

Hebb Net

Randy Cahya Wihandika, S.ST., M.Kom.

Topik Hari Ini

- Hebb net
- Studi kasus Hebb net: logika AND
- Studi kasus Hebb net: pengenalan karakter

Hebb Rule

- Metode pelatihan (*training*) jaringan saraf yang pertama kali ada dan yang paling sederhana
 - Dengan cara mengubah nilai-nilai bobot
- Proses pelatihan hanya dilakukan sebanyak satu *epoch*
- Sebuah JST single-layer yang dilatih menggunakan Hebb rule disebut **Hebb net**



Sumber: <https://open.library.ubc.ca>

Donald Hebb (1904–1985)

Organization

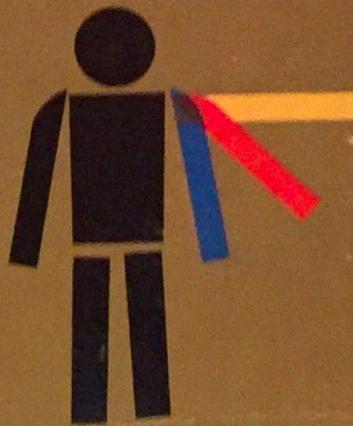
of

Behavior

by

D. O. Hebb

*Stimulus and response — and what occurs
in the brain in the interval between them*



Algoritme Pelatihan Hebb Rule

1. Inisialisasi semua bobot dengan nilai 0
2. Untuk setiap data latih s dan target t , lakukan langkah 3–5
3. Set nilai aktivasi setiap neuron input:
$$x_i = s_i$$
4. Set nilai aktivasi neuron *output*.
$$y = t$$
5. Ubah nilai bobot:

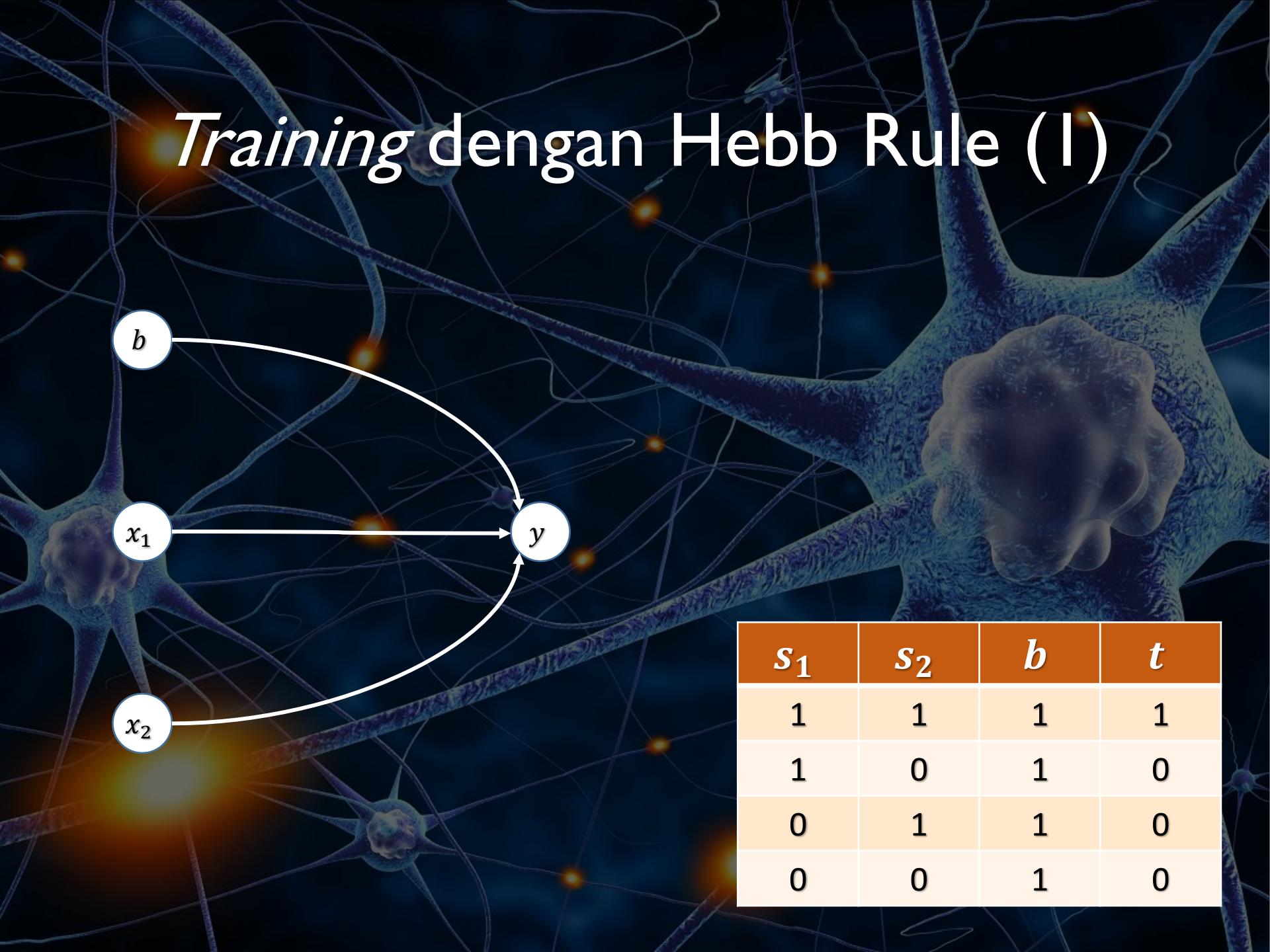
$$w'_i = w_i + x_i y$$

Training dengan Hebb Rule (I)

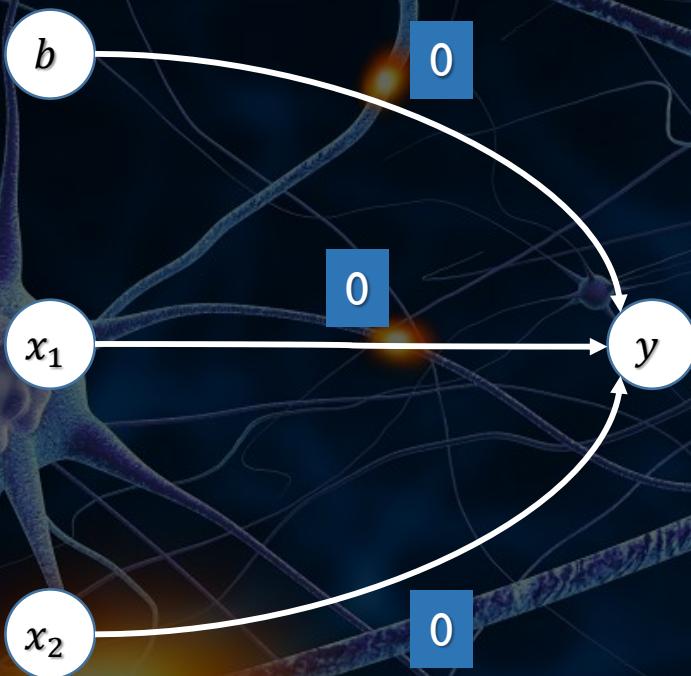
- Data *training* untuk logika AND dengan input dan target biner:

