

13. ПОЛУЧЕНИЕ ИЗОБРАЖЕНИЯ. ОСНОВНЫЕ ПОНЯТИЯ ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

Визуальная информация преобразуется в электрические сигналы с помощью видеодатчиков. После пространственной дискретизации и квантования по амплитуде эти сигналы дают цифровое изображение.

Основными устройствами, используемыми в системах компьютерного видения, являются телевизионные камеры, состоящие из электронной трубки или твердотельного чувствительного элемента с соответствующими электронными блоками. Твердотельные чувствительные элементы представляются, как правило, приборами с зарядовой связью (ПЗС). Они обладают рядом преимуществ перед электронными трубками (например, имеют меньшие вес и размеры, больший ресурс и малую энергоемкость). Однако, разрешающая способность некоторых трубок пока выше возможности твердотельных камер.

Трубка видикон представляет собой цилиндрическую стеклянную оболочку, содержащую с одного конца электронную пушку, а с другого - экран и мишень. Электронный луч фокусируется и отклоняется с помощью напряжения, прикладываемого к катушкам. Отклоняющий контур обеспечивает сканирование луча по внутренней поверхности мишени для считывания изображения. В процессе сканирования в металлическом слое образуется ток, который подается на выводы трубки. Величина тока пропорциональна числу перемещающихся электронов и следовательно интенсивности светового потока в соответствующем месте. Это изменение тока во время сканирования электронного луча после его обработки в электронном блоке камеры формирует видеосигнал, пропорциональный интенсивности входного изображения. Причем, электронный луч, сканирует по всей поверхности мишени 30 раз в 1 секунду. Полный объем сканирования (называемый кадром) состоит из 525 линий, 480 из которых содержат информацию об изображении.

При рассмотрении устройств на ПЗС удобно подразделить датчики на два типа: датчики линейного сканирования и датчики с плоскостной структурой. Основной частью ПЗС-датчиков является ряд кремниевых чувствительных элементов, называемых фотоячейками. Фотоны от отображаемого объекта проходят через входную прозрачную полукристаллическую кремниевую структуру и образуют пары электрон-дырка. Полученные фотоэлектроны собираются на фотоячейках, при этом величина заряда на каждой фотоячейке пропорциональна соответствующей интенсивности светового потока.

ПЗС-датчики с плоскостной структурой аналогичны датчикам линейного сканирования, с тем отличием, что в них фотоячейки расположены в форме матрицы, а между рядами фотоячеек имеется комбинация переходных регистров. Разрешающая способность датчиков с плоскостной структурой составляет 32×32 элементов на нижней границе и 256×256 элементов для среднего диапазона.

Современные ПЗС-датчики имеют 1024×1024 элементов и больше.

Обозначим через $f(x,y)$ двумерное изображение, получаемое телевизионной камерой или ПЗС-датчиком, дающим изображение:

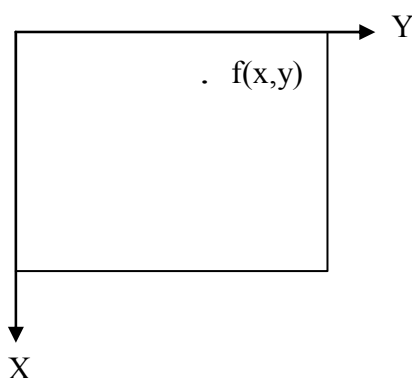


Рис. 13.1

Здесь x и y — пространственные координаты, т. е. координаты плоскости изображения, а величина f в произвольной точке (x,y) пропорциональна яркости (интенсивности) изображения в этой точке.

Для получения удобной с точки зрения вычисления формы, функции изображения $f(x,y)$, ее необходимо дискретизировать как пространственно, так и по амплитуде (интенсивности). Дискретизацию по пространственным координатам (x,y) будем называть дискретизацией изображения, а амплитудную дискретизацию – квантованием по интенсивности или квантованием по уровню серого. Последний термин означает тот факт, что изображение меняется по тону от черного до белого в серых оттенках.

Предположим, что непрерывное изображение дискретизировано равномерно на N строк и M столбцов, причем каждая дискретная величина проквантована по интенсивности. Такая система называется цифровым изображением и может быть представлена в виде:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \dots & \dots & \dots & \dots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix}$$

где x и y – дискретные переменные:

$$x = 0, 1, \dots, N-1; y = 0, 1, \dots, M-1.$$

Каждый элемент системы называется элементом изображения, элементом картинки или пикселом.

В соответствии с рис.13.1 $f(0,0)$ является пикселем начала координат изображения, $f(0,1)$ – правый от него пиксел, и так далее.

Обычно на практике величины N , M и число уровней дискретной интенсивности каждого квантованного пиксела задают в виде целых степеней числа 2.

Степень дискретизации и число уровней интенсивности, необходимые для получения требуемого в системе изображения объекта зависят от самого изображения и от вида его использования.

Необходимо отметить, что для получения качественной черно-белой телевизионной картинки требуется 512×512 пикселей со 128 уровнями интенсивности. Как правило, универсальная система компьютерного видения должна иметь, как минимум, разрешающую способность порядка 256×256 пикселей с 64 уровнями интенсивности.

14. ОСНОВНЫЕ ПРЕОБРАЗОВАНИЯ ИЗОБРАЖЕНИЯ

Основными преобразованиями при обработке изображений являются вращение, изменение масштаба и смещение (сдвиг) изображения. Все преобразования записываются в трехмерной декартовой системе координат, в которой координаты точки записываются как (X,Y,Z) . В случае двумерных изображений для обозначения координат пиксела используется сокращенная запись (x,y) .

Смещение (сдвиг) изображения

Предположим, что требуется сместить точку с координатами (X,Y,Z) в новое место, используя перемещения (X_0,Y_0,Z_0) . Смещение выполняется в соответствии с соотношениями:

$$X^*=X+X_0, Y^*=Y+Y_0, Z^*=Z+Z_0,$$

где X^*, Y^*, Z^* - координаты новой точки.

В матричном виде это можно записать как $V^*=TV$, то есть:

$$\begin{bmatrix} X^* \\ Y^* \\ Z^* \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & X_0 \\ 0 & 1 & 0 & Y_0 \\ 0 & 0 & 1 & Z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Изменение масштаба изображения

Масштабирование на коэффициенты S_x, S_y, S_z по осям X, Y, Z производится с помощью матрицы преобразования:

$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Вращение изображения

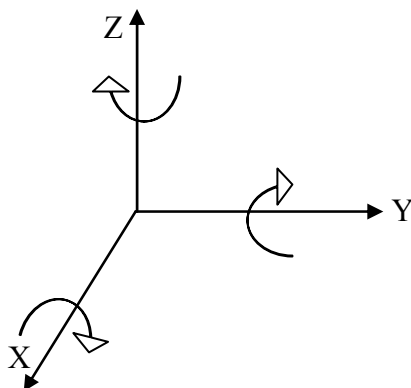


Рис. 13.2

Вращение точки относительно координатной оси Z на угол θ реализуется с помощью преобразования

$$R\theta = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Вращение точки вокруг оси X на угол α выполняется с помощью преобразования

$$R\alpha = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha & 0 \\ 0 & -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Вращение точки вокруг оси Y на угол β реализуется с помощью преобразования

$$R\beta = \begin{bmatrix} \cos \beta & 0 & -\sin \beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \beta & 0 & \cos \beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

15. ОСНОВНЫЕ ВЗАИМОСВЯЗИ МЕЖДУ ПИКСЕЛАМИ ИЗОБРАЖЕНИЯ. МЕТРИЧЕСКИЕ СВОЙСТВА ИЗОБРАЖЕНИЯ

Основные взаимосвязи между пикселями изображения

Пиксел p с координатами (x, y) имеет четыре горизонтальных и вертикальных соседних пиксела с координатами:

$$(x+1, y), (x-1, y), (x, y+1), (x, y-1)$$

Эта группа пикселей, называемая четыре соседа p , обозначается через $N_4(p)$. Четыре диагональных соседних пиксела по отношению к пикселу p имеют координаты

$$(x+1, y+1), (x+1, y-1), (x-1, y+1), (x-1, y-1)$$

и обозначаются через $N_d(p)$. Эти четыре точки вместе с четырьмя указанными выше называются восемь соседей p и обозначаются через $N_8(p)$. Некоторые из точек $N_4(p)$, $N_d(p)$, $N_8(p)$ могут выходить за пределы изображения, если (x, y) находится на границе изображения.

Рассмотрим три типа связей пикселей:

1. Четырехсвязный. Два пиксела p и q с определенными значениями интенсивности являются четырехсвязными, если q относится к группе $N_4(p)$.
2. Восьмисвязный. Два пиксела p и q со значениями интенсивности из V , где V -ряд значений интенсивности, являются восьмисвязными, если q относится к группе $N_8(p)$.
3. m -связный (смешанная связь). Два пиксела p и q со значениями интенсивностей из V являются m -связными, если:
 - а) q относится к группе $N_4(p)$

б) q относится к группе $N_d(p)$ и множество $N_4(p) \cap N_4(q)$ – пустое.

Иными словами, это множество пикселей, являющихся четырьмя соседними, как по отношению к p , так и по отношению к q со значениями интенсивности из V .

Метрические свойства для изображений

Расстояния между двумя векторами X и Y размерностью $n \times 1$ удовлетворяют трем аксиомам:

- 1) $d(X, Y) = 0$, если $X = Y$
- 2) $d(X, Y) = d(Y, X)$
- 3) $d(X, Y) \leq d(X, Z) + d(Z, Y)$, для всех Z

Наиболее важной метрикой является l^p -метрика

$$l^p(x, y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{1/p} \quad \text{для } p \geq 1$$

Если $p=2$, мы получим классическую Евклидову метрику.

Определим для пикселей p, q, z соответственно с координатами (x, y) , (s, t) и (u, v) функцию расстояния или метрику D следующим образом:

- 1) $D(p, q) \geq 0$ [$D(p, q) = 0$, если $p = q$]
- 2) $D(p, q) = D(q, p)$ (симметрия)
- 3) $D(p, z) \leq D(p, q) + D(q, z)$ (неравенство треугольника)

Евклидово расстояние между p и q определяется из соотношения:

$$D_E(p, q) = [(x-s)^2 + (y-t)^2]^{1/2}$$

При измерении этого расстояния пиксели, имеющие расстояние? меньшее или равное некоторой величине r от (x,y) располагаются в окружности радиусом r с центром в (x,y) .

Расстояние D_4 , называемое модульным между p и q , определяется соотношением:

$$D_4(p,q) = |x-s| + |y-t|$$

Метрику D_4 иногда называют метрикой городских кварталов, из-за того, что расстояния измеряются параллельно осям координат.

Покажем, что функция D_4 удовлетворяет к примеру, условию 3. Запишем соотношение:

$$D_4(p,q) + D_4(q,z) = |x-s| + |y-t| + |s-u| + |t-v|$$

Но, согласно неравенству треугольника для вещественных чисел будем иметь:

$$|x-s| + |s-u| \geq |x-u|,$$

$$|y-t| + |t-v| \geq |y-v|,$$

$$D_4(p,q) + D_4(q,z) \geq |x-u| + |y-v| \geq D_4(p,z), \text{ т.е.}$$

$$D_4(p,z) \leq D_4(p,q) + D_4(q,z), \text{ что и требовалось доказать.}$$

В этом случае пиксели, имеющие расстояние D_4 , меньшее или равное некоторой величине r от (x,y) образуют ромбовидную структуру с центром в (x,y) (центральная точка). Например, пиксели с расстоянием $D_4 < 2$ от (x,y) образуют следующие контуры с равными от центра расстояниями

```

      2
    2 1 2
  2 1 0 1 2
    2 1 2
      2

```

Расстояние D_8 , называемое также шахматным между p и q определяется по формуле:

$$D_8(p,q) = \max \{ |x-s|, |y-t| \}$$

При этом пиксели с расстоянием, меньшим или равным некоторой величине r образуют квадрат с центром в (x,y) . Например, пиксели с расстоянием $D_8 \leq 2$ от центральной точки образуют следующие равноудаленные от центра контуры:

```

  2 2 2 2 2
  2 1 1 1 2
  2 1 0 1 2
  2 1 1 1 2
  2 2 2 2 2

```

Геометрические места точек, удаленных на единичное расстояние от начала координат, построенные для различных метрик будут иметь вид:

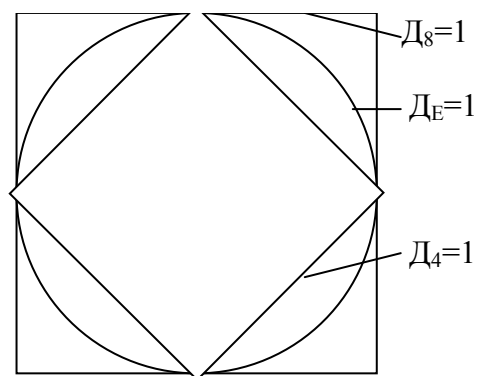


Рис. 15.1

16. МЕТОДЫ ПРОСТРАНСТВЕННОЙ И ЧАСТОТНОЙ ОБЛАСТИ ПРИМЕНИТЕЛЬНО К ПРЕДВАРИТЕЛЬНО ОБРАБОТКЕ ИЗОБРАЖЕНИЯ

К пространственной области относится совокупность пикселей, составляющих изображение. Функция предварительной обработки в пространственной области записывается в виде:

$$g(x,y) = h[f(x,y)],$$

где $f(x,y)$ - входное изображение,

$g(x,y)$ – выходное (обработанное) изображение,

h – оператор функции f , определенный в некоторой области (x,y) .

Основным подходом при определении окрестности точки (x,y) является использование квадратной или прямоугольной области части изображения с центром в точке (x,y) . Центр этой части изображения перемещается от пиксела к пикселу, начиная, например, от левого верхнего угла. При этом для получения $g(x,y)$ оператор применяется для каждого положения (x,y) . Хотя используются иногда и другие формы окрестности (например, круг), квадратные формы более предпочтительны из-за простоты их реализации.

Один из наиболее применяемых методов пространственной области основан на использовании так называемых масок свертки (шаблонов, окон, фильтров). Обычно маска представляет собой небольшую (например, размерность 3×3) двумерную систему, коэффициенты которой выбираются таким образом, чтобы обнаружить заданное свойство изображения.

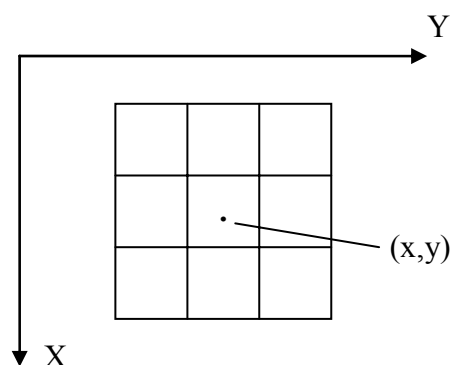


Рис. 16.1

На рис.м 16.1 представлена окрестность размерностью 3*3 точки (x,y) изображения. Предположим для начала, что дано изображение с постоянной интенсивностью, которое содержит отдельные, удаленные друг от друга пиксели с отличной от фона интенсивностью. Эти точки могут быть обнаружены маской, показанной на рис. 16.2

-1	-1	-1
-1	8	-1
-1	-1	-1

Рис. 16.2

Процесс заключается в следующем. Центр маски, помеченный цифрой 8, перемещается по изображению определенным образом. При совпадении центра маски с положением каждого пиксела производится умножение значений всех пикселей, находящихся под маской, на соответствующий коэффициент на маске, т.е. значение пиксела под центром маски умножается на 8, а значения восьми соседних пикселей умножаются на -1. Затем результаты этих умножений суммируются. Если все пиксели под маской имеют одинаковые значения (постоянный фон), то сумма будет равна нулю. Если же центр маски разместится над точкой с другой интенсивностью, сумма будет отлична от нуля. В случае размещения указанной точки вне центра, сумма также будет отлична от нуля, но на меньшую величину. Это

различие может быть устранено путем сравнения значения суммы с пороговым значением.

Если величины w_1, w_2, \dots, w_9 представляют собой коэффициенты, маски пиксела (x, y) и его восьми соседей (рис.16.3), то описанный выше алгоритм можно представить как выполнение следующей операции на окрестности 3×3 точки (x, y) .

w_1 $(x-1, y-1)$	w_2 $(x-1, y)$	w_3 $(x-1, y+1)$
w_4 $(x, y-1)$	w_5 (x, y)	w_6 $(x, y+1)$
w_7 $(x+1, y-1)$	w_8 $(x+1, y)$	w_9 $(x+1, y+1)$

Рис. 16.3

$$\begin{aligned}
 h[f(x, y)] = & w_1 f(x-1, y-1) + w_2 f(x-1, y) + w_3 f(x-1, y+1) + \\
 & + w_4 f(x, y-1) + w_5 f(x, y) + w_6 f(x, y+1) + w_7 f(x+1, y-1) + \\
 & + w_8 f(x+1, y) + w_9 f(x+1, y+1)
 \end{aligned}
 \tag{16.1}$$

Методы частотной области используют для своей реализации алгоритмы прямого и обратного двумерного преобразований.

$$\begin{aligned}
 F(u, v) &= \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux+vy)/N} \\
 f(x, y) &= \sum_{U=0}^{N-1} \sum_{V=0}^{N-1} F(u, v) e^{j2\pi(ux+vy)/N}
 \end{aligned}
 \tag{16.2}$$

При вычислении прямого двумерного преобразования Фурье, сначала с использованием техники одномерного преобразования Фурье оперируют строками массива изображения, а затем одномерные преобразования осуществляются над столбцами (построчно-столбцовый алгоритм).

Операция сглаживания используется для снижения шума и других помех, которые могут появиться на изображении в результате его дискретизации и квантования.

Усреднение окрестности является прямым методом пространственной области для сглаживания изображения. Для изображения $f(x,y)$ процесс заключается в получении сглаженного изображения $g(x,y)$, интенсивность которого в каждой точке (x,y) равна усредненному значению интенсивности пикселей функции f , содержащихся в заданной окрестности точки (x,y) . Иными словами сглаженное изображение получается с использованием соотношения:

$$g(x,y) = \frac{1}{P} \sum_{n,m \in S} f(n,m)$$

для всех x,y функции $f(x,y)$, где p - общее число точек в окрестности.

Для окрестности размерностью $3*3$ указанное соотношение является частным случаем выражения (16.1) при $w_i=1/9$.

17. СЕГМЕНТАЦИЯ ИЗОБРАЖЕНИЯ ПОСРЕДСТВОМ ВЫДЕЛЕНИЯ ГРАНИЦ ОБЛАСТЕЙ

Сегментация означает выделение областей, однородных по своим яркостным, цветовым или текстурным свойствам, либо посредством выделения их границ, либо путем разметки внутренних точек. В свою очередь, можно выделить два класса, которыми определяются методы выделения границ: пространственное дифференцирование и высокочастотная фильтрация. Причем общим для указанных методов является стремление рассматривать границу (край), как область резкого перепада функции яркости изображения.

Метод пространственного дифференцирования основан на предположении о том, что граничные (краевые) точки имеют большую величину модуля градиента функции $f(x,y)$. Градиент изображения $f(x,y)$ в точке (x,y) определяется как двумерный вектор:

$$\overline{G[f(x,y)]} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad (17.1)$$

7

Из векторного анализа известно, что вектор G указывает направление максимального изменения функции f в точке (x,y) . Величина этого вектора определяется как:

$$G[f(x,y)] = [G_x^2 + G_y^2]^{1/2} = \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2} \quad (17.2)$$

На практике, как правило, градиент аппроксимируется абсолютными значениями

$$G[f(x,y)] \approx |G_x| + |G_y| \quad (17.3)$$

Как видно из указанных соотношений, вычисление градиента основано на нахождении первых производных $\partial f/\partial x$, $\partial f/\partial y$.

Для цифрового изображения это можно сделать несколькими путями. Один из подходов состоит в использовании разности между соседними пикселями.

$$G_x = \partial f/\partial x = f(x,y) - f(x-1,y) \quad (17.4)$$

$$G_y = \partial f/\partial y = f(x,y) - f(x,y-1)$$

Несколько более сложный способ включает пиксели в окрестности размерностью 3*3 и более.

Структура метода пространственного дифференцирования может быть представлена следующим образом

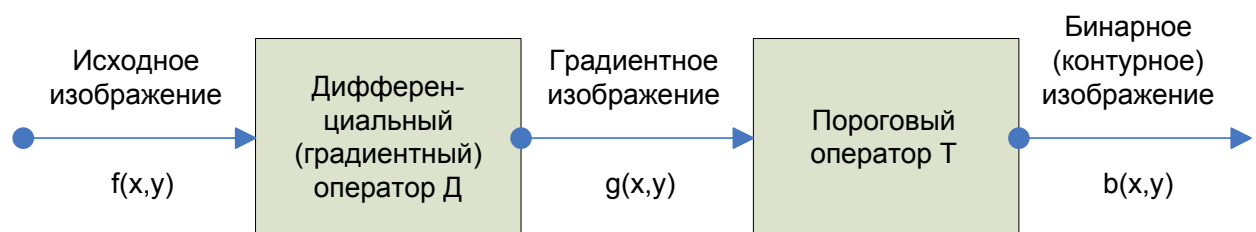


Рис. 7

Получаемое после преобразования Д поле $g(x,y)$ называется градиентным изображением или изображением с усиленными границами. Обработка градиентного изображения осуществляется с помощью порогового оператора согласно правилу:

$$b(x,y) = \begin{cases} 1 & \text{при } g(x,y) \geq T \\ 0 & \text{при } g(x,y) < T \end{cases}$$

где T – величина постоянного или меняющегося по определенному правилу порога.

Вычислительная реализация данного метода сводится к синтезу численных алгоритмов оценки частных производных в некоторой точке изображения и выполнению далее несложных арифметических операций.

Пусть через $f(m,n)$ и $g(m,n)$, где $m, n = \overline{0, N-1}$ обозначены дискретизированные исходное и градиентное изображения и в качестве дискретного аналога соотношения (17.2) выступает:

$$g(m,n) = [d_1^2(m,n) + d_2^2(m,n)]^{1/2}$$

или его упрощенный вычислительный вариант:

$$g(m,n) = |d_1(m,n)| + |d_2(m,n)|$$

В этих соотношениях $d_1(m,n)$ и $d_2(m,n)$ обозначают оценки частных производных функции яркости в точке (m,n) .

Обозначим через F_{mn} и F_{mn} векторы

$$(F_{mn})^T = (f_{mn}, f_{m,n+1}, f_{m+1,n}, f_{m+1,n+1})$$

$$(F_{mn})^T = (f_{m-1,n-1}, f_{m-1,n}, f_{m-1,n+1}, f_{m,n-1}, f_{m,n}, f_{m,n+1}, f_{m+1,n-1}, f_{m+1,n}, f_{m+1,n+1}),$$

представляющие собой упорядоченные по строкам элементы окрестности точки (m,n) , размерностью $2*2$ и $3*3$.

