Chapter 1

Introduction

Sample Solutions

Solutions

Exercise 1.1

a. $N_5(t) = N_3(t\text{-}1)$ OR $N_4(t\text{-}1)$

$N_3(t\text{-}1) = N_1(t\text{-}2)$ AND $N_2(t\text{-}2)$

$N_4(t\text{-}1) = N_1(t\text{-}2)$ AND NOT $N_2(t\text{-}2)$

so

$N_5(t) = [N_1(t\text{-}2)$ AND $N_2(t\text{-}2)]$ OR $[N_1(t\text{-}2)$ AND NOT $N_2(t\text{-}2)]$

(Note that this reduces to $N_5(t) = N_1(t\text{-}2)$.)

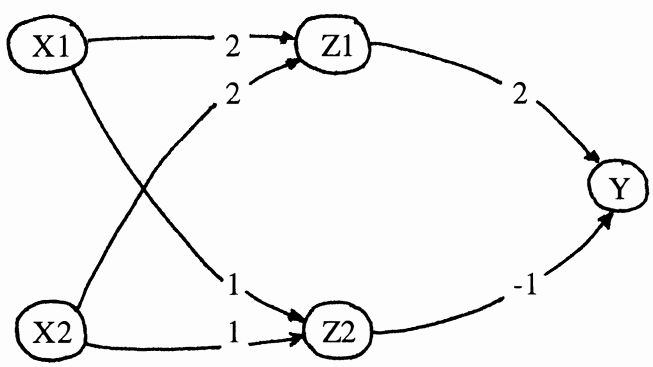
b.

time	activations
t = 0	$N_1 = 1$, $N_2 = 0$.
t = 1	$N_3 = N_1(0)$ AND $N_2(0) = 1$ AND $0 = 0$
	$N_4 = N_1(0)$ AND NOT $N_2(0) = 1$ AND NOT $0 = 1$
t = 2	$N_5 = N_3(1)$ OR $N_4(1) = 0$ OR $1 = 1$

Exercise 1.2

x_1 XOR x_2 = (x_1 OR x_2) AND NOT (x_1 AND x_2)

which corresponds to the net



Both this net and the net in Figure 1.17 involve 1 hidden layer of 2 units; this net uses 3 different basic neuron types, the net in Figure 1.17 uses only 2.

Exercise 1.3

The modified requirements can be expressed as

$$y_2(t) = \{x_2(t\text{-}3) \text{ AND } x_2(t\text{-}2)\} \text{ AND } x_2(t\text{-}1).$$

The requirement for y_1 remains unchanged:

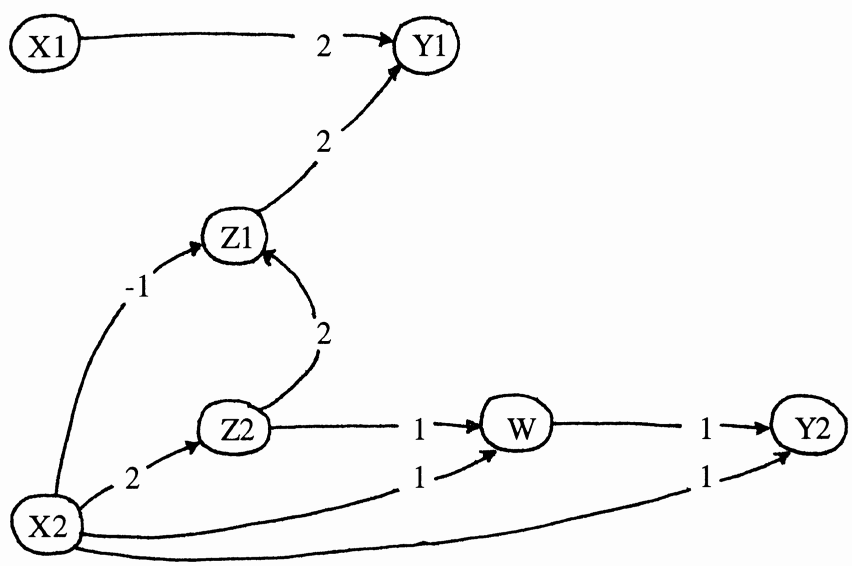
$$y_1(t) = x_1(t\text{-}1) \text{ OR } \{x_2(t\text{-}3) \text{ AND NOT } x_2(t\text{-}2)\}.$$

