Глава 3
ГИБРИДНЫЕ СЕТИ

Каждая разновидность систем искусственного интеллекта имеет свои особенности, например, по возможностям обучения, обобщения и выработки выводов, что делает ее наиболее пригодной для решения одного класса задач и менее пригодной — для другого.

Например, нейронные сети хороши для задач распознавания образов, но весьма неудобны для выяснения вопроса, *как они такое распознавание осуществляют.* Они могут автоматически приобретать знания, но процесс их обучения зачастую происходит достаточно медленно, а анализ обученной сети весьма сложен (обученная сеть обычно — черный ящик для пользователя). При этом какую-либо априорную информацию (знания эксперта) для ускорения процесса ее обучения в нейронную сеть ввести невозможно.

Системы с нечеткой логикой, напротив, хороши для объяснения получаемых с их помощью выводов, но *они не могут автоматически приобретать* знания для использования их в механизмах выводов. Необходимость разбиения универсальных множеств на отдельные области, как правило, ограничивает количество входных переменных в таких системах небольшим значением.

Вообще говоря, теоретически, системы с нечеткой логикой и искусственные нейронные сети эквивалентны друг другу, однако, в соответствии с изложенным выше, на практике у них имеются свои собственные достоинства и недостатки. Данное соображение легло в основу аппарата гибридных сетей, в которых выводы делаются на основе аппарата нечеткой логики, но соответствующие функции принадлежности подстраиваются с использованием алгоритмов обучения нейронных сетей, например, алгоритма обратного распространения ошибки. Такие системы не только используют априорную информацию, но могут приобретать новые знания и для пользователя являются логически прозрачными.

3.1. Основные понятия и определения гибридных сетей

Для пояснения сущности гибридных сетей, рассмотрим еще раз простую нейронную сеть, имеющую два входа и только один нейрон (рис. 3.1).

Здесь входные сигналы x_i «взаимодействуют» с весами w_i, образуя произведения

$$p_i = x_i w_i, \quad i = 1, 2.$$

Такая частная информация (произведения) объединяются с использованием операции суммирования, образуя вход net нейрона:

$$\mathrm{net} = p_1 + p_2 = w_1 x_1 + w_2 x_2.$$

Выход нейрона образуется в результате преобразования входа net некоторой активационной функцией:

$$y = f(\mathrm{net}) = f(w_1 x_1 + w_2 x_2),$$

например, сигмоидного типа

$$f(x) = \frac{1}{1 + e^{-x}}.$$

Приведенную однонейронную сеть, в которой используются операции умножения, суммирования и сигмоидная функция активации, будем называть стандартной нейронной сетью.

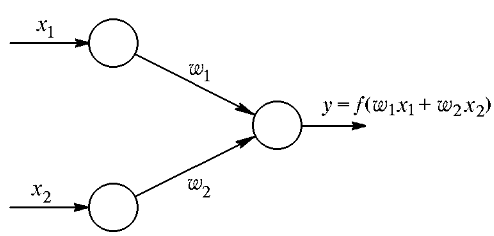

Рис. 3.1. Элементарная НС

В случае применения других операций, таких как *t-норма* или *t-конорма* (см. гл. 1), придем к нейронной сети, которая будет называться *гибридной*.

Определение. Гибридная нейронная сеть — это нейронная сеть с четкими сигналами, весами и активационной функцией, но с объединением x_i и w_i, p_1 и p_2 с использованием *t-нормы*, *t-конормы* или некоторых других непрерывных операций.

Входы, выходы и веса гибридной нейронной сети — вещественные числа, принадлежащие отрезку $[0, 1]$.

Рассмотрим следующие примеры элементарных гибридных нейронных сетей.