В качестве масок (шаблонов) можно использовать разностный оператор:

$$W_1 = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}; W_2 = \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}$$

тогда:

$$d_1(m,n) = [f(m,n) - f(m,n+1)] + [f(m+1,n) - f(m+1,n+1)]$$

$$d_2(m,n) = [f(m+1,n) - f(m,n)] + [f(m+1,n+1) - f(m,n+1)]$$

Оператор Робертса

$$W_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}; W_2 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$d_1(m,n) = [f(m,n) - f(m+1,n+1)]$$

$$d_2(m,n) = [f(m+1,n) - f(m,n+1)]$$

Оператор Превитта

$$W_1 = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}; W_2 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$d_1(m,n) = [f(m-1,n-1) + f(m,n-1) + f(m+1,n-1)] - [f(m-1,n+1) + f(m,n+1) + f(m+1,n+1)]$$

$$d_2(m,n) = [f(m+1,n-1) + f(m+1,n) + f(m+1,n+1)] - [f(m-1,n-1) + f(m-1,n) + f(m-1,n+1)]$$

Оператор Собеля

$$W_1 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}; W_2 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$d_1(m,n) = [f(m-1,n-1) + 2f(m,n-1) + f(m+1,n-1)] - [f(m-1,n+1) + 2f(m,n+1) + f(m+1,n+1)]$$

$$d_2(m,n) = [f(m+1,n-1) + 2f(m+1,n) + f(m+1,n+1)] - \\ - [f(m-1,n-1) + 2f(m-1,n) + f(m-1,n+1)]$$

18. ОСНОВНЫЕ СВОЙСТВА ДВУМЕРНОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

Сдвиг

Пара преобразований Фурье обладает следующими трансляционными (сдвиговыми) свойствами

$$f(x,y)e^{j2\pi(u_0 x/M+v_0 y/N)} \Leftrightarrow F(u-u_0, v-v_0)$$

$$f(x-x_0, y-y_0) \Leftrightarrow F(u, v)e^{-j2\pi(x_0 u/M+y_0 v/N)},$$

где двойная стрелка использована для обозначения того, что соответствующие функции образуют пару Фурье-преобразований.

При $u_0=M/2$, $v_0=N/2$ будем иметь

$$e^{j2\pi(u_0 x/M+v_0 y/N)} = e^{j\pi(x+y)} = (-1)^{x+y}$$

В этом случае будем иметь

$$f(x,y)(-1)^{x+y} \Leftrightarrow F(u-M/2, v-N/2)$$

и аналогично при $x_0=M/2$; $y_0=N/2$

$$f(x-M/2, y-N/2) \Leftrightarrow F(u, v)(-1)^{u+v}$$

Из определения преобразования Фурье следует, что

$$F[f_1(x,y)+f_2(x,y)] = F[f_1(x,y)] + F[f_2(x,y)] \text{ и}$$

$$F[f_1(x,y)*f_2(x,y)] \neq F[f_1(x,y)]*F[f_2(x,y)]$$

Иными словами, преобразование Фурье обладает свойством дистрибутивности по отношению к сложению, но не обладает этим свойством по отношению к умножению. Указанное относится и к обратному Фурье-преобразованию.

Аналогично, если a, b – две постоянные, то

$$f(ax, by) \Leftrightarrow aF(u, v)$$

$$f(ax, by) \Leftrightarrow \frac{1}{|ab|} F(u/a, v/b), \quad ab \neq 0$$

(изменение масштаба)

Поворот

Введем понятие координаты

$$x = r \cdot \cos \theta, \quad y = r \cdot \sin \theta$$

$$u = \omega \cdot \cos \varphi$$

$$v = \omega \cdot \sin \varphi$$

И обозначим через $f(r, \theta)$ и $F(\omega, \varphi)$ функции, рассматриваемые как функции полярных координат соответственно.

$$\text{Тогда } f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0)$$

Это выражение указывает на то, что поворот функции $f(x, y)$ на угол θ_0 приводит к повороту функции $F(u, v)$ на тот же угол.

Дискретное преобразование Фурье обладает следующими свойствами периодичности

$$F(u, v) = F(u + M, v) = F(u, v + N) = F(u + M, v + N)$$

Обратное Фурье-преобразование также периодически:

$$f(x,y)=f(x+M,y)=f(x,y+N)=f(x+M,y+N)$$

Свойство симметрии относительно операции комплексного сопряжения

$$F(u,v)=F^*(-u,-v)$$

Отсюда следует, что спектр центрально симметричен относительно начала координат

$$|F(u,v)| = |F(-u,-v)|$$

В случае двумерного преобразования спектр определяется как

$$|F(u,v)| = [R^2(u,v)+I^2(u,v)]^{1/2}$$

где $R=\text{Re}(F)$, $I=\text{Im}(F)$

Фаза определяется как $\Phi(u,v)=\text{arctg}[I(u,v)/R(u,v)]$

Энергетический спектр вычисляется как $P(u,v)=|F(u,v)|^2$, а среднее значение будет равно

$$\bar{f}(x,y) = F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)$$

Представление в полярных координатах имеет вид

$$F(u,v) = |F(u,v)| e^{-j\Phi(u,v)}$$

Свойство разделения переменных означает, что для вычисления двумерного преобразования Фурье необходимо вначале вычислить одномерные преобразования по каждой строке изображения, а затем вычислить одномерные преобразования по каждому столбцу полученного промежуточного результата. Изменение порядка вычислений (сначала по столбцам, затем по строкам) приводит к тому же результату.

Дискретная свертка двух функций $f(x,y)$ и $h(x,y)$ размерностью $M \times N$ обозначается как $f(x,y) * h(x,y)$ и определяется как

$$f(x,y) * h(x,y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) h(x-m, y-n)$$

Используя двумерное Фурье-представление, будем иметь

$$f(x,y) * h(x,y) \Leftrightarrow F(u,v) H(u,v)$$

$$f(x,y) h(x,y) \Leftrightarrow F(u,v) * H(u,v)$$

При вычислении двумерной свертки необходимо использовать технологию дополнения нулями.

Пусть мы имеем два изображения $f(x,y)$ и $h(x,y)$ размерами $A \times B$ и $C \times D$ соответственно. Как и в одномерном случае необходимо рассматривать эти массивы как периодические с некоторыми периодами P в направлении x и Q в направлении y . Причем выбор

$$P \geq A+C-1$$

$$Q \geq B+D-1$$

позволяет избежать ошибок перекрытия при вычислении двумерной свертки. При этом дополнение нулями функций $f(x,y)$, $h(x,y)$ в основном периоде осуществляется следующим образом

$$fe(x,y) = \begin{cases} f(x,y) & 0 \leq x \leq A-1 \quad 0 \leq Y \leq B-1 \\ 0 & A \leq x \leq P \quad B \leq Y \leq Q \end{cases}$$

$$he(x,y) = \begin{cases} h(x,y) & 0 \leq x \leq C-1 \quad 0 \leq Y \leq D-1 \\ 0 & C \leq x \leq P \quad D \leq Y \leq Q \end{cases}$$

Корреляционная функция двух функций $f(x,y)$ и $h(x,y)$ определяется соотношением

$$f(x,y) \circ h(x,y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m,n) h(x+m, y+n)$$

где f^* обозначает функцию, комплексно сопряженную функции f . Обычно мы имеем дело с действительными функциями и в этом случае $f^*=f$. Корреляция от свертки отличается комплексным сопряжением (в общем случае) и заменой минусов на плюсы во всех членах под знаком суммы. В остальном вычисление корреляционной функции идентично вычислению свертки, включая необходимость дополнения нулями.

Тогда теорема корреляции

$$f(x,y) \circ h(x,y) \Leftrightarrow F^*(u,v) H(u,v)$$

означает, что корреляционная функция в пространственной области может быть получена как результат обратного Фурье-преобразования, примененного к произведению $F^*(u,v) H(u,v)$, где F^* - функция, комплексно сопряженная к F .

Аналогично вычисление корреляционной функции в частотной области сводится к умножению в пространственной области.

$$F(u,v) \circ H(u,v) \Leftrightarrow f^*(x,y)h(x,y)$$

Указанные два утверждения составляют содержание теоремы о корреляции, при чем все функции должны быть дополнены нулями.

Следует отметить, что свертка является связующим звеном между процедурами фильтрации в пространственной и частотной областях, а основная задача, для решения которой используется корреляция – это задача совмещения.

Изображение $f(x,y)$ участвующее в совмещении состоит, как правило, из некоторых объектов или областей. Если мы хотим определить, содержит ли изображение f конкретный интересующий нас объект (область), то мы формируем отдельное изображение $h(x,y)$ этого объекта (обычно называемое эталоном) и вычисляем корреляционную функцию изображения и эталона. Тогда, если данное изображение содержит объект, совмещенный с эталоном, то корреляция двух функций будет иметь максимум, расположенный там, где обнаружено соответствие между эталоном h и изображением. При решении большинства практических задач совмещения необходимы процедуры предварительной обработки, такие как утоньшение, масштабирование, выравнивание (эквализация), однако основой является вычисление корреляции. Вместо термина корреляция очень часто используется термин «взаимная корреляция» чтобы подчеркнуть, что корреляция вычисляется между различными изображениями. Если изображения сравниваемые совпадают, то

$$f(x,y) \circ f(x,y) \Leftrightarrow F^*(u,v)F(u,v) \Leftrightarrow |F(u,v)|^2$$

Это соотношение представляет собой энергетический спектр.

Аналогично

$$|f(x,y)|^2 \Leftrightarrow F(u,v) \circ F(u,v)$$

19. ОСНОВЫ ФИЛЬТРАЦИИ В ЧАСТОТНОЙ ОБЛАСТИ

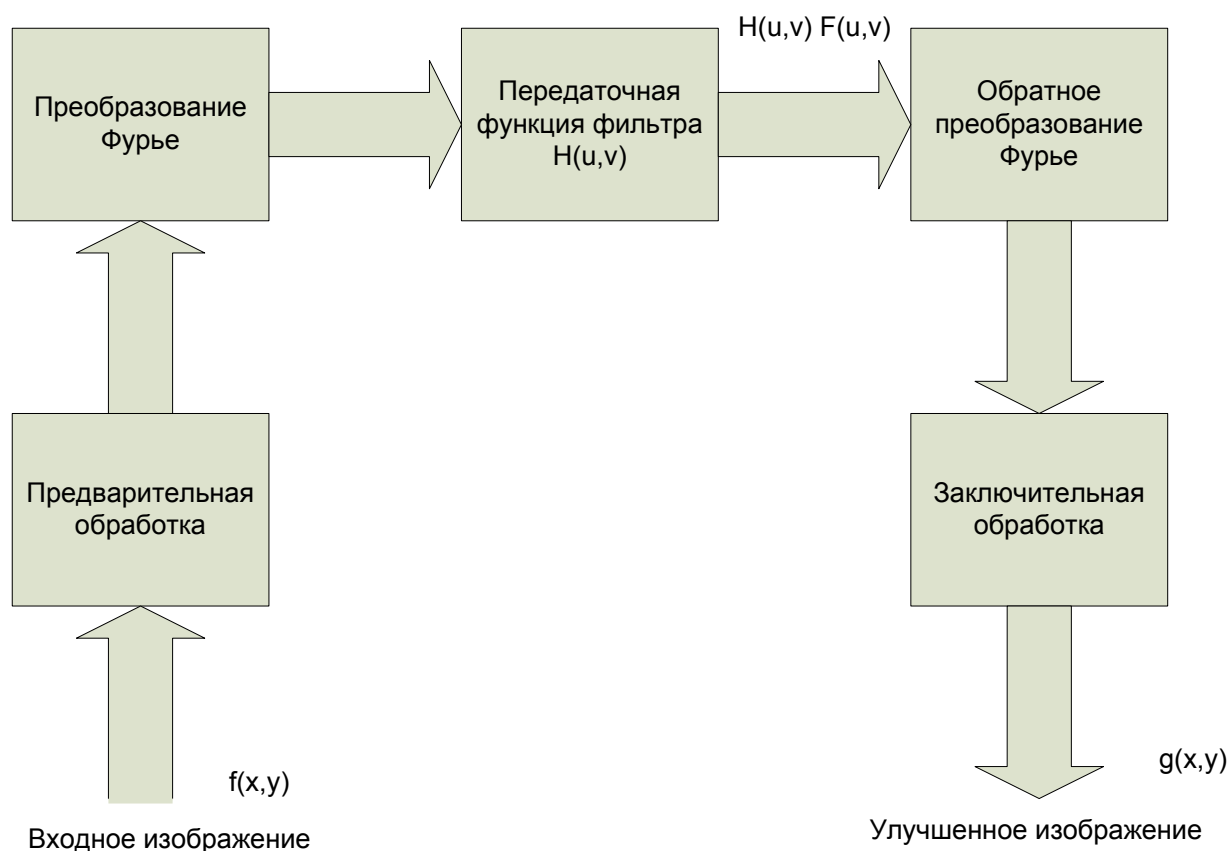


Рис. 19.1

$$F[f(x,y)(-1)^{x+y}] = F(u-M/2, v-N/2) \quad (19.1)$$

Процедура фильтрации в частотной области состоит из следующих шагов:

1. Исходное изображение умножается на $(-1)^{x+y}$, чтобы его Фурье-преобразование в соответствии с (19.1) оказалось центрированным.
2. Вычисляется прямое ДПФ $F(u,v)$ изображения, полученного после шага 1.
3. Функция $F(u,v)$ умножается на $H(u,v)$ (передаточная функция фильтра).
4. Вычисляется обратное ДПФ от результата шага 3.
5. Выделяется вещественная часть результата шага 4.

6. Результат шага 5 умножается на $(-1)^{x+y}$

Причина, по которой множитель $H(u,v)$ называется фильтром, состоит в том, что он подавляет некоторые частоты преобразования, оставляя при этом другие без изменения.

Аналогия с повседневной жизнью возникает при рассмотрении сетчатого фильтра (сито), который пропускает некоторые предметы и не пропускает другие в соответствии с их размерами.

Пусть $f(x,y)$ обозначает входное изображение после шага 1 и пусть $F(u,v)$ есть его Фурье-образ. Тогда Фурье-образ выходного изображения определяется соотношением

$$G(u,v)=H(u,v)F(u,v)$$

Умножение функций двух переменных H и F осуществляется поэлементно. Это означает, что первый элемент H умножается на первый элемент F , второй элемент H – на второй элемент F и так далее.

Фильтрованное изображение получается вычислением обратного преобразования Фурье от Фурье-образа $G(u,v)$

$$g_{\text{фильтр}}(x,y)=F^{-1}[G(u,v)]$$

Искомое изображение получается выделением действительной части из последнего результата и умножения на $(-1)^{x+y}$, чтобы скомпенсировать эффект от умножения входного изображения на ту же величину.

Обратное Фурье-преобразование в общем случае является комплексным. Однако, в случае вещественного входного изображения и вещественной передаточной функции фильтра мнимые части всех значений обратного Фурье-преобразования должны равняться нулю. Однако в связи с ошибками округления при вычислениях на практике значения обратного Фурье-преобразования, как правило, содержат паразитную мнимую

составляющую, которой необходимо при решении прикладных задач пренебречь.

Процедура фильтрации, приведенная на рис.19.1, представлена в общем виде, включая предварительную и заключительную обработку. Помимо умножения на $(-1)^{x+y}$ такая обработка может включать обрезание входного изображения так, чтобы его размеры приняли ближайшие четные значения по отношению к исходным (это необходимо для правильного центрирования Фурье-преобразования), яркостное масштабирование, преобразование форматов входных и выходных данных. Возможны многоступенчатые процедуры фильтрации, а также разнообразные операции предварительной и заключительной обработки. Существенным является то, что метод фильтрации основан на некотором изменении Фурье-образа изображения посредством передаточной функции фильтра и последующем обращении результата для получения обработанного выходного изображения.

Фильтры, основанные на порядковых статистиках

Фильтры, основанные на порядковых статистиках, относятся к классу нелинейных пространственных фильтров. Отклик фильтра определяется предварительным упорядочиванием (ранжированием), значений пикселей, покрываемых маской фильтра и последующим выбором значения, находящегося на определенной позиции упорядоченной последовательности (т. е. имеющего определенный ранг). При этом фильтрация сводится к замещению исходного значения пикселя (в центре маски) на полученное значение отклика фильтра. Из указанного класса наиболее известен медианный фильтр, который заменяет значение пикселя на значение медианы распределения яркости всех пикселей в окрестности, включая исходный. Медианные фильтры особенно эффективны при фильтрации импульсных шумов, иногда называемых шумами «соль и перец», которые выглядят как наложение на изображение случайных черных и белых точек.

Медиана набора чисел есть такое число ξ , что половина чисел из набора меньше или равны ξ , а другая половина – больше и равны ξ . Чтобы выполнить медианную фильтрацию для элемента изображения, необходимо сначала упорядочить по возрастанию значения пикселей внутри окрестности, затем найти значение медианы и наконец присвоить полученное значение обрабатываемому элементу. Так для окрестности размером 3×3 медианой будет пятое значение по величине, для окрестности 5×5 – тринадцатое значение и так далее.

Аналитически действие медианного фильтра определяется как

$$\hat{f}(x, y) = \underset{s, t \in S_{xy}}{med} \{g(s, t)\}$$

Причем при вычислении медианы значение в самой точке (т. е. в центре окружности) также учитывается.

Если несколько элементов в окрестности имеют одинаковые значения, то значения будут сгруппированы. Например, пусть в окрестности 3×3 имеют место следующие значения (10, 20, 20, 20, 15, 20, 20, 25, 100). После упорядочения они будут расположены следующим образом (10, 15, 20, 20, 20, 20, 20, 25, 100), а, следовательно, медианой будет значение 20.

Хотя медианный фильтр значительно более распространен в обработке изображений, чем остальные виды фильтров, тем не менее, он не является единственным. Медиана представляет собой 50-ый процентиль упорядоченного набора чисел.

Широкая популярность медианных фильтров обусловлена тем, что они прекрасно приспособлены для подавления импульсных шумов и при этом приводят к меньшему размыванию по сравнению с линейными сглаживающими фильтрами.

Использование 100-го процентиля приводит к фильтру, основанному на выборе максимального значения (фильтр максимума), который задается выражением

$$\hat{f}(x, y) = \max_{(s, t) \in S_{xy}} \{g(s, t)\}$$

Такой фильтр полезен при обнаружении наиболее ярких точек на изображении.

Использование 0-го процентиля приводит к фильтру, основанному на выборе минимального значения (фильтр минимума)

$$\hat{f}(x, y) = \min_{(s, t) \in S_{xy}} \{g(s, t)\}$$

Такой фильтр полезен при обнаружении наиболее темных точек на изображении.

Применение фильтра срединной точки заключается в вычислении среднего между максимальным и минимальным значениями в соответствующей окрестности

$$\hat{f}(x, y) = \frac{1}{2} [\max_{(s, t) \in S_{xy}} \{g(s, t)\} + \min_{(s, t) \in S_{xy}} \{g(s, t)\}]$$

Этот фильтр объединяет методы порядковых статистик и усреднения. Этот фильтр лучше всего работает при наличии таких случайно распределенных шумов, как гауссов или равномерный.

Предположим, что мы удалили $d/2$ наименьших и $d/2$ наибольших значений яркости из множества всех значений функции $g(s, t)$ в окрестности S_{xy} . Пусть $g_r(s, t)$ представляет оставшиеся элементы изображения, количество которых равно $(mn-d)$.

Фильтр, действие которого заключается в усреднении оставшихся значений, называется фильтром усеченного среднего

$$\hat{f}(x, y) = \frac{1}{mn - d} \sum_{(s, t) \in S_{xy}} q_r(s, t),$$

где $d = \overline{0, mn - 1}$

При $d=0$ фильтр усеченного среднего сводится к среднеарифметическому фильтру. В случае $d/2=(mn-1)/2$ фильтр трансформируется в медианный.

20. СЖАТИЕ ДАННЫХ

Важной областью применения ортогональных преобразований является сжатие данных. Если дискретный сигнал содержит N отсчётов, то его можно рассматривать как точку N -мерного пространства. Тогда каждый отсчёт является координатой N -мерного вектора данных. Для более эффективного представления можно осуществить ортогональное преобразование X , что приводит к:

$$Y = T * X,$$

где Y и T – вектор коэффициентов преобразования и матрица преобразования соответственно. Целью сжатия данных является выбор подмножества M координат вектора Y , где M существенно меньше N . Остальные $(N - M)$ координат можно отбросить, не вызывая существенной ошибки при восстановлении сигнала по M координатам вектора Y . Следовательно, сравнивать ортогональные преобразования следует в соответствии с некоторым критерием ошибки. Одним из часто используемых является критерий среднеквадратической ошибки.

Поиск оптимального преобразования

Найдём ортогональное преобразование, которое с одной стороны будет обеспечивать представление сигнала, а с другой стороны будет оптимальным в смысле среднеквадратичного критерия.

Пусть T - ортогональное преобразование, заданное в виде:

$$T = [\phi_1 \ \phi_2 \ \dots \ \phi_N],$$

где ϕ – векторы размерностью $N \times 1$. Для удобства базисные векторы $\{\phi_m\} \ m=\overline{1, N}$, будем считать вещественнозначными и ортонормированными.

Для каждого вектора X , принадлежащего данному классу векторов исходных данных получим:

$$Y = T * X, \quad (20.1)$$

$$\text{где } X = [x_1 \ x_2 \ \dots \ x_N], \ Y = [y_1 \ y_2 \ \dots \ y_N]$$

Из соотношения (20.1) :

$$X = T^1 Y = [y_1 \ y_2 \ \dots \ y_N] Y, \text{ что можно записать как}$$

$$X = y_1 \phi_1 + y_2 \phi_2 + \dots + y_N \phi_N = \sum_{i=1}^N y_i \phi_i$$

Желательно сохранить подмножество $\{y_1, y_2, \dots, y_M\}$ координат Y и при этом получить оценку X . Это может быть осуществлено заменой остальных $(N - M)$ координат Y заранее выбранными константами, что приводит к

$$\tilde{X}(M) = \sum_{i=1}^M y_i \phi_i + \sum_{i=M+1}^N b_i \phi_i,$$

где $\tilde{X}(M)$ обозначает оценку X . Ошибку, возникающую при отбрасывании $N - M$ координат можно представить в виде

$$\Delta X = X - \tilde{X}(M) \text{ т.е.}$$

$$\Delta X = X - \sum_{i=1}^M y_i \phi_i - \sum_{i=M+1}^N b_i \phi_i = \sum_{i=M+1}^N y_i \phi_i - \sum_{i=M+1}^N b_i \phi_i = \sum_{i=M+1}^N (y_i - b_i) \phi_i.$$

Таким образом, среднеквадратичная ошибка $\varepsilon(M)$ определяется в виде

$$\varepsilon(M) = \varepsilon\{\|\Delta x\|^2\} = \varepsilon\{(\Delta x)^T \Delta x\}$$

С учётом подстановки получим

$$\varepsilon(M) = \varepsilon\left\{\sum_{i=M+1}^N \sum_{j=M+1}^N (y_i - b_i)^T (y_j - b_j) \phi_i^T \phi_j\right\},$$

что в упрощённом виде можно записать как

$$\varepsilon(M) = \sum_{i=M+1}^N E\{(y_i - b_i)^2\}$$

Из последнего соотношения следует, что для каждого выбора ϕ_i и b_i получаем определённое значение $\varepsilon(M)$. Требуется определить такую комбинацию этих величин, чтобы минимизировать $\varepsilon(M)$. Процедура выбора оптимальных ϕ_i и b_i разбивается соответственно на 2 этапа.

Первый этап

Оптимальное значение b_i определяется из выражения

$$\frac{\partial}{\partial b_i} E\{(y_i - b_i)^2\} = -2[E\{y_i\} - b_i] = 0$$

Тогда $E\{y_i\} = b_i$.