The network is modified by the addition of a hidden unit, W, such that

$$w(t) = z_2(t\text{-}1) \text{ AND } x_2(t\text{-}1)$$
$$= x_2(t\text{-}2) \text{ AND } x_2(t\text{-}1)$$

so that

$$y_2(t) = w(t\text{-}1) \text{ AND } x_2(t\text{-}1).$$



Exercise 1.4

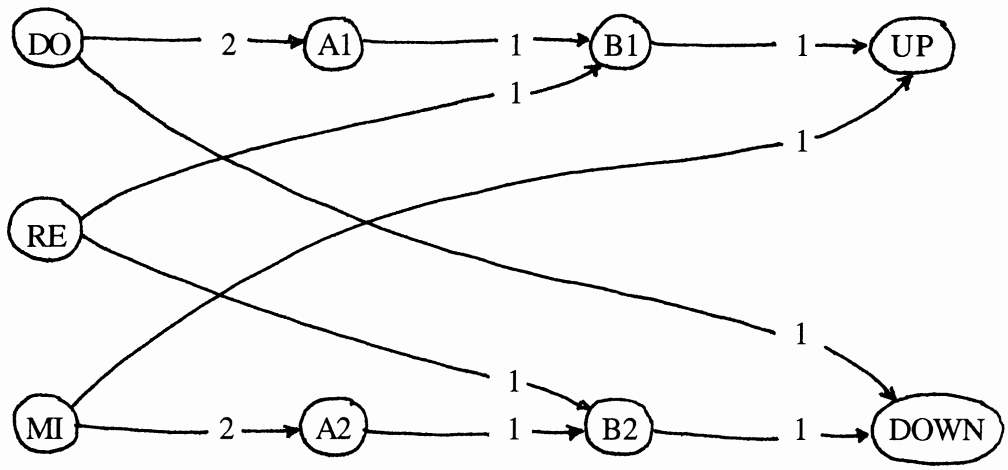
If heat is applied, heat will be perceived one time step later and will continue to be perceived until one time step after the heat is removed.

If cold is applied for one time step and removed, heat will be perceived two time steps later, but only for one time step.

If cold is applied for two or more steps, cold will be perceived two time steps after the application. This perception of cold will continue until one time step after the cold is removed. Heat will then be perceived for one time step (i.e. two time steps after the cold stimulus is removed).

Exercise 1.5

A simple network to recognize an "upscale" when the sequence DO, RE, MI is input and to recognize "downscale" when MI, RE, DO is input is shown below.



The analysis for the upscale portion of the net is as follows; the downscale portion is similar.

The desired response is

UP(t)	=	MI(t-1) AND [RE(t-2) AND DO(t-3)]
	=	MI(t-1) AND B1(t-1)
B1(t-1)	=	RE(t-2) AND A1(t-2)
A1(t-2)	=	DO(t-3)

Note that this network does not require "clean playing" of the scale segment; for example, further refinements could require that RE(t-1), DO(t-2), and DO(t-1) be 0 (in addition to MI(t-1), RE(t-2), and DO(t-3) being 1) for the upscale neuron to give a positive response.

Chapter 2

Simple Neural Nets for Pattern Classification

Sample Solutions

Solutions

Hebb Net

Exercise 2.1

The weight changes to store each input : target pair are given in the following table.

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	1	1	-1	1
-1	1	1	1	-1	1	1
-1	-1	1	-1	1	1	-1

The final weights are the sum of the changes.

w_1	w_2	b
0	0	0

Exercise 2.2

The following logic function has no positive responses.

y = FALSE

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	-1	-1	1	-1
-1	1	1	-1	1	-1	-1
-1	-1	1	-1	1	1	-1

The final weights are the sum of the changes.

w_1	w_2	b
0	0	-4

The following logic functions have exactly 1 positive response.

$y = x_1$ AND x_2

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	-1	-1	1	-1
-1	1	1	-1	1	-1	-1
-1	-1	1	-1	1	1	-1

The final weights are the sum of the changes.

w_1	w_2	b
2	2	-2

$y = x_1$ AND NOT x_2

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	1	1	-1	1
-1	1	1	-1	1	-1	-1
-1	-1	1	-1	1	1	-1

The final weights are the sum of the changes.

w_1	w_2	b
2	-2	-2

$y = x_2$ AND NOT x_1

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	-1	-1	1	-1
-1	1	1	1	-1	1	1
-1	-1	1	-1	1	1	-1

The final weights are the sum of the changes.

w_1	w_2	b
-2	2	-2

$Y =$ NOT x_1 AND NOT $x_2 =$ NOT (x_1 OR x_2)

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	-1	-1	1	-1
-1	1	1	-1	1	-1	-1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
-2	-2	-2

The following logic functions have exactly 2 positive responses.