Нечеткий нейрон «И». Сигналы x_i и веса w_i в данном случае объединяются с помощью треугольной конормы:

$$p_i = \mathrm{S}(w_i,\, x_i), \quad i = 1,\, 2,$$

а выход образуется с применением треугольной нормы (рис. 3.2):

$$y = \mathrm{AND}(p_1,\, p_2) = \mathrm{T}(p_1,\, p_2) = \mathrm{T}\big(\mathrm{S}(w_1,\, x_1),\, \mathrm{S}(w_2,\, x_2)\big).$$

Если принять

$$\mathrm{T} = \min, \quad \mathrm{S} = \max,$$

тогда нечеткий нейрон «И» реализует композицию min-max:

$$y = \min\,(w_1 \vee x_1,\, w_2 \vee x_2).$$

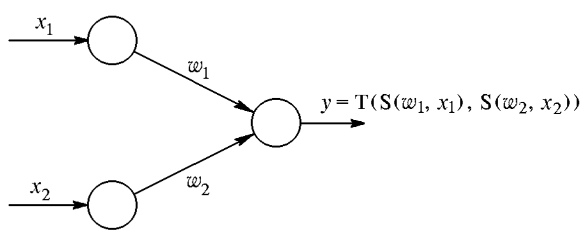

Рис. 3.2. Структура гибридного нейрона «И»

Нечеткий нейрон «ИЛИ». Сигналы x_i и веса w_i здесь объединяются с помощью треугольной нормы:

$$p_i = \mathrm{T}(w_i,\, x_i), \quad i = 1,\, 2,$$

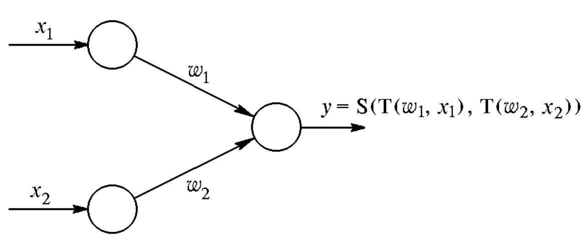

Рис. 3.3. Нечеткий нейрон «ИЛИ»

а выход образуется с применением треугольной конормы (см. рис. 3.3):

$$y = \mathrm{OR}(p_1,\, p_2) = \mathrm{S}(p_1,\, p_2) = \mathrm{S}\big(\mathrm{T}(w_1,\, x_1),\, \mathrm{T}(w_2,\, x_2)\big).$$

Если принять

$$\mathrm{T} = \min, \quad \mathrm{S} = \max,$$

тогда нечеткий нейрон «ИЛИ» реализует композицию max-min:

$$y = \max\,(w_1 \wedge x_1,\ w_2 \wedge x_2).$$

3.2. Алгоритмы обучения и использования гибридных сетей

Опишем типовой подход к построению алгоритмов обучения и использования гибридных нейронных сетей.

Предположим, что гибридной сетью должно быть реализовано (неизвестное) отображение

$$y^k = f(\mathbf{x}^k) = f(x_1^k,\ x_2^k,\ \ldots,\ x_n^k), \quad k = 1,\ 2,\ \ldots,\ N,$$

при наличии обучающего множества

$$\{(\mathbf{x}^1,\ y^1),\ \ldots,\ (\mathbf{x}^N,\ y^N)\}.$$

Для моделирования неизвестного отображения f используем ранее рассмотренный упрощенный алгоритм нечеткого вывода, применяя следующую форму записи предикатных правил:

Π_i: если x_1 есть A_{i1} и x_2 есть A_{i2} и ... и x_n есть A_{in}, тогда $y = z_i$, $i = 1,\ 2,\ \ldots,\ m$,

где A_{ij} — нечеткие числа треугольной формы, z_i — вещественные числа, определяя степень истинности i-го правила с помощью операции умножения (Larsen):

$$\alpha_i = \prod_{j=1}^{n} \mathrm{A}_{ij}(x_j^k)$$

(здесь можно использовать и другие представления для моделирования логического оператора «И») и определяя выход нечеткой системы дискретным аналогом центроидного метода:

$$\mathrm{o}^k = \frac{\sum\limits_{i=1}^{m} \alpha_i z_i}{\sum\limits_{i=1}^{m} \alpha_i}.$$

Введение функции ошибки для k-го предъявленного образца вида

$$E_k = \frac{1}{2}(\mathrm{o}^k - y^k)^2$$

позволяет, далее, как в обычных (стандартных) нейронных сетях использовать градиентный метод для подстройки параметров заданных предикатных правил. Так, величины z_i можно корректировать по соотношению

$$z_i := z_i - \eta \frac{\partial E_k}{\partial z_i} = z_i - \eta(\mathrm{o}^k - y^k)\frac{\alpha_i}{\alpha_1 + \alpha_2 + \ldots + \alpha_m},$$
$$i = 1,\, 2,\, \ldots,\, m,$$

где η, как и раньше, — константа, характеризующая скорость обучения.