В свою очередь $y_i = \phi_i \cdot x$.

Таким образом

$$b_i = \phi_i E\{x\} = \phi_i \bar{x} \quad \text{где } E\{x\} = \bar{x}$$

Так как разность $(y_i - b_i)$ является скалярной, то $\varepsilon(M)$ можно записать как

$$\varepsilon(M) = \sum_{i=M+1}^N E\{(y_i - b_i)(y_i - b)^T\} \quad (20.2)$$

Подстановка

$$y_i = \phi_i x \quad b_i = \phi_i \bar{x} \quad \text{в (20.1) приводит к}$$

$$\varepsilon(M) = \sum_{i=M+1}^N \phi_i E\{(x - \bar{x})(x - \bar{x})^T\} \phi_i^T$$

Так как $\sum_x = E\{(x - \bar{x})(x - \bar{x})^T\}$ является ковариационной матрицей x , то получим

$$\varepsilon(M) = \sum_{i=M+1}^N \phi_i \sum_x \phi_i^T \quad (20.3)$$

Второй этап

Для нахождения оптимального ϕ_i следует не только минимизировать $\varepsilon(M)$ по отношению к ϕ_i , но и удовлетворить ограничению

$$\phi_i^T \phi_i = 1$$

При этом, используя метод множителей Лагранжа, получим

$$\sum_x \phi_i^T = \phi_i^T \lambda_i \quad (20.4)$$

Подставляя (20.4) в (20.3), получим минимальное значение среднеквадратической ошибки в виде

$$E_{\min}(M) = \sum_{i=M+1}^N \lambda_i$$

Таким образом, разложение

$x = \sum_{i=1}^N y_i \phi_i$ представляет собой разложение по собственным векторам ковариационной матрицы. Это разложение называется разложением Карунена-Лоэва. В литературе по статистике задачи по минимизации $\varepsilon(M)$ называются факторным анализом или анализом главных компонент.

Таким образом можно сделать 2 вывода:

1. Преобразование Карунена-Лоэва является оптимальным для представления сигналов по отношению к критерию среднеквадратичной ошибки.
2. Поскольку $Y = TX$, то ковариационная функция в области изображения \sum_Y определяется как

$$\sum_Y = T \sum_X T^{-1} = T \sum_X T^T$$

Так как T состоит из собственных векторов \sum_X , то

$$\sum_Y = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_N)$$

где λ_i - собственные значения \sum_X . Так как \sum_Y - диагональная матрица, то координаты вектора преобразованных данных y_i в выражении $y_i = \phi_i^T x$ некоррелированы.

Из выражения $E_{\min}(M) = \sum_{i=M+1}^N \lambda_i$ следует, что эффективность коэффициента преобразования y_i для представления вектора данных X определяется соответствующим ему собственным значением. Если какой-либо коэффициент y_k не учитывается, то среднеквадратичная ошибка увеличивается на соответствующее собственное значение λ_k . Таким образом,

необходимо выбрать множество y_i соответствующее M наибольшим собственным значениям, а остальные y_i отбросить, так как их можно заменить константами $b_i \quad i = \overline{M+1, N}$.

Так как $b_i = \phi_i \bar{X}$, то остальные y_i можно приравнять к нулю, если исходные данные центрированы, то есть $\bar{X} = 0$.

Так как собственные значения являются элементами \sum_Y , стоящими на главной диагонали, то они соответствуют дисперсиям коэффициентов преобразования y_i . Для всех остальных преобразований \sum_Y содержит ненулевые внедиагональные элементы. Поэтому естественным критерием при выборе множества оставляемых коэффициентов преобразования является сохранение M коэффициентов с наибольшими дисперсиями, а остальные $(N - M)$ коэффициентов можно отбросить. Приведённый выше критерий выбора коэффициентов преобразования носит название дисперсионного критерия.

Выше предполагалось, что X принадлежит одному классу сигналов, но все рассуждения легко распространяются на несколько классов. Отличие заключается в том, что базисные векторы для соответствующего преобразования Карунена-Лоэва являются собственными векторами общей матрицы \sum_X^* , а не \sum_X . Матрица \sum_X^* определяется как:

$$\sum_X^* = p_1 \sum_{X1} + p_2 \sum_{X2} + \dots + p_K \sum_{XK},$$

Где K – число классов сигналов,

\sum_{Xi} – ковариационная матрица i -ого класса с априорной вероятностью

p_i .

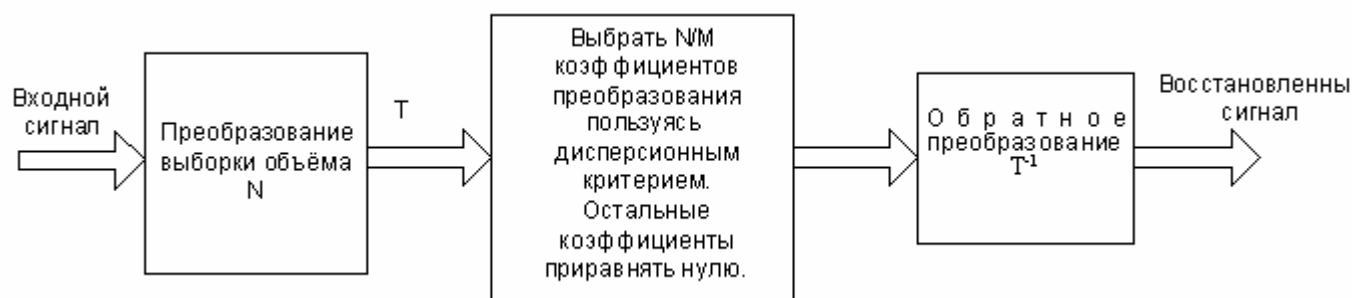


Рис. 20.1 Модель сжатия данных в соотношении M:1.

При цифровой обработке изображений каждый элемент обычно кодируется в виде слова, содержащего 6 бит. Таким образом, элемент изображения можно представить десятичным числом от 1 до 64 или от 0 до 63 (то есть 64 уровнями). Закодированные, таким образом, данные обычно обрабатываются блоками размером $N \times N$, как показано на рис.

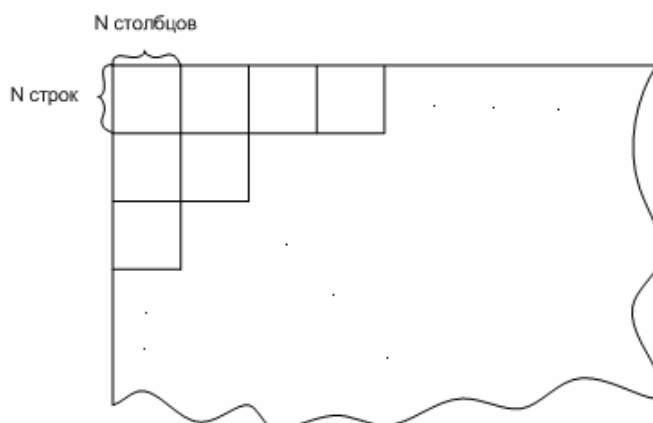


Рис. 20.2

Элемент изображения в i -ой строке и j -ом столбце можно представить в виде случайной величины $f(i, j)$, а матрицу ($N \times N$) случайных величин как $[f(i, j)]$. Тогда двумерное преобразование $[f(i, j)]$ и обратное преобразование можно записать как:

$$[F(u, v)] = T[f(i, j)]T^T$$

$$[f(i, j)] = T' [F(u, v)] T,$$

где $[F(u, v)]$ - матрица коэффициентов преобразования, T – матрица преобразования.

Обозначим через $\sigma^2(i, j)$ и $\sigma^2(u, v)$ дисперсии $f(i, j)$ и $F(u, v)$ соответственно. Если функция распределения дисперсии $\sigma^2(u, v)$ неизвестна для изображения или класса изображений, подвергаемых преобразованию, её получают обычно моделированием. Один из подходов заключается в том что изображение описывается статистически как Марковский процесс первого порядка, а строки и столбцы обрабатываются независимо. Если дисперсия каждого столбца и строки равна σ^2 , то ковариационные матрицы для строк и столбцов можно записать как

$$\sum_k = \sigma^2 R_k$$

$$\text{где } R_k = \begin{vmatrix} 1 & \rho_k & \rho_k^2 & \dots & \rho_k^{N-1} \\ \rho_k & 1 & \rho_k & \rho_k^2 & \dots \\ \rho_k^2 & \rho_k & 1 & \rho_k & \dots \\ \dots & \dots & \dots & \dots & \dots \\ \rho_k^{N-1} & \dots & \dots & \dots & 1 \end{vmatrix}$$

а ρ_1 и ρ_2 - коэффициенты корреляции для строк и столбцов случайных величин соответственно

Тогда ковариационные матрицы в области преобразований, соответствующие \sum_k запишутся как:

$$\tilde{\sum}_k = \sigma^2 [T R_k T'] \quad k=1,2$$

В свою очередь, функция распределения дисперсии $\sigma^2(u, v)$ определяется как

$$\tilde{\sigma}^2(u, v) = \tilde{\Sigma}_1(u, u) \tilde{\Sigma}_2(v, v)$$

Рассмотрим простейший способ сжатия данных, называемый зональным кодированием, который реализуется в 3 этапа:

1. Получают двумерное преобразование данного изображения, путём обработки блоками ($N \times N$).
2. Из N^2 коэффициентов преобразования сохраняют N^2/m коэффициентов, обладающих наибольшими двумерными дисперсиями $\sigma^2(u, v)$. Все остальные коэффициенты приравниваются нулю.
3. Каждый из N^2/m сохранённых коэффициентов кодируется с помощью k бит. Затем восстанавливается соответствующий ($N \times N$) блок с помощью обратного преобразования.

Так как кодируется только N^2/m коэффициентов преобразования, а не N^2 исходных элементов изображения для каждого ($N \times N$) блока, то среднее число бит на элемент восстановленного изображения равняется R/m .

Следует отметить, что при заданном m выбор типа преобразования определяется объёмом вычислений, объёмом требуемой аппаратуры и требованиями, предъявляемыми к качеству изображения.

21. ВЫБОР ПРИЗНАКОВ И РАСПОЗНАВАНИЕ ОБРАЗОВ.

ПРИНЦИП ОБУЧЕНИЯ

Задача распознавания образов заключается в классификации некоторой группы объектов на основе определённых требований.

Требования эти могут быть различными , так как в различных ситуациях возникает необходимость в определённом типе классификатора. К примеру при распознавании английских букв образуется 26 классов образов. Однако, чтобы отличить при распознавании буквы от китайских иероглифов нужны лишь два класса образов.

Проблема распознавания получила широкое распространение при решении различных задач в различных областях народного хозяйства: распознавание рукописных букв или слов, медицинская диагностика, диагностика неисправностей в технических системах, обнаружение цели , прогноз погоды , классификация сейсмических волн.

Простейший подход к распознаванию образов заключается в сопоставлении с эталоном. В этом случае некоторое множество образов по одному из каждого класса образов хранится в памяти машины. Входной, подлежащий распознаванию образ (неизвестного класса) сравнивается с эталоном каждого класса . Классификация основывается на заранее выбранном критерии соответствия или критерии подобия. Иными словами, если входной образ лучше соответствует эталону i -го класса образов , чем любому другому эталону , то входной образ классифицируется как принадлежащий i -му классу образов.

Более совершенный подход заключается в том , что вместо сравнения входного образа с эталонами, классификация основывается на некотором множестве отобранных замеров , называемых информативными признаками, ,которые предполагаются инвариантными или малочувствительными по отношению к изменениям и искажениям .

В этом случае распознавание образов можно рассматривать состоящим из двух задач.

Первая задача заключается в выборе подмножества признаков и их упорядочиванию в заданном множестве измерений. Вторая задача заключается в классификации (принятии решения о принадлежности входного образа тому или иному классу), которая основывается на измерениях отобранных признаков.

Упрощенная блок-схема системы распознавания приведена на рис.1.

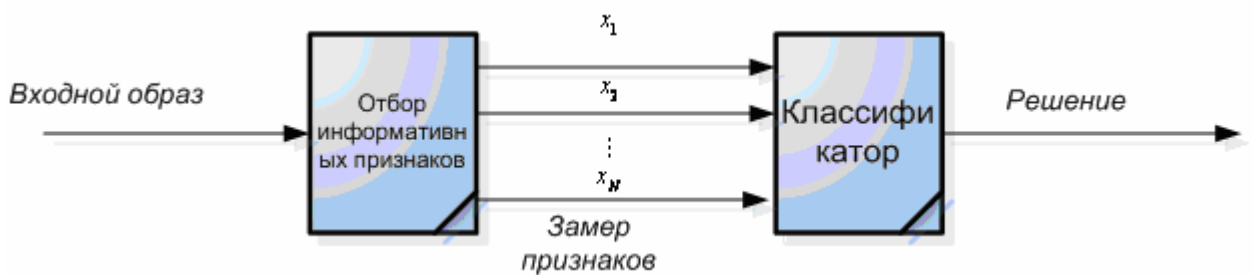


Рис. 21.1

Таким образом, подход, заключающийся в сопоставлении с эталоном, можно рассматривать как частный случай второго подхода- подхода измерения признаков, при этом эталоны хранятся в виде измеренных признаков, а в классификаторе используется специальный критерий сопоставления.

Математически задача классификации может быть сформулирована с помощью разделяющей функции.

Пусть $\omega_1, \omega_2, \dots, \omega_m$ обозначают m возможных классов образов, подлежащих распознаванию и пусть $X' = [x_1, x_2, \dots, x_n]$ вектор замеров признаков, где x_k предоставляет k -ый замер.

Тогда разделяющая функция $D_j(x)$, относящаяся к классу образов $\omega_j, j = \overline{1, m}$, такова, что если входной образ представленный вектором признаков X , принадлежит классу ω_i , то величина $D_i(x)$ должна быть

наибольшей. Пусть $X \sim \omega_i$ обозначает, что вектор признаков X входного образа принадлежит классу ω_i . Тогда можно записать, что для всех $X \sim \omega_i$

$$D_i(x) > D_j(x); i, j = \overline{1, m} \quad i \neq j \quad (21.1)$$

Таким образом, в пространстве признаков Ω_x граница разбиений, называемая решающей границей между областями, относящимися соответственно к классам ω_i, ω_j , выражается уравнением

$$D_i(x) - D_j(x) = 0$$

Общая схема классификатора, использующего критерий (21.1) и типичный двумерный пример приведены на рис. 21.2 и рис. 21.3.

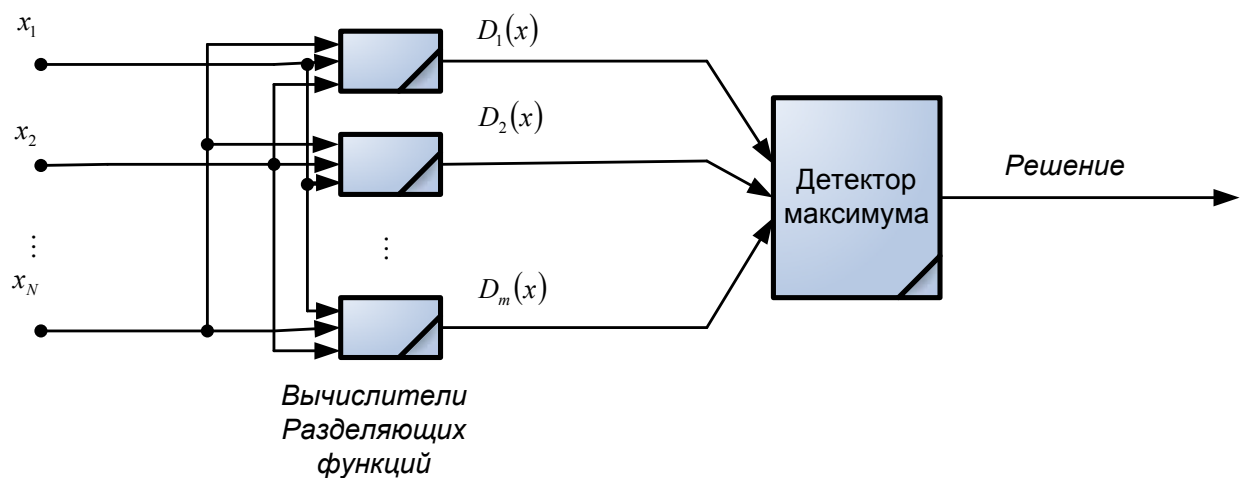


Рис. 21.2

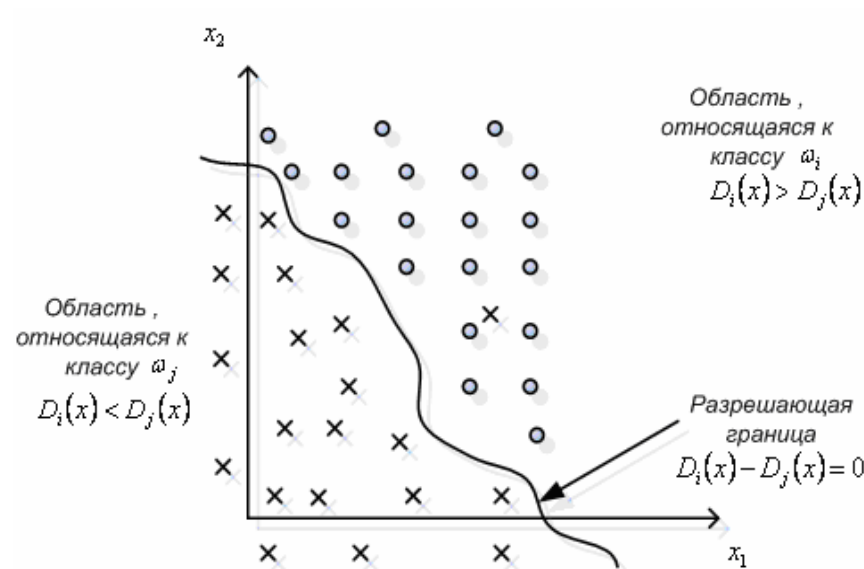


Рис. 21.3

22. ВИДЫ РАЗДЕЛЯЮЩИХ ФУНКЦИЙ. КЛАССИФИКАТОР ПО МИНИМАЛЬНОМУ РАССТОЯНИЮ

А. Линейные разделяющие функции.

В этом случае в качестве $D_i(x)$ используется линейная комбинация измеренных признаков x_1, x_2, \dots, x_n

$$D_i(x) = \sum_{k=1}^n w_{ki} x_k + w_{i,n+1}$$

Решающая граница между областями w_i и w_j в пространстве признаков Ωx имеет вид

$$D_i(x) - D_j(x) = \sum_{k=1}^n w_k x_k + w_{n+1} = 0 \quad (22.1)$$

где $w_k = w_{ki} - w_{kj}$,
 $w_{n+1} = w_{i,n+1} - w_{j,n+1}$

Уравнение (22.1) представляет собой уравнение гиперплоскости в пространстве признаков Ωx .

Общая схема вычислителя линейной разделяющей функции представлена на рис. 22.1.

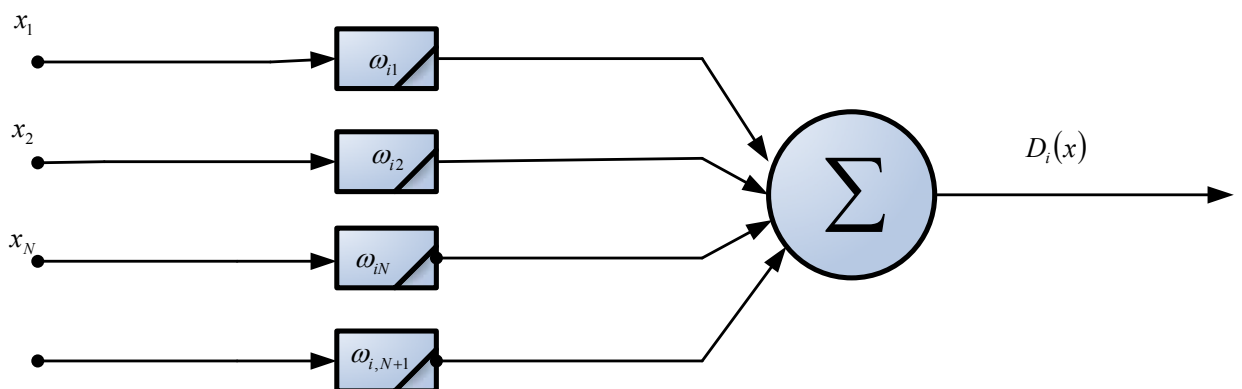


Рис. 22.1

Если $m = 2$, то согласно (22.1) $i, j = 1, 2$ и в качестве классификатора, использующего линейную разделяющую функцию, можно применить пороговый логический элемент.

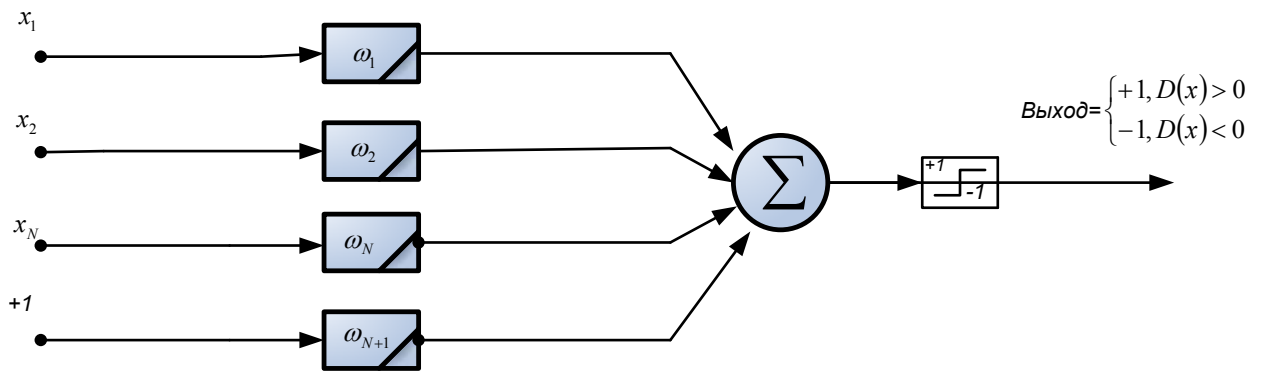


Рис. 22.2

Из этого рисунка получим, что если выходной сигнал $= +1$, т.е. при $D(x) > 0$, то $X \sim \omega_1$; если выходной сигнал $= -1$, т.е. $D(x) < 0$, то $X \sim \omega_2$.

Когда число классов образов больше двух, т.е. $m > 2$ можно применить параллельное соединение нескольких пороговых логических элементов. При этом комбинации выходных сигналов M пороговых логических элементов достаточны для различения m классов при $2^M \geq m$.

В. Классификатор по минимальному расстоянию.

Важный класс составляют линейные классификаторы, в которых в качестве критерия классификации используется расстояние между входным образом и множеством опорных векторов или эталонных точек в пространстве признаков. Предположим, что задано m опорных векторов R_1, R_2, \dots, R_m , где R_j соответствует классу образов ω_j . При классификации по минимальному расстоянию относительно R_1, R_2, \dots, R_m входной сигнал X предполагается принадлежащим ω_i , т.е.

$X \sim \omega_i$, если $|X - R_i|$ минимально, где $|X - R_i|$ есть расстояние между X и R_i .