$y = x_1$

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	1	1	-1	1
-1	1	1	-1	1	-1	-1
-1	-1	1	-1	1	1	-1
The final weights are the sum of the changes.				w_1	w_2	b
				4	0	0

$y = x_2$

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	-1	-1	1	-1
-1	1	1	1	-1	1	1
-1	-1	1	-1	1	1	-1
The final weights are the sum of the changes.				w_1	w_2	b
				0	4	0

Y = NOT (x_1 XOR x_2) This function is not linearly separable.

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	-1	-1	1	-1
-1	1	1	-1	1	-1	-1
-1	-1	1	1	-1	-1	1
The final weights are the sum of the changes.				w_1	w_2	b
				0	0	0

$y = x_1$ XOR x_2 This function is not linearly separable.

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	1	1	-1	1
-1	1	1	1	-1	1	1
-1	-1	1	-1	1	1	-1
The final weights are the sum of the changes.				w_1	w_2	b
				0	0	0

y = NOT x_2

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	1	1	-1	1
-1	1	1	-1	1	-1	-1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
0	-4	0

y = NOT x_1

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	-1	-1	1	-1
-1	1	1	1	-1	1	1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
-4	0	0

The following logic functions have exactly 3 positive responses.

y = NOT (x_1 AND x_2) = (NOT x_1) OR (NOT x_2)

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	-1	-1	-1	-1
1	-1	1	1	1	-1	1
-1	1	1	1	-1	1	1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
-2	-2	2

y = NOT (x_1 AND NOT x_2) = (NOT x_1) OR x_2

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	-1	-1	1	-1
-1	1	1	1	-1	1	1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
-2	2	2

$y = \text{NOT } (x_2 \text{ AND NOT } x_1) = x_1 \text{ OR } (\text{NOT } x_2)$

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	1	1	-1	1
-1	1	1	-1	1	-1	-1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
2	-2	2

$y = x_1 \text{ OR } x_2$

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	1	1	-1	1
-1	1	1	1	-1	1	1
-1	-1	1	-1	1	1	-1

The final weights are the sum of the changes.

w_1	w_2	b
2	2	2

The following logic function has 4 positive responses.

$y = \text{TRUE}$

x_1	x_2		t	Δw_1	Δw_2	Δb
1	1	1	1	1	1	1
1	-1	1	1	1	-1	1
-1	1	1	1	-1	1	1
-1	-1	1	1	-1	-1	1

The final weights are the sum of the changes.

w_1	w_2	b
0	0	4

Exercise 2.3

The weights are as found in Example 2.8:

2 -2 -2 -2 2 -2 2 0 2 -2 -2 0 2 0 -2 -2 2 0 2 -2 2 -2 -2 -2 2

The first input pattern is:

1 -1 -1 -1 1 -1 1 -1 1 -1 -1 -1 1 -1 -1 -1 1 -1 1 -1 1 -1 -1 -1 1

One example of a vector formed from this pattern with 20 components missing is:

1 -1 -1 -1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

The response of the net for this input pattern is:

(1)(2) + (-1)(-2) + (-1)(-2) + (-1)(-2) + (1)(2) = 10 -> 1 (the target value for the first pattern).

One example of a vector formed from this pattern with 10 components wrong is:

1 -1 -1 -1 1 -1 1 -1 1 -1 -1 -1 1 -1 -1 1 -1 1 -1 1 -1 1 1 1 -1

The response of the net for this input pattern is 6 -> 1 (the target value for the first pattern).

The additional pattern for "not X" is

-1 -1 1 -1 -1 -1 1 -1 1 -1 -1 1 1 1 -1 1 -1 -1 -1 1 1 -1 -1 -1 1

with target value of -1.

The new weight vector for storing all three of these associations is

3 -1 -3 -1 3 -1 1 1 1 -1 -1 -1 1 -1 -1 -3 3 1 3 -3 1 -1 -1 -1 1

with bias of -1.

The bias indicates that there has been one more example of a "not X" than of an "X".

Exercise 2.4

There are lots of possibilities for this.

