Algorithm 1: The Perceptron Training Algorithm

Step 1: ***Initialization***

Set initial weights $w_1, w_2, \ldots, w_n$ and threshold θ to random numbers in the range [-0.5, 0.5]. (e.g., $\theta = 0.2$, $\alpha = 0.1$)

Step 2: ***Activation***

Activate the perceptron by applying inputs $x_1(p), x_2(p), \ldots, x_n(p)$ and **desired output** $Y_d(p)$, where iteration p refers to the pth training example presented to the perceptron ($p = 1, 2, \ldots$). Calculate the **actual output** $Y(p)$ at iteration $p = 1$

$$Y(p) = step\left[\sum_{i=1}^{n} x_i(p) w_i(p) - \theta\right], \; // \; Y(X) = step[X] = \begin{cases} 1 & \text{if } X \geq 0 \\ 0 & \text{if } X < 0 \end{cases} \qquad (6.6)$$

where n is the number of the perceptron inputs, and ***step*** is a **step activation function**.

Step 3: ***Weight training*** (***learning***)

Update the **weights** of the perceptron

$$w_i(p+1) = w_i(p) + \Delta w_i(p), \qquad (6.7)$$

where $\Delta w_i(p)$ is the **weight correction** at iteration p.

The **weight correction** is computed by the **delta rule**:

$$\Delta w_i(p) = \alpha \times x_i(p) \times e(p), \qquad (6.8)$$

where α is the **learning rate**, $\alpha \in (0, 1]$; $e(p) = Y_d(p) - Y(p)$, where $p = 1, 2, 3, \ldots$

(α can be fixed for all iterations changed for different iterations)

Step 4: ***Iteration***

Increase iteration p by one, go back to Step 2 and repeat the process until convergence.
// until error $e = 0$

• A single-layer two-input perceptron is shown in Figure 6.5.

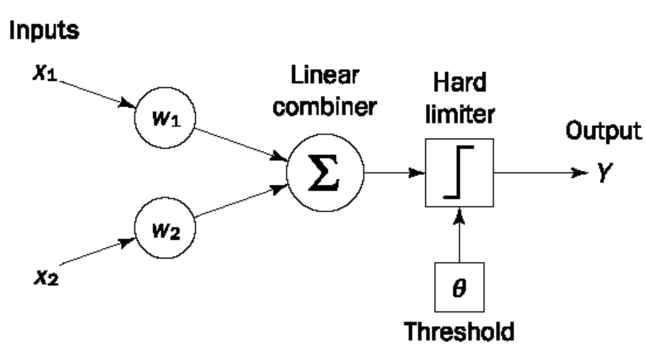

Figure 6.5 Single-layer two-input perceptron

Table 6.3 Example of perceptron learning: the logical operation AND

Epoch	Inputs		Desired output	Initial weights		Actual output	Error	Final weights	
	x_1	x_2	Y_d	w_1	w_2	Y	e	w_1	w_2
1	0	0	0	0.3	-0.1	0	0	0.3	-0.1
	0	1	0	0.3	-0.1	0	0	0.3	-0.1
	1	0	0	0.3	-0.1	1	-1	0.2	-0.1
	1	1	1	0.2	-0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	-1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	-1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	-1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

Threshold $\theta = 0.2$, learning rate $\alpha = 0.1$

• The sequence of four input patterns representing an **epoch**. The four input patterns (i.e., training examples) are $(x_1, x_2) = (0, 0)$, $(x_1, x_2) = (0, 1)$, $(x_1, x_2) = (1, 0)$, $(x_1, x_2) = (1, 1)$.

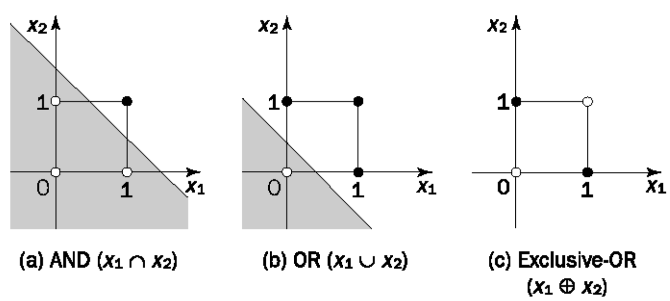

Figure 6.7 Two-dimensional plots of basic logical operations

$$X = \sum_{i=1}^{n} x_i w_i \qquad (6.1)$$

$$Y = \begin{cases} +1 & \text{if } X \geq \theta \\ -1 & \text{if } X < \theta \end{cases}$$

$$Y = sign\left[\sum_{i=1}^{n} x_i w_i - \theta\right] \qquad (6.2)$$

$$\sum_{i=1}^{n} x_i w_i - \theta = 0 \qquad (6.3)$$

$e(p) = Y_d(p) - Y(p)$, where $p = 1, 2, 3, \ldots$ (6.4)

$w_i(p + 1) = w_i(p) + \alpha \times x_i(p) \times e(p)$ (6.5)