Путин Павел Александрович, группа 7-1

Лабораторная работа № 4

Вариант № 9

Распознавание образов, описываемых бинарными признаками

Цель работы

Синтезировать алгоритмы распознавания образов, описываемых бинарными признаками. Исследовать синтезированные алгоритмы распознавания с точки зрения ожидаемых потерь и ошибок.

Задание

Получить у преподавателя вариант задания и написать код, реализующий алгоритм распознавания образов, заданных бинарными изображениями. Проведите имитационное моделирование алгоритма, в ходе которого рассчитайте значения вероятности ошибок распознавания для трех различных случаев априорных вероятностей гипотез:

- $p(\omega_1) > p(\omega_2)$;
- $p(\omega_1) = p(\omega_2)$;
- $p(\omega_1) < p(\omega_2)$;

Сравните полученные вероятности ошибок их со значениями, вычисленными теоретически. Провести анализ полученных результатов и представить его в виде выводов по проделанной работе.

Получить матрицы ошибок (теоретическую и экспериментальную) распознавания трех образов (первых букв имени фамилии и отчества исполнителя). Использовать разделяющую функцию вида (5.36) (раздел учебника 5.3.3). Значение вероятности искажения символа pI = 0.6.

Код программы (внесённые изменения в шаблон кода выделены)

```
% Файл pr54_rec_bin. Синтез и анализ алгоритмов распознавания образов по дискретным
% признакам (на примере распознавания бинарных изображений)
% ПРИМЕР ДЛЯ ТРЁХ КЛАССОВ С ВЫВОДОМ МАТРИЦ ОШИБОК
clear all;
close all;
%% 1. Задание исходных данных
n = 35; % количество признаков (исходя из размера изображений)
M = 3; % ЗДЕСЬ M меняется на 3
s = zeros(n, M); % количество классов и эталонные описания
% 1.1. Задание эталонов классов (ЗДЕСЬ добавляется описание третей буквы)
letterP = [1 1 1 1 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ]';
letterY = [1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 1 1 1 1 ...
           0 0 0 0 1 ...
           0 0 0 0 1 ...
           1 1 1 1 1 ]';
letterA = [1 1 1 1 1 ...   % Добавляется третья буква
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ...
           1 1 1 1 1 ...
           1 0 0 0 1 ...
           1 0 0 0 1 ]';
s(:, 1) = letterP;
s(:, 2) = letterY;
s(:, 3) = letterA; % Добавляется третья буква
% pw1 < pw2 < pw3
% pw1 < pw2 = pw3
% pw1 < pw2 > pw3
% pw1 = pw2 < pw3
% pw1 = pw2 = pw3
% pw1 = pw2 > pw3
% pw1 > pw2 < pw3
% pw1 > pw2 = pw3
% pw1 > pw2 > pw3
PW = [0.2 0.3 0.5; % pw1 < pw2 < pw3
    0.2 0.4 0.4; % pw1 < pw2 = pw3
    0.2 0.5 0.3; % pw1 < pw2 > pw3
    0.3 0.3 0.4; % pw1 = pw2 < pw3
    0.333333 0.333333 0.333334; % pw1 = pw2 = pw3
    0.4 0.4 0.2; % pw1 = pw2 > pw3
    0.4 0.2 0.3; % pw1 > pw2 < pw3
    0.4 0.3 0.3; % pw1 > pw2 = pw3
    0.5 0.3 0.2];% pw1 > pw2 > pw3
% 1.2. Задание параметров эксперимента (ЗДЕСЬ добавляется третье значение в pw)
```

```
for pwi = 1:9
  pw = PW(pwi,:); %априорные вероятности гипотез
  disp('Априорные вероятности классов');
  disp(pw);
  np = sum(pw);
  pw = pw / np; % исключение некорректного задания априорных вероятностей
  N = 20;  % количество шагов изменения варьируемого параметра - pi
  K = 1000; % количество реализаций
  pi = zeros(1, N); % массив вероятностей искажения символа
  % 1.4. Матрицы ошибок (ЗДЕСЬ добавляется теоретическая матрица ошибок)
  Pc_ = zeros(M); % экспериментальная матрица вероятностей ошибок
  Pt = zeros(M); % МАТРИЦЫ теперь двумерные
  pI = 0.6; % ЗДЕСЬ задается фиксированное значение вероятности искажения (pI)
  % 2.Синтез решающего правила и расчет теоретических вероятностей ошибок
  if pI == 0
    pI = 0.0001;