x_1	x_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



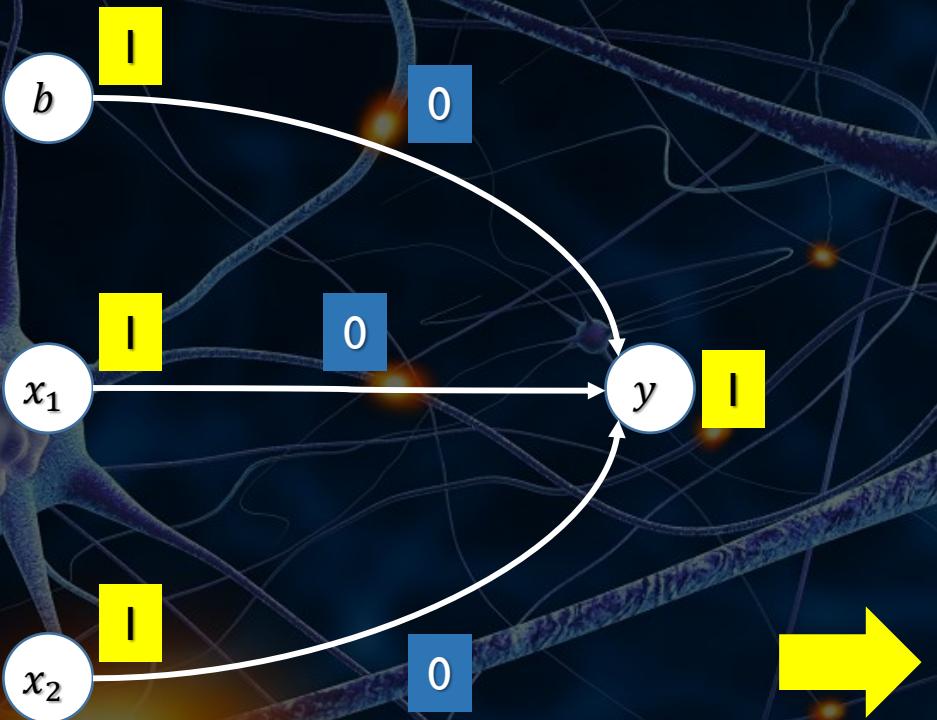
Training dengan Hebb Rule (I)



Inisialisasi semua bobot
dengan nilai 0

s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)

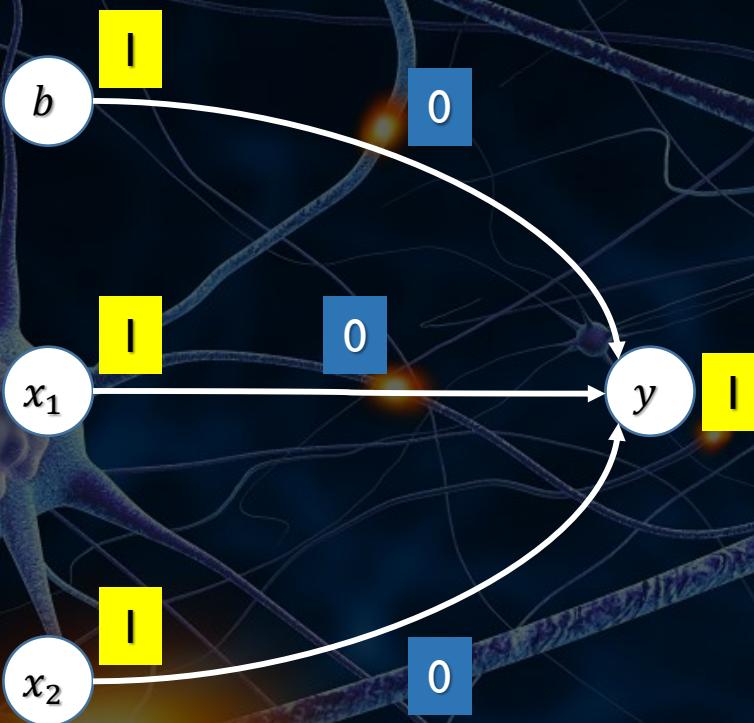


Set nilai aktivasi:

$$x_i = s_i \\ y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



Ubah bobot:

$$w'_i = w_i + x_i y$$

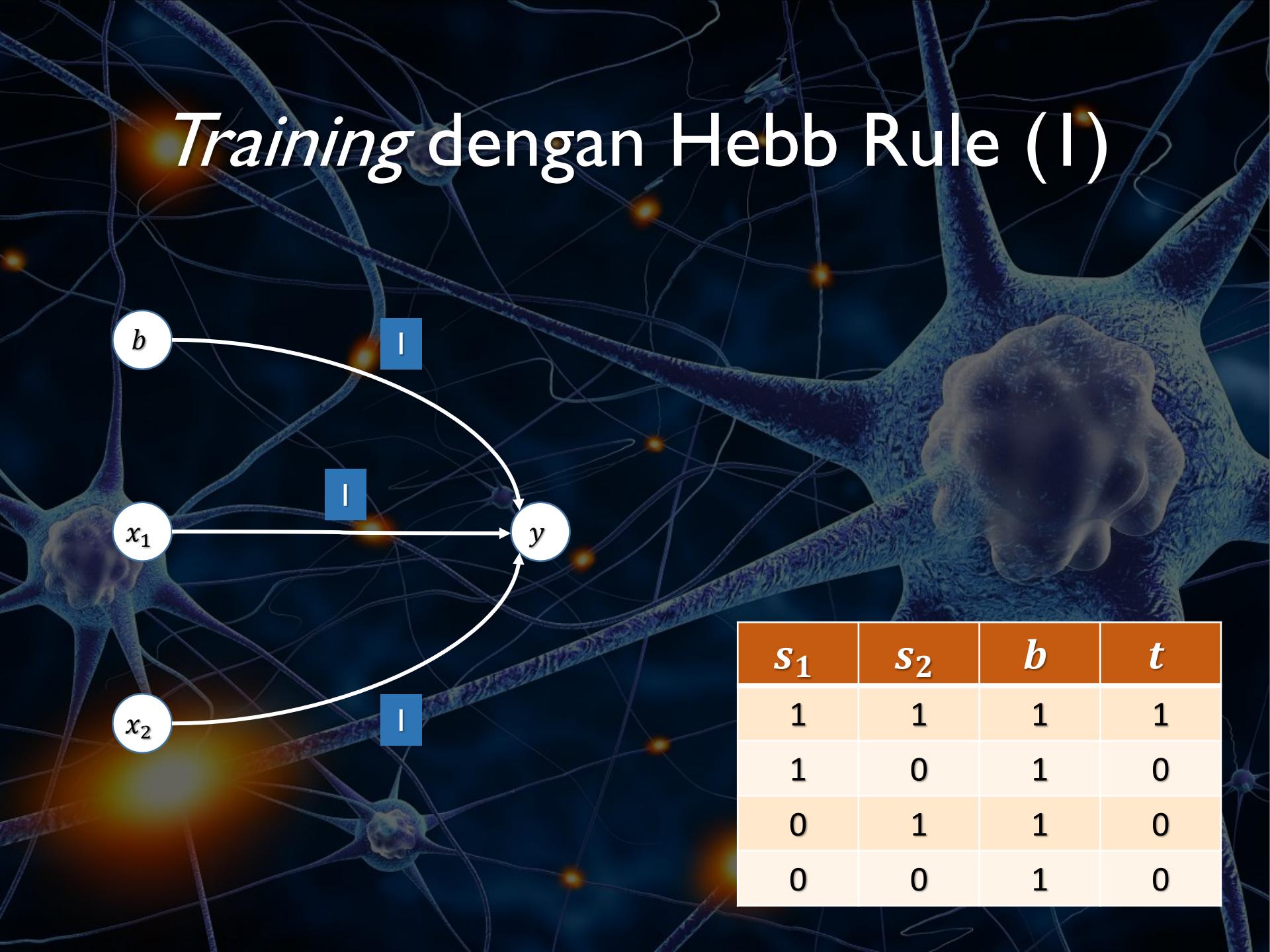
$$b' = 0 + 1 = 1$$

$$w'_1 = 0 + 1 \cdot 1 = 1$$

$$w'_2 = 0 + 1 \cdot 1 = 1$$

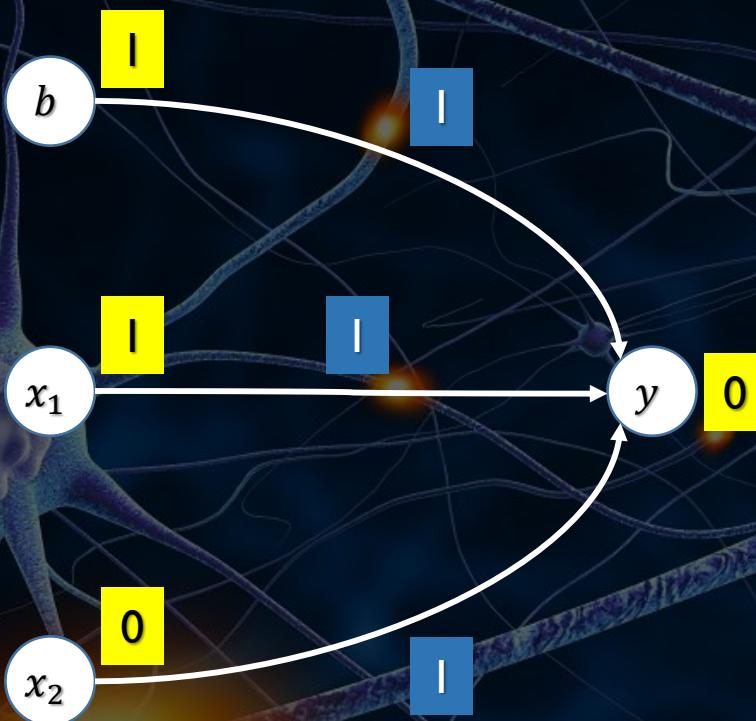
s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)

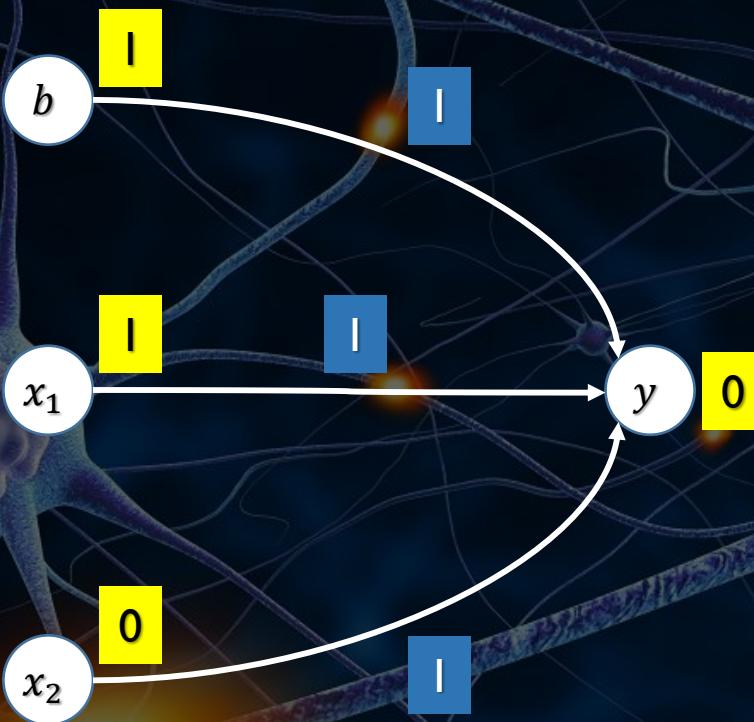


Set nilai aktivasi:

$$x_i = s_i \\ y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



Ubah bobot:

$$w'_i = w_i + x_i y$$

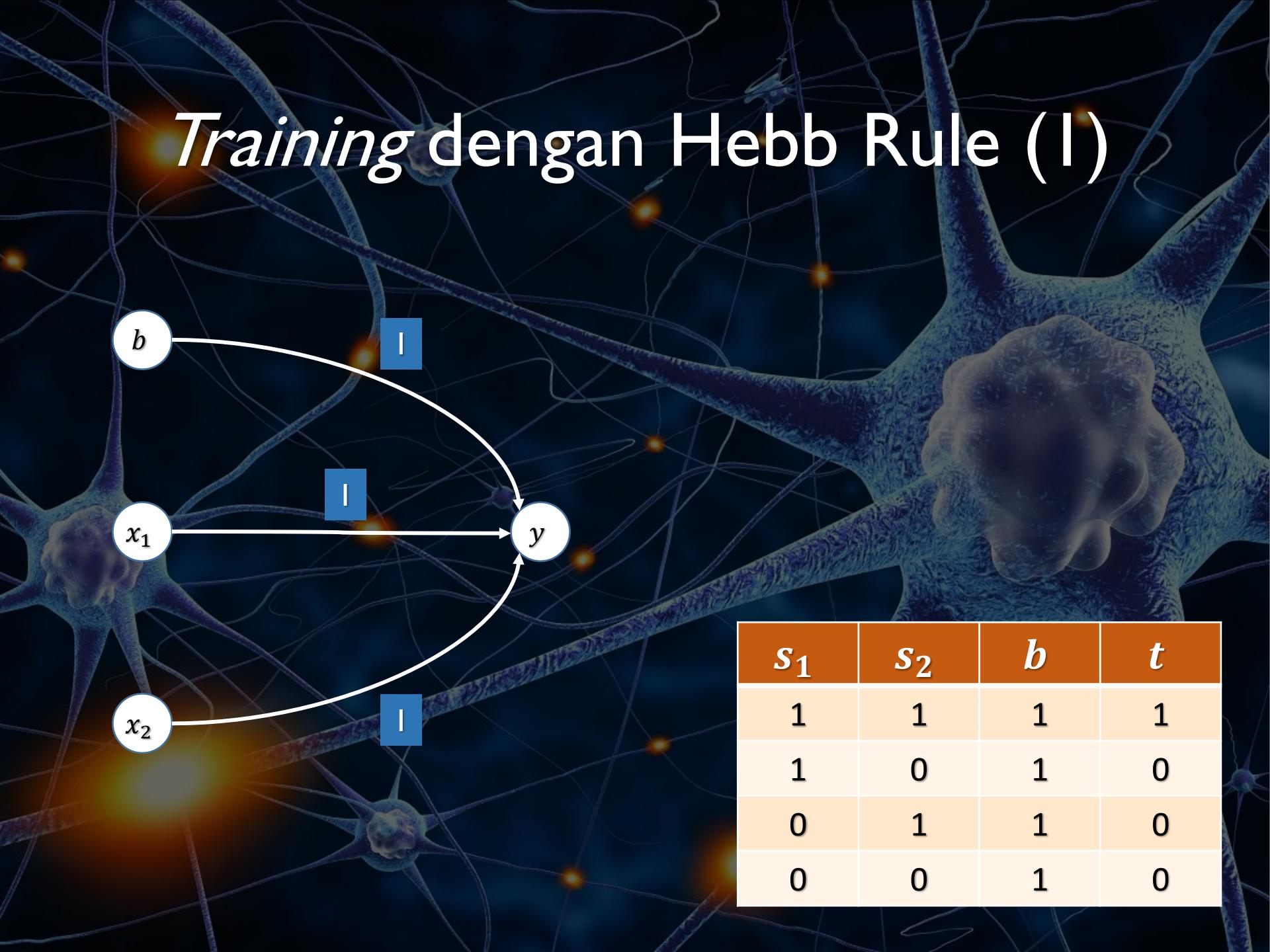
$$b' = 1 + 0 = 1$$

$$w'_1 = 1 + 1.0 = 1$$

$$w'_2 = 1 + 0.0 = 1$$

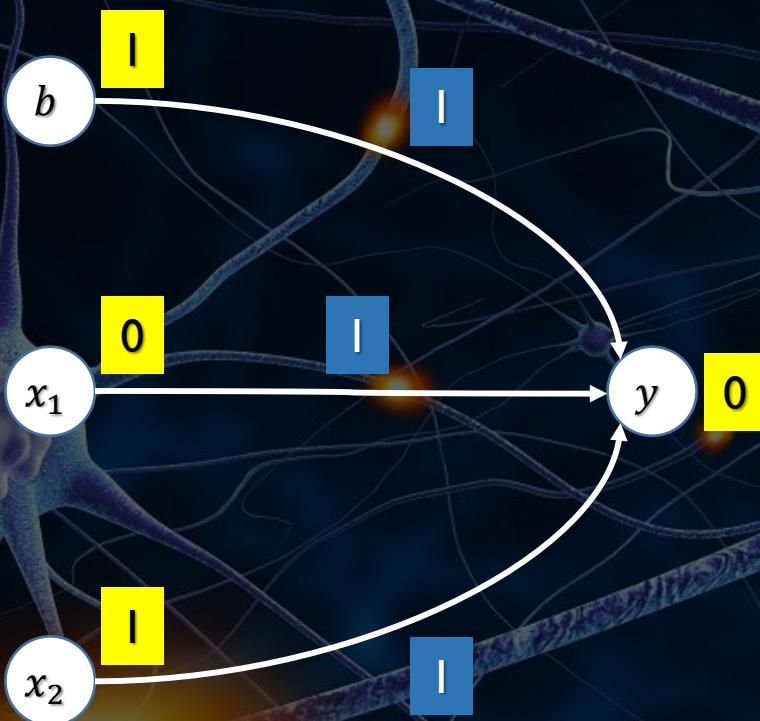
s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)

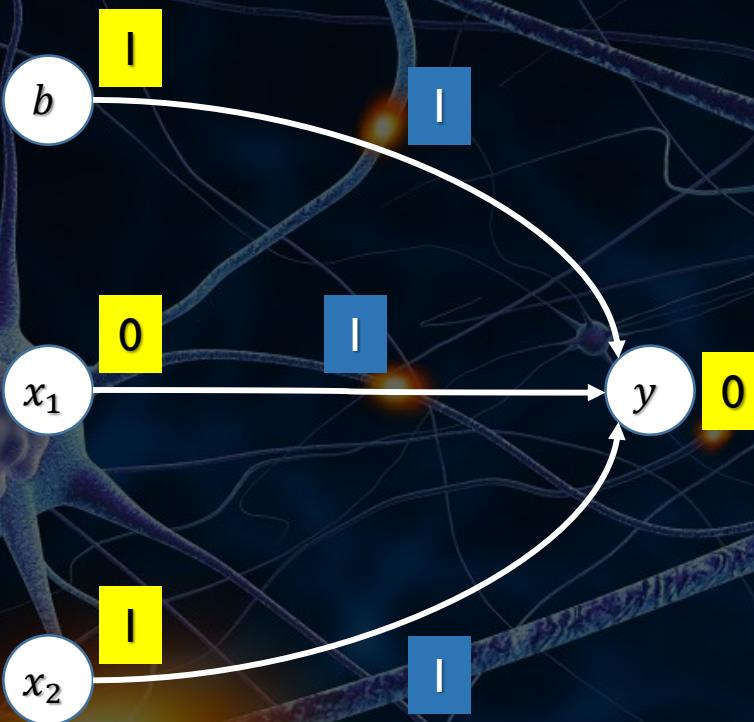


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



Ubah bobot:

$$w'_i = w_i + x_i y$$

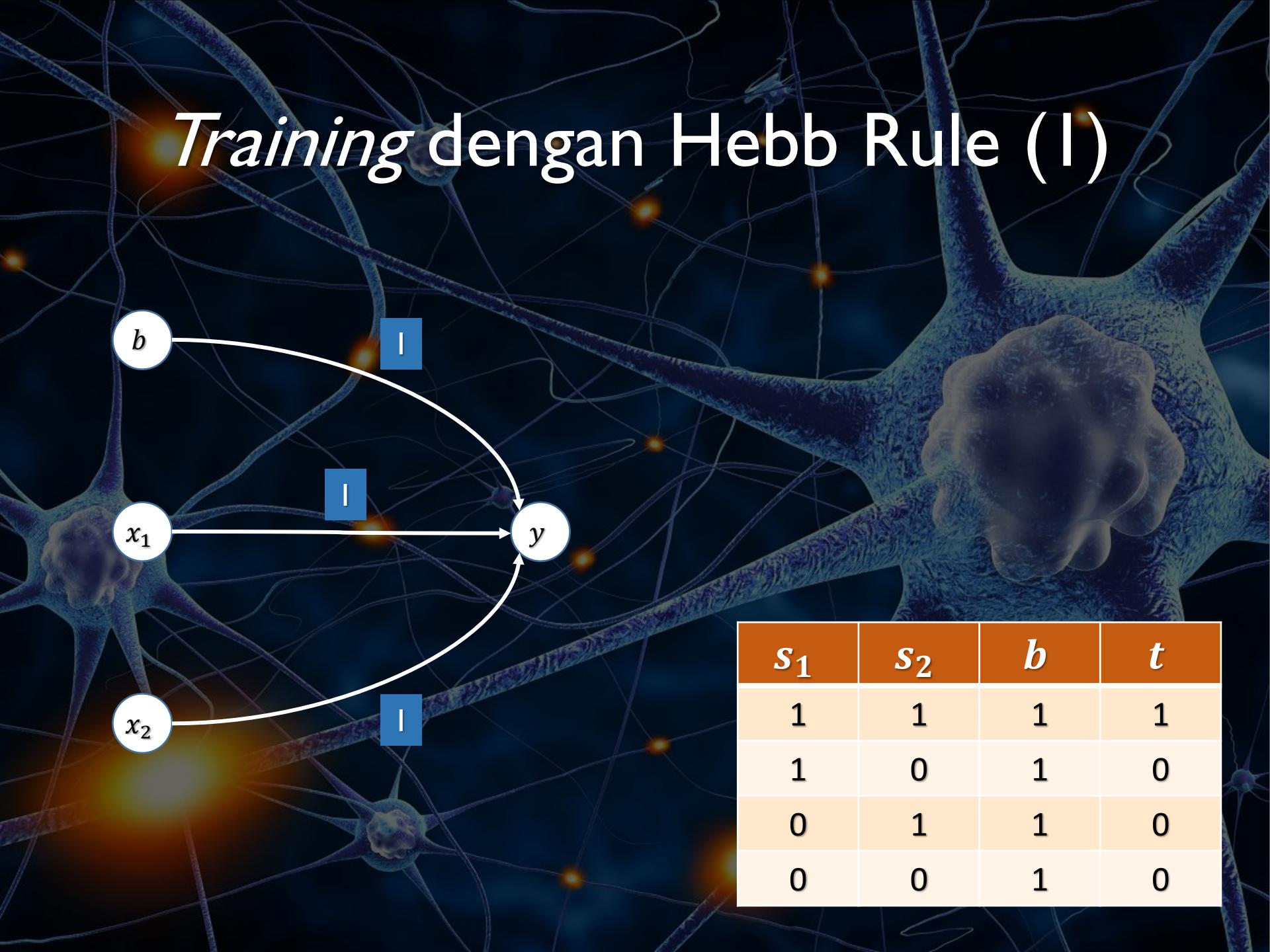
$$b' = 1 + 0 = 1$$

$$w'_1 = 1 + 0.0 = 1$$

$$w'_2 = 1 + 1.0 = 1$$

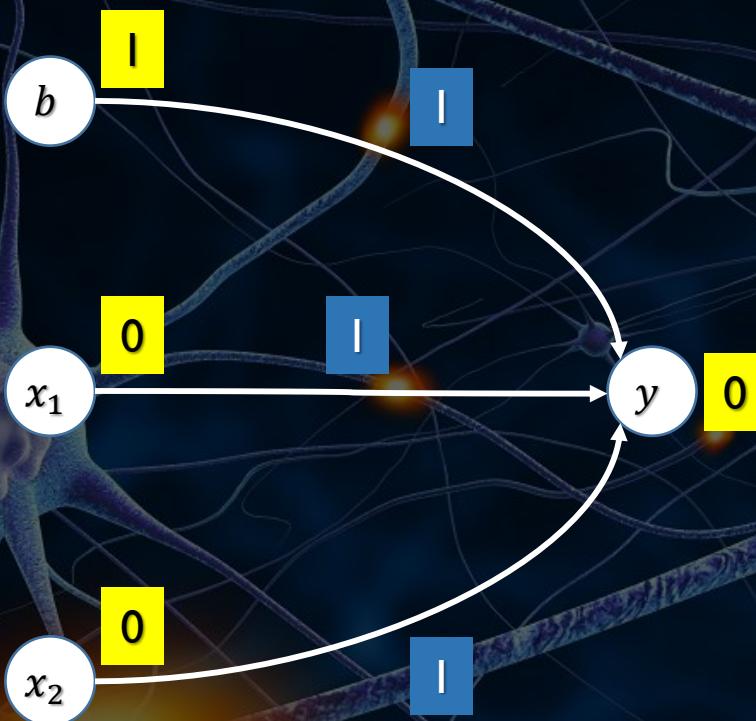
s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



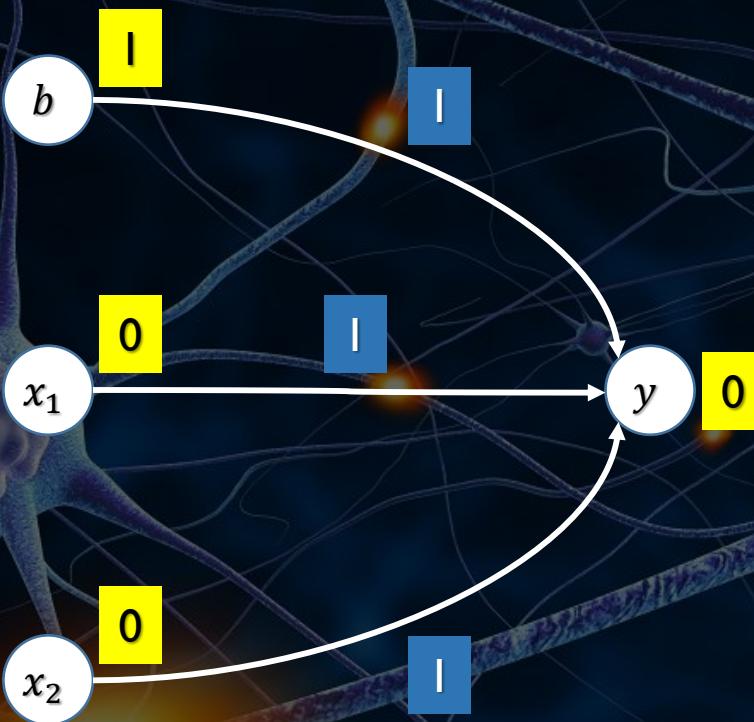
Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$



s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

Training dengan Hebb Rule (I)



Ubah bobot:

$$w'_i = w_i + x_i y$$

$$b' = 1 + 0 = 1$$

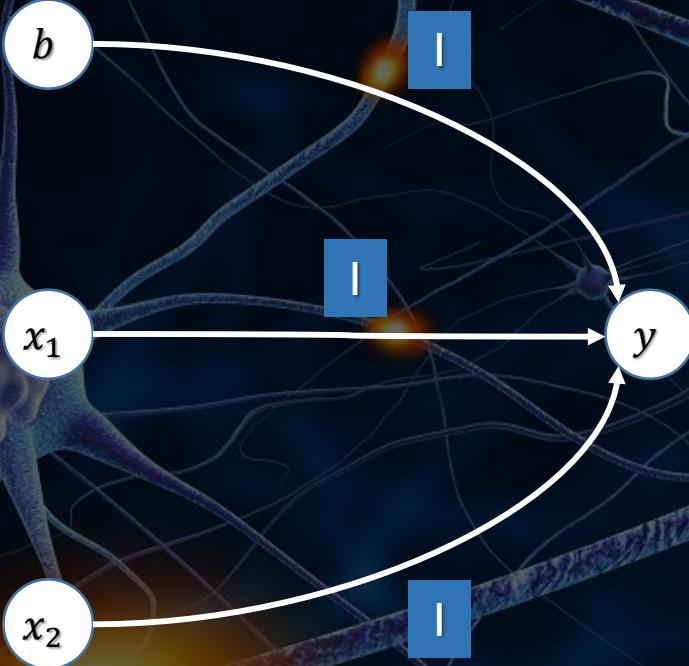
$$w'_1 = 1 + 0.0 = 1$$

$$w'_2 = 1 + 0.0 = 1$$

s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

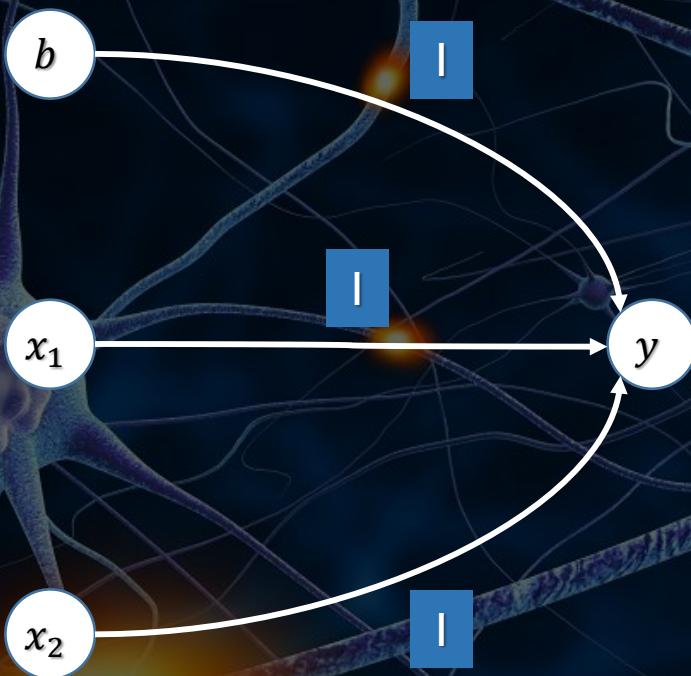
Training dengan Hebb Rule (I)

Proses *training* selesai



s_1	s_2	b	t
1	1	1	1
1	0	1	0
0	1	1	0
0	0	1	0

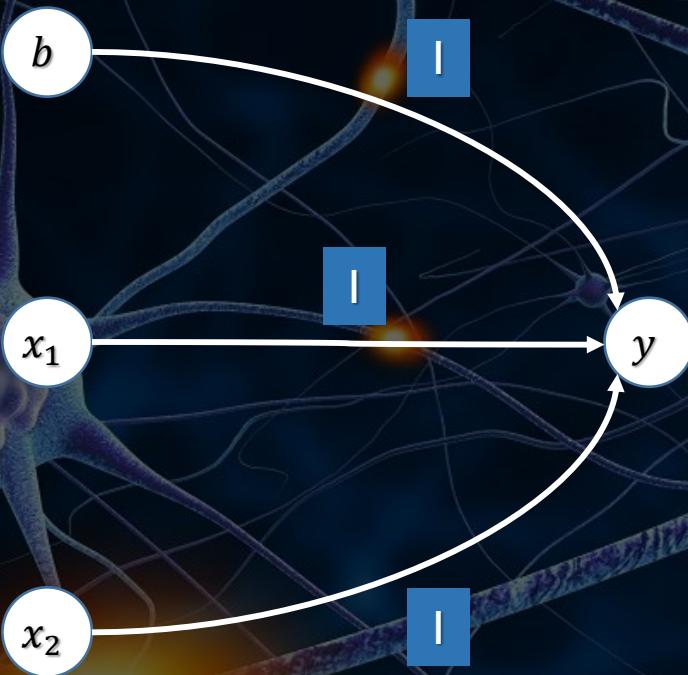
Training dengan Hebb Rule (I)



Aplikasi input logika
AND ke jaringan hasil
training

x_1	x_2	y
1	1	
1	0	
0	1	
0	0	

Training dengan Hebb Rule (I)



Training dengan data latih dan target biner memberikan hasil yang salah

x_1	x_2	y
1	1	1
1	0	1
0	1	1
0	0	1

Training dengan Hebb Rule (I)

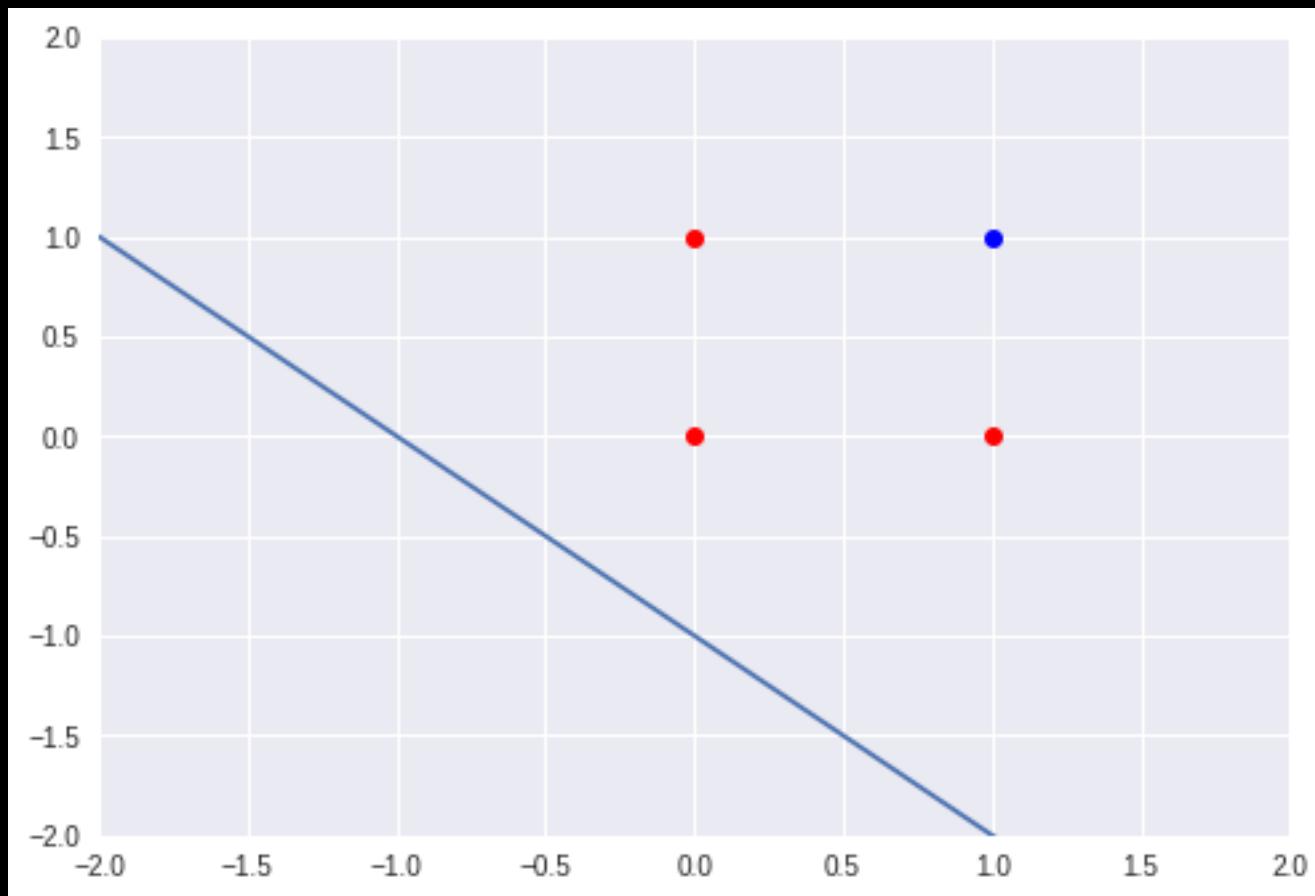
- Persamaan *decision boundary* yang dihasilkan:

$$b + x_1 w_1 + x_2 w_2 = 0$$

$$1 + x_1 + x_2 = 0$$

$$x_2 = -x_1 - 1$$

Training dengan Hebb Rule (I)

