

ошибка первого рода -  $\alpha = p(x \notin \delta / z \in \delta)$ ;

ошибка второго рода -  $\beta = p(x \in \delta / z \notin \delta)$ ;

первая достоверность -  $D_1 = p(x \in \delta / z \in \delta)$ ;

вторая достоверность -  $D_2 = p(x \notin \delta / z \notin \delta)$ ,

где  $x, z$  - измеренное и действительное значение признака распознавания соответственно.

Разобьем множество значений признаков на области  $\mu_1$  и  $\mu_2$ . Область  $\mu_1$  включает значения, находящиеся в допуске  $\delta$ , а  $\mu_2$  - не в допуске.

Тогда можно записать  $\alpha = p(\gamma_2 / \mu_1)$ ,  $\beta = p(\gamma_1 / \mu_2)$ ,  $D_1 = p(\gamma_1 / \mu_1)$ ,  $D_2 = p(\gamma_2 / \mu_2)$ .

Выразим апостериорные вероятности  $p(\mu_m/\gamma_l)$  через априорные по формуле Байеса :

$$p(\mu_m/\gamma_l) = \frac{p(\mu_m) p(\gamma_l/\mu_m)}{p(\mu_1) p(\gamma_l/\mu_1) + p(\mu_2) p(\gamma_l/\mu_2)} \quad (6)$$

и, взяв  $p(\mu_1)=p(\mu_2)=0,5$ , получаем:

$$\begin{aligned} p(\mu_1/\gamma_1) &= \frac{D_1}{D_1 + \beta}, & p(\mu_2/\gamma_1) &= \frac{\beta}{D_1 + \beta}, \\ p(\mu_1/\gamma_2) &= \frac{\alpha}{\alpha + D_2}, & p(\mu_2/\gamma_2) &= \frac{p_2 D_2}{p_1 \alpha + p_2 D_2}. \end{aligned} \quad (7)$$

После подстановки (7) в (6) получим формулу для вычисления КФЭ по Шеннону :

$$\begin{aligned} E = 1 + \frac{1}{2} & \left( \frac{\alpha}{\alpha + D_2} \log_2 \frac{\alpha}{\alpha + D_2} + \frac{D_1}{D_1 + \beta} \log_2 \frac{D_1}{D_1 + \beta} + \right. \\ & \left. + \frac{\beta}{D_1 + \beta} \log_2 \frac{\beta}{D_1 + \beta} + \frac{D_2}{\alpha + D_2} \log_2 \frac{D_2}{\alpha + D_2} \right). \end{aligned} \quad (8)$$

Структурная схема алгоритма вычисления информационного критерия функциональной эффективности обучения по параллельному способу обработки обучающей матрицы в процессе построения оптимального контейнера класса показано на рис. 5.1.

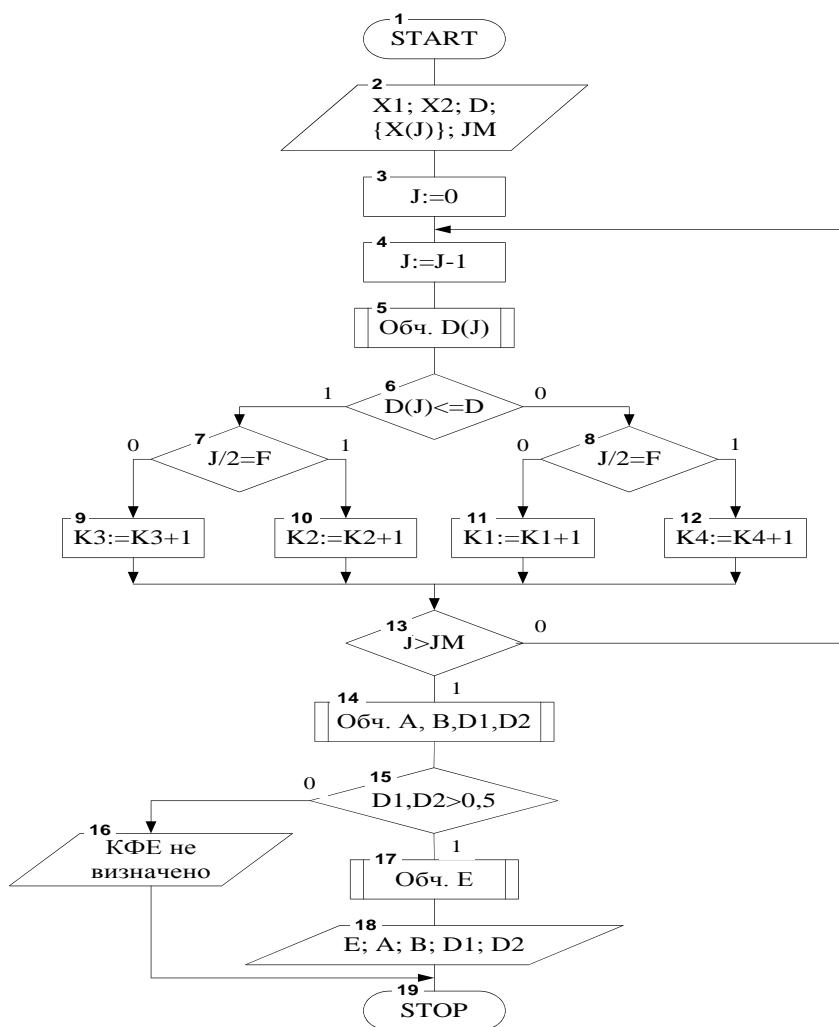


Рисунок 2. – Структурная схема вычисления информационного критерия функциональной эффективности обучения системы распознавания

На рис. 2 приведены следующие входные данные:  $X_1$ ,  $X_2$  – эталонные двоичные векторы классов  $X_1^o$  и  $X_2^o$  соответственно;

$\{X(N)\}$  – обучающаяся матрица, состоящая из реализаций этих классов;  $N = 1, \overline{NM}$ , где  $NM$  – объем репрезентативной обучающейся выборки,  $D$  – радиус контейнера класса  $X_1^o$ . Выходные данные:  $E$  – значение КФЭ;  $\alpha, \beta, D_1, D_2$  – значения точностных характеристик процесса обучения: ошибки первого и второго рода, первая и вторая достоверности соответственно.