Более детально алгоритм настройки рассмотрим на примере системы, включающей два правила:

Π_1: если x есть A_1, тогда $y = z_1$,

Π_2: если x есть A_2, тогда $y = z_2$,

при этом предполагается, что нечеткие понятия A_1 («малый») и A_2 («большой») имеют сигмоидные функции принадлежности

$$\mathrm{A}_1(x) = \frac{1}{1 + e^{b_1(x-a_1)}}, \quad \mathrm{A}_2(x) = \frac{1}{1 + e^{b_2(x-a_2)}},$$

характеризующиеся параметрами a_1, a_2, b_1, b_2.

Степени истинности правил определяются в данном случае соотношениями

$$\alpha_1 = \mathrm{A}_1(x) = \frac{1}{1 + e^{b_1(x-a_1)}}, \quad \alpha_2 = \mathrm{A}_2(x) = \frac{1}{1 + e^{b_2(x-a_2)}},$$

а выход системы — выражением

$$\mathrm{o} = \frac{\alpha_1 z_1 + \alpha_2 z_2}{\alpha_1 + \alpha_2} = \frac{\mathrm{A}_1(x) z_1 + \mathrm{A}_2(x) z_2}{\mathrm{A}_1(x) + \mathrm{A}_2(x)}.$$

Предположим, что имеется обучающее множество $\{(x_1, y_1), \ldots$ $\ldots, (x^N, y^N)\}$, отображающее неизвестную функцию f.

Требуется: осуществить такую настройку параметров системы a_1, a_2, b_1, b_2, z_1, z_2, при которой обеспечивается наилучшая аппроксимация данной функции.

Решение. В данном случае функция ошибки может быть записана в форме

$$E_k = E_k(a_1, b_1, a_2, b_2, z_1, z_2) = \frac{1}{2}\big(\mathrm{o}^k(a_1, b_1, a_2, b_2, z_1, z_2) - y^k\big)^2.$$

Используя далее тот же подход, что и при выводе алгоритма обратного распространения ошибки (см. § 2.5), запишем:

$$z_1 := z_1 - \eta \frac{\partial E_k}{\partial z_1} = z_1 - \eta(\mathrm{o}^k - y^k) \frac{\alpha_1}{\alpha_1 + \alpha_2} = \\ = z_1 - \eta(\mathrm{o}^k - y^k) \frac{\mathrm{A}_1(x^k)}{\mathrm{A}_1(x^k) + \mathrm{A}_2(x^k)},$$

$$z_2 := z_2 - \eta \frac{\partial E_k}{\partial z_2} = z_2 - \eta(\mathrm{o}^k - y^k) \frac{\alpha_2}{\alpha_1 + \alpha_2} = \\ = z_2 - \eta(\mathrm{o}^k - y^k) \frac{\mathrm{A}_2(x^k)}{\mathrm{A}_1(x^k) + \mathrm{A}_2(x^k)}.$$

Аналогичным путем могут быть получены развернутые выражения для коррекции коэффициентов a_1, a_2, b_1, b_2. Исходные

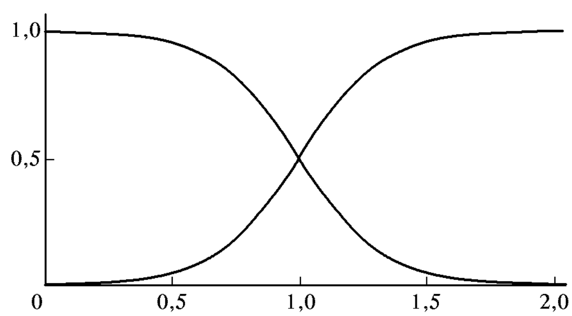

Рис. 3.4. Симметричные функции принадлежности

соотношения таковы:

$$a_1 := a_1 - \eta \frac{\partial E_k}{\partial a_1}, \quad a_2 := a_2 - \eta \frac{\partial E_k}{\partial a_2},$$

$$b_1 := b_1 - \eta \frac{\partial E_k}{\partial b_1}, \quad b_2 := b_2 - \eta \frac{\partial E_k}{\partial b_2}.$$