Расстояние можно определить ,например, следующим образом

$$|X - R_i| = \left[(X - R_i)^T (X - R_i) \right]^{1/2}, \quad (22.2)$$

где индекс T определяет операцию транспонирования вектора.

Из последнего соотношения следует, что

$$|X - R_i|^2 = X^T X - X^T R_i - X R_i^T + R_i^T R_i$$

Так как $X^T X$ не зависит от i , то соответствующая разделяющая функция для классификатора по минимальному расстоянию имеет вид

$$D_i(x) = X^T R_i + X R_i^T - R_i^T R_i, \quad i = \overline{1, m}$$

Как видно из этого соотношения, классификатор по минимальному расстоянию является линейной функцией. В свою очередь, свойства классификатора по минимальному расстоянию конечно зависят от того, как выбраны опорные векторы.

С. Кусочно-линейная разделяющая функция.

Изложенная выше идея может быть распространена на классификацию по минимальным расстояниям до m множеств опорных векторов. Пусть R_1, R_2, \dots, R_m обозначают m множеств опорных векторов, относящихся соответственно к классам $\omega_1, \omega_2, \dots, \omega_m$ и пусть опорные векторы в R_j обозначены через $R_j^{(k)}$, т.е.

$$R_j^{(k)} \in R_j, \quad k = 1, \dots, U_j$$

где U_j -число опорных векторов множества R_j . Определим расстояние между вектором входных признаков и R_j следующим образом.

$$d(X, R_j) = \min_{k=1, \dots, U_j} |X - R_i^{(k)}|,$$

То есть расстояние между X и R_j равно наименьшему из расстояний между X и каждым вектором в R_j . Такой классификатор будет относить входной сигнал к классу образов, которому соответствует ближайшее множество векторов.

Если расстояние между X и $R_j^{(k)}$ определить согласно (22.1) разделяющая функция в данном случае имеет вид

$$D_i(x) = \max_{k=1, \dots, U_j} \{X^T R_i^{(k)} + X(R_i^{(k)})^T - (R_i^{(k)})^T R_i^{(k)}\} \quad i = \overline{1, m}$$

$$\text{Пусть } D_i^{(k)}(x) = X^T R_i^{(k)} + X(R_i^{(k)})^T - (R_i^{(k)})^T R_i^{(k)},$$

Тогда

$$D_i(x) = \max_{k=1, \dots, U_j} \{D_i^{(k)}(x)\}$$

Следует отметить, что $D_i^{(k)}(x)$ является линейной комбинацией признаков. Поэтому указанный классификатор часто называют кусочно-линейным классификатором.

Д. Полиномиальная разделяющая функция.

Полиномиальная функция r -ой степени может быть представлена в виде

$$D_i(x) = w_{i1}f_1(x) + w_{i2}f_2(x) + \dots + w_{iL}f_L(x) + w_{i,L+1},$$

где $f_i(x)$ является формой

$$x_{k_1}^{n_1} x_{k_2}^{n_2} \dots x_{k_r}^{n_r} \text{ при } \begin{cases} k_1, k_2, \dots, k_r = 1, \dots, N \\ n_1, n_2, \dots, n_r = 0, 1 \end{cases}$$

Решающая граница между двумя классами также имеет форму полинома r -ой степени. В частности, если $r=2$, решающая функция называется квадратичной.

В этом случае

$$f_j(x) = x_{k_1}^{n_1} x_{k_2}^{n_2} \text{ при } \begin{cases} k_1, k_2 = 1, \dots, N \\ n_1, n_2 = 0, 1 \end{cases}$$

Разделяющая функция будет иметь вид

$$D_i(x) = \sum_{k=1}^N w_{kk} x_k^2 + \sum_{j=1}^{N-1} \sum_{k=j+1}^N w_{kj} x_j x_k + \sum_{j=1}^N w_j x_j + w_{i,L+1},$$

где $L = 1/2N(N+3)$

В общем случае границей для квадратичных разделяющих функций является гиперboloид. В частных случаях это будут гиперсфера, гиперэллипсоид и гиперэллипсоидальный цилиндр. Общая схема вычислителя квадратичного разделения показана на рис. 22.3

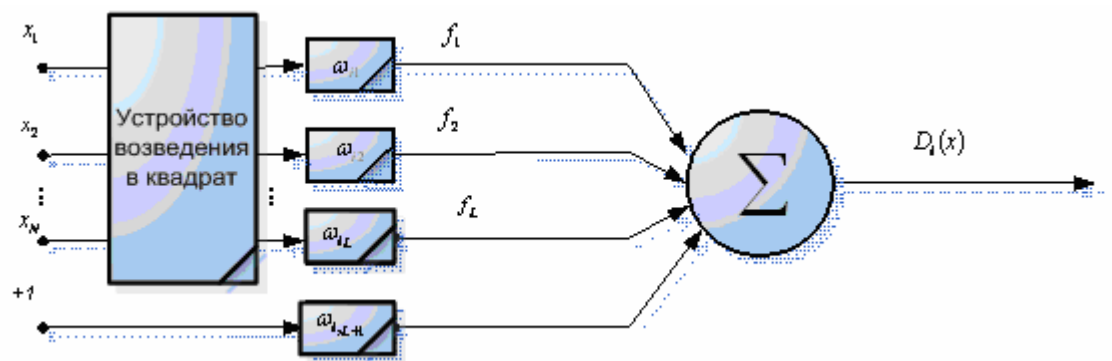


Рис. 22.3

23. СТРУКТУРА СИСТЕМЫ РАСПОЗНАВАНИЯ ОБРАЗОВ С ПРИМЕНЕНИЕМ ОРТОГОНАЛЬНЫХ ПРЕОБРАЗОВАНИЙ.

ПРИНЦИП ОБУЧЕНИЯ В ДВУХКЛАССОВОЙ СИСТЕМЕ РАСПОЗНАВАНИЯ

Систему распознавания образов с применением ортогональных преобразований можно представить в виде

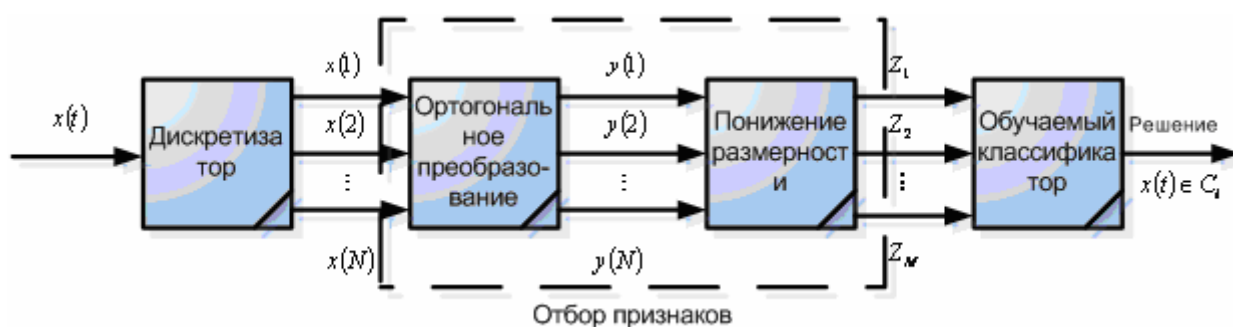


Рис. 23.1

Через $x(t)$ обозначается сигнал, принадлежащий одному из K классов C_1, C_2, \dots, C_K .

На первом этапе выбора осуществляется ортогональное преобразование. Вторым этапом выбора признака является понижение размерности, после чего получаем подмножество M признаков z_1, z_2, \dots, z_M из $\{Y(M)\} = \{Y(1), Y(2), \dots, Y(N)\}$ причём $M \ll N$.

Понижать размерность следует таким образом, чтобы сопутствующее этому увеличение ошибки классификации было относительно невелико. Классификатор, изображенный на рис. 23.1 является решающим устройством, которое обучается с целью классификации входного сигнала $x(t)$, принадлежащего к одному из K классов.

Принцип обучения наилучшим образом можно объяснить с помощью простого примера. Предположим, что мы желаем обучить классификатор автоматически классифицировать образ Z , принадлежащий либо классу C_1 ,

либо классу C_2 . Предположим также что обучающее множество (т.е. множество истинная классификация которого известна) состоит из следующего множества двумерных образов Z_{ij} , где Z_{ij} обозначает j -ый образ, принадлежащий $C_i, i=1,2$

$$C_1: Z_{11} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}; Z_{12} = \begin{bmatrix} 6 \\ 5 \end{bmatrix}; Z_{13} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}; Z_{14} = \begin{bmatrix} 6 \\ 7 \end{bmatrix}; Z_{15} = \begin{bmatrix} 7 \\ 5 \end{bmatrix}$$

$$C_2: Z_{21} = \begin{bmatrix} 0 \\ 3 \end{bmatrix}; Z_{22} = \begin{bmatrix} -1 \\ 3 \end{bmatrix}; Z_{23} = \begin{bmatrix} -2 \\ 3 \end{bmatrix}; Z_{24} = \begin{bmatrix} -3 \\ 3 \end{bmatrix}; Z_{25} = \begin{bmatrix} -4 \\ 3 \end{bmatrix}$$

Образы $\{Z_{ij}\}$, принадлежащие классам C_1 и C_2 , располагаются в двумерном пространстве признаков, как показано на рис. 23.2

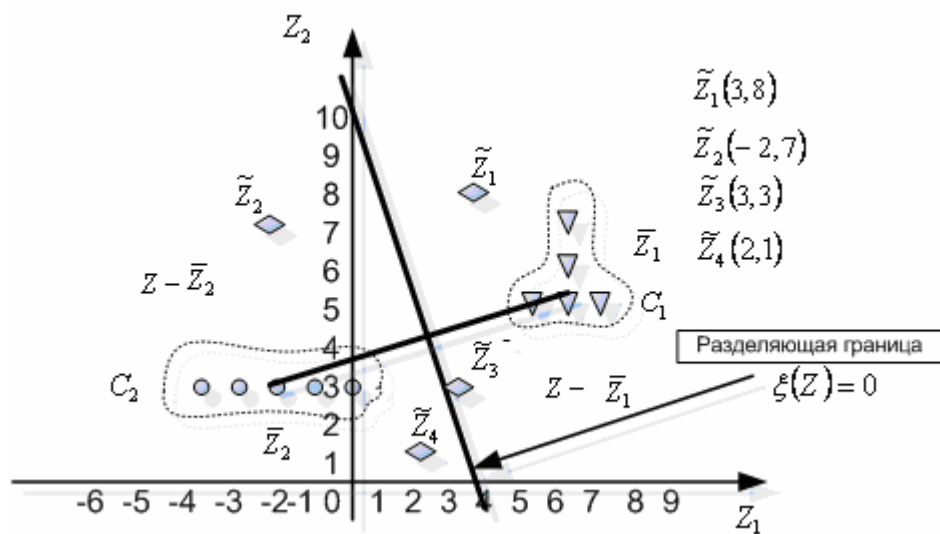


Рис. 23.2

Пусть \bar{Z}_1 и \bar{Z}_2 - средние векторы образов, связанные с C_1 и C_2 соответственно. Тогда

$$\bar{Z}_i = \frac{1}{5} \sum_{j=1}^5 Z_{ij}, i=1,2$$

$$\bar{Z}_1 = \begin{bmatrix} 6 \\ 5,6 \end{bmatrix}, \bar{Z}_2 = \begin{bmatrix} -2 \\ 3 \end{bmatrix} \quad (23.1)$$

Из рис. 23.2 видно, что наиболее целесообразной решающей границей (т.е. линией на плоскости), разделяющей классы C_1 и C_2 , является срединный перпендикуляр к прямой, соединяющей \bar{Z}_1 и \bar{Z}_2 . Следующим шагом является описание решающей границы с помощью уравнения.

Рассмотрим любую точку Z , принадлежащую решающей границе, как показано на рис. Так как решающая граница является срединным перпендикуляром к прямой, соединяющей \bar{Z}_1 и \bar{Z}_2 , то

$$\|Z - \bar{Z}_1\|^2 = \|Z - \bar{Z}_2\|^2 \text{ или в упрощенном виде}$$

$$(\bar{Z}_1 - \bar{Z}_2)' Z = \frac{1}{2} \{ \|\bar{Z}_1\|^2 - \|\bar{Z}_2\|^2 \} \quad (23.2)$$

Подставляя (23.1) в (23.2), получим уравнение для решающей границы в виде

$$8Z_1 + 2,6Z_2 = 27,18 \quad (23.3)$$

Величина $0,5(\|\bar{Z}_1\|^2 - \|\bar{Z}_2\|^2) = 27,18$ называется порогом классификатора.

Уравнение (2) даёт всю информацию, необходимую для создания классификатора. Основной характеристикой, представляющей классификатор, является дискриминантная функция $\xi(Z)$, которая определяется как

$$\xi(Z) = 8Z_1 + 2,6Z_2 - 27,18.$$

Возникает вопрос, что в действительности делает $\xi(Z)$? Чтобы ответить на этот вопрос, рассмотрим несколько испытательных образов (т.е. образов точная классификация которых неизвестна), которые обозначим как \tilde{Z}_k ($k = 1, 2, 3, 4$).

Вычислим теперь значение $\xi(Z)$ для каждого из испытательных образов

$$\tilde{Z}_1 : \xi(Z) = 8(3) + 2,6(8) - 27,18 > 0$$

$$\tilde{Z}_2 : \xi(Z) = 8(-2) + 2,6(7) - 27,18 < 0$$

$$\tilde{Z}_3 : \xi(Z) = 8(3) + 2,6(3) - 27,18 > 0$$

$$\tilde{Z}_4 : \xi(Z) = 8(2) + 2,6(1) - 27,18 < 0$$

Рассмотрение последнего выражения приводит к выводу, что когда образ \tilde{Z} лежит справа от решающей границы, то $\xi(Z) > 0$ и наоборот, когда образ \tilde{Z} лежит слева от решающей границы, то $\xi(Z) < 0$. Все точки лежащие справа от границы ближе к \tilde{Z}_1 , а точки, лежащие слева от границы ближе к \tilde{Z}_2 . Таким образом, благодаря дискриминантной функции $\xi(Z)$ получим следующее простое решающее правило:

если $\xi(Z) > 0$, то $Z \in C_1$

если $\xi(Z) < 0$, то $Z \in C_2$

Описанный выше классификатор, называемый элементом линейной пороговой логики, реализуется как показано на рис. 23.3.

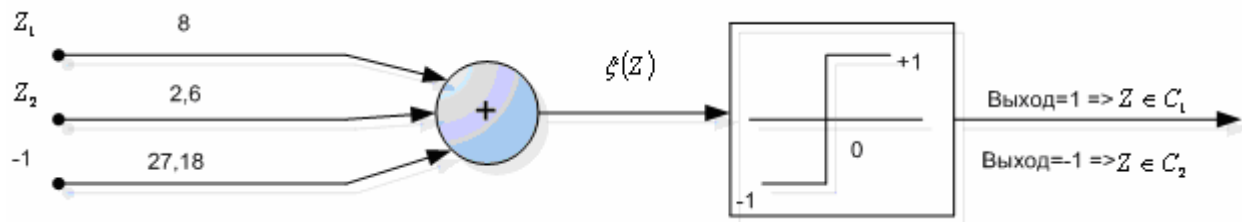


Рис. 23.3

Получив дискриминантную функцию $\xi(Z)$, говорят, что классификатор обучен, т.е. способен классифицировать образы с помощью соответствующего решающего правила. Элемент пороговой логики является классификатором, работающим по критерию минимума расстояния, так как решающее правило может быть сформулировано также следующим образом:

если Z ближе к \bar{Z}_1 , то $Z \in C_1$

и если Z ближе к $\overline{Z_2}$, то $Z \in C_2$

Рассмотрим элемент линейной пороговой логики, на который воздействуют $(d \times 1)$ образы, где $d > 2$. В этом случае $Z' = [Z_1 Z_2 \dots Z_d]$. Соответствующая дискриминантная функция записывается как

$$\xi(Z) = w_1 Z_1 + w_2 Z_2 + \dots + w_d Z_d - \theta$$

где w_i - веса или параметры классификатора, а θ - порог. Веса w_i и θ получаются из обучающего множества. Границей в этом случае является гиперплоскость, определяемая как

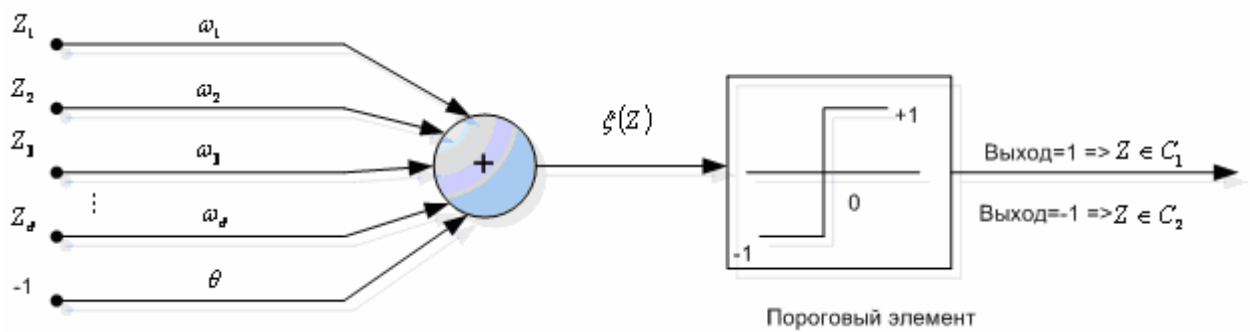


Рис. 23.4

24. ЗАДАЧА ТРЕХ КЛАССОВ ОБРАЗОВ ПО КРИТЕРИЮ МИНИМАЛЬНОГО РАССТОЯНИЯ

Ранее рассматривались аспекты так называемой задачи двух классов . Распространим эти понятия на общий случай трёх классов: C_1, C_2, C_3 . Пусть обучающее множество записывается как

$$\begin{aligned} C_1 : Z_{11} &= \begin{bmatrix} 0 \\ 3 \end{bmatrix}; Z_{12} = \begin{bmatrix} -1 \\ 3 \end{bmatrix}; Z_{13} = \begin{bmatrix} -2 \\ 3 \end{bmatrix}; Z_{14} = \begin{bmatrix} -3 \\ 3 \end{bmatrix}; Z_{15} = \begin{bmatrix} -4 \\ 3 \end{bmatrix} \\ C_2 : Z_{21} &= \begin{bmatrix} 5 \\ 5 \end{bmatrix}; Z_{22} = \begin{bmatrix} 6 \\ 5 \end{bmatrix}; Z_{23} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}; Z_{24} = \begin{bmatrix} 6 \\ 7 \end{bmatrix}; Z_{25} = \begin{bmatrix} 7 \\ 5 \end{bmatrix} \\ C_3 : Z_{31} &= \begin{bmatrix} 6 \\ -1 \end{bmatrix}; Z_{32} = \begin{bmatrix} 7 \\ 0 \end{bmatrix}; Z_{33} = \begin{bmatrix} 8 \\ 1 \end{bmatrix}; Z_{34} = \begin{bmatrix} 9 \\ 1 \end{bmatrix}; Z_{35} = \begin{bmatrix} 10 \\ 1 \end{bmatrix} \end{aligned}$$

В этом случае средние векторы образов равны

$$\bar{Z}_1 = \begin{bmatrix} -2 \\ 3 \end{bmatrix}; \bar{Z}_2 = \begin{bmatrix} 6 \\ 5,6 \end{bmatrix}; \bar{Z}_3 = \begin{bmatrix} 8 \\ 0,4 \end{bmatrix}$$

Классификатор, работающий по минимуму расстояния имеет следующее решающее правило : данный образ Z принадлежит C_i , если Z ближе всего к \bar{Z}_i , $i=1,2,3$.

Пусть D_i обозначает расстояние образа Z от \bar{Z}_i . Тогда получим

$$D_i^2 = \|Z - \bar{Z}_i\|^2 = (Z - \bar{Z}_i)^T (Z - \bar{Z}_i)$$

Упрощение D_i^2 приводит к

$$D_i^2 = Z^T Z - Z^T \bar{Z}_i - \bar{Z}_i^T Z + \bar{Z}_i^T \bar{Z}_i = \|Z\|^2 - 2 \left(\bar{Z}_i^T Z - \frac{1}{2} \|\bar{Z}_i\|^2 \right) \quad (24.1)$$

Очевидно, что D_i^2 минимально, когда величина $\left\{ \overline{Z}_i^T Z - \frac{1}{2} \|\overline{Z}_i\|^2 \right\}$ максимальна. Поэтому вместо вычисления D_i^2 по формуле (24.1) проще потребовать, чтобы в классификаторе вычислялось значение $\left\{ \overline{Z}_i^T Z - \frac{1}{2} \|\overline{Z}_i\|^2 \right\}$. Классификатор в этом случае описывается дискриминантными функциями.

$$\xi_i(Z) = \left\{ \overline{Z}_i^T Z - \frac{1}{2} \|\overline{Z}_i\|^2 \right\} \quad i = 1, 2, 3.$$

Подстановка численных значений \overline{Z}_i и $\|\overline{Z}_i\|^2$ приводит к

$$\begin{aligned} \xi_1(Z) &= -2Z_1 + 3Z_2 - 6,5 \\ \xi_2(Z) &= 6Z_1 + 5,6Z_2 - 33,68 \\ \xi_3(Z) &= 8Z_1 + 0,4Z_2 - 32,08 \end{aligned}$$

Таким образом, классификатор вычисляет три числа: $\xi_1(Z)$, $\xi_2(Z)$ и $\xi_3(Z)$, а затем сравнивает их. Классификатор относит Z к классу C_1 , если $\xi_1(Z)$ максимально, к классу C_2 , если $\xi_2(Z)$ максимально и к классу C_3 , если $\xi_3(Z)$ максимально.

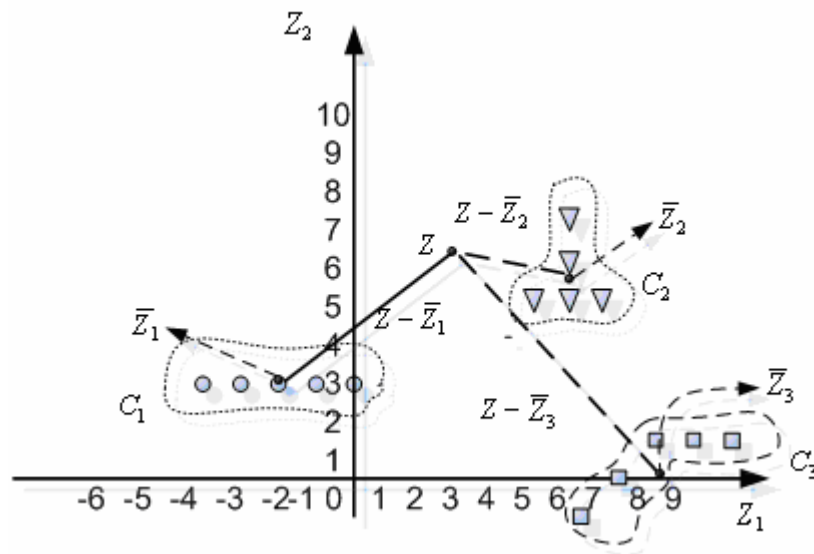


Рис. 24.1 Двумерное пространство признаков связанное с C_1, C_2, C_3 .