Блок 5 вычисляет при каждом испытании кодовое расстояние  $D(J)$  путем сложения по модулю два вектора  $X_1$  с текущим вектором-реализацией  $X(J)$  и подсчета количества единиц в полученной сумме. При каждом нечетном испытании определяется расстояние  $D(J)$  между вектором  $X_1$  и реализацией своего класса, а на каждом четном – между вектором  $X_1$  и реализацией другого класса. Вычисление коэффициентов  $K_1, K_2, K_3$  и  $K_4$  осуществляется по следующему алгоритму (блоки 6 – 12):

а) сравнения (блок 6): если  $D(J) \leq D$  (реализация принадлежит области класса  $X_1^o$ ), то при нечетном испытании вычисляется  $K_1 := K_1 + 1$  ("своя" реализация), а при парном –  $K_3 := K_3 + 1$  ("чужая" реализация). Определение четности или нечетности реализаций осуществляют блоки 7 и 8, которые проверяют выполнение условия  $J/2 = F$ , где  $F$  – целое число. Если условие выполняется, то испытание четное, иначе – нечетное. Если  $D(J) > D$  (реализация не принадлежит области класса  $X_1^o$ ), то при нечетном испытании вычисляется коэффициент  $K_2 := K_2 + 1$  ("своя" реализация), а при четном –  $K_4 := K_4 + 1$  ("чужая" реализация);

б) сравнение (блок 13):  $J > JM$  (обработано все реализации обучающей матрицы), тогда вычисляются оценки точностных характеристик по (14), иначе обрабатывается следующая реализация (блок 4);

в) при выполнении условия блока 15 : ( $D_1 > 0,5$  и  $D_2 > 0,5$ ) вычисляется информационный критерий, иначе выдается сообщение «КФЭ не определен».



**Задание 1.** Вычислить значение точностных характеристик.

**Задание 2.** Программно реализовать функцию вычисления значения критерия функциональной эффективности по Кульбаку при указанных геометрических параметрах классов обучения.

**Задание 3.** Программно реализовать функцию вычисления значения критерия функциональной эффективности по Шеннону при указанных геометрических параметрах классов обучения.

**Задание 4.** Выделить рабочую область определения критерия функциональной эффективности на графике и в таблице.

### Порядок выполнения работы

1. Вычислить значения точностных характеристик при указанных геометрических параметрах классов обучения:

$$D_1^{(k)} = \frac{K_1^{(k)}}{n}, \quad \alpha^{(k)} = \frac{K_2^{(k)}}{n}, \quad \beta^{(k)} = \frac{K_3^{(k)}}{n}, \quad D_2^{(k)} = \frac{K_4^{(k)}}{n}.$$

2. Программно реализовать функцию вычисления значения критерия функциональной эффективности при указанных геометрических параметрах классов обучения. В качестве критерия использовать критерий Кульбака:

$$J_m^{(k)} = \frac{1}{n} \log_2 \left\{ \frac{2n + 10^{-r} - [K_2^{(k)} + K_3^{(k)}]}{[K_2^{(k)} + K_3^{(k)}] + 10^{-r}} \right\} * [n - (K_2^{(k)} + K_3^{(k)})], \quad (9)$$

$K_1^{(k)}, K_2^{(k)}$  – количество событий, которые означают соответственно принадлежность и непринадлежность реализаций образа контейнеру  $K_{1,k}^o$ , если действительно  $\{x_1^{(j)}\} \in X_1^o$ ;

$K_3^{(k)}, K_4^{(k)}$  – количество событий, которые означают

соответственно принадлежность и непринадлежность реализаций контейнеру  $K_{1,k}^o$ , если они на самом деле принадлежат классу  $X_2^o$ ;

$k$  — шаг обучения системы распознавания;

$r$  — число цифр в мантиссе значения критерия;

$n$  — количество реализаций.

3. Программно реализовать функцию вычисления значения критерия функциональной эффективности при указанных геометрических параметрах классов обучения. Использовать информационный критерий Шеннона для двоихальтернативного решение:

$$E_1^{(k)} = 1 + \frac{1}{2} \left( \frac{K_1^{(k)}}{K_1^{(k)} + K_3^{(k)}} \log_2 \frac{K_1^{(k)}}{K_1^{(k)} + K_3^{(k)}} + \frac{K_2^{(k)}}{K_2^{(k)} + K_4^{(k)}} \log_2 \frac{K_2^{(k)}}{K_2^{(k)} + K_4^{(k)}} + \frac{K_3^{(k)}}{K_1^{(k)} + K_3^{(k)}} \log_2 \frac{K_3^{(k)}}{K_1^{(k)} + K_3^{(k)}} + \frac{K_4^{(k)}}{K_2^{(k)} + K_4^{(k)}} \log_2 \frac{K_4^{(k)}}{K_2^{(k)} + K_4^{(k)}} \right).$$

4. Построить графики зависимости информационного критерия от параметров оптимизации.

5. Провести проверку и анализ полученных результатов.

### **Пример функции вычисления точностных характеристик и значение критерия функциональной эффективности**

// ---расчет точностных характеристик---

function to4nist(\$nom)

```
{
    for($i=0; $i<100;$i++)
    {
        $k1[$i]=0;
        $k2[$i]=0;
        $k3[$i]=0;
        $k4[$i]=0;
    }
}
```

```

for($i=0; $i<100; $i++)
{
    for($j=0; $j<100; $j++)
    {
        if ($_SESSION['kr'][$nom][0][$j]<=$i)
            $k1[$i]=$k1[$i]+1;
        else
            $k3[$i]=$k3[$i]+1;
        if ($_SESSION['kr'][$nom][1][$j]<=$i)
            $k2[$i]=$k2[$i]+1;
        else
            $k4[$i]=$k4[$i]+1;
    }
    $_SESSION['D1'][$nom][$i]=$k1[$i]/100;
    $_SESSION['a'][$nom][$i]=$k3[$i]/100;
    $_SESSION['b'][$nom][$i]=$k2[$i]/100;
    $_SESSION['D2'][$nom][$i]=$k4[$i]/100;
}
for ($i=0; $i<100; $i++)
{{
    if
(($_SESSION['D1'][$nom][$i]>=0.5)&&($_SESSION['D2'][$nom][$i]>=0.5))
    {
        $a=$_SESSION['a'][$nom][$i];
        $b=$_SESSION['b'][$nom][$i];
        $temp=(log((2-($a+$b)))/($a+$b))/log(2))*(1-($a+$b));//
критерий Кульбака

        // $temp=1+0.5*(($a/($a+$D2))*(log($a/($a+$D2)))/log(2)+($D1/($D1+$b))*(
log($D1/($D1+$b)))/log(2)+($b/($D1+$b))*(log($b/($D1+$b)))/log(2)+($D2/($a+$D2
))*(log($D2/($a+$D2)))/log(2)); //критерий Шеннона
        $_SESSION['J'][$nom][$i]=$temp;
    }
    else
        $_SESSION['J'][$nom][$i]=0;
}
}

$max_J=0;
for ($i=0; $i<100; $i++)