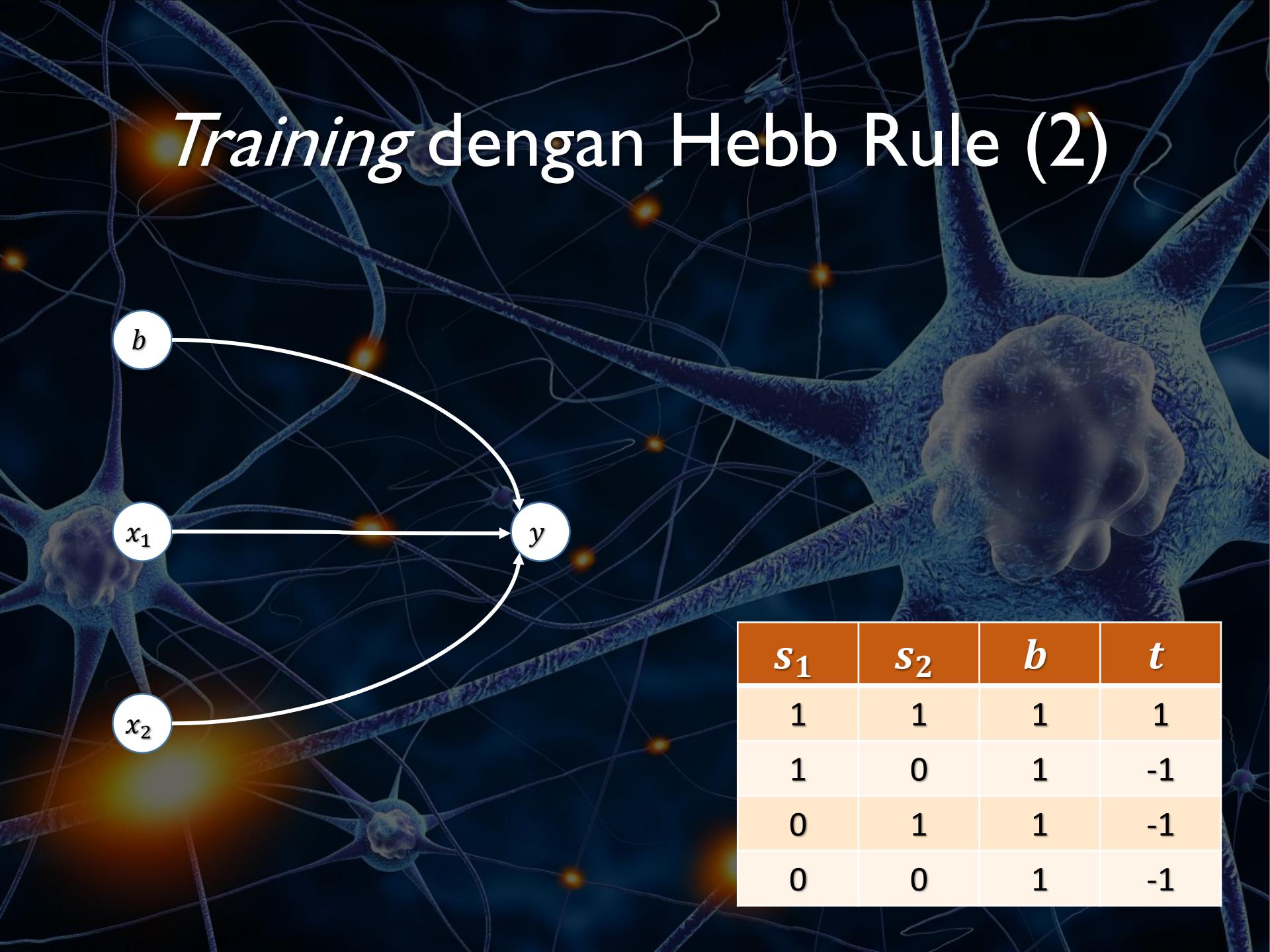


Training dengan Hebb Rule (2)

- Data *training* untuk logika AND dengan data **input biner** dan **target bipolar**:

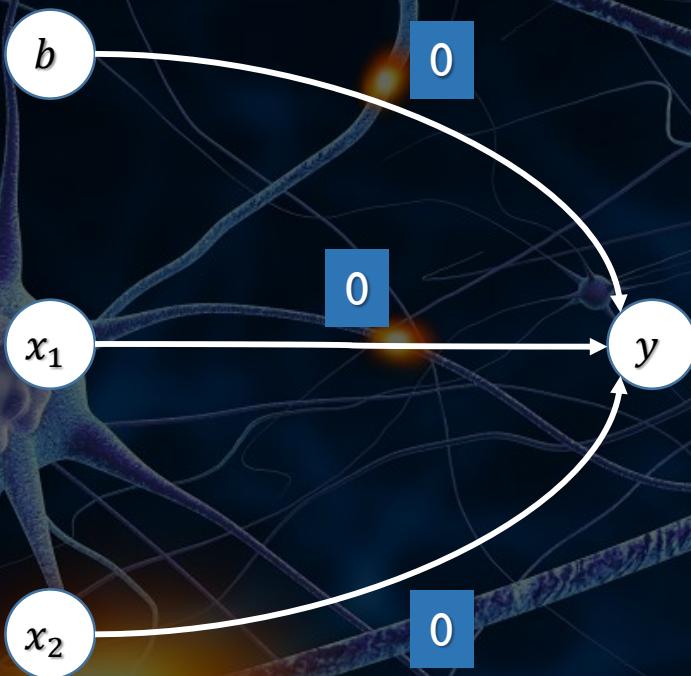
x_1	x_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