  end % регуляризация разделяющей функции
  if pI == 0.5
    pI = 0.4999;
  end
  pI_ = 1 - pI;
  s_ = 1 - s; % получение инвертированных изображений
  G1 = zeros(1, n);
  G2 = zeros(1, n);
  % Попарное сравнение классов
  for ii = 1 : M - 1
    for jj = ii + 1 : M
      % 2.1. Вычисление порога принятия решений
      ns = sum(abs(s(:, ii) - s(:, jj))); % общее количество несовпадающих
элементов
      l0_ = log(pw(jj) / pw(ii)); % порог принятия решения
      L0  = log(pw(jj) / pw(ii)) / (2 * log(pI_) - 2 * log(pI)) + ns / 2;
      L0r = floor(L0);
      % 2.2. Вычисление коэффициентов разделяющей функции (этот пункт отсюда можно
убрать)
      for k = 1 : n
         G1(1, k) = log((s(k, ii) * pI_ + s_(k, ii) * pI) / (s(k, jj) * pI_ + s_(k,
jj) * pI));
         G2(1, k) = log((s(k, ii) * pI + s_(k, ii) * pI_) / (s(k, jj) * pI + s_(k,
jj) * pI_));
      end
      % 2.3. Определение вероятностей перепутывания
      if pI < 0.5 % расчет вероятностей ошибок
        Pt(ii,jj) = binocdf(L0r, ns, 1-pI);
        Pt(jj,ii) = 1 - binocdf(L0r, ns, pI);
      else
        Pt(ii,jj) = 1 - binocdf(L0r, ns, 1 - pI);
        Pt(jj,ii) = binocdf(L0r, ns, pI);
      end
    end
  end
  % ЗДЕСЬ добавляется строчка с вычислением диагональных элементов теор. матрицы
ошибок
  % Вычисление вероятностей правильного распознавания (диагональные элементы)
  Pt = Pt + diag(ones(3, 1) - sum(Pt, 2));
```

```
  %% 3. Тестирование алгоритма методом статистических испытаний
  for kk = 1 : K % цикл по числу реализаций
    for i = 1 : M % цикл по классам
      % 3.1. Моделирование искажения
      x = s(:, i);
      r = rand(n, 1);
      ir = find(r < pI);
      x(ir) = 1 - x(ir); % искажение элементов - инверсия в случайных точках
      x_ = 1 - x;
      % 3.2. Классификация искаженного образа (попарное сравнение классов)
      % ЗДЕСЬ добавляется инициализация массива для хранения
      % результатов попарных сравнений классов; п.3.2 помещается
      % внутрь двойного цикла вместе с п.2.1 и 2.2.
      iais = []; % результаты попарных сравнений (индексы классов)
      for ii = 1 : M - 1
    for jj = ii + 1 : M
      % Копия 2.1. Вычисление порога принятия решений
          ns = sum(abs(s(:, ii) - s(:, jj))); % общее количество несовпадающих
элементов
          l0_ = log(pw(jj) / pw(ii)); % порог принятия решения
          L0  = log(pw(jj) / pw(ii)) / (2 * log(pI_) - 2 * log(pI)) + ns / 2;
          L0r = floor(L0);
          % Копия 2.2.Вычисление коэффициентов разделяющей функции
          for k = 1 : n % вычисление коэффициентов разделяющей функции
             G1(1, k) = log((s(k, ii) * pI_ + s_(k, ii) * pI) / (s(k, jj) * pI_ +
s_(k, jj) * pI));
             G2(1, k) = log((s(k, ii) * pI + s_(k, ii) * pI_) / (s(k, jj) * pI +
s_(k, jj) * pI_));
          end
          % 3.2. Классификация искаженного образа
          u = G1 * x + G2 * x_ - l0_; % вычисление значения разделяющих функций
          if u > 0
            iai = ii;
          else
            iai = jj;
          end
          % ЗДЕСЬ Запоминаем результат попарного сравнения
          iais = [iais, iai];
        end
      end
      % ЗДЕСЬ выбираем индекс класса, за который проголосовали
      % большинство парных классификаторов
      id = mode(iais);   % самый часто повторяющийся индекс
      % 3.3. Фиксация результата распознавания
      Pc_(i, id) = Pc_(i, id) + 1; % фиксация результата распознавания
    end
  end
  Pc_ = Pc_ / K; % Здесь можно убрать индексы при Pc_
  disp('Теоретическая матрица ошибок')
  disp(Pt)
  disp('Экспериментальная матрица ошибок')
  disp(Pc_)
end
```