Конечные выражения являются достаточно громоздкими, но могут быть упрощены в случае, если функции принадлежности имеют вид

$$\mathrm{A}_1(x) = \frac{1}{1 + e^{-b(x-a)}}, \quad \mathrm{A}_2(x) = \frac{1}{1 + e^{b(x-a)}}.$$

Данные функции характеризуются всего двумя параметрами (a и b), в определенном смысле являются симметричными (см. рис. 3.4) и удовлетворяют уравнению

$$\mathrm{A}_1(x) + \mathrm{A}_2(x) = 1.$$

Заметим, что из последнего и ранее полученных уравнений следует:

$$z_1 := z_1 - \eta \frac{\partial E_k}{\partial z_1} = z_1 - \eta(\mathrm{o}^k - y^k)\frac{\alpha_1}{\alpha_1 + \alpha_2} = \\ = z_1 - \eta(\mathrm{o}^k - y^k)\mathrm{A}_1(x^k),$$

$$z_2 := z_2 - \eta \frac{\partial E_k}{\partial z_2} = z_2 - \eta(\mathrm{o}^k - y^k)\frac{\alpha_2}{\alpha_1 + \alpha_2} = \\ = z_2 - \eta(\mathrm{o}^k - y^k)\mathrm{A}_2(x^k).$$

Последующие выкладки таковы:

$$a := a - \eta \frac{\partial E_k(a,\, b)}{\partial a},$$

где

$$\frac{\partial E_k(a,\, b)}{\partial a} = (\mathrm{o}^k - y^k)\frac{\partial \mathrm{o}^k}{\partial a} = \\ = (\mathrm{o}^k - y^k)\frac{\partial}{\partial a}\big(z_1\mathrm{A}_1(x^k) + z_2\mathrm{A}_2(x^k)\big) = \\ = (\mathrm{o}^k - y^k)\frac{\partial}{\partial a}\big(z_1\mathrm{A}_1(x^k) + z_2(1 - \mathrm{A}_1(x^k))\big) = \\ = (\mathrm{o}^k - y^k)(z_1 - z_2)\frac{\partial \mathrm{A}_1(x^k)}{\partial a} = (\mathrm{o}^k - y^k)(z_1 - z_2)b \times \\ \times \frac{e^{b(x^k - a)}}{(1 + e^{b(x^k - a)})^2} = (\mathrm{o}^k - y^k)(z_1 - z_2)b\mathrm{A}_1(x^k)\big(1 - \mathrm{A}_1(x^k)\big) = \\ = (\mathrm{o}^k - y^k)(z_1 - z_2)b\mathrm{A}_1(x^k)\mathrm{A}_1(x^k),$$

и

$$b := b - \eta \frac{\partial E_k(a,\, b)}{\partial b},$$

где

$$\frac{\partial E_k(a,\,b)}{\partial b} = (\mathrm{o}^k - y^k)(z_1 - z_2)\frac{\partial}{\partial b}\frac{1}{1+e^{b(x^k-a)}} =$$
$$= (\mathrm{o}^k - y^k)(z_1 - z_2)(x^k - a)\mathrm{A}_1(x^k)\big(1 - \mathrm{A}_1(x^k)\big) =$$
$$= (\mathrm{o}^k - y^k)(z_1 - z_2)(x^k - a)\mathrm{A}_1(x^k)\mathrm{A}_1(x^k).$$

Приведенные выкладки, как представляется, полностью иллюстрируют идеи алгоритмов обучения и использования гибридной сети.