Exercise 2.5

x_1	x_2	x_3	x_4	t	Δw_1	Δw_2	Δw_3	Δw_4
1	1	1	1	1	1	1	1	1
-1	1	-1	-1	1	-1	1	-1	-1
1	1	1	-1	-1	-1	-1	-1	1
1	-1	-1	1	-1	-1	1	1	-1

The final weights are

w_1	w_2	w_3	w_4
-2	2	0	0

The response of the net for each training input vector:

(1 1 1 1) (-2 2 0 0) = 0

(-1 1 -1 -1) (-2 2 0 0) = 4

(1 1 1 -1) (-2 2 0 0) = 0

(1 -1 -1 1) (-2 2 0 0) = -4

The use of a bias would not improve the ability of the net to correctly classify the training patterns using weights from Hebb learning, because the bias would be 0 after this training.

Exercise 2.6

a The bipolar form of these associations are

x_1	x_2	x_3	t	Δw_1	Δw_2	Δw_3
1	-1	1	1	1	-1	1
1	1	-1	-1	-1	-1	1

The final weights are	w_1	w_2	w_3
	0	-2	2

b. Using binary input patterns, the response of the net is correct for each of the training patterns.

(1 0 1) (0 -2 2) = 2 -> 1
(1 1 0) (0 -2 2) = -2 -> 0

c. Using bipolar input patterns, the response of the net is correct for each of the training patterns.

(1 -1 1) (0 -2 2) = 4 -> 1
(1 1 -1) (0 -2 2) = -4 -> 0

d. The following shows the response of the net to noisy patterns, grouped according to their formation from the training patterns. The zero responses are inconclusive; all other responses are correct.

Using noisy versions of the first bipolar input pattern:

One component missing:

(0 -1 1) (0 -2 2) = 2 -> 1
(1 0 1) (0 -2 2) = 2 -> 1
(1 -1 0) (0 -2 2) = 2 -> 1

Two components missing:

(1 0 0) (0 -2 2) = 0 -> 0
(0 -1 0) (0 -2 2) = 2 -> 1
(0 0 1) (0 -2 2) = 2 -> 1

Using noisy version of the second bipolar input pattern:

One component missing:

(0 1 -1) (0 -2 2) = 0 -> 0
(1 0 -1) (0 -2 2) = -2 -> -1
(1 1 0) (0 -2 2) = -2 -> -1

Two components missing:

(1 0 0) (0 -2 2) = 0 -> 0
(0 1 0) (0 -2 2) = -2 -> -1
(0 0 -1) (0 -2 2) = -2 -> -1

One component wrong (is this one wrong in the first pattern or in the second pattern???):

(1 1 1) (0 -2 2) = 0 -> 0

Perceptron

Exercise 2.7

Graph the lines $x_1 w_1 + x_2 w_2 + b = 0$ for

w_1	w_2	b
1	1	1
0	2	0
1	1	-1

Graph

$x_1 + x_2 + 1 = 0$ $\quad$ $x_2 = 0$ $\quad$ $x_1 + x_2 - 1 = 0$

Exercise 2.8

The training process for the AND function with an arbitrary $\alpha > 0$, and intital weights of 0:

Input (x_1 x_2 1)	Net	Out	Target	Weight Changes	Weights (w_1 w_2 b)
					(0 0 0)
(1 1 1)	0	0	1	(α α α)	(α α α)
(1 -1 1)	α	1	-1	($-\alpha$ α $-\alpha$)	(0 2α 0)
(-1 1 1)	2α	1	-1	(α $-\alpha$ $-\alpha$)	(α α $-\alpha$)
(-1 -1 1)	-3α	-1	-1	(0 0 0)	(α α $-\alpha$)

Note that at each stage of learning the separating line is exactly the same as for the case with $\alpha = 1$.

b. If the response of the net is incorrect for a given training input, the weights are updated according to $\tilde{w}(\text{new}) = \tilde{w}(\text{old}) + \alpha\, \tilde{x}$. We now show that the sequence of input training vectors for which a weight change occurs is finite. If $\tilde{x}(0)$ is the first training vector for which an error occurred, then $\tilde{w}(1) = \tilde{w}(0) + \alpha\, \tilde{x}(0)$ (where by assumption $\tilde{x}(0) \cdot \tilde{w}(0) \le 0$).