```

```

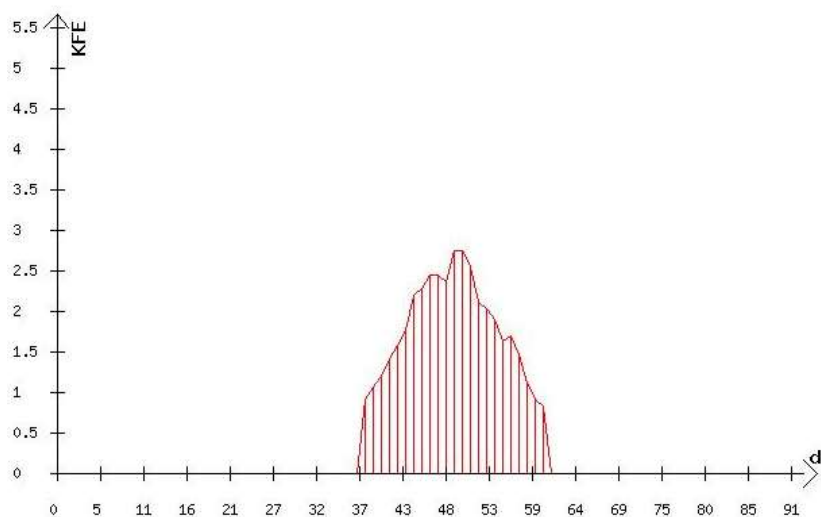
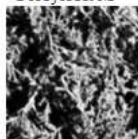
{
    if ($max_J < $_SESSION['J'][$nom][$i])
        $max_J = $_SESSION['J'][$nom][$i];
}
// ---запись точностных характеристик в файл---
$file1=fopen("matr/har".$nom.".txt","w");
fwrite($file1,"D1 \t D2 \t a \t b \r\n");
for($i=0;$i<100;$i++){
    $d1=$_SESSION['D1'][$nom][$i];
    $d2=$_SESSION['D2'][$nom][$i];
    $a=$_SESSION['a'][$nom][$i];
    $b=$_SESSION['b'][$nom][$i];
    fwrite($file1,"$d1 \t $d2 \t $a \t $b \r\n");
}
fclose($file1);
$file1=fopen("matr/kfe".$nom.".txt","w");
for ($i=0; $i<100;$i++)
{
    $temp=$_SESSION['J'][$nom][$i];
    fwrite($file1,"$i \t\t $temp");
}
fclose($file1);
return $max_J;
}