Другим примером может служить система, имеющая следующую базу знаний:

П$_1$: если x_1 есть L_1 и x_2 есть L_2 и x_3 есть L_3, тогда y есть H,

П$_2$: если x_1 есть H_1 и x_2 есть H_2 и x_3 есть L_3, тогда y есть M,

П$_3$: если x_1 есть H_1 и x_2 есть H_2 и x_3 есть H_3, тогда y есть S,

где x_1, x_2, x_3 — входные переменные, y — выход системы, L_1, L_2, L_3, H_1, H_2, H_3, H, M, S — некоторые нечеткие множества с функциями принадлежности сигмоидного типа:

$$\mathrm{L}_j(t) = \frac{1}{1+e^{b_j(t-c_j)}}, \quad \mathrm{H}_j(t) = \frac{1}{1+e^{-b_j(t-c_j)}}, \quad j = 1,\,2,\,3,$$
$$\mathrm{H}(t) = \frac{1}{1+e^{-b_4(t-c_4+c_5)}}, \quad \mathrm{M}(t) = \frac{1}{1+e^{-b_4(t-c_4)}},$$
$$\mathrm{S}(t) = \frac{1}{1+e^{b_4(t-c_4)}}.$$

Для определения выходной переменной используется алгоритм вывода Tsukamoto (см. выше), т. е.

1) подсчитываются значения истинности предпосылок для каждого правила:

$$\alpha_1 = \mathrm{L}_1(a_1) \wedge \mathrm{L}_2(a_2) \wedge \mathrm{L}_3(a_3),$$
$$\alpha_2 = \mathrm{H}_1(a_1) \wedge \mathrm{H}_2(a_2) \wedge \mathrm{L}_3(a_3),$$
$$\alpha_3 = \mathrm{H}_1(a_1) \wedge \mathrm{H}_2(a_2) \wedge \mathrm{H}_3(a_3),$$

где в данном случае a_1, a_2, a_3 — текущие значения входов системы;

2) для каждого правила определяются частные выходы:

$$z_1 = \mathrm{B}^{-1}(\alpha_1) = c_4 + c_5 + \frac{1}{b_4}\ln\frac{1-\alpha_1}{\alpha_1},$$
$$z_2 = \mathrm{B}^{-1}(\alpha_2) = c_4 + \frac{1}{b_4}\ln\frac{1-\alpha_2}{\alpha_2},$$
$$z_3 = \mathrm{B}^{-1}(\alpha_3) = c_4 + \frac{1}{b_4}\ln\frac{1-\alpha_3}{\alpha_3};$$

3) находится общий выход системы:

$$z_0 = \frac{\alpha_1 z_1 + \alpha_2 z_2 + \alpha_3 z_3}{\alpha_1 + \alpha_2 + \alpha_3}.$$

Изложенный процесс иллюстрируется рис. 3.5.

Гибридная нейронная сеть, отражающая приведенный механизм вывода, представлена на рис. 3.6. Заметим, что сети с подоб-

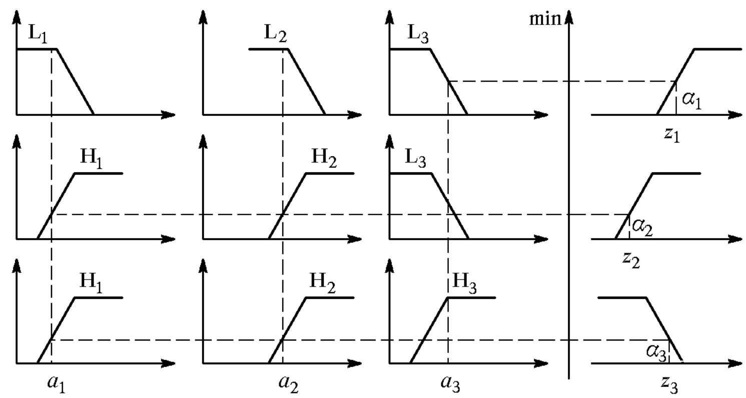

Рис. 3.5. Иллюстрация алгоритма вывода Tsukamoto

ной архитектурой в англоязычной литературе получили название ANFIS (Adaptive Neuro-Fuzzy Inference System).

Данная сеть может быть описана следующим образом.