Результаты выполнения задания

Таблица 1 - Значения матриц ошибок

Априорные вероятности	**Теоретическая**			**Экспериментальная**		
$p(\omega_1) < p(\omega_2) < p(\omega_3)$ 0,2 0,3 0,5	-0.0305	0.2465	0.7840	0.1170	0.2370	0.6460
	0.2465	0.4460	0.3075	0.0170	0.7140	0.2690
	0.0640	0.1501	0.7859	0.0280	0.1290	0.8430
$p(\omega_1) < p(\omega_2) = p(\omega_3)$ 0.2 0.4 0.4	-0.2512	0.4672	0.7840	0.1060	0.2480	0.6460
	0.0994	0.7505	0.1501	0.0220	0.7050	0.2730
	0.0640	0.3075	0.6285	0.0310	0.1680	0.8010
$p(\omega_1) < p(\omega_2) > p(\omega_3)$ 0.2 0.5 0.3	0.1808	0.4672	0.3520	0.1010	0.3880	0.5110
	0.0994	0.7505	0.1501	0.0190	0.8390	0.1420
	0.3520	0.3075	0.3405	0.0350	0.2830	0.6820
$p(\omega_1) = p(\omega_2) < p(\omega_3)$ 0.3 0.3 0.4	0.4015	0.2465	0.3520	0.4850	0.2050	0.3100
	0.2465	0.4460	0.3075	0.1750	0.6400	0.1850
	0.3520	0.1501	0.4979	0.2730	0.1610	0.5660
$p(\omega_1) = p(\omega_2) = p(\omega_3)$ 0.333333 0.333333 0.333334	0.4015	0.2465	0.3520	0.4780	0.1930	0.3290
	0.2465	0.4460	0.3075	0.1480	0.6870	0.1650
	0.3520	0.1501	0.4979	0.2800	0.1360	0.5840
$p(\omega_1) = p(\omega_2) > p(\omega_3)$ 0.4 0.4 0.2	0.6895	0.2465	0.0640	0.7020	0.2500	0.0480
	0.2465	0.6034	0.1501	0.2210	0.7520	0.0270
	0.7840	0.3075	-0.0915	0.6040	0.2000	0.1960
$p(\omega_1) > p(\omega_2) < p(\omega_3)$ 0.4 0.2 0.3	0.5486	0.0994	0.3520	0.5860	0.1010	0.3130
	0.4672	0.2252	0.3075	0.3140	0.5180	0.1680
	0.3520	0.1501	0.4979	0.3270	0.0770	0.5960
$p(\omega_1) > p(\omega_2) = p(\omega_3)$ 0.4 0.3 0.3	0.4015	0.2465	0.3520	0.5140	0.1800	0.3060
	0.2465	0.6034	0.1501	0.1650	0.6580	0.1770
	0.3520	0.3075	0.3405	0.2590	0.1500	0.5910
$p(\omega_1) > p(\omega_2) > p(\omega_3)$ 0.5 0.3 0.2	0.8366	0.0994	0.0640	0.8320	0.1060	0.0620
	0.4672	0.3826	0.1501	0.4560	0.5160	0.0280
	0.7840	0.3075	-0.0915	0.7150	0.0980	0.1870

Выводы

1. Симметричный вид зависимости вероятности ошибки распознавания от вероятности искажения символа объясняется тем, что по мере увеличения вероятности искажения искажённая картинка становится всё более похожей на инвертированное исходное изображение. Инвертированное изображение также учитывается при распознавании образа.