```

## Графическое отображение значения критерия функциональной эффективности

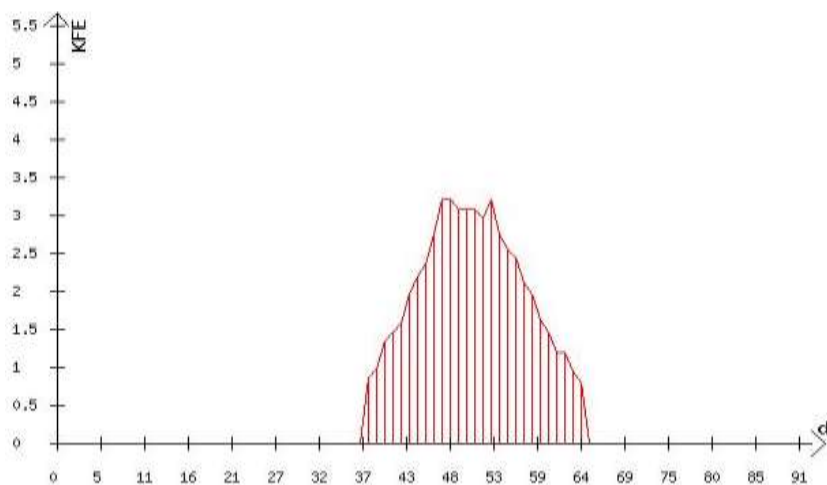
Межцентровое кодовое расстояние =72

Рисунок 5.2



| d  | D1   | D2   | Alfa | Beta | KFE    |
|----|------|------|------|------|--------|
| 32 | 0.3  | 1    | 0.7  | 0    | 0      |
| 33 | 0.31 | 1    | 0.69 | 0    | 0      |
| 34 | 0.34 | 1    | 0.66 | 0    | 0      |
| 35 | 0.37 | 1    | 0.63 | 0    | 0      |
| 36 | 0.42 | 1    | 0.58 | 0    | 0      |
| 37 | 0.47 | 1    | 0.53 | 0    | 0      |
| 38 | 0.53 | 1    | 0.47 | 0    | 0.9025 |
| 39 | 0.57 | 1    | 0.43 | 0    | 1.065  |
| 40 | 0.6  | 1    | 0.4  | 0    | 1.2    |
| 41 | 0.64 | 1    | 0.36 | 0    | 1.4001 |
| 42 | 0.67 | 1    | 0.33 | 0    | 1.5673 |
| 43 | 0.7  | 1    | 0.3  | 0    | 1.7518 |
| 44 | 0.76 | 1    | 0.24 | 0    | 2.1846 |
| 45 | 0.79 | 0.98 | 0.21 | 0.02 | 2.2669 |
| 46 | 0.82 | 0.97 | 0.18 | 0.03 | 2.4423 |
| 47 | 0.83 | 0.96 | 0.17 | 0.04 | 2.4423 |
| 48 | 0.84 | 0.94 | 0.16 | 0.06 | 2.3527 |
| 49 | 0.89 | 0.93 | 0.11 | 0.07 | 2.7371 |
| 50 | 0.9  | 0.92 | 0.1  | 0.08 | 2.7371 |
| 51 | 0.9  | 0.9  | 0.1  | 0.1  | 2.5359 |
| 52 | 0.92 | 0.83 | 0.08 | 0.17 | 2.1055 |
| 53 | 0.94 | 0.8  | 0.06 | 0.2  | 2.0295 |
| 54 | 0.95 | 0.77 | 0.05 | 0.23 | 1.8856 |
| 55 | 0.96 | 0.72 | 0.04 | 0.28 | 1.6268 |
| 56 | 0.98 | 0.71 | 0.02 | 0.29 | 1.6882 |
| 57 | 0.99 | 0.66 | 0.01 | 0.34 | 1.4541 |
| 58 | 0.99 | 0.59 | 0.01 | 0.41 | 1.1086 |
| 59 | 0.99 | 0.54 | 0.01 | 0.46 | 0.9025 |
| 60 | 0.99 | 0.52 | 0.01 | 0.48 | 0.8281 |
| 61 | 1    | 0.45 | 0    | 0.55 | 0      |
| 62 | 1    | 0.44 | 0    | 0.56 | 0      |
| 63 | 1    | 0.4  | 0    | 0.6  | 0      |
| 64 | 1    | 0.34 | 0    | 0.66 | 0      |
| 65 | 1    | 0.3  | 0    | 0.7  | 0      |

Рисунок 5.3



| d  | D1   | D2   | Alfa | Beta | KFE    |
|----|------|------|------|------|--------|
| 32 | 0.28 | 1    | 0.72 | 0    | 0      |
| 33 | 0.33 | 1    | 0.67 | 0    | 0      |
| 34 | 0.38 | 1    | 0.62 | 0    | 0      |
| 35 | 0.42 | 1    | 0.58 | 0    | 0      |
| 36 | 0.46 | 1    | 0.54 | 0    | 0      |
| 37 | 0.46 | 1    | 0.54 | 0    | 0      |
| 38 | 0.52 | 1    | 0.48 | 0    | 0.8647 |
| 39 | 0.55 | 1    | 0.45 | 0    | 0.9813 |
| 40 | 0.63 | 1    | 0.37 | 0    | 1.3477 |
| 41 | 0.65 | 1    | 0.35 | 0    | 1.4541 |
| 42 | 0.68 | 0.99 | 0.32 | 0.01 | 1.5673 |
| 43 | 0.74 | 0.99 | 0.26 | 0.01 | 1.9562 |
| 44 | 0.78 | 0.98 | 0.22 | 0.02 | 2.1846 |
| 45 | 0.81 | 0.97 | 0.19 | 0.03 | 2.3527 |
| 46 | 0.85 | 0.97 | 0.15 | 0.03 | 2.7371 |
| 47 | 0.9  | 0.96 | 0.1  | 0.04 | 3.2094 |
| 48 | 0.91 | 0.95 | 0.09 | 0.05 | 3.2094 |
| 49 | 0.92 | 0.93 | 0.08 | 0.07 | 3.0808 |
| 50 | 0.93 | 0.92 | 0.07 | 0.08 | 3.0808 |
| 51 | 0.94 | 0.91 | 0.06 | 0.09 | 3.0808 |
| 52 | 0.97 | 0.87 | 0.03 | 0.13 | 2.9598 |
| 53 | 0.99 | 0.87 | 0.01 | 0.13 | 3.2094 |
| 54 | 0.99 | 0.83 | 0.01 | 0.17 | 2.7371 |
| 55 | 0.99 | 0.81 | 0.01 | 0.19 | 2.5359 |
| 56 | 1    | 0.79 | 0    | 0.21 | 2.4423 |
| 57 | 1    | 0.75 | 0    | 0.25 | 2.1055 |
| 58 | 1    | 0.73 | 0    | 0.27 | 1.9562 |
| 59 | 1    | 0.68 | 0    | 0.32 | 1.6268 |
| 60 | 1    | 0.65 | 0    | 0.35 | 1.4541 |
| 61 | 1    | 0.6  | 0    | 0.4  | 1.2    |
| 62 | 1    | 0.6  | 0    | 0.4  | 1.2    |
| 63 | 1    | 0.54 | 0    | 0.46 | 0.9413 |
| 64 | 1    | 0.5  | 0    | 0.5  | 0.7925 |
| 65 | 1    | 0.46 | 0    | 0.54 | 0      |

### Контрольные вопросы

1. Что называется первой достоверностью?
2. Что называется второй достоверностью?
3. Что называется ошибкой первого рода?
4. Что называется ошибкой второго рода?
5. Обоснуйте целесообразность использования информационного КФЭ для оценки функциональной эффективности обобщающей интеллектуальной системы.

## Практическое занятие 6. Оптимизация системы контрольных допусков на признаки распознавания

**Цель** - разработать и программно реализовать алгоритм оптимизации геометрических параметров контейнеров классов распознавания.

### Теоретические сведения

Рассмотрим алгоритм параллельной оптимизации системы контрольных допусков на признаки распознавания в рамках МФСИ, На рис. 3 показано симметричное (двустороннее) поле допусков на значение признака  $y_{mi}^{(j)}, i = \overline{1, N}$ .

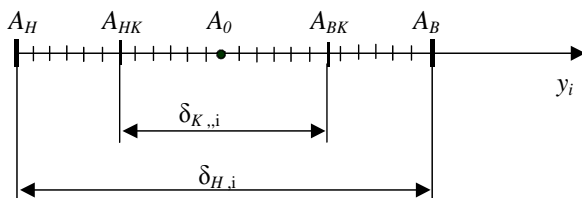


Рисунок 3. – Симметричное поле допусков на значение признака распознавания

Здесь  $A_0$  – номинальное значение признака  $y_i$ ;

$A_H, A_B$  – нижний и верхний нормированные допуски соответственно;

$A_{HK}, A_{BK}$  – нижний и верхний контрольные допуски соответственно;

$\delta_{H,i}$  – нормированное поле допусков;

$\delta_{K,i}$  – контрольное поле допусков.