If another error occurs, we denote the vector $\tilde{x}(1)$.

$$\tilde{w}(2) = \tilde{w}(1) + \alpha\, \tilde{x}(1) \qquad \text{(where by assumption } \tilde{x}(1) \cdot \tilde{w}(1) \le 0\text{), etc.}$$

Combining the successive weight changes gives

$$\tilde{w}(k) = \tilde{w}(0) + \alpha\, \tilde{x}(0) + \alpha\, \tilde{x}(1) + \alpha\, \tilde{x}(2) + \dots + \alpha\, \tilde{x}(k-1)$$

We now show that k cannot be arbitrarily large.

Let $\tilde{w}^*$ be a weight vector such that $\tilde{x} \cdot \tilde{w}^* > 0$ for all training vectors in F. By assumption such a weight vector exists! Let $m = \min\{ \tilde{x} \cdot \tilde{w}^* \}$ where the minimum is taken over all training vectors in F; this minimum exists as long as there are only finitely many training vectors. Now

$$\tilde{w}(k) \cdot \tilde{w}^* = [\tilde{w}(0) + \alpha\, \tilde{x}(0) + \alpha\, \tilde{x}(1) + \alpha\, \tilde{x}(2) \;\ldots\; + \alpha\, \tilde{x}(k\text{-}1)] \cdot \tilde{w}^*$$

$$\geq \tilde{w}(0) \cdot \tilde{w}^* + k\, \alpha\, m$$

since $\tilde{x}(i) \cdot \tilde{w}^* \geq m$ for each i, $1 \leq i \leq P$.

By the Cauchy-Schwartz inequality

$$\|\tilde{w}(k)\|^2 \geq \frac{(\tilde{w}(k) \cdot \tilde{w}^*)^2}{\|\tilde{w}^*\|^2} \geq \frac{(\tilde{w}(0) \cdot \tilde{w}^* + k\, \alpha\, m)^2}{\|\tilde{w}^*\|^2} .$$

This shows that the squared length of the weight vector grows faster than k^2 where k is the number of time the weights have changed.

However, to show that the length cannot continue to grow indefinitely, consider

$$\tilde{w}(k) = \tilde{w}(k\text{-}1) + \alpha\, \tilde{x}(k\text{-}1) ,$$

together with the fact that

$$\tilde{x}(k\text{-}1) \cdot \tilde{w}(k\text{-}1) \leq 0.$$

By simple algebra

$$\|\tilde{w}(k)\|^2 = \|\tilde{w}(k\text{-}1)\|^2 + 2\alpha\, \tilde{x}(k\text{-}1) \cdot \tilde{w}(k\text{-}1) + \alpha^2 \|\tilde{x}(k\text{-}1)\|^2$$

$$\leq \|\tilde{w}(k\text{-}1)\|^2 + \alpha^2 \|\tilde{x}(k\text{-}1)\|^2$$

Now let M = max $\{\|\tilde{x}\|^2$ for all x in the training set$\}$; then

$$\|\tilde{w}(k)\|^2 \leq \|\tilde{w}(k\text{-}1)\|^2 + \alpha^2 \|\tilde{x}(k\text{-}1)\|^2$$

$$\leq \|\tilde{w}(k\text{-}2)\|^2 + \alpha^2 \|\tilde{x}(k\text{-}2)\|^2 + \alpha^2 \|\tilde{x}(k\text{-}1)\|^2$$

$$\ldots$$

$$\leq \|\tilde{w}(0)\|^2 + \alpha^2 \|\tilde{x}(0)\|^2 + \cdots + \alpha^2 \|\tilde{x}(k\text{-}1)\|^2$$

$$\leq \|\tilde{w}(0)\|^2 + k\, \alpha^2\, M .$$

Thus the squared length grows less rapidly than linearly in k.

Combining these two inequalities shows that the number of times that the weights may change is bounded. Specifically,

$$\frac{(\tilde{w}(0) \cdot \tilde{w}^* + k\, \alpha\, m)^2}{\|\tilde{w}^*\|^2} \leq \|\tilde{w}(k)\|^2 \leq \|\tilde{w}(0)\|^2 + k\, \alpha^2\, M$$

Again, to simplify the algebra, assume (without loss of generality) that $\tilde{w}(0) = 0$. The maximum possible number of times the weights may change is then given by

$$\frac{(k\, \alpha\, m)^2}{\|\tilde{w}^*\|^2} \leq k\, \alpha^2\, M$$

or

$$k \leq \frac{M\, \|\tilde{w}^*\|^2}{m^2}.$$