1. *Слой* 1 (*Layer* 1). Выходы узлов этого слоя представляют собой значения функций принадлежности при конкретных (заданных) значениях входов.

2. *Слой* 2 (*Layer* 2). Выходами нейронов этого слоя являются степени истинности предпосылок каждого правила базы знаний системы, вычисляемые по формулам:

$$\alpha_1 = \mathrm{L}_1(a_1) \wedge \mathrm{L}_2(a_2) \wedge \mathrm{L}_3(a_3),$$
$$\alpha_2 = \mathrm{H}_1(a_1) \wedge \mathrm{H}_2(a_2) \wedge \mathrm{L}_3(a_3),$$
$$\alpha_3 = \mathrm{H}_1(a_1) \wedge \mathrm{H}_2(a_2) \wedge \mathrm{H}_3(a_3).$$

Все нейроны этого слоя обозначены буквой T, что означает, что они могут реализовывать произвольную t-норму для моделирования операции «И».

3. *Слой* 3 (*Layer* 3). Нейроны этого слоя (обозначены буквой N) вычисляют величины:

$$\beta_1 = \frac{\alpha_1}{\alpha_1 + \alpha_2 + \alpha_3}, \quad \beta_2 = \frac{\alpha_2}{\alpha_1 + \alpha_2 + \alpha_3}, \quad \beta_3 = \frac{\alpha_3}{\alpha_1 + \alpha_2 + \alpha_3}.$$

4. *Слой* 4 (*Layer* 4). Нейроны данного слоя выполняют операции:

$$\beta_1 z_1 = \beta_1 \mathrm{H}^{-1}(a_1), \quad \beta_2 z_2 = \beta_2 \mathrm{M}^{-1}(a_2), \quad \beta_3 z_3 = \beta_3 \mathrm{S}^{-1}(a_3).$$

5. *Слой* 5 (*Layer* 5). Единственный нейрон этого слоя вычисляет выход сети:

$$z_0 = \beta_1 z_1 + \beta_2 z_2 + \beta_3 z_3.$$

Корректировка параметров системы здесь производится в соответствии с ранее рассмотренным подходом. Так, например, на-

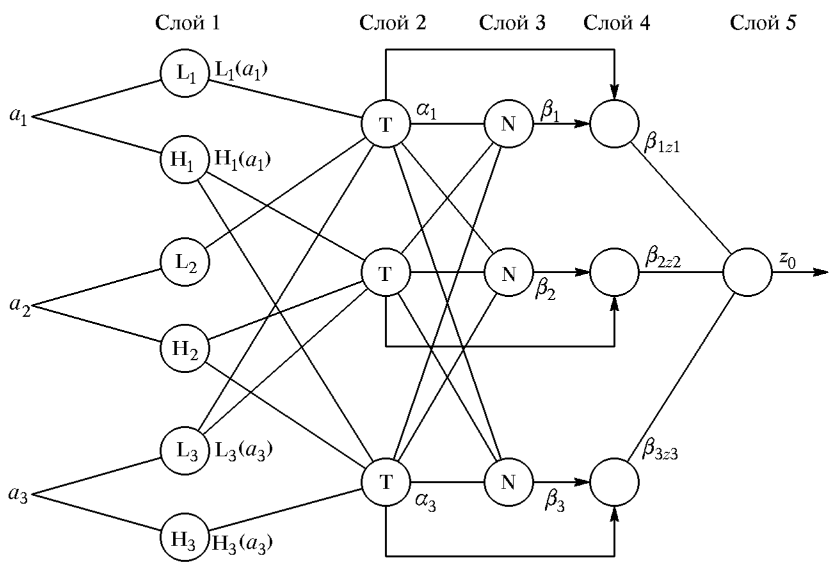

Рис. 3.6. Структура гибридной нейронной сети (архитектура ANFIS)

стройка коэффициентов b_4, c_4 и c_5 — по формулам:

$$b_4 := b_4 - \eta \frac{\partial E_k}{\partial b_4} = b_4 - \frac{\eta}{b_4^2} \delta_k \frac{\alpha_1 + \alpha_2 - \alpha_3}{\alpha_1 + \alpha_2 + \alpha_3},$$

$$c_4 := c_4 - \eta \frac{\partial E_k}{\partial c_4} = c_4 + \eta \delta_k \frac{\alpha_1 + \alpha_2 + \alpha_3}{\alpha_1 + \alpha_2 + \alpha_3} = c_4 + \eta \delta_k,$$

$$c_5 := c_5 - \eta \frac{\partial E_k}{\partial c_5} = c_5 + \eta \delta_k \frac{\alpha_1}{\alpha_1 + \alpha_2 + \alpha_3},$$

где $\delta_k = y^k - \mathrm{o}^k$.