Существует несколько возможных стратегий изменения поля допусков  $\delta_{K,i}$ , среди которых выделим две основные:

• симметричная стратегия  $S_1(\overset{\rightarrow}{\leftarrow} \text{var } A_{HK}, \overset{\leftarrow}{\rightarrow} \text{var } A_{BK})$ , которая оправдана, например, при условии подтверждения разведывательным анализом совпадения номинального значения  $A_0$  с теоретическим центром рассеивания значений обучающей выборки  $\{y_{m,i}^{(j)} \vee j = \overline{1, n}\}$ ;

• асимметричная стратегия  $S_2(\overset{\leftarrow}{\rightarrow} \text{var } A_{HK}, \overset{\rightarrow}{\leftarrow} \text{var } A_{BK})$ , которая имеет место при отклонении значения  $A_0$  от центра рассеивания значений выборки  $\{y_{m,i}^{(j)} \vee j = \overline{1, n}\}$ .

Задача оптимизации контрольных допусков на признаки распознавания является частичной задачей информационного синтеза, в которой необходимо определить экстремальные значения параметра поля контрольных допусков  $\delta = |A_0 - A_{HK}|$ :

$$\delta^* = \arg \max_{G_\delta} \{ \max_{G_\Omega} \{ \max_{G_d} \overline{E} \} \},$$

где  $G_\delta, G_\Omega, G_d$  – допустимые области значений параметра поля контрольных допусков  $\delta$ , пространства признаков распознавания и радиусов контейнеров классов распознавания соответственно.

Алгоритм оптимизации контрольных допусков, как и других параметров обучения в рамках МФСИ, заключается в



приближении глобального максимума информационного критерия оптимизации к предельному его значению в рабочей области значений функции критерия.

**Задание 1.** Разработать алгоритм последовательной оптимизации системы контрольных допусков на признаки распознавания. На каждом шаге оптимизации вычислить оптимальные значения геометрических параметров плана обучения.

**Задание 2.** Разработать систему визуализации процесса оптимизации системы контрольных допусков и геометрических параметров плана обучения.

### **Порядок выполнения работы**

1. В качестве входных данных использовать бинарную обучающую матрицу яркости  $\{X[J, I, K]\}$  и эталонные векторы-реализации изображений.
2. Определить область значений параметра оптимизации  $\delta \in [0; \delta_H / 2]$ , где  $\delta_H$  – нормированное (эксплуатационное) поле допусков. Для черно-белых изображений рекомендуется выбирать  $\delta_{\max} \geq 20$  градаций яркости.
3. Программно реализовать итерационную процедуру параллельной оптимизации системы контрольных допусков по информационному критерию Шеннона.
4. Построить графики зависимости информационного критерия от параметров оптимизации. Провести проверку и анализ полученных результатов.

**Пример функции оптимизации системы контрольных допусков и радиусов контейнеров классов распознавания**

```
//--оптимизация контрольных допусков---
```

```
$radius=0.5;
    s4it (0,$delta,$radius);
    s4it (1,$delta,$radius);
for ($delta=30;$delta<=50;$delta++)
{
    optimize (0,$delta,$radius);
    optimize (1,$delta,$radius);
    for ($i=0; $i<2; $i++)
    {
        for ($j=0; $j<2;$j++)
        {
            sosedtabl($i,$j);
        }
    }
    sosed(2);
    for($i=0; $i<2; $i++)
    {
        skr($i);
        $max_J[$i][$delta]=to4nist($i);
        $klass=$i+1;
    }
}
$t_max_J=0;
$opt_delta=0;
    for ($delta=30;$delta<=50;$delta++)
{
    $summ=$max_J[0][$delta]+$max_J[1][$delta];
    if ($t_max_J<=$summ)
    {
        $t_max_J=$summ;
        $opt_delta=$delta;
    }
}
s4it (0,$opt_delta,$radius);
s4it (1,$opt_delta,$radius);
```

```
// ---поиск оптимального радиуса---
```

```

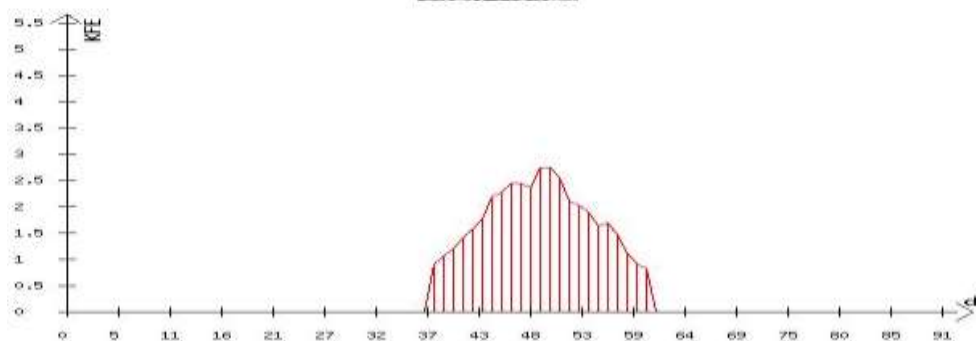
function dop($kol)
{
    for($k=0; $k<$kol; $k++)
    {
        for($i=0; $i<100;$i++)
        {
            if
            (($_SESSION['D1'][$k][$i]>0.5)&&($_SESSION['D2'][$k][$i]>0.5))
            {
                $zn=$_SESSION['J'][$k][$i];
                $_SESSION['DOPT'][$k]=$i+1;
            }
            for($i=0; $i<100;$i++)
            for($i=0; $i<100;$i++)
            {
                if
                (($_SESSION['J'][$k][$i]>$zn)&&($_SESSION['D1'][$k][$i]>0.5)&&($_SESSION['D2'][$k][$i]>0.5))
                {
                    $zn=$_SESSION['J'][$k][$i];
                    $_SESSION['DOPT'][$k]=$i+1;
                }
            }
        }
    }
}

```

### **Графическое отображение процесса оптимизации системы контрольных допусков и геометрических параметров контейнеров классов обучения**