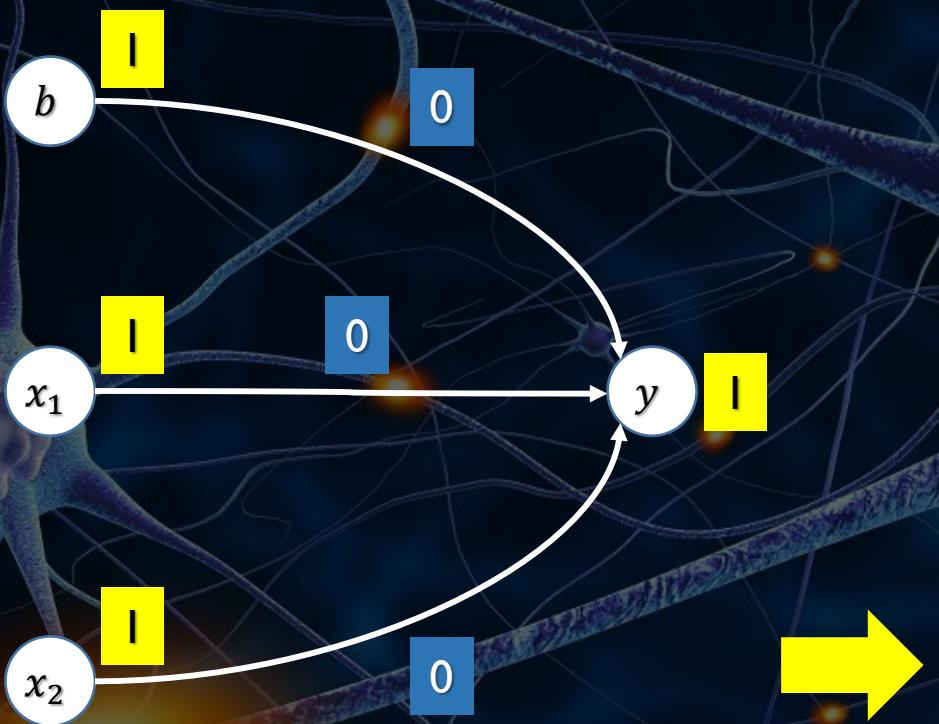
Training dengan Hebb Rule (2)



Inisialisasi semua bobot
dengan nilai 0

s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)

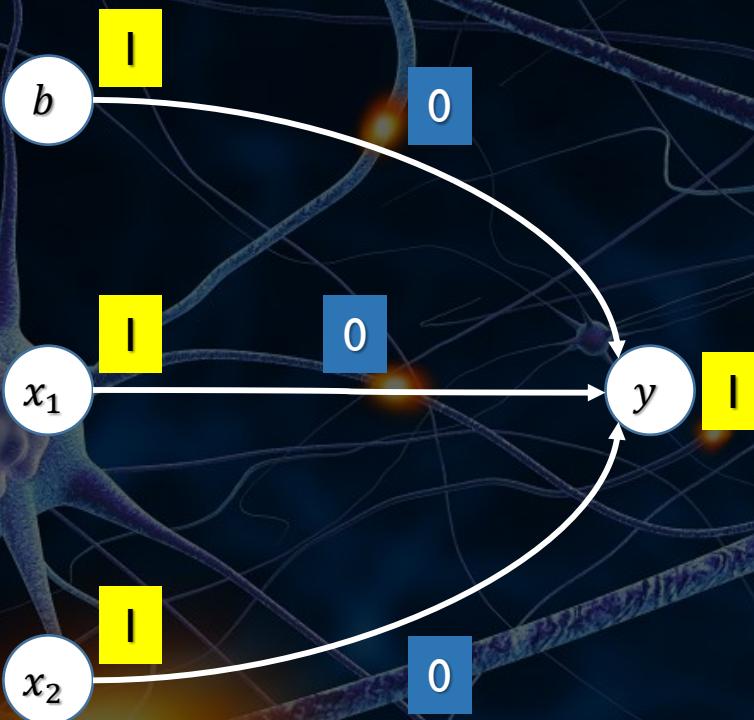


Set nilai aktivasi:

$$x_i = s_i \\ y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



Ubah bobot:

$$w'_i = w_i + x_i y$$

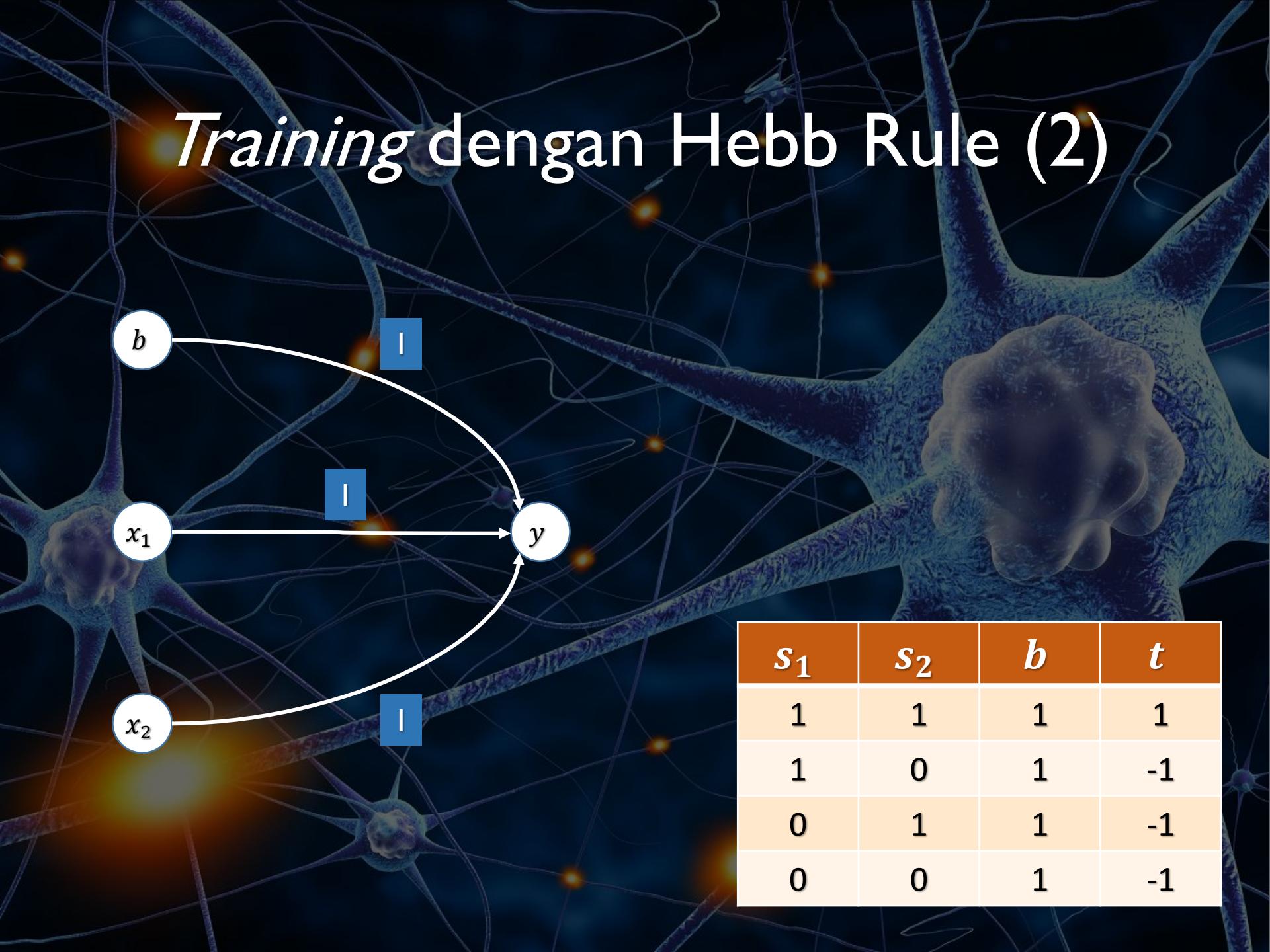
$$b' = 0 + 1 = 1$$

$$w'_1 = 0 + 1 \cdot 1 = 1$$

$$w'_2 = 0 + 1 \cdot 1 = 1$$