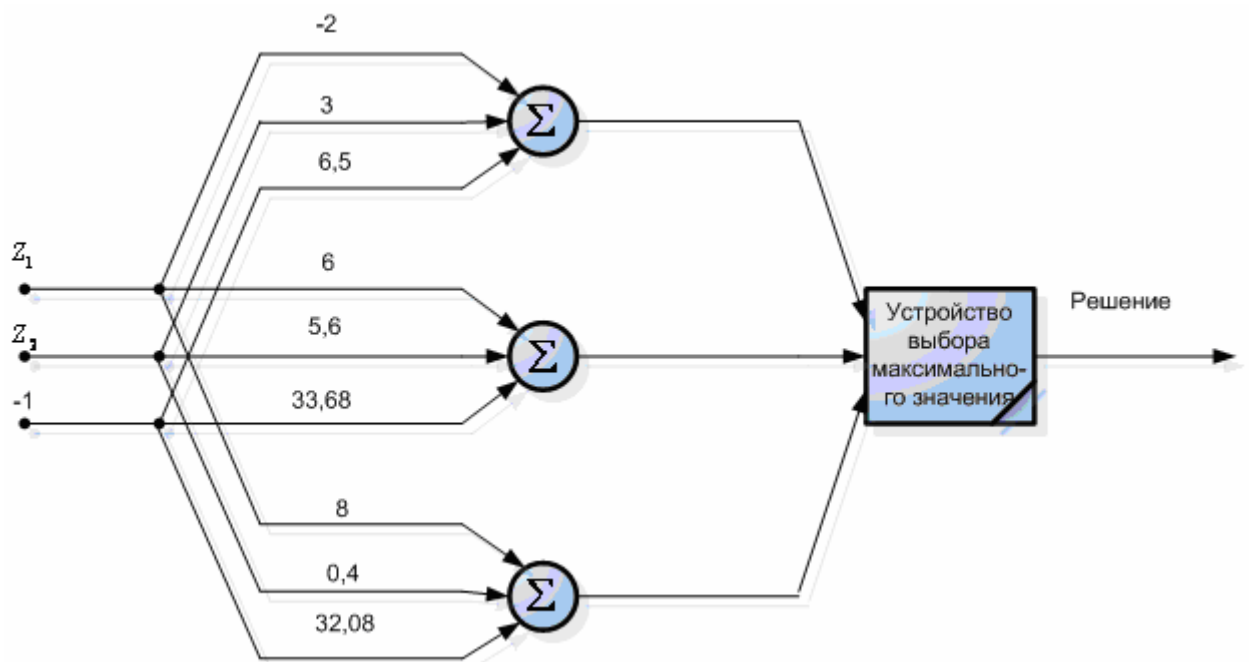


Рис. 24.2 Классификатор для трёх классов образов, работающий по критерию минимального расстояния.

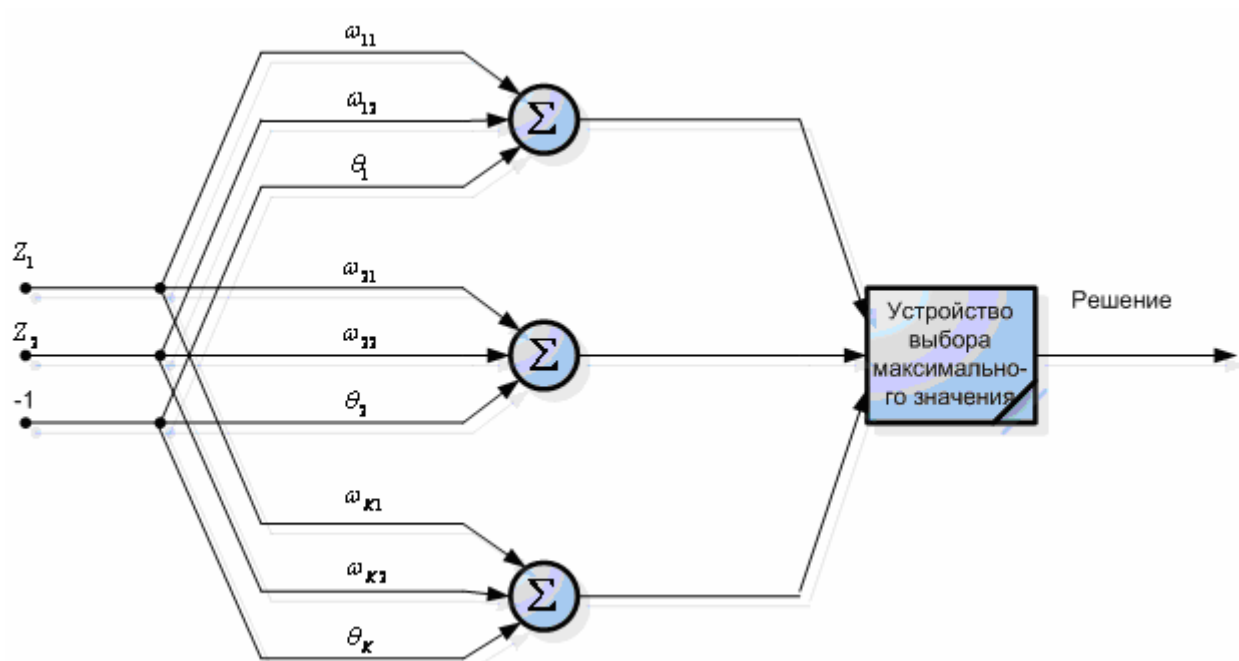


Рис. 24.3 Классификатор для K классов образов , работающий по критерию минимального расстояния.

25. МЕТОД ОТОБРАЖЕНИЯ ПО АЛГОРИТМУ НАИМЕНЬШИХ КВАДРАТОВ

При обсуждении классификаторов, работающих по критерию наименьшего расстояния, предполагалось, что классы образов в пространстве признаков группируются вокруг соответствующих им средних \bar{Z}_i ($i=1,2,\dots,K$). Однако, возможен и другой подход. При этом классификатор должен в первую очередь отображать образы в пространство решений, в котором образы, принадлежащие C_i обязательно группируются вокруг заранее выбранной точки V_i . Преобразование A , которое позволяет осуществлять это отображение из пространства признаков в пространство решений в общем случае выбирается таким, чтобы общая среднеквадратичная ошибка была минимальной. Для классификации некоторого образа этот образ сначала отображается в пространство решений, а затем классифицируется как принадлежащий C_{i_0} , если он отображён ближе к точке V_{i_0} .

Введём отображение по методу наименьших квадратов, на котором основываются классификаторы с минимальным среднеквадратичным расстоянием.

Рассмотрим множество M -мерных образов Z_{ij} , $j=1,2,\dots,N_i$, которые должны отображаться в определённую точку в K -мерном пространстве, обозначаемую $V_i = [v_1, v_2, \dots, v_k]$.

Найдём преобразование A , которое отображает $\{Z_{ij}\}$ в точку V_i , таким образом, чтобы общая среднеквадратичная ошибка, вызываемая отображением, была минимальной.

Обозначим результат отображения образа Z_{ij} через L_{ij} . Тогда соответствующий вектор ошибки будет равен

$$\varepsilon_j = L_{ij} - V_i = AZ_{ij} - V_i \quad (25.1)$$

Из выражения (25.1) следует , что общая среднеквадратичная ошибка при отображении Z_{ij} в V_i определяется как

$$\varepsilon = \frac{1}{N_i} \sum_{j=1}^{N_i} \left\| \varepsilon_j \right\|^2 \quad (25.2)$$

Подстановка (25.1) в (25.2) приводит к

$$\begin{aligned} \varepsilon &= \frac{1}{N_i} \sum_{j=1}^{N_i} \left\| AZ_{ij} - V_i \right\|^2 = \frac{1}{N_i} \sum_{j=1}^{N_i} (AZ_{ij} - V_i)' (AZ_{ij} - V_i) = \\ &= \frac{1}{N_i} \sum_{j=1}^{N_i} (AZ_{ij})' AZ_{ij} - 2(AZ_{ij})' V_i + V_i' V_i = \\ &= \frac{1}{N_i} \sum_{j=1}^{N_i} \left\{ Z_{ij}' A' AZ_{ij} - 2Z_{ij}' A' V_i + \|V_i\|^2 \right\} \end{aligned} \quad (25.3)$$

Так как A должно быть выбрано так , чтобы ε было минимальным, то оно получается в результате решения уравнения $\nabla_A \varepsilon = 0$,

что приведёт к

$$\frac{1}{N_i} \sum_{j=1}^{N_i} \nabla_A \left\{ Z_{ij}' A' AZ_{ij} \right\} - \frac{2}{N_i} \sum_{j=1}^{N_i} \nabla_A \left\{ Z_{ij}' A' V_i \right\} + \frac{1}{N_i} \sum_{j=1}^{N_i} \nabla_A \left\{ \|V_i\|^2 \right\} = 0 \quad (25.4)$$

Поскольку

$$\begin{aligned} \nabla_A \left\{ Z_{ij}' A' AZ_{ij} \right\} &= 2A \left(Z_{ij} Z_{ij}' \right), \\ \nabla_A \left\{ Z_{ij}' A' V_i \right\} &= V_i Z_{ij}', \\ \nabla_A \left\{ \|V_i\|^2 \right\} &= 0 \end{aligned}$$

то подстановка в (25.4) приведёт к

$$A \left[\frac{2}{N_i} \sum_{j=1}^{N_i} (Z_{ij} Z_{ij}') \right] = \frac{2}{N_i} \sum_{j=1}^{N_i} V_i Z_{ij}',$$

что позволяет определить A как

$$A = \left[\sum_{j=1}^{N_i} V_i Z'_{ij} \right] \left[\sum_{j=1}^{N_i} (Z_{ij} Z'_{ij}) \right]^{-1}$$

Рассмотрим пример . Пусть множество $\{Z_{ij}\}$ имеет вид

$$Z_{i1} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}; Z_{i2} = \begin{bmatrix} 5 \\ 6 \end{bmatrix}; Z_{i3} = \begin{bmatrix} 5 \\ 7 \end{bmatrix}; Z_{i4} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}; Z_{i5} = \begin{bmatrix} 6 \\ 7 \end{bmatrix}$$

что соответствует $N_i = 5$. Пусть $V_i = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ тогда

$$\begin{aligned} \sum_{j=1}^5 V_i Z'_{ij} &= \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 5 & 5 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 5 & 6 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 5 & 7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 6 & 6 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 6 & 7 \end{bmatrix} = \\ &= \begin{bmatrix} 5 & 5 \\ 5 & 5 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 5 & 6 \end{bmatrix} + \begin{bmatrix} 5 & 7 \\ 5 & 7 \end{bmatrix} + \begin{bmatrix} 6 & 6 \\ 6 & 6 \end{bmatrix} + \begin{bmatrix} 6 & 7 \\ 6 & 7 \end{bmatrix} = \begin{bmatrix} 27 & 31 \\ 27 & 31 \end{bmatrix} \\ \sum_{j=1}^5 Z_{ij} Z'_{ij} &= \begin{bmatrix} 25 & 25 \\ 25 & 25 \end{bmatrix} + \begin{bmatrix} 25 & 30 \\ 30 & 36 \end{bmatrix} + \begin{bmatrix} 25 & 35 \\ 35 & 49 \end{bmatrix} + \begin{bmatrix} 36 & 36 \\ 36 & 36 \end{bmatrix} + \begin{bmatrix} 36 & 42 \\ 42 & 49 \end{bmatrix} = \begin{bmatrix} 147 & 168 \\ 168 & 195 \end{bmatrix} \end{aligned}$$

Тогда

$$A = \begin{Bmatrix} 0,129 & 0,0475 \\ 0,129 & 0,0475 \end{Bmatrix}$$

Далее вычислим

$$\begin{aligned} L_{ij} &= A Z_{ij} \\ L_{i1} &= \begin{bmatrix} 0,883 \\ 0,883 \end{bmatrix}; L_{i2} = \begin{bmatrix} 0,930 \\ 0,930 \end{bmatrix}; L_{i3} = \begin{bmatrix} 0,978 \\ 0,978 \end{bmatrix}; L_{i4} = \begin{bmatrix} 1,06 \\ 1,06 \end{bmatrix}; L_{i5} = \begin{bmatrix} 1,11 \\ 1,11 \end{bmatrix} \end{aligned}$$

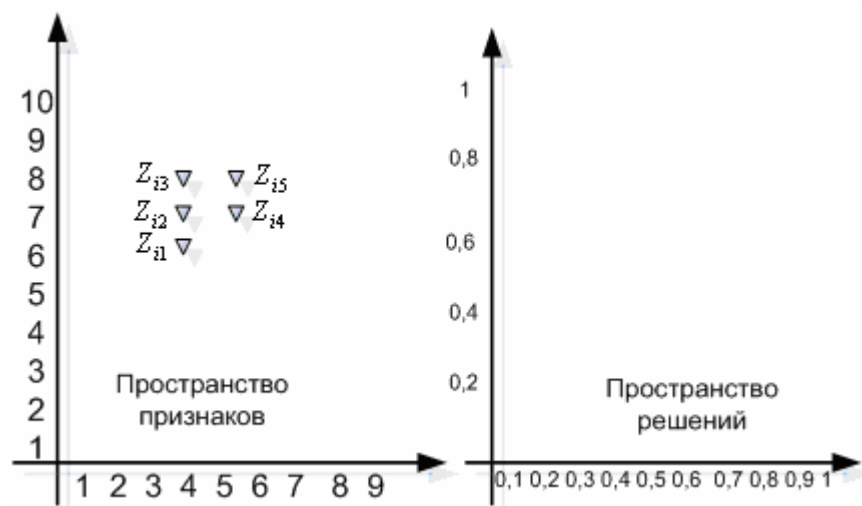


Рис. 25.1

Понятие расширенного пространства признаков необходимо при разработке классификатора с минимальным среднеквадратичным расстоянием.

Такое понятие вытекает из определения дискриминантной функции

$$\xi(Z) = w_1 Z_1 + w_2 Z_2 + \dots + w_d Z_d - \theta$$

В матричном виде это соотношение можно записать как

$$\xi(Z) = W' \hat{Z},$$

где $W' = [w_1 w_2, \dots, w_d \theta]$; $\hat{Z}' = [Z_1 Z_2, \dots, Z_d - 1] = [Z' - 1]$

Из последнего соотношения следует, что \hat{Z} можно получить из данного образа Z приписыванием к нему дополнительной координаты, равной -1. Пространство, состоящее из $(d+1)$ -мерных образов Z , называется расширенным пространством признаков.

26. КЛАССИФИКАТОР ДЛЯ РАСПОЗНАВАНИЯ ТРЁХ КЛАССОВ ОБРАЗОВ ПО КРИТЕРИЮ НАИМЕНЬШЕГО СРЕДНЕКВАДРАТИЧНОГО РАССТОЯНИЯ

Классификатор для распознавания трёх классов, работающий по критерию наименьшего среднеквадратичного расстояния можно представить иллюстративно следующим образом

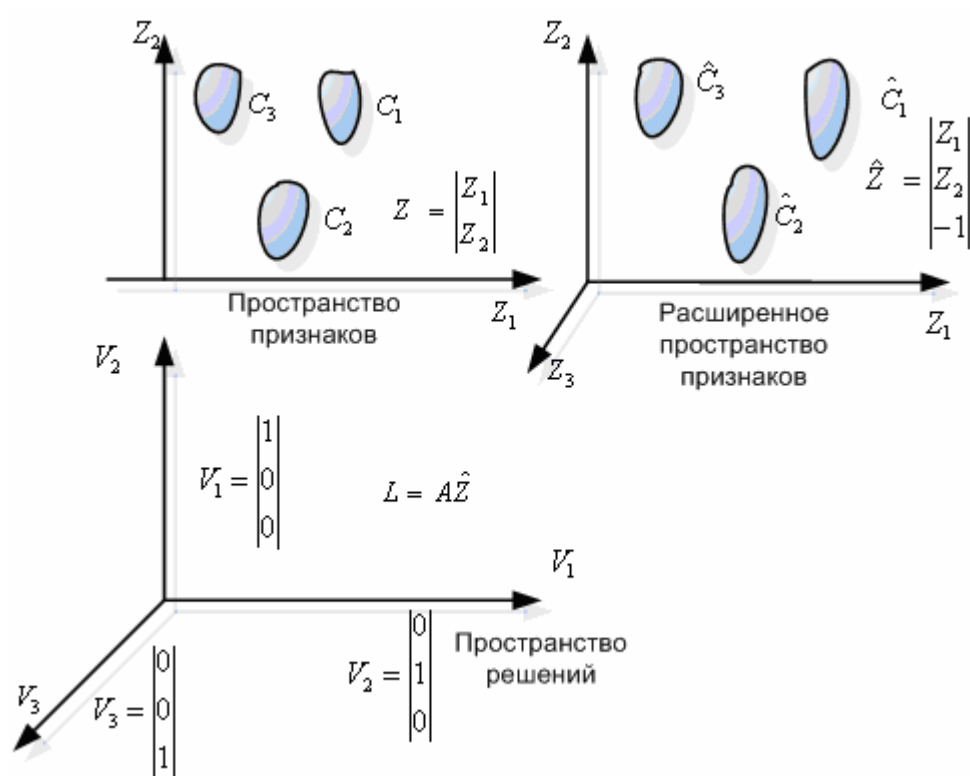


Рис. 26.1

Определяем преобразование A , отображающее $\hat{Z} \in C_k$; $k=1,2,3$ в V_k в смысле наименьших квадратов. Выберем в качестве вершины трёх единичных векторов, как показано на рис. Обозначим априорные вероятности классов C_k через P_k . Преобразование A можно непосредственно найти из соотношения

$$A \left[\frac{P_1}{N_1} \sum_{j=1}^{N_1} \hat{Z}_{1j} \hat{Z}'_{1j} + \frac{P_2}{N_2} \sum_{j=1}^{N_2} \hat{Z}_{2j} \hat{Z}'_{2j} + \frac{P_3}{N_3} \sum_{j=1}^{N_3} \hat{Z}_{3j} \hat{Z}'_{3j} \right] =$$

$$= \frac{P_1}{N_1} \sum_{j=1}^{N_1} V_1 \hat{Z}'_{1j} + \frac{P_2}{N_2} \sum_{j=1}^{N_2} V_2 \hat{Z}'_{2j} + \frac{P_3}{N_3} \sum_{j=1}^{N_3} V_3 \hat{Z}'_{3j}$$

Решая это уравнение относительно A , получим

$$A = S_{\hat{Z}}^{-1} \hat{Z} \hat{Z}';$$

$$\text{где } S_{\hat{Z}} = \sum_{i=1}^3 \sum_{j=1}^{N_i} \frac{P_i}{N_i} (V_i \hat{Z}'_{ij}) = E\{\hat{Z} \hat{Z}'\} \quad (26.1)$$

$$S_{\hat{Z}\hat{Z}} = \sum_{i=1}^3 \sum_{j=1}^{N_i} \frac{P_i}{N_i} (\hat{Z}_{ij} \hat{Z}'_{ij}) = E\{\hat{Z} \hat{Z}'\} \quad (26.2)$$

Очевидно, что $S_{\hat{Z}}$ и $S_{\hat{Z}\hat{Z}}$ являются матрицами взаимной корреляции и автокорреляции соответственно. Следует отметить, что матрица A имеет размерность $[3 \times (d+1)]$. Для классификации образа $\hat{Z}' = [Z_1 Z_2, \dots, Z_d - 1]$ классификатор вычисляет в первую очередь $L = A \hat{Z}$, а затем отображает \hat{Z} в пространство решений. При этом применяется следующее решающее правило по критерию минимума расстояния: если L наиболее близко к V_{i_0} , то Z классифицируется как принадлежащий C_{i_0} . Расстояния, которые вычисляет классификатор определяются как

$$D_i^2 = \|L - V_i\|^2, i = \overline{1,3}; \text{ т.е.}$$

$$D_i^2 = (L - V_i)' (L - V_i) = \|L\|^2 - 2V_i' L + \|V_i\|^2$$

Так как $\|V_i\|^2 = 1$, то D_i^2 будет минимально, когда $V_i' L$ максимально. Поэтому вместо D_i^2 в последнем соотношении достаточно, чтобы классификатор вычислял

$$d_i = V_i' L \quad (26.3)$$

Подставляя $V_1' = [1 \ 0 \ 0]$, $V_2' = [0 \ 1 \ 0]$, $V_3' = [0 \ 0 \ 1]$ и $L = A\hat{Z}$ в (26.3) получаем

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} A\hat{Z} = A\hat{Z}$$

Матрица преобразования обозначается как

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1d} & \theta_1 \\ a_{21} & a_{22} & \dots & a_{2d} & \theta_2 \\ a_{31} & a_{32} & \dots & a_{3d} & \theta_3 \end{bmatrix},$$

где d_i представляют собой следующие дискриминантные функции, которые определяют классификатор

$$\xi(Z) = d_i = a_{i1}Z_1 + a_{i2}Z_2 + \dots + a_{id}Z_d - \theta_i.$$

Из последнего соотношения следует, что классификатор полностью определяется матрицей преобразования A , которая получается из множества обучающих образов. Для классификации данного образа классификатор вычисляет три числа $\xi_1(Z)$, $\xi_2(Z)$ и $\xi_3(Z)$.

Если $\max\{\xi_i(Z)\} = \xi_{i_0}(Z)$, то этот образ приписывается C_{i_0} .