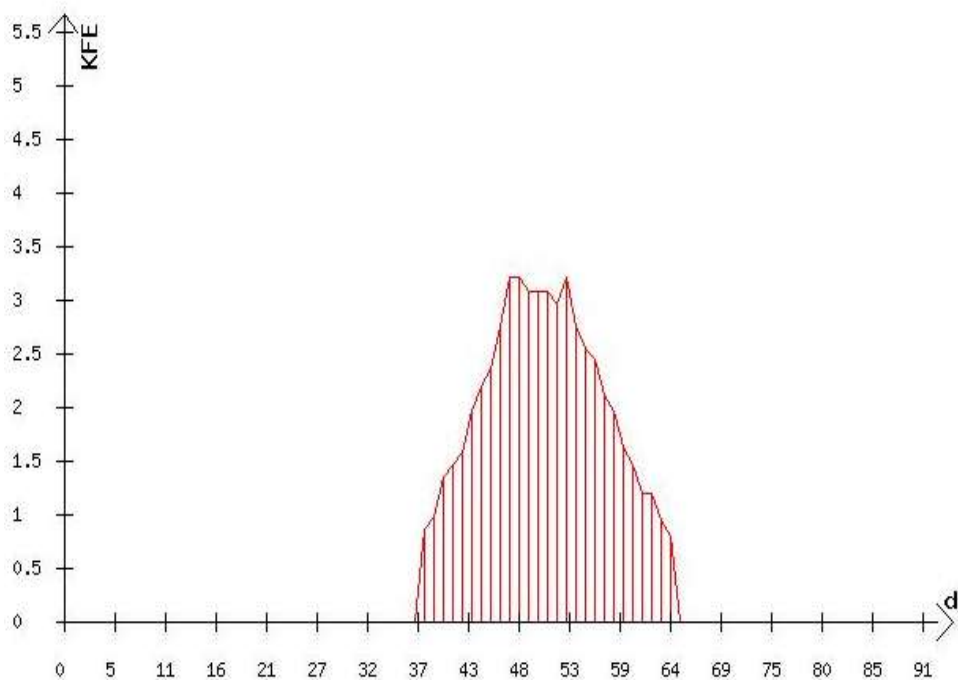
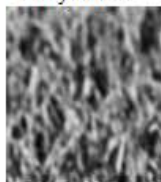
Значение КФЭ = 2.737053 для 1-го класса максимально при дельта=40  
 Значение КФЭ = 3.209351 для 2-го класса максимально при дельта=40  
 Межцентровое кодовое расстояние =89

Рисунок 2



| d  | D1   | D2   | Alfa | Beta | KFE    |
|----|------|------|------|------|--------|
| 32 | 0.3  | 1    | 0.7  | 0    | 0      |
| 33 | 0.31 | 1    | 0.69 | 0    | 0      |
| 34 | 0.34 | 1    | 0.66 | 0    | 0      |
| 35 | 0.37 | 1    | 0.63 | 0    | 0      |
| 36 | 0.42 | 1    | 0.58 | 0    | 0      |
| 37 | 0.47 | 1    | 0.53 | 0    | 0      |
| 38 | 0.53 | 1    | 0.47 | 0    | 0.9025 |
| 39 | 0.57 | 1    | 0.43 | 0    | 1.065  |
| 40 | 0.6  | 1    | 0.4  | 0    | 1.2    |
| 41 | 0.64 | 1    | 0.36 | 0    | 1.4001 |
| 42 | 0.67 | 1    | 0.33 | 0    | 1.5673 |
| 43 | 0.7  | 1    | 0.3  | 0    | 1.7518 |
| 44 | 0.76 | 1    | 0.24 | 0    | 2.1846 |
| 45 | 0.79 | 0.98 | 0.21 | 0.02 | 2.2669 |
| 46 | 0.82 | 0.97 | 0.18 | 0.03 | 2.4423 |
| 47 | 0.83 | 0.96 | 0.17 | 0.04 | 2.4423 |
| 48 | 0.84 | 0.94 | 0.16 | 0.06 | 2.3527 |
| 49 | 0.89 | 0.93 | 0.11 | 0.07 | 2.7371 |
| 50 | 0.9  | 0.92 | 0.1  | 0.08 | 2.7371 |
| 51 | 0.9  | 0.9  | 0.1  | 0.1  | 2.5359 |
| 52 | 0.92 | 0.83 | 0.08 | 0.17 | 2.1055 |
| 53 | 0.94 | 0.8  | 0.06 | 0.2  | 2.0295 |
| 54 | 0.95 | 0.77 | 0.05 | 0.23 | 1.8856 |
| 55 | 0.96 | 0.72 | 0.04 | 0.28 | 1.6268 |
| 56 | 0.98 | 0.71 | 0.02 | 0.29 | 1.6882 |
| 57 | 0.99 | 0.66 | 0.01 | 0.34 | 1.4541 |
| 58 | 0.99 | 0.59 | 0.01 | 0.41 | 1.1086 |
| 59 | 0.99 | 0.54 | 0.01 | 0.46 | 0.9025 |
| 60 | 0.99 | 0.52 | 0.01 | 0.48 | 0.8281 |
| 61 | 1    | 0.45 | 0    | 0.53 | 0      |
| 62 | 1    | 0.44 | 0    | 0.56 | 0      |
| 63 | 1    | 0.4  | 0    | 0.6  | 0      |
| 64 | 1    | 0.34 | 0    | 0.66 | 0      |
| 65 | 1    | 0.3  | 0    | 0.7  | 0      |