Соответствующие выражения могут быть получены и для остальных коэффициентов.

3.3. Нечеткий гибридный классификатор

Рассмотрим теперь, как с помощью гибридной системы решается задача классификации, т.е. отнесение объекта, характеризующегося набором признаков, к некоторому классу.

Одна из возможных структур для решения подобной задачи приведена на рис. 3.7.

Предполагается, что здесь объект характеризуется двумя количественными признаками x_1 и x_2 и относится к одному из двух

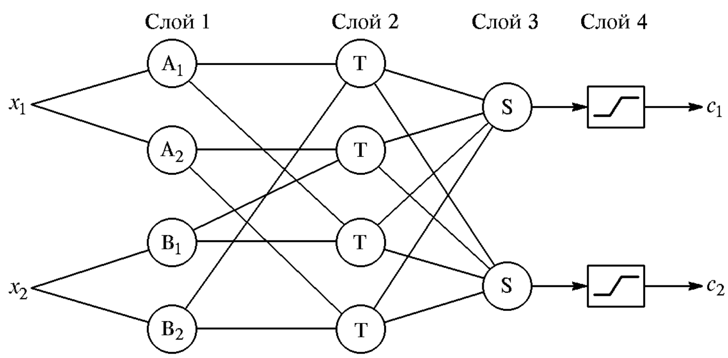

Рис. 3.7. Гибридная сеть для решения задачи классификации

классов — c_1 или c_2. Каждый вход представляется двумя лингвистическими понятиями, что позволяет ограничиться всего четырьмя правилами.

Сеть может быть описана следующим образом.

1. *Слой* 1 (*Layer* 1). Выходы узлов данного слоя — это степени принадлежности входных переменных определенным для них нечетким множествам A_1, A_2, B_1, B_2.

В данном случае выбраны функции принадлежности колоколообразного вида

$$\mathrm{A}_i(t) = \exp\big(-(t - a_{i1}/b_{i1})^2/2\big)$$

с набором параметров a_{i1}, a_{i2}, b_{i1}, b_{i2}.

Значения данных параметров корректируются в процессе обучения сети (основанном на градиентном методе).

2. *Слой* 2 (*Layer* 2). Каждый нейрон этого слоя является нейроном типа рассмотренного выше гибридного (нечеткого) нейрона «И».

3. *Слой* 3 (*Layer* 3). Нейроны данного слоя являются обычными (стандартными) нейронами, входами которых являются линейные (взвешенные) комбинации выходов нейронов предыдущего

слоя, а выходы формируются с использованием активационных функций сигмоидного типа. Эти выходы трактуются как степени принадлежности предъявленного объекта первому или второму классу.

Алгоритм обучения данной сети, в принципе, не отличается от рассмотренных.

3.4. Программная реализация моделей нечеткой логики, нейросетевых и гибридных

В настоящее время существует огромное количество программных продуктов, позволяющих реализовывать нейросетевые структуры (так называемых программ-нейроимитаторов). Значительно меньше программ, работающих с моделями нечеткой логики (и они пока практически недоступны отечественным специалистам). Отметим в этой связи, что для целей проектирования и использования как нейросетевых моделей, так и нечетких и гибридных крайне удобной является достаточно распространенная в нашей стране математическая система MATLAB (версии 5.2 и 5.3), точнее два инструментальных средства этой системы: пакеты Neural Networks Toolbox (нейронные сети) и Fuzzy Logic Toolbox (пакет нечеткой логики), которые и будут ниже подробно рассмотрены.