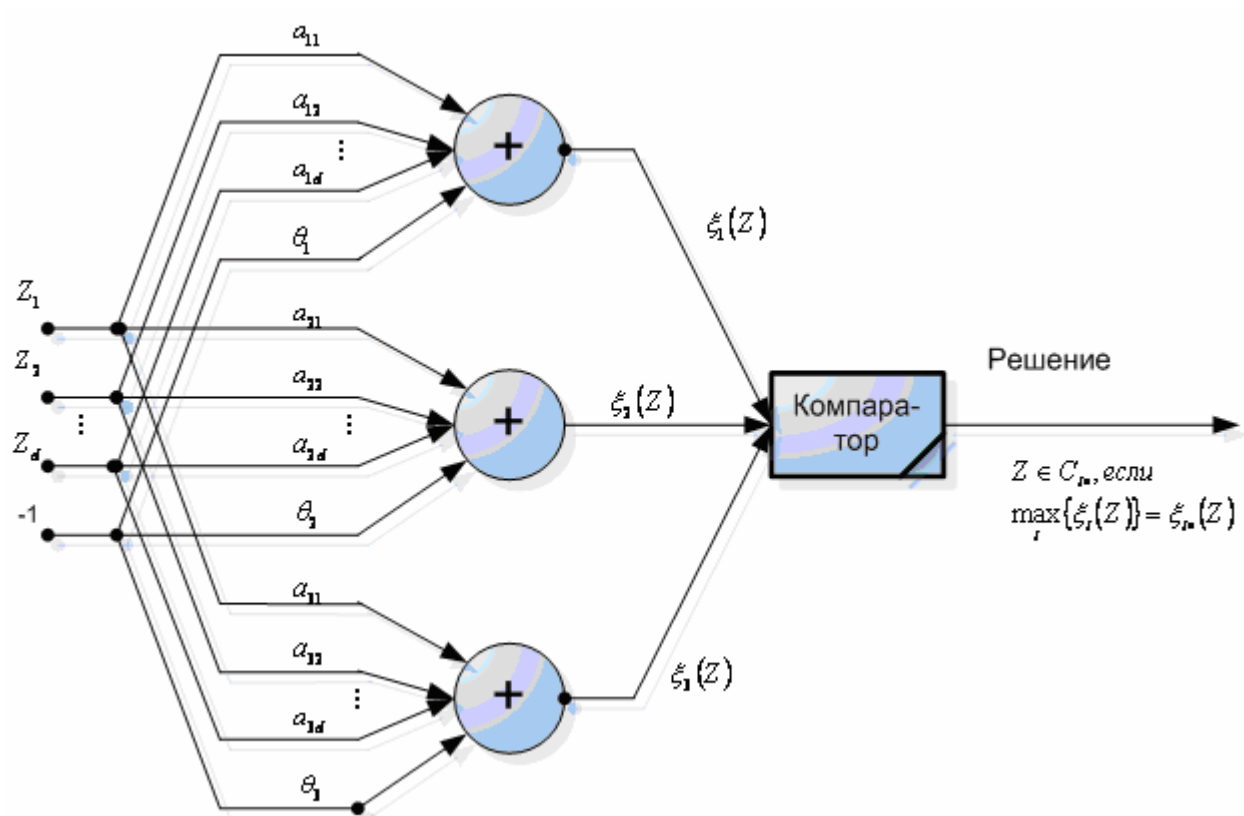


Рис. 26.2 Структура классификатора для трех классов

27. КЛАССИФИКАТОР ДЛЯ РАСПОЗНАВАНИЯ K КЛАССОВ ОБРАЗОВ ПО КРИТЕРИЮ НАИМЕНЬШЕГО СРЕДНЕКВАДРАТНОГО РАССТОЯНИЯ

Задача трёх классов может быть обобщена для случая K классов. Расширенные образы, принадлежащие C_i отображаются в вершине единичного K -вектора V_i

$$V_i = [0 \cdots 0 1 0 \cdots 0],$$

где ненулевой элемент «1» находится в i -ой позиции V_i , $i = \overline{1, K}$. Вместо матрицы преобразования A размерностью $3 \times (d+1)$ получаем матрицу размерностью $K \times (d+1)$

$$A = S_{vx} S_{xx}, \text{ где}$$

$$S_{vz} = \sum_{i=1}^K \sum_{j=1}^{N_i} P_i / N_i (V_i \hat{Z}_{ij}') = E(V \hat{Z}')$$

$$S_{zz} = \sum_{i=1}^K \sum_{j=1}^{N_i} P_i / N_i (\hat{Z}_{ij} \hat{Z}_{ij}') = E(\hat{Z} \hat{Z}')$$

Таким образом, дискриминантные функции, определяющие классификатор, имеют вид

$$\xi_i(z) = a_{i1}z_1 + a_{i2}z_2 + \cdots + a_{id}z_d - \theta_i, i = \overline{1, K}$$

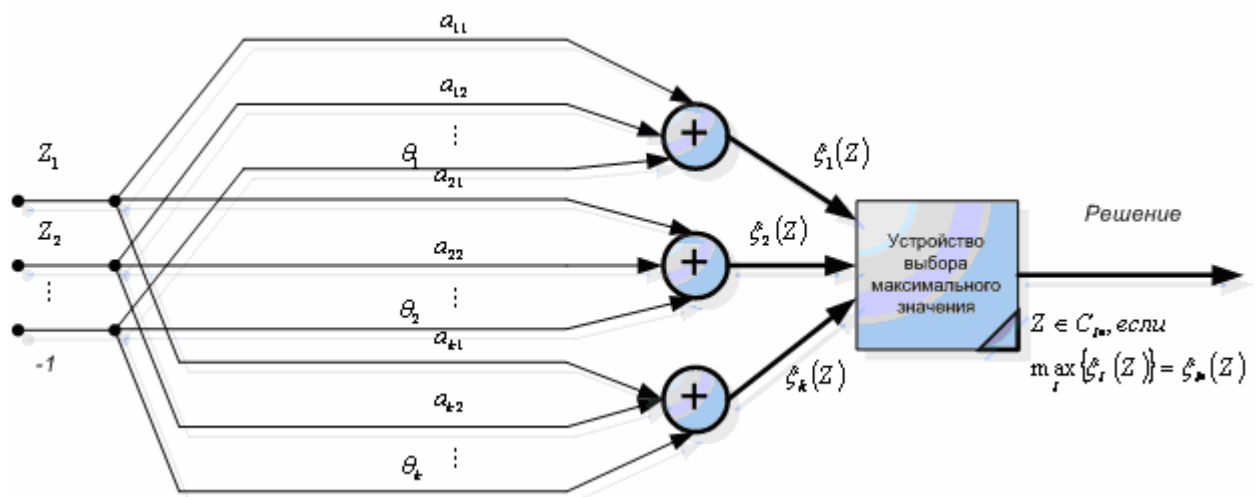


Рис. 27.1 Структура классификатора к классов

При обсуждении классификаторов по минимуму расстояния мы ограничивались рассмотрением только линейных классификаторов, т.е. классификаторов, которые обладают линейными разделяющими границами в пространстве признаков. Соответствующую процедуру обучения нетрудно получить для случая квадратичных разделяющих границ, но при этом реализация классификатора усложняется. Квадратичный классификатор определяется дискриминантными функциями вида

$$\xi_i(z) = \sum_{j=1}^d w_{ij} z_j^2 + \sum_{j=1}^{d-1} \sum_{k=j+1}^d w_{jk} z_j z_k + \sum_{j=1}^d w_j z_j - \theta_i, i = \overline{1, K} \quad (27.1)$$

Из последнего соотношения следует, что квадратичная дискриминантная функция $\xi_i(z)$ имеет $[(d+1)/(d+2)]/2$ весовых коэффициентов или параметров, а именно:

d параметров, соответствующих коэффициентам при $z_j^2 - w_{jj}$

d параметров, соответствующих коэффициентам при $z_j - w_j$

$d(d-1)/2$ параметров, соответствующих коэффициентам при $z_j z_k - w_{jk}, j \neq k$

порог- θ_i

Приведённые выше параметры определяются в процессе обучения .

Для пояснения реализации классификатора, связанного с вычислением (27.1) определим Q -мерный вектор G , координаты которого f_1, f_2, \dots, f_Q являются функциями z_i , $i = \overline{1, d}$. Первые d координат вектора G имеют вид $z_1^2, z_2^2, \dots, z_d^2$, следующие $[d(d-1)]/2$ координаты всем парам $z_1 z_2, z_1 z_3, \dots, z_{d-1} z_d$; последние α координат представляют собой z_1, z_2, \dots, z_d .

Тогда реализация классификатора будет иметь вид

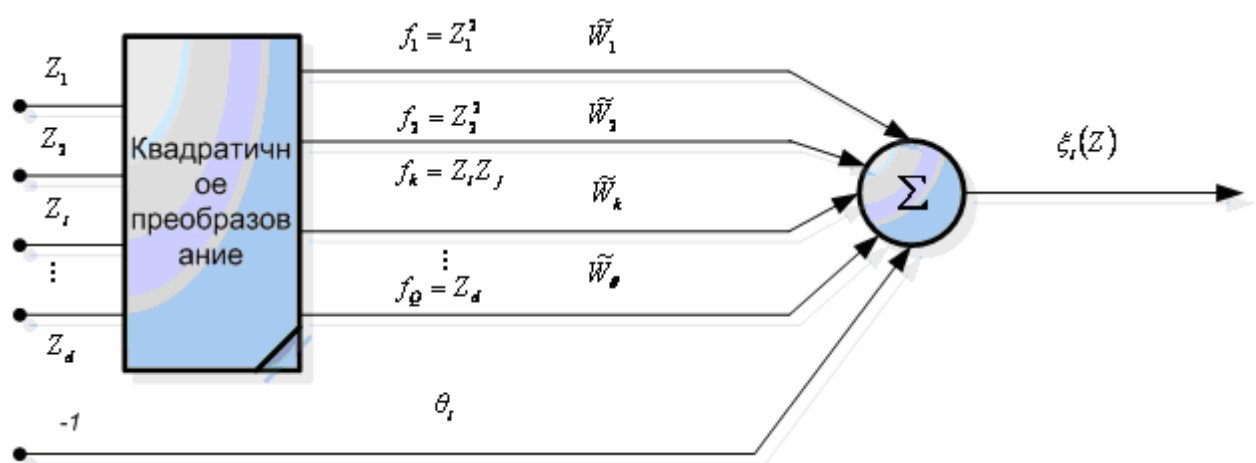
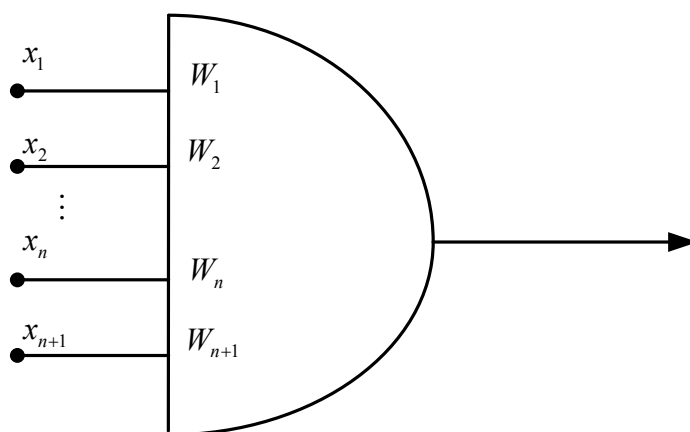


Рис. 27.2

$$\xi_i(z) = \tilde{w}_1 f_1 + \tilde{w}_2 f_2 + \dots + \tilde{w}_Q f_Q - \Theta_i$$

28. МОДЕЛЬ НЕЙРОННОЙ СЕТИ. ОБУЧАЮЩИЙ АЛГОРИТМ ДЛЯ ПЕРЦЕПТРОНА

При разбиении на два класса реализация классификатора может быть осуществлена с помощью порогового устройства



$$R = 1, \text{ если } W'X > T$$

$$R = -1, \text{ если } W'X < T$$

Рис. 28.1

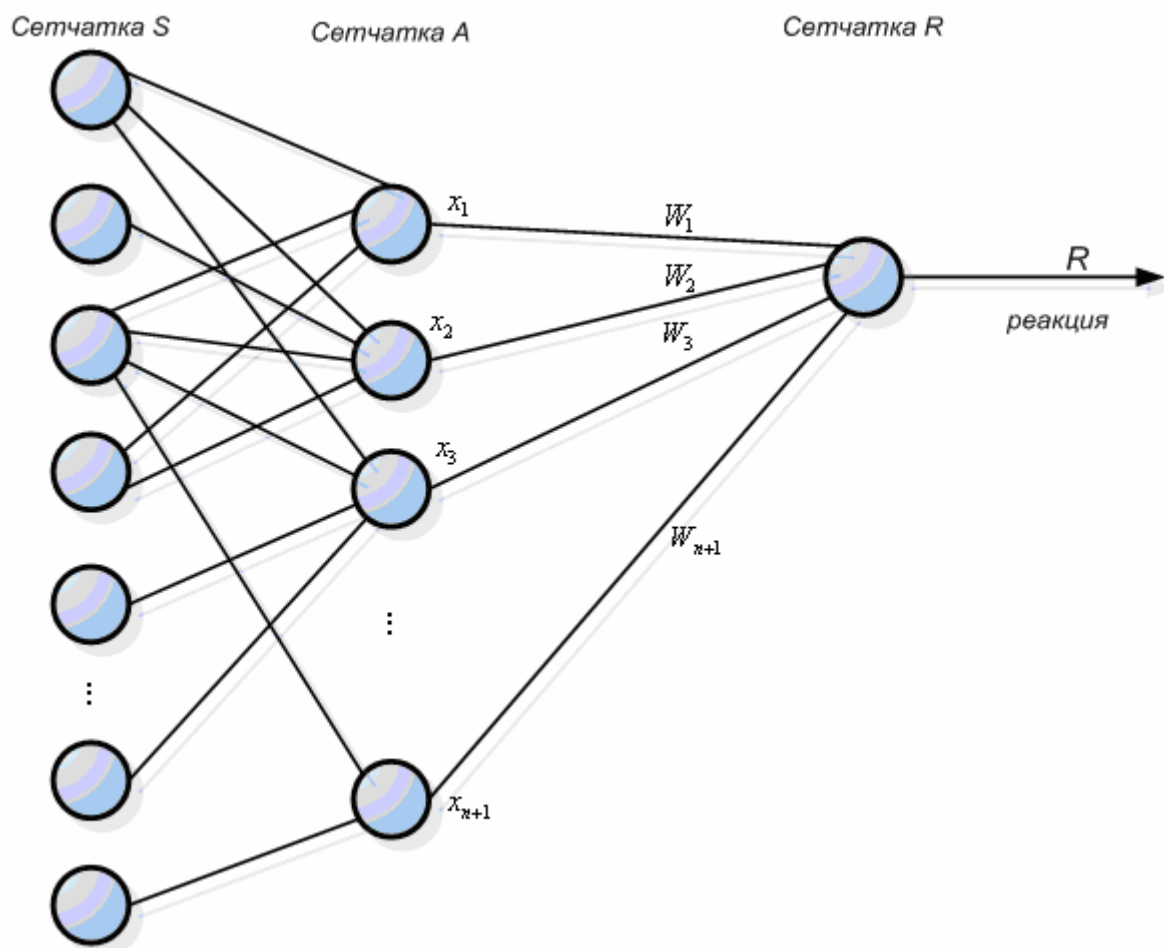
Эти две выходные величины (реакции порогового устройства) обычно обозначаются как 1 и -1 соответственно

Если заданы два множества образов, принадлежащих соответственно классам ω_1, ω_2 , то решение ищется в виде вектора весов W , обладающего тем свойством, что для всех образов класса ω_1 выполняется условие $W'X > 0$, а для всех образов класса ω_2 - условие $W'X < 0$. Если образы класса ω_2 умножить на -1 , то эквивалентное условие $W'X > 0$ становится общим для всех образов.

Причём класс устройств, предложенных в начале 80-х годов американским учёным Розенблаттом и называемых перцептронами,

представляет естественную и обладающую большими возможностями модель процесса обучения машин.

Основная модель пещептрона, обеспечивающая отнесение образа к одному из заданных классов приведена на рис. 28.2.



Устройство состоит из сетчатки S сенсорных элементов , которые случайным образом соединены с ассоциативными элементами второй сетчатки A . Каждый из элементов второй сетчатки воспроизводит выходной сигнал только в том случае ,если достаточное число сенсорных элементов, соединённых с его входом, находится в возбуждённом состоянии. Сенсорные элементы можно рассматривать в качестве устройств, с помощью которых вся система воспринимает из внешней среды стимулы ,т.е. как некие измерительные устройства ,а ассоциативные элементы –как входную часть системы.

Реакция всей системы пропорциональна сумме взятых с определёнными весами реакций элементов ассоциативной сетчатки. Таким образом, обозначив через x_i реакцию i -го ассоциативного элемента и через w_i - соответствующий вес, реакцию системы можно записать как

$$R = \sum_{i=1}^{n+1} W_i' x_i = W'X$$

Если $R > 0$, значит предъявленный системе образ принадлежит классу ω_1 , если $R < 0$, то образ относится к классу ω_2 .

Отметим, что перцептронная модель представляет собой за исключением сенсорной сетчатки не что иное, как реализацию линейной решающей функции.

Схему, приведённую на рис 28.2, легко распространить на случай разделения на несколько классов посредством увеличения числа реагирующих элементов в R -сетчатке.

Классификация проводится обычным способом: рассматриваются значения реакций R_1, R_2, \dots, R_M , где M - число классов и образ причисляется к классу ω_i , если $R_i > R_j$ для всех $i \neq j$.

Основную модель можно также легко распространить на случай нелинейных решающих функций введением соответствующих нелинейных преобразователей между сетчатками A и R .

Обучающий алгоритм для перцептрона сводится к простой схеме итеративного определения вектора весов W . Дадим краткое определение этой схемы, которую обычно называют алгоритмом перцептрона. Заданы два обучающие множества, представляющие классы ω_1 и ω_2 соответственно.

Пусть $W(1)$ начальный вектор весов, который выбирается произвольно. В таком случае k -ый шаг обучения выглядит следующим образом

Если $x(k) \in \omega_1$ и $W'(k)x(k) \leq 0$, то вектор весов $W(k)$ заменяется вектором

$$W(k+1)=W(k)+cx(k), (1) \quad (28.1)$$

где c -корректирующее приращение

Если $x(k) \in \omega_2$ и $W'(k)x(k) \geq 0$, то $W(k)$ заменяется вектором

$$W(k+1)=W(k)-cx(k), \quad (28.2)$$

в противном случае $W(k)$ не изменяется ,т.е.

$$W(k+1)=W(k)$$

Иными словами, алгоритм вносит изменения в вектор весов W в том и только том случае ,если образ предъявленный на k -м шаге был при выполнении этого шага неправильно классифицирован с помощью соответствующего вектора весов . Причём корректирующее приращение должно быть положительным .

Очевидно, что алгоритм перцептрона является процедурой типа «подкрепление - наказание» ,причём подкреплением за правильную классификацию образа служит отсутствие наказания .Иными словами, если образ классифицирован правильно, то система подкрепляется тем, что в вектор весов W не вносятся никаких изменений.

С другой стороны если образ классифицирован неправильно и произведение $W'(k)x(k)$ оказывается меньше нуля , когда оно должно быть больше нуля, система наказывается увеличением значения вектора весов $W(k)$ на величину, пропорциональную $x(k)$.Точно так же, если произведение $W'(k)x(k)$ оказывается больше нуля , когда оно должно быть меньше нуля, система наказывается противоположным образом.

Сходимость алгоритма наступает при правильной классификации всех образов с помощью некоторого вектора весов. Если заданные классы

линейно разделены , то алгоритм перцептрона сходится за конечное число итераций.

Пример: рассмотрим образы представленные на рис. 28.1 и 28.2

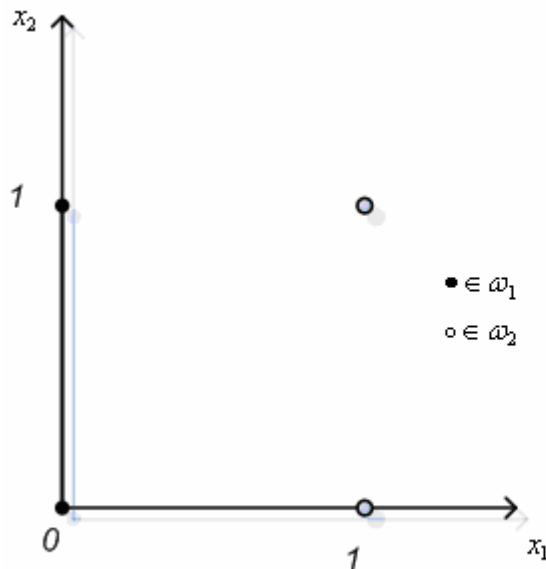


Рис. 28.1

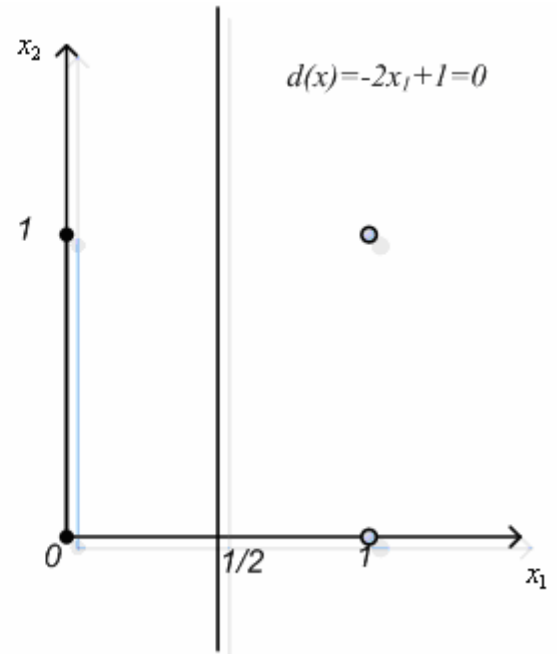


Рис. 28.2

Следует применить к этим образам алгоритм перцептрона с тем, чтобы с его помощью определить весовой вектор решения. Осмотр образов показывает, что два заданных класса линейно разделимы и следовательно применение алгоритма окажется успешным.

До начала работы алгоритма пополним все образы. При этом рассматриваемые классы обратятся

$$\begin{array}{ll} \omega_1: \{0 \ 0 \ 1\} & \omega_2: \{1 \ 0 \ 1\} \\ \omega_1: \{0 \ 1 \ 1\} & \omega_2: \{1 \ 1 \ 1\} \end{array}$$

Задав $c=1$, $W(1)=0$ и предъявив образы в указанном выше порядке , получим (по шагам)

$$W'(1)x(1) = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 0$$

$$W(2) = W(1) + x(1) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(2)x(2) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = I$$

$$W(3) = W(2) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(3)x(3) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = I$$

$$W(4) = W(3) - x(3) = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$$

$$W'(4)x(4) = \begin{pmatrix} -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = -I$$

$$W(5) = W(4) = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$$

В связи с ошибками классификации коррекция вектора весов производилась на 1-м и 3-м в соответствии с формулами (28.1) и (28.2).

Так как получаемый результат можно считать искомым решением только в том случае, когда алгоритм осуществит без ошибок полный цикл

итераций по всем образам, обучающее множество следует предъявить ещё раз. Процесс обучения системы продолжается при $x_5=x_1, x_6=x_2, x_7=x_3, x_8=x_4$.

Второй цикл итерации приводит к следующим результатам:

$$W'(5)x(5) = \begin{pmatrix} -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 0$$

$$W(6) = W(5) + x(5) = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(6)x(6) = \begin{pmatrix} -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = 1$$

$$W(7) = W(6) = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(7)x(7) = \begin{pmatrix} -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = -1 + 0 + 1 = 0$$

$$W(8) = W(7) - x(7) = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \\ 0 \end{pmatrix}$$

$$W'(8)x(8) = \begin{pmatrix} -2 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = -2$$

$$W(9) = W(8) = \begin{pmatrix} -2 \\ 0 \\ 0 \end{pmatrix}$$

Поскольку в данном цикле итерации совершено две ошибки, все образы предъявляются ещё раз.

$$W'(9)x(9) = \begin{pmatrix} -2 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 0$$

$$W(10) = W(9) + x(9) = \begin{pmatrix} -2 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(10)x(10) = \begin{pmatrix} -2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = -2 \times 0 + 0 \times 1 + 1 \times 1 = 1$$

$$W(11) = W(10) = \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(11)x(11) = \begin{pmatrix} -2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = -2 \times 1 + 0 \times 0 + 1 \times 1 = -1$$

$$W(12) = W(11) = \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(12)x(12) = \begin{pmatrix} -2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = -1$$

$$W(13) = W(12) = \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

Поскольку в данном цикле итераций совершена одна ошибка, все образы предъявляется ещё раз.

$$W'(13)x(13) = \begin{pmatrix} -2 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = 1$$

$$W(14)=W(13)=\begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(14)x(14)=\begin{pmatrix} -2 & 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}=I$$

$$W(15)=W(14)=\begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(15)x(15)=\begin{pmatrix} -2 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}=-2\times I+I\times I=-I$$

$$W(16)=W(15)=\begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

$$W'(16)x(16)=\begin{pmatrix} -2 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}=-2\times I+I\times I=-I$$

$$W(17)=W(16)=\begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix}$$

Итак за четыре итерации все образы классифицируются правильно.
Вектор решения имеет вид

$$W=\begin{pmatrix} -2 & 0 & 1 \end{pmatrix}$$

Соответствующей решающей функцией будет

$$d(x)=-2x_I+I$$

Приравнивание этой функции к нулю даёт уравнение разделяющей границы.

Умножив все образы класса ω_2 на -1 ,алгоритм перцептрона можно записать как

$$W(k+1) = \begin{cases} W(k), & \text{если } W'(k)x(k) > 0 \\ W(k) + cx(k), & \text{если } W'(k)x(k) \leq 0 \end{cases}$$

где с-положительное корректирующее приращение.

Обобщающий алгоритм перцептрона

Рассмотрим М классов $\omega_1, \omega_2, \dots, \omega_M$.

Пусть на k-ой итерации процедуры обучения системе предъявляется образ $x(k)$, принадлежащий классу ω_i . Вычисляются значения М решающих функций

$$d_j[x(k)] = W_j'(k)x(k); \quad j=1, 2, \dots, M.$$

Затем если выполняются условия

$$d_i[x(k)] > d_j[x(k)], \quad j = \overline{1, M} \quad i \neq j$$

то векторы весов не изменяются, т.е.

$$W_j(k+1) = W_j(k),$$

Допустим с другой стороны , что для некоторого l

$$d_i[x(k)] \leq d_l[x(k)]$$

В этом случае производятся следующие коррекции весов

$$W_i(k+1) = W_i(k) + cx(k)$$

$$W_l(k+1) = W_l(k) - cx(k)$$

$$W_j(k+1) = W_j(k)$$

$$j = 1, 2, \dots, M; j \neq i; j \neq l$$

где c -положительная константа.

Если при рассмотрении этого случая классы разделимы, то этот алгоритм сходится за конечное число итераций при произвольных начальных векторах $W_i(1)$.

29. ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ

29.1. ОБЩИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Приведем одно из определений искусственных нейронных сетей:

Искусственная нейронная сеть (ИНС) – это существенно параллельно распределенный процессор, который обладает способностью к сохранению и репрезентации опытного знания. Она сходна с мозгом в двух аспектах:

- знание приобретается сетью в процессе обучения;
- для сохранения знания используются силы межнейронных соединений, называемые также синаптическими соединениями.

Работа нейронной сети (НС) состоит в преобразовании входных сигналов во времени, в результате чего меняется внутреннее состояние сети и формируются выходные воздействия. Обычно НС оперирует цифровыми, а не символьными величинами. Большинство моделей НС требуют обучения. В общем случае, **обучение** — это такой выбор параметров сети, при котором сеть лучше всего справляется с поставленной проблемой. Обучение — это задача многомерной оптимизации и для ее решения существует множество алгоритмов.

Современные искусственные НС демонстрируют такие ценные свойства, как:

1. *Обучаемость*. Выбрав одну из моделей НС, создав сеть и выполнив алгоритм обучения, мы можем обучить сеть решению задачи, которая ей по силам.

2. *Способность к обобщению*. После обучения сеть становится нечувствительной к малым изменениям входных сигналов (шуму или вариациям входных образов) и дает правильный результат на выходе.

3. *Способность к абстрагированию.* Если предъявить сети несколько искаженных вариантов входного образа, то сеть сама может создать на выходе идеальный образ, с которым она никогда не встречалась.

К задачам, успешно решаемым НС на данном этапе их развития относятся:

- распознавание зрительных, слуховых образов;
- ассоциативный поиск информации и создание ассоциативных моделей, синтез речи, формирование естественного языка;
- формирование моделей и различных нелинейных и трудно описываемых математически систем, прогнозирование развития этих систем во времени;
- системы управления и регулирования с предсказанием;
- разнообразные конечные автоматы: системы массового обслуживания и коммутации, телекоммуникационные системы;
- принятие решений и диагностика, исключаящие логический вывод, особенно в областях, где отсутствуют четкие математические модели: в медицине, криминалистике, финансовой сфере.

На рис. 29.1 показана структура пары типичных биологических нейронов. Дендриты идут от тела нервной клетки к другим нейронам, где они принимают сигналы в точках соединения, называемых синапсами. Принятые синапсом входные сигналы подводятся к телу нейрона. Здесь они суммируются, причем одни входы стремятся возбудить нейрон, другие – воспрепятствовать его возбуждению. Когда суммарное возбуждение в теле нейрона превышает некоторый порог, нейрон возбуждается, посылая по аксону сигнал другим нейронам. У этой основной функциональной схемы много усложнений и исключений, тем не менее большинство искусственных нейронных сетей моделируют лишь эти простые свойства.

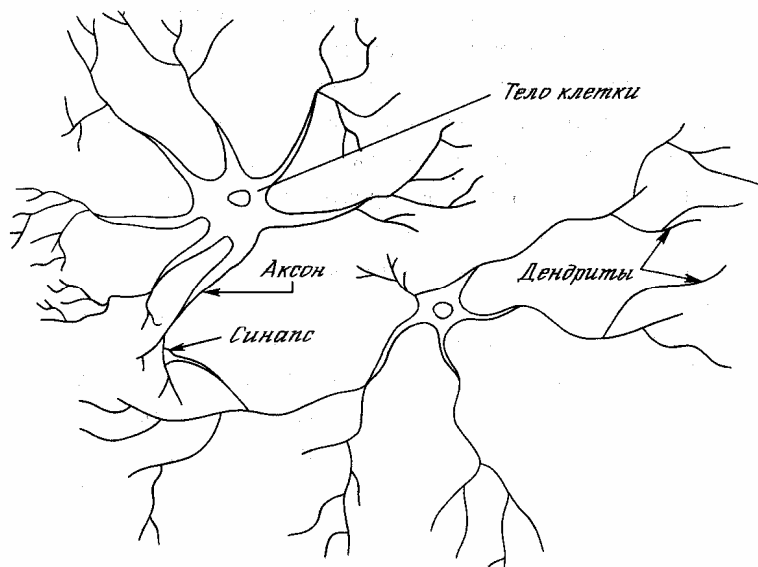


Рис. 7.1. Биологический нейрон

Основным элементом нейронной сети является **нейрон**, который осуществляет операцию нелинейного преобразования суммы произведений входных сигналов на весовые коэффициенты:

$$y = f\left(\sum_{i=1}^n w_i x_i + T\right), \quad (29.1)$$

где $X=(x_1, x_2, \dots, x_n)$ – вектор входного сигнала; $W=(w_1, w_2, \dots, w_n)$ – весовой вектор; T – порог; f – так называемая функция активации.

Схема нейронного элемента изображена на рис 29.2. Каждому i -му входу нейрона соответствует весовой коэффициент w_i (синапс), который характеризует силу синаптической связи по аналогии с биологическим нейроном.

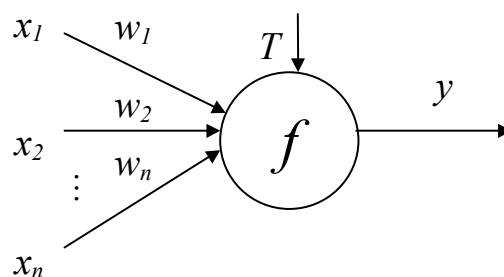


Рис. 7.2. Искусственный нейрон

Весовой вектор W , порог T и пороговая функция f определяют поведение любого нейрона – его реакцию на входные данные. Величина веса w_i определяет степень влияния входа i на выход нейрона, а знак – характер влияния. Каждый из входов связан с некоторым источником информации (рецептор, формирующий признаки, распределительная ячейка, выход другого нейрона). Положительные веса характерны для возбуждающих связей, способствующих повышению активности нейрона. Отрицательные, как правило, соответствуют тормозящим связям.

Порог, если он используется, является характеристикой, задающей начальный уровень активности (при нулевом входе) и помогающей настроить нейрон на пороговую функцию. Изменение порога эквивалентно сдвигу пороговой функции по оси абсцисс. Ряд авторов вводят дополнительный вход нейрона x_0 , всегда равный 1 , и обозначают порог как его вес w_0 . Это позволяет упростить выражение (29.1) и математическую запись некоторых алгоритмов обучения. Однако на практике при программной реализации это не приводит к экономии времени и способствует ошибкам, кроме этого, порог может настраиваться иначе, чем весовой вектор.

Функция активации используется для ограничения выхода нейрона в заданном диапазоне и для нелинейного преобразования взвешенной суммы. Последнее позволяет нейронному классификатору аппроксимировать любую нелинейную границу между классами в пространстве образов. Функция

активации выбирается для конкретной задачи и является неизменной характеристикой отдельного нейрона.

Могут использоваться следующие функции активации и их гибриды:

1) линейная функция $y = Ax$;

2) пороговая функция $y = \begin{cases} 1, x > 0 \\ 0, x \leq 0 \end{cases}$;

3) биполярная пороговая функция $y = \begin{cases} 1, x > 0 \\ -1, x \leq 0 \end{cases}$;

4) сигмоидная функция (рис. 29.2)

$$y = \frac{1}{(1 + e^{-x})}; \quad (29.2)$$

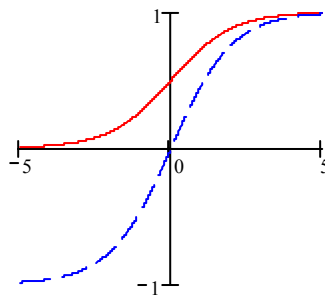


Рис. 7.2. Сигмоидные функции активации

5) биполярная сигмоидная функция (см. рис. 29.2)

$$y = \frac{2}{(1 + e^{-x})} - 1; \quad (29.3)$$

6) гиперболический тангенс (рис. 29.3)

$$y = th(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \quad (29.4)$$

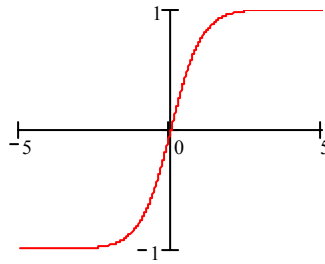


Рис. 7.3. Гиперболический тангенс

Для классификаторов чаще всего применяют функции (29.2) – (29.4), поскольку они нелинейные и хорошо дифференцируются.

Веса и порог конкретного нейрона являются настраиваемыми параметрами. Они содержат знания нейрона, определяющие его поведение. Процесс настройки этих знаний с целью получения нужного поведения называется обучением.

Нейронная сеть – совокупность нейронных элементов и связей между ними. Слоем нейронной сети называется множество нейронных элементов, на которые в каждый такт времени параллельно поступает информация от других нейронных элементов сети. Помимо слоев нейронов часто используется понятие входного распределительного слоя. Распределительный слой передает входные сигналы на первый обрабатывающий слой нейронных элементов (рис. 29.4).

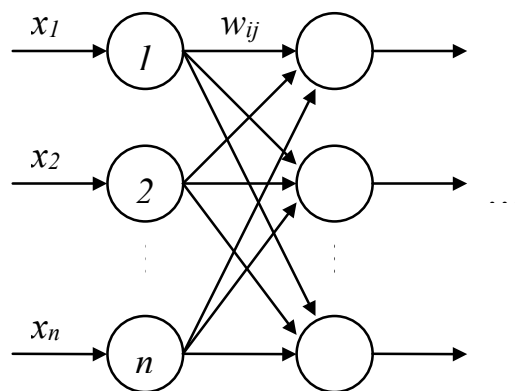


Рис. 29.4. Распределительный слой

У любой нейронной сети можно выделить 2 режима работы: обучение и воспроизведение. На этапе обучения настраиваются веса и пороги всех слоев. Воспроизведение является этапом обработки информации, следующим за обучением, при этом веса и пороги, как правило, не изменяются. Большое влияние на функционирование сети оказывает ее топология – архитектура слоев и связей между нейронами.

На способе обработки информации решающим образом сказывается наличие или отсутствие в сети обратных связей. Если обратных связей нет (каждый нейрон получает информацию только от нейронов предыдущих слоев), то обработка информации происходит в одном направлении за число тактов, равное числу слоев. При наличии обратных связей информация может проходить через сеть много раз до достижения какого-либо условия. В случае, если условие достигнуто, говорят, что сеть стабилизировалась. В общем случае сходимость не гарантируется. Тем не менее наличие обратных связей позволяет решать задачи с привлечением меньшего числа нейронов, что ускоряет процесс обучения.

Для лучшего понимания данного вопроса приведем одну из возможных классификаций НС в зависимости от различных характеристик:

1. По типу входной информации:
 - аналоговые НС (используют информацию в форме действительных чисел);
 - двоичные НС (оперируют с информацией, представленной в двоичном виде);
2. По характеру обучения:
 - с учителем (известно входное пространство решений НС);
 - без учителя (НС формирует выходное пространство решений только на основе входных воздействий – самоорганизующиеся сети);
3. По характеру настройки синапсов:
 - сети с фиксированными связями (весовые коэффициенты НС выбираются сразу, исходя из условия задачи);
 - сети с динамическими связями (в процессе обучения происходит настройка синаптических связей);
4. По методу обучения
 - НС с алгоритмом обратного распространения ошибки;
 - НС с конкурентным обучением;
 - НС, использующие правило Хебба;
 - НС с гибридным обучением, в которых используются различные алгоритмы обучения;
5. По характеру связей:
 - НС с прямыми связями;
 - НС с обратным распространением информации;
6. По архитектуре и обучению:
 - персептронные сети с прямыми связями;
 - самоорганизующиеся НС (НС Кохонена, НС адаптивного резонанса, рециркуляционные НС);

- НС с обратными связями (НС Хопфилда, НС Хэмминга, двунаправленная ассоциативная память, рекуррентные НС);
- гибридные НС (НС встречного распространения, НС с радиально-базисной функцией активации).

29.2. НЕЙРОННАЯ СЕТЬ ХОПФИЛДА

Сеть Хопфилда – однослойная, симметричная, нелинейная, автоассоциативная нейронная сеть, которая запоминает бинарные / биполярные образы. Сеть характеризуется наличием обратных связей. Топология сети Хопфилда показана на рис. 7.5. Информация с выхода каждого нейрона поступает на вход всех остальных нейронов. Образы для данной модификации сети Хопфилда кодируются биполярным вектором, состоящим из 1 и -1 .

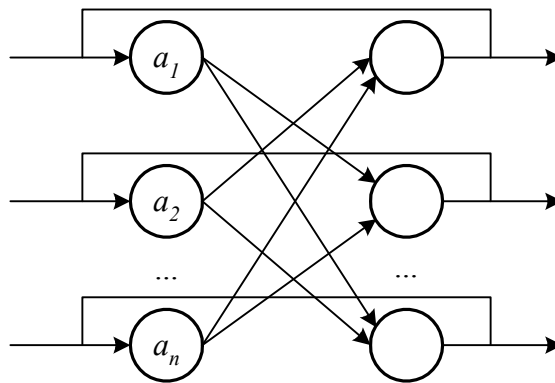


Рис. 7.5. Топология сети Хопфилда

Обучение. Обучение сети осуществляется в соответствии с соотношением

$$w_{ij} = \begin{cases} \sum_{k=1}^m a_i^k a_j^k, & i \neq j, \text{ для } i, j = \overline{1, n}, \\ 0, & i = j \end{cases} \quad (29.5)$$

где w_{ij} – вес связи от i -го нейрона к j -му;

n – количество нейронов в сети;

m – количество образов, используемых для обучения сети;

a_i^k – i -й элемент k -го образа из обучающей выборки.

Матрица весовых коэффициентов

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ \dots & \dots & \dots & \dots \\ w_{n1} & w_{n2} & \dots & w_{nn} \end{bmatrix}. \quad (29.6)$$

В качестве матрицы весовых коэффициентов Хопфилд использовал симметричную матрицу ($w_{ij}=w_{ji}$) с нулевой главной диагональю ($w_{ii}=0$). Последнее условие соответствует отсутствию обратной связи нейронного элемента на себя. В качестве функции активации нейронных элементов может использоваться как пороговая, так и непрерывная функция, например сигмоидная или гиперболический тангенс.

Будем рассматривать нейронную сеть Хопфилда с дискретным временем. Тогда при использовании пороговой функции активации она называется нейронной сетью с дискретным состоянием и временем. Нейронная сеть с непрерывной функцией активации называется нейронной сетью с непрерывным состоянием и дискретным временем. При использовании непрерывного времени модель Хопфилда называется непрерывной.

Для описания функционирования таких сетей Хопфилд использовал аппарат статистической физики. При этом каждый нейрон имеет два состояния активности $(1, -1)$, которые аналогичны значениям спина некоторой частицы. Весовой коэффициент w_{ji} можно интерпретировать как вклад поля j – частицы в величину потенциала i – частицы. Хопфилд показал, что поведение такой сети аналогично поведению лизингового спинового стекла. При этом он ввел понятие вычислительной энергии, которую можно интерпретировать в виде ландшафта с долинами и впадинами. Структура соединений сети определяет очертания ландшафта. Нейронная сеть выполняет вычисления, следуя по пути, уменьшающему вычислительную энергию сети. Это происходит до тех пор, пока путь не приведет на дно впадины. Данный процесс аналогичен скатыванию капли жидкости по склону, когда она минимизирует свою потенциальную энергию в поле тяготения. Впадины и долины в сети Хопфилда соответствуют наборам информации, которую хранит сеть. Если процесс начинается с приближенной или неполной информации, то он следует по пути, который ведет к ближайшей впадине. Это соответствует операции ассоциативного распознавания.

Матрица весов является диагонально симметричной, причем все диагональные элементы равны 0.

Воспроизведение. Нейронная сеть Хопфилда может функционировать синхронно и асинхронно. Для воспроизведения используется соотношение

$$a_i(t+1) = f\left(\sum_{j=1}^n w_{ij} a_j(t)\right), \quad (29.7)$$

где $a_j(t)$ – выход j -го нейрона в момент времени t , а f – бинарная / биполярная функция активации;

$$f(x) = \begin{cases} 1 & x > 0, \\ -1 & x \leq 0. \end{cases} \quad (29.8)$$

При работе в синхронном режиме на один такт работы сети все нейроны одновременно меняют состояние. В случае асинхронной работы состояние меняет только один случайный нейрон. Итерации продолжаются до тех пор, пока сеть не придет в стабильное состояние.

Во время воспроизведения исходным вектором $a(0)$ является некоторый тестовый образ, не совпадающий с образами из обучающей выборки. В процессе функционирования по формуле (29.8) сеть должна прийти в состояние, соответствующее образу из обучающей выборки, наиболее похожему на тестовый.

Максимальное количество образов, которое можно запомнить в матрице W не превышает

$$m = \frac{n}{2 \ln n + \ln \ln n}, \quad (29.9)$$

где n – количество нейронов, что следует отнести к недостаткам этой сети.

29.3. МНОГОСЛОЙНЫЙ ПЕРСЕПТРОН

Многослойный персептрон является сетью с прямым распространением сигнала (без обратных связей), обучаемой с учителем. Такая сеть способна аппроксимировать любую непрерывную функцию или границу между классами со сколь угодно высокой точностью. Для этого достаточно одного скрытого слоя нейронов с сигмоидной функцией активации (29.2) – (29.4), т.е. многослойный персептрон обычно состоит из 3

слоев: первого распределительного, второго скрытого и третьего выходного (рис. 29.6).

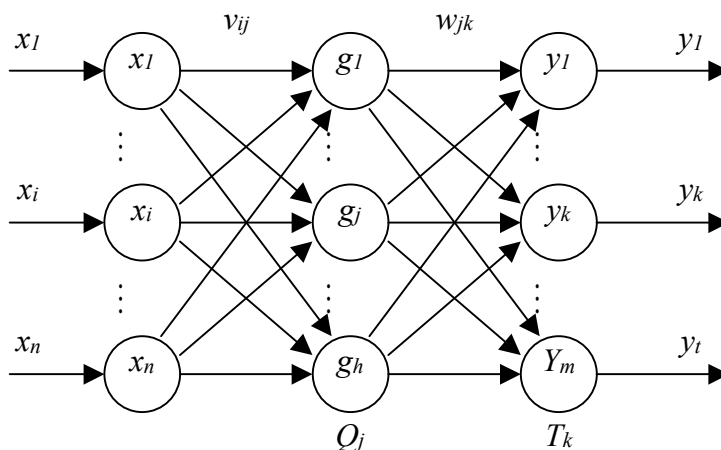


Рис. 29.6. Многослойный персептрон

Такая сеть имеет n входов и n нейронов распределительного слоя, h нейронов скрытого слоя и m выходных нейронов. Используются две матрицы весов: скрытого слоя v размером $n \times h$ и выходного слоя w размером $h \times m$. Кроме этого, с каждым слоем нейронов связан массив порогов: Q – для скрытого слоя, T – для выходного. Эти данные представляют собой знания сети, настраиваемые в процессе обучения и определяющие ее поведение. Персептрон функционирует по следующим формулам:

$$g_j = f\left(\sum_{i=1}^n v_{ij} x_i + Q_j\right), \quad (29.10)$$

$$y_k = f\left(\sum_{j=1}^h w_{jk} g_j + T_k\right). \quad (29.11)$$

В качестве функции активации используется одна из функций (29.2) – (29.4). Вид функции определяет диапазон чисел, в котором работает сеть. В дальнейшем будет использоваться сигмоидная функция (29.2), имеющая область значений от 0 до 1.

Обучение с учителем ставит перед сетью задачу обобщить p примеров, заданных парами векторов (x^r, y^r) , $r = \overline{1, p}$. Вектор $x^r = (x_1^r, x_2^r, \dots, x_l^r, \dots, x_n^r)$ в случае задачи классификации задает входной образ (вектор признаков), а вектор $y^r = (y_1^r, y_2^r, \dots, y_k^r, \dots, y_m^r)$, задающий эталонный выход, должен кодировать номер класса. При этом есть множество вариантов кодирования. Оптимальным представляется кодирование, когда номер класса определяется позицией единичной компоненты в векторе y^r , а все остальные компоненты равны 0. Каждый выходной нейрон соответствует одному классу. Такой способ позволяет при классификации определять вероятность каждого класса по величине на выходе соответствующего нейрона (чем ближе к единице, тем вероятность больше).

Обучение персептрона проводится с помощью алгоритма обратного распространения ошибки, который минимизирует среднеквадратичную ошибку нейронной сети. Для этого с целью настройки синаптических связей используется метод градиентного спуска в пространстве весовых коэффициентов и порогов нейронной сети. Рассмотрим алгоритм обратного распространения ошибки.

1. На первом этапе происходит начальная инициализация знаний сети. Простейший вариант такой инициализации – присвоить всем весам и порогам случайные значения из диапазона $[-1, 1]$.

2. Далее для каждой пары векторов (x^r, y^r) выполняется следующее:

- 2.1. Для входного вектора рассчитываются выходы нейронов скрытого слоя и выходы сети по формулам (29.10), (29.11).

2.2. Происходит коррекция знаний сети, при этом главное значение имеет отклонение реально полученного выхода сети y от идеального вектора y^r . Согласно методу градиентного спуска, изменение весовых коэффициентов и порогов нейронной сети происходит по следующим формулам:

$$w_{jk}(t+1) = w_{jk}(t) + \alpha \frac{\partial E}{\partial w_{jk}(t)}, \quad (29.12)$$

$$T_k(t+1) = T_k(t) + \alpha \frac{\partial E}{\partial T_k(t)}, \quad (29.13)$$

где E - среднеквадратичная ошибка нейронной сети для одного образа, а α – параметр, определяющий скорость обучения. Формулы записаны в терминах выходного слоя, аналогично выглядят формулы для скрытого слоя. Среднеквадратичная ошибка сети вычисляется как

$$E = \frac{1}{2} \sum_{k=1}^m (y_k^r - y_k)^2. \quad (29.14)$$

Ошибка k -го нейрона выходного слоя определяется как

$$d_k = \frac{\partial E}{\partial y_k} = y_k^r - y_k. \quad (29.15)$$

Выразим производные из формул (29.12), (29.13) через легко вычисляемые величины. Определим взвешенную сумму, аргумент функции активации как

$$S_k = \sum_{j=1}^h w_{jk} g_j + T_k. \quad (29.16)$$

$\frac{\partial E}{\partial w_{jk}}$ можно представить как:

$$\frac{\partial E}{\partial w_{jk}} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial S_k} \frac{\partial S_k}{\partial w_{jk}}, \quad (29.17)$$

где $\frac{\partial E}{\partial y_k} = d_k = y_k^r - y_k$ – ошибка k -го нейрона; $\frac{\partial y_k}{\partial S_k} = f'(S_k)$ – производная функции активации; $\frac{\partial S_k}{\partial w_{jk}} = g_j$ – значение j -го нейрона предыдущего слоя.

Получаем

$$\frac{\partial E}{\partial w_{jk}} = d_k f'(S_k) g_j. \quad (29.18)$$

Аналогично, с учетом того, что $\frac{\partial S_k}{\partial T_k} = 1$, получаем

$$\frac{\partial E}{\partial T_k} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial S_k} \frac{\partial S_k}{\partial T_k} = d_k f'(S_k). \quad (29.19)$$

Веса и пороги скрытого слоя также корректируются по формулам, аналогичным (29.12), (29.13), с учетом (29.18) и (29.19). При этом главной трудностью является определение ошибки нейрона скрытого слоя. Эту ошибку явно определить по формуле, аналогичной (29.15) нельзя, однако существует возможность рассчитать ее через ошибки нейронов выходного слоя (отсюда произошло название алгоритма обратного распространения ошибки):

$$e_j = \frac{\partial E}{\partial g_j} = \sum_{k=1}^m \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial S_k} \frac{\partial S_k}{\partial g_j} = \sum_{k=1}^m d_k f'(S_k) w_{jk} . \quad (29.20)$$

Производные от функций активации тоже легко рассчитываются, например, для сигмоидной функции получаем

$$f'(S_k) = \left(\frac{1}{(1 + e^{-S})} \right)' = f(S_k)(1 - f(S_k)) = y_k(1 - y_k) . \quad (29.21)$$

Аналогично:

$$(th(S_k))' = 1 - y_k^2 ;$$

$$\left(\frac{2}{(1 + e^{-x})} - 1 \right)' = \frac{(1 - y_k^2)}{2} .$$

Таким образом, можно записать окончательные выражения (29.12), (29.13) для двух слоев, использующих сигмоидную функцию:

$$w_{jk} = w_{jk} + \alpha y_k (1 - y_k) d_k g_j , \quad (29.22)$$

$$T_k = T_k + \alpha y_k (1 - y_k) d_k , \quad (29.23)$$

$$v_{ij} = v_{ij} + \beta g_j (1 - g_j) e_j x_i , \quad (29.24)$$

$$Q_j = Q_j + \beta g_j (1 - g_j) e_j . \quad (29.25)$$

В эти формулы вводится дополнительный параметр – β – скорость обучения скрытого слоя, который может отличаться от аналогичного параметра для выходного слоя. Рекомендуется изменять скорости обучения обратно пропорционально количеству шагов 2 алгоритма обучения, однако это не всегда оправдывает себя на практике.

3. После того, как коррекция знаний произведена для каждой пары векторов, можно оценить степень успешности обучения сети для определения момента завершения алгоритма. Для этого можно использовать в качестве критерия максимальную по модулю ошибку на выходе d_k , полученную на шаге 2. Условием прекращения обучения в этом случае будет

$$\max |d_k| < D, k = \overline{1, m}, r = \overline{1, p}, \quad (29.26)$$

где D – достаточно маленькая константа – величина максимальной ошибки, которую требуется достичь в процессе обучения. Если условие (29.26) не выполняется, то шаг 2 повторяется.

Способность персептрона-классификатора разделять образы в пространстве признаков прежде всего зависит от его скрытого слоя. Именно на этот слой возлагается задача сделать множество классов линейно разделимым для успешной работы выходного слоя. Очевидно, что чем больше нейронов в скрытом слое, тем большее количество примеров этот слой может разделять. Кроме этого, увеличение числа признаков входных образов также способствует успешному их разделению в пространстве признаков. Однако увеличение этих параметров приводит к росту ошибок сети и времени обучения. Увеличение размерности входов n приводит к росту ошибки аппроксимации сети, возникающей из-за обобщения данных. Увеличение числа нейронов h скрытого слоя приводит к росту ошибки, связанной со сложностью модели. Персептрону легче провести функцию через эталонные точки, однако при этом обобщающая способность сети ухудшается. Он хуже предсказывает поведение аппроксимируемой функции

на образах, не входящих в обучающую выборку. Такое состояние сети называется переобучением.

Оптимальное соотношение между этими параметрами оценивают как

$$h \sim \sqrt{\frac{p}{n}}. \quad (29.27)$$

Эксперименты показывают, что обучение максимально успешно проходит на множестве классов, хорошо (желательно линейно) разделенных в пространстве признаков. Это достигается удачным подбором информативных признаков. Если классы в пространстве признаков хорошо кластеризуются, т.е. образы каждого класса составляют компактную группу, достаточно удаленную от других групп, то есть возможность уменьшить размер обучающей выборки p (используются только центры кластеров) и затем уменьшить число нейронов h . Это приводит к ускорению обучения и улучшает работу классификатора.

Другой проблемой является то, что алгоритм градиентного спуска не гарантирует нахождение глобального минимума среднеквадратичной ошибки сети (7.14), а гарантируется определение только локального минимума. Проблемы, возникающие в процессе градиентного спуска, можно проанализировать на примере функции ошибки, схематически изображенной на рис. 29.7.

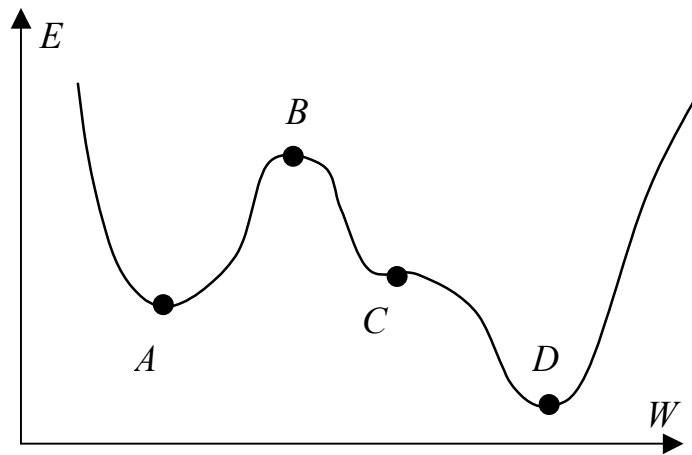


Рис. 29.7. Схематическая функция ошибки сети

На рис. 29.7 показаны четыре критические точки, производная функции ошибки в которых близка к нулю. Точка A соответствует локальному минимуму. Признаком достижения локального минимума в процессе обучения является полное прекращение уменьшения ошибки (29.26). В этом случае может помочь повторное обучение с другим начальным распределением знаний (тут может помочь случайная инициализация). Точка B – локальный максимум. В случае попадания в окрестность такой точки скорость резко падает, затем снова быстро растет. Не существует способа предсказать, в какую сторону лучше двигаться из точки с производной, близкой к нулю. Точка C – точка перегиба, характеризуется длительным уменьшением скорости. Точка D – глобальный минимум – цель алгоритма.

Существуют многочисленные способы оптимизации метода градиентного спуска, призванные улучшить поведение алгоритма в подобных критических точках. Эффективной модификацией является введение момента, накапливающего влияние градиента на веса со временем. Тогда величина Δw (29.12) в момент времени t будет вычисляться как

$$\Delta w(t) = \alpha \frac{\partial E}{\partial w} + \mu \Delta w(t-1), \quad (29.28)$$

где μ - параметр, определяющий величину влияния момента. С использованием (29.28) скорость изменения весов возрастает на участках с постоянным знаком производной. В окрестностях минимума скорость резко падает за счет колебания знака.

Достоинства алгоритма – большая скорость в точках перегиба, возможность по инерции преодолевать небольшие локальные минимумы. Недостатки – еще один параметр, величину которого следует подбирать и настраивать.

Этот и другие алгоритмы оптимизации обучения персептрона позволяют улучшить работу сети в условиях плохой сходимости. Однако они усложняют процесс обучения, не гарантируя в то же время полного успеха во всех случаях. Успех обучения классификатора зависит от самого алгоритма обучения и качества обучающей выборки.

29.4. СЕТЬ РБФ

Сеть РБФ (радиальная базисная функция) является аналогом многослойного персептрона (рис. 29.8). Скорость обучения такой сети гораздо выше, причем допускается полностью аналитический подход к расчету весовых коэффициентов. Однако эти положительные моменты сопровождаются рядом недостатков, главным из которых является ухудшение точности аппроксимации. Сеть обладает хорошей обобщающей способностью только для ограниченного класса аппроксимируемых функций. В качестве классификатора такая сеть может с успехом применяться в случае хорошей кластеризации классов в пространстве признаков.

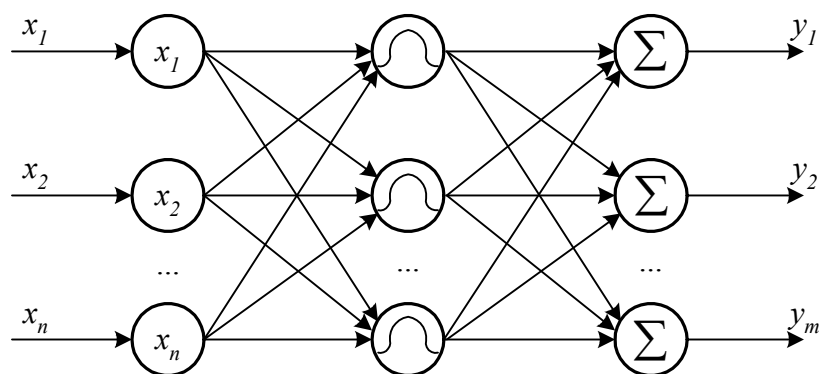


Рис. 29.8 Топология сети РБФ

Упрощение работы и обучения достигается за счет введения вместо скрытого слоя нейронов слоя РБФ ячеек. Классический закон, по которому такая ячейка функционирует, определяется формулой гауссова колокола:

$$g_j = \exp\left(\frac{-|x - t^j|^2}{\sigma_j^2}\right), \quad (29.29)$$

где x – входной вектор; t^j – вектор, определяющий математическое ожидание (центр кластера в пространстве признаков) РБФ ячейки; σ_j – среднеквадратическое отклонение или параметр, зависящий от величины разброса образов класса от его центра (рис. 29.8). В данном выражении евклидово расстояние между векторами x и t^j вычисляется как

$$|x - t^j|^2 = (x_1 - t_1^j)^2 + (x_2 - t_2^j)^2 + \dots + (x_n - t_n^j)^2.$$

Обучение. РБФ ячейки обучаются путем подбора центра и отклонения каждой из них. Для классификатора в качестве центра выбирается центр кластера в пространстве признаков, компактно содержащего образы одного и того же класса. В простейшем случае, если класс задается одним идеальным

образом, этот образ и будет являться вектором t – центром РБФ ячейки. Параметр разброса каждой ячейки выбирается в зависимости от величины радиуса кластера или расстояния до соседних центров. Ряд авторов рекомендует выбирать σ как половину расстояния до ближайшего центра ячейки, соответствующей другому классу. Количество РБФ ячеек выбирается таким образом, чтобы покрыть гауссовыми колоколами все классы.

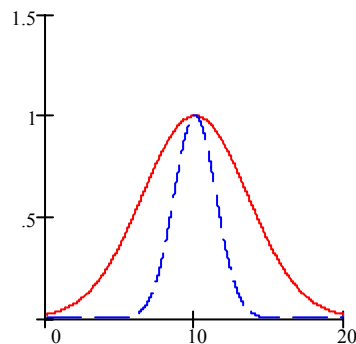


Рис. 7.9. Примеры функций РБФ ячеек с одинаковым центром и разным разбросом

Выходной слой РБФ сети обычно состоит из суммирующих ячеек

$$y_k = \sum_{j=1}^h w_{jk} g_j. \quad (29.30)$$

Это позволяет при определенных условиях использовать систему линейных уравнений для определения весов выходного слоя. В других обстоятельствах можно использовать алгоритм градиентного спуска для настройки весов выходного слоя (функция активации в данном случае линейная). С учетом того, что обучается только выходной слой нейронной сети, а скрытый уже настроен, обучение такой сети проходит на порядок быстрее, чем аналогичного многослойного персептрона.

Формула (29.22) упрощается до

$$w_{jk} = w_{jk} + \alpha d_k g_j, \quad (29.31)$$

поскольку функция активации в выходном слое сети РБФ линейная и ее производная равна 1. Для коррекции весовых коэффициентов выходных суммирующих ячеек используется только (29.31).

Контроль завершения алгоритма обучения производится аналогично.

Воспроизведение. Сеть функционирует по формулам (29.29) и (29.30).

Очевидно, что функция на выходе РБФ сети будет представлять собой суперпозицию гауссовых колоколов. В этом заключается ограничение данного класса классификаторов. Кроме этого, при неудачном выборе признаков большой проблемой является выбор количества РБФ ячеек, определение их центров и отклонения. С уменьшением числа РБФ ячеек улучшается обобщение данных сетью, но могут проявляться большие ошибки в эталонных точках.

29.5. КОНКУРЕНТНАЯ НЕЙРОННАЯ СЕТЬ

Самоорганизующиеся нейронные сети обучаются без учителя. Они способны адаптироваться к входным данным, используя содержащиеся в этих данных зависимости. Такие сети используются для нахождения более компактного описания данных (сжатия), кластеризации, выделения признаков.

Конкурентная сеть является простейшей самоорганизующейся нейронной сетью (рис. 29.10).

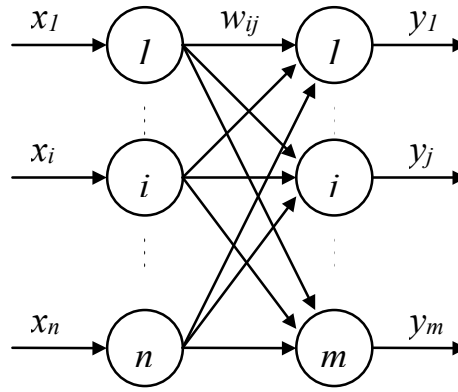


Рис. 29.10. Конкурентная нейронная сеть

Первый слой является распределительным. Нейроны второго слоя функционируют по формуле

$$y_j = \sum_{i=1}^n w_{ij} x_i = |w^j| |x| \cos \alpha, \quad (29.32)$$

где $x = (x_1, x_2, \dots, x_i, \dots, x_n)$ – входной вектор; $w^j = (w_{1j}, w_{2j}, \dots, w_{ij}, \dots, w_{nj})$ – вектор весовых коэффициентов нейрона, а $|x|$ и $|w^j|$ – их модули, α – угол между ними.

Обучение. При обучении нейронной сети при подаче каждого входного вектора определяется нейрон-победитель, для которого (29.32) максимально. Для этого нейрона синаптические связи усиливаются по формуле

$$w_{ij}(t+1) = w_{ij}(t) + \beta(x_i - w_{ij}(t)), \quad (29.33)$$

где β – скорость обучения.

Смысл этой формулы в том, что вектор весовых коэффициентов нейрона – победителя “поворачивается” в сторону входного вектора, тем самым активность нейрона усиливается. Удобно работать с нормированными

входными и весовыми векторами, когда их модуль равен 1. Нормировка уравнивает шансы в конкуренции нейронов с разным модулем вектора весовых коэффициентов. Выражение (29.32) для нормированных векторов будет выглядеть как

$$y_j = \sum_{i=1}^n w_{ij} x_i = \cos \alpha, \quad (29.34)$$

а выражение (7.33):

$$w_{ij}(t+1) = \frac{w_{ij}(t) + \beta(x_i - w_{ij}(t))}{|w^j(t) + \beta(x - w^j(t))|}. \quad (29.35)$$

Случайное начальное распределение весовых коэффициентов может привести к тому, что некоторые нейроны никогда не станут победителями, так как их весовые векторы окажутся удаленными от всех входных векторов. Существует ряд модификаций алгоритма обучения, позволяющих устранить этот недостаток. Хорошие результаты на практике показало частотно-зависимое конкурентное обучение. Согласно нему, нейрон-победитель определяется по минимуму произведения евклидова расстояния между входным и весовым вектором и количеством побед данного нейрона f_j :

$$d_v = \min_j (|x - w^j| f_j). \quad (29.36)$$

Шансы нейрона на победу уменьшаются с количеством побед, что дает преимущество другим нейронам.

Конкурентное обучение продолжается до тех пор, пока максимум евклидова расстояния между любым входным вектором и

соответствующим ему вектором весов нейрона-победителя не достигнет заданного малого значения.

Конкурентная сеть позволяет разбить входную выборку нормированных векторов на m (количество выходных нейронов сети) кластеров, расположенных на поверхности гиперсферы в пространстве признаков единичного радиуса. Входные векторы, приводящие к победе одного и того же нейрона, относят к одному кластеру.

Кохонен предложил внести в правило конкурентного обучения (29.33) информацию о расположении нейронов в выходном слое. Для этого нейроны упорядочиваются в одномерные или двухмерные решетки. Вводится функция, корректирующая изменение весов в зависимости от расстояния до нейрона-победителя $h(t, k, j)$ – сила влияния между нейроном-победителем k и нейроном j в момент времени t . Для $j=k$ эта функция всегда равна 1 и уменьшается с ростом расстояния между k и j в решетке. С течением времени радиус влияния обычно сужается. С использованием этой функции веса меняются для всех нейронов сети, а не только для нейрона-победителя:

$$w_{ij}(t+1) = w_{ij}(t) + \beta h(t, k, j)(x_i - w_{ij}(t)). \quad (29.37)$$

В качестве функции $h(t, k, j)$ может использоваться гауссовый колокол с параметром σ , зависящим от времени или функция вида “мексиканская шляпа”.

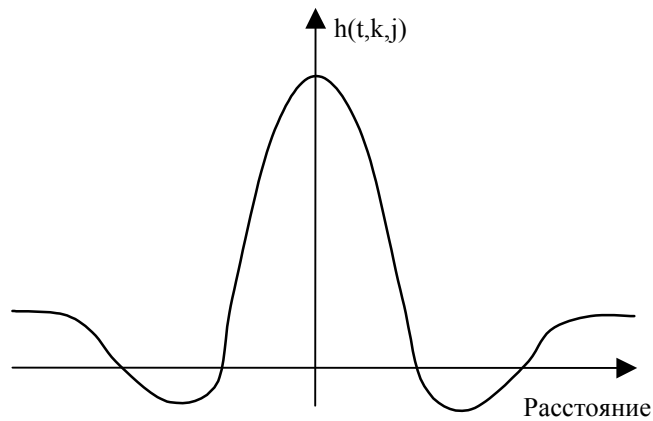


Рис. 29.11. Пример функции расстояния в сетях Кохонена

В результате модификации конкурентного обучения сеть Кохонена не только кластеризует входные примеры, но и упорядочивает их в виде одномерной или двухмерной решетки. Это позволяет получить дополнительную информацию о близости кластеров. Если два кластера проецируются на соседние нейроны решетки, это говорит об их близости в исходном пространстве признаков. Обратное неверно. Из-за уменьшения размерности пространства отношение близости сохраняется только для ограниченного числа кластеров.

ЛИТЕРАТУРА

1. У. Прэтт. Цифровая обработка изображений. В 2-х томах. М.: Мир, 1982.
2. Гонсалес Р., Вудс Р. Цифровая обработка изображений. - М.: Техносфера, 2005.
3. Сэломон Д. Сжатие данных, изображений и звука: Учеб. пособие для вузов. - М.: Техносфера, 2004.
4. Уэлстид С. Фракталы и вейвлеты для сжатия изображений в действии: Учеб. пособие. - М.: Триумф. 2003.
5. Бутаков Е.А. и др. Обработка изображений на ЭВМ. М.: Радио и связь, 1987.
6. СБИС для распознавания образов и обработки изображений. М.: Мир, 1988.
7. К. Фу. Последовательные методы в распознавании образов и обучении машин. М.: Наука, 1971.
8. Р. Дуда, П. Харт. Распознавание образов и анализ сцен. М.: Мир, 1976.
9. Павлидис Т. Алгоритмы машинной графики и обработки изображений. М.: Радио и связь, 1986.
10. Мартинес Ф. Синтез изображений. Принципы, аппаратное и программное обеспечение. М.: Радио и связь, 1990.
11. Ярославский Л.П. Введение в цифровую обработку изображений. М.: Радио и связь, 1979.
12. Верхаген К., Дейн Р., Грун Ф. Распознавание образов: состояние и перспектива. М.: Радио и связь, 1982.
13. Горелик А.Л., Гуревич И.Б., Скрипник В.А. Современное состояние проблемы распознавания. М.: Радио и связь, 1985.
14. Дж. Бендат, А. Пирсол. Прикладной анализ случайных данных. М.: Мир, 1989.
15. Гольдберг Л.М., Матюшкин Б.Д., Поляк М.Н. Цифровая обработка сигналов. М.: Радио и связь, 1990.
16. Горелик А.Л., Скрипник В.А. Методы распознавания. М.: Высшая школа, 1989.
17. Q. Tian, Y.Faiman and S.H. Lee. Comparison of statistical pattern recognition algorithms for hybrid processing, 1 Linear mapping algorithms// Journal of the optical society of America, V.5 №10, 1988, p.1655-1669.
18. O. Tian, V. Faiman and S.Y. Lee. Comparison of statistical pattern recognition algorithms for hybrid processing. 2. Statistical pattern

- recognition algorithms// Journal of the optical society of America, v.5, №10, 1988, pp.1670-1680.
19. Кун С. Матричные методы обработки в СБИС. М. «Мир», 1990.
 20. Соколов Е.Н., Вайткявичюс Г.Г. Нейроинтеллект: от нейрона к нейрокомпьютеру. М.: Наука, 1989.
 21. Ярославский Л.П. Цифровая голография. М.: Наука, 1992.
 22. Василенко Г.И., Тараторин А.М. Восстановление изображений. М.: Радио и связь, 1986.
 23. Фу К. Структурные методы распознавания образов. М.: Мир, 1974.
 24. Фоли Дж., Дэм А. Основы интерактивной машинной графики в 2-х томах. М.: Мир, 1985.
 25. Садыхов Р.Х., Чеголин П.М., Шмерко В.П. Методы и средства обработки сигналов в дискретных базисах. Минск, Наука и техника, 1987.
 26. В.А. Головкин. Нейронные сети: обучение, организация и применение. Москва, 2001.