Рисунок 6.3



| d  | D1   | D2   | Alfa | Beta | KFE    |
|----|------|------|------|------|--------|
| 32 | 0.28 | 1    | 0.72 | 0    | 0      |
| 33 | 0.33 | 1    | 0.67 | 0    | 0      |
| 34 | 0.38 | 1    | 0.62 | 0    | 0      |
| 35 | 0.42 | 1    | 0.58 | 0    | 0      |
| 36 | 0.46 | 1    | 0.54 | 0    | 0      |
| 37 | 0.46 | 1    | 0.54 | 0    | 0      |
| 38 | 0.52 | 1    | 0.48 | 0    | 0.8647 |
| 39 | 0.55 | 1    | 0.45 | 0    | 0.9813 |
| 40 | 0.63 | 1    | 0.37 | 0    | 1.3477 |
| 41 | 0.65 | 1    | 0.35 | 0    | 1.4541 |
| 42 | 0.68 | 0.99 | 0.32 | 0.01 | 1.5673 |
| 43 | 0.74 | 0.99 | 0.26 | 0.01 | 1.9562 |
| 44 | 0.78 | 0.98 | 0.22 | 0.02 | 2.1846 |
| 45 | 0.81 | 0.97 | 0.19 | 0.03 | 2.3527 |
| 46 | 0.85 | 0.97 | 0.15 | 0.03 | 2.7371 |
| 47 | 0.9  | 0.96 | 0.1  | 0.04 | 3.2094 |
| 48 | 0.91 | 0.95 | 0.09 | 0.05 | 3.2094 |
| 49 | 0.92 | 0.93 | 0.08 | 0.07 | 3.0808 |
| 50 | 0.93 | 0.92 | 0.07 | 0.08 | 3.0808 |
| 51 | 0.94 | 0.91 | 0.06 | 0.09 | 3.0808 |
| 52 | 0.97 | 0.87 | 0.03 | 0.13 | 2.9598 |
| 53 | 0.99 | 0.87 | 0.01 | 0.13 | 3.2094 |
| 54 | 0.99 | 0.83 | 0.01 | 0.17 | 2.7371 |
| 55 | 0.99 | 0.81 | 0.01 | 0.19 | 2.5359 |
| 56 | 1    | 0.79 | 0    | 0.21 | 2.4423 |
| 57 | 1    | 0.75 | 0    | 0.25 | 2.1055 |
| 58 | 1    | 0.73 | 0    | 0.27 | 1.9562 |
| 59 | 1    | 0.68 | 0    | 0.32 | 1.6268 |
| 60 | 1    | 0.65 | 0    | 0.35 | 1.4541 |
| 61 | 1    | 0.6  | 0    | 0.4  | 1.2    |
| 62 | 1    | 0.6  | 0    | 0.4  | 1.2    |
| 63 | 1    | 0.54 | 0    | 0.46 | 0.9413 |
| 64 | 1    | 0.5  | 0    | 0.5  | 0.7925 |
| 65 | 1    | 0.46 | 0    | 0.54 | 0      |


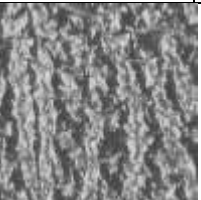
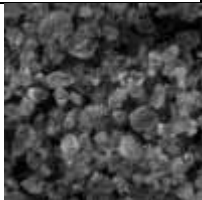

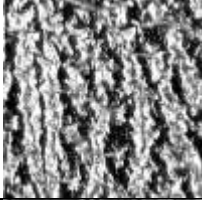
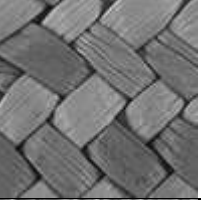
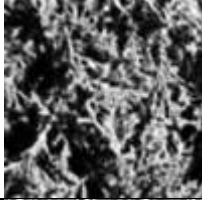


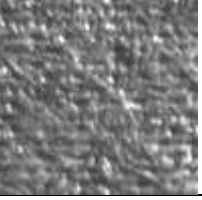
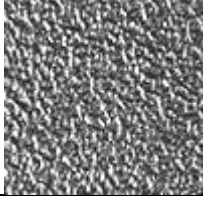
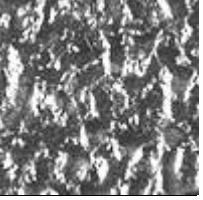
### Контрольные вопросы

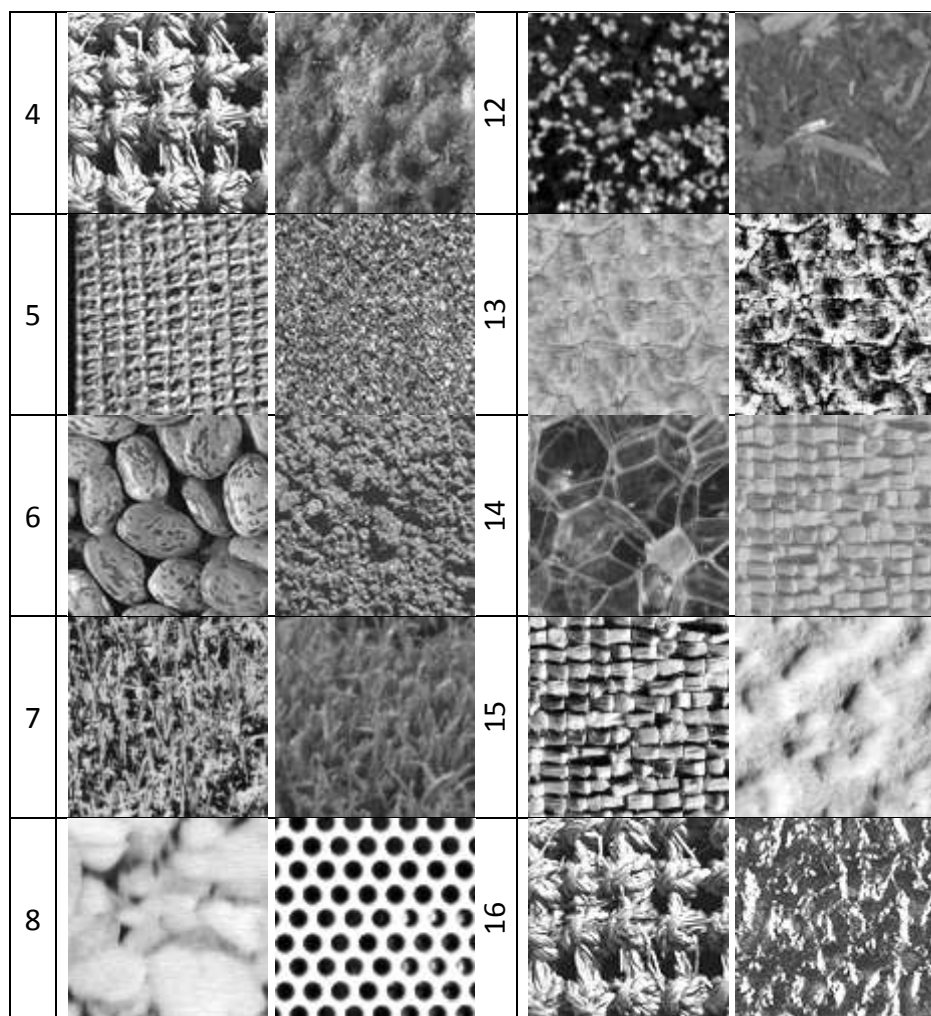
1. Что такое «рабочая область» при оптимизации геометрических параметров классов распознавания? Какова ее роль в этом процессе?
2. На сколько отличаются значения оптимальных геометрических параметров классов обучения, полученные с использованием различных информационных критериев?
3. Какие компоненты РНР были использованы при создании системы визуализации процесса оптимизации геометрических параметров плана обучения?

## Практическое занятие 7. Реализация алгоритма обучения интеллектуальной системы

**Цель** - разработать и программно реализовать алгоритм обучения с оптимизацией контрольных допусков системы распознавания.

**Задача 1.** Провести обучение с оптимизацией системы контрольных допусков и геометрических параметров классов распознавания. Обучающую матрицу сформировать, используя текстуры.

| Номер<br>варианта | 1-й класс   | 2-й класс   | Номер<br>варианта | 1-й класс   | 2-й класс  |
|-------------------|---|---|-------------------|---|--|
| 1                 |    |    | 9                 |    |    |
| 2                 |   |   | 10                |   |   |
| 3                 |  |  | 11                |  |  |



### Порядок выполнения работы

1. Записать тему и цель работы.
2. Провести обучение с оптимизацией системы контрольных допусков и геометрических параметров классов распознавания.
3. Провести проверку и анализ полученных результатов.



### Контрольные вопросы

1. Как вычислить оптимальные значения геометрических параметров классов и системы контрольных допусков на признаки распознавания?
2. Как получить графические изображения бинарных обучающих матриц при начальных и оптимальных параметрах обучения?
3. Как происходит изменение критерия функциональной эффективности в процессе оптимизации системы контрольных допусков и геометрических параметров классов распознавания?

## Практическое занятие 8. Реализация алгоритма экзамена

**Цель** - разработать и программно реализовать алгоритм экзамена системы распознавания.

### Теоретические сведения

Алгоритмы экзамена по МФСИ могут иметь разную структуру в зависимости от распределения реализаций образа. Обязательным условием их реализации является обеспечение равных условий структурированности и параметров формирования как для обучающей, так и для экзаменационной матриц.

Реализации алгоритма экзамена:

1. Формирование счетчика  $m := m + 1$  классов распознавания.
2. Формирование счетчика числа реализаций, которые распознаются  $j := j + 1$ .
3. Вычисления кодового расстояния  $d(x_m^* \oplus x^{(j)})$ .
4. Вычисления функции принадлежности:

$$\mu_m = 1 - \frac{d(x_m^* \oplus x^{(j)})}{d_m^*}.$$

5. Сравнения: если  $j \leq n$ , то выполняется шаг 2, иначе - шаг 6.

6. Сравнения: если  $m \leq M$ , то выполняется шаг 1, иначе- шаг 7.

7.Определение класса  $X_m^o$ , к которому принадлежит экзаменационная реализация, например, при условии

$$\mu_m^* = \max_{\{m\}} \bar{\mu}_m,$$

где  $\bar{\mu}_m = \frac{1}{n} \sum_{j=1}^n \mu_{m,j}$  - усредненное значение функции

принадлежности для реализаций класса  $X_m^o$ , или выдача уведомления : «Класс не определен», если  $\bar{\mu}_m \leq c$ , где  $c$  – пороговое значение.

**Задание 1.** Сформировать экзаменационную матрицу.

**Задача 2.** Программно реализовать алгоритм экзамена интеллектуальной системы.

### Порядок выполнения работы

1. Сформировать экзаменационную матрицу. Для формирования можно использовать обучающиеся матрицы, которые были определенным образом деформированы.
2. Программно реализовать алгоритм экзамена интеллектуальной системы.

### Пример функции этапа экзамена

//---функция экзамена---

```

function ekzamen($kol)
{
    $nom_str=rand(0,99);
    $nom_matr=rand(0,$kol-1);
    for($j=0; $j<100; $j++)
    {
        $XP[$j]=$_SESSION['obuch_matr'][$nom_matr][$nom_str][$j];
    }
    for($j=0; $j<100; $j++)
    {
        if
        (($XP[$j]>=$ndk_1[$nom_matr][$j])&&($XP[$j]<=$vdk_1[$nom_matr][$j]))
        {
            $XP_bin[$j]=1;
            $_SESSION['xp'][$j]=1;
        }
        else
        {
            $XP_bin[$j]=0;
            $_SESSION['xp'][$j]=0;
        }
    }
    for ($k=0; $k<$kol; $k++)
    {
        $DD[$k]=0;
        for($j=0; $j<100; $j++)
        {
            if($XP_bin[$j]!=$_SESSION['etalon'][$k][$j])
                $DD[$k]++;
        }
        if ($_SESSION['DOPT'][$k]!=0)
            $F[$k]=1-($DD[$k]/$_SESSION['DOPT'][$k]);
        else
            $F[$k]=1;
    }
    $max_f=$F[0];
    $ind_f=0;
    for($k=0; $k<$kol; $k++)
    {
        if($F[$k]>$max_f)

```



4. Как формируется бинарная обучающая матрица для изображения классов?
5. Объясните преимущество обработки изображений в дискретном пространстве признаков распознавания.
6. Что определяет в дискретном пространстве признаков распознавания координата вершины двоичного эталонного вектора реализации класса распознавания?
7. Как найти межцентровое кодовое расстояние для классов, контейнеры которых построены в радиальном базисе?
8. Обоснуйте целесообразность построения контейнеров классов распознавания для изображений-текстур.
9. Какую структуру имеет входное математическое описание системы распознавания изображений?

### **Список рекомендованной литературы**

1. Довбиш А. С. Основи проектування інтелектуальних систем : навчальний посібник / А. С. Довбиш. — Суми : СумДУ, 2009. — 171 с.
2. Проектирование систем управления на ЭВМ //А. Ю. Соколов, Ю. Н. Соколов, В. М. Ильюшко, М. М. Митрахович, Д. Н. Гайсёнок // под ред. Ю. Н. Соколова. — Харьков : «ХАИ», 2005. — 590 с.
3. Учебник PHP.—URL: <http://www.phpbook.org.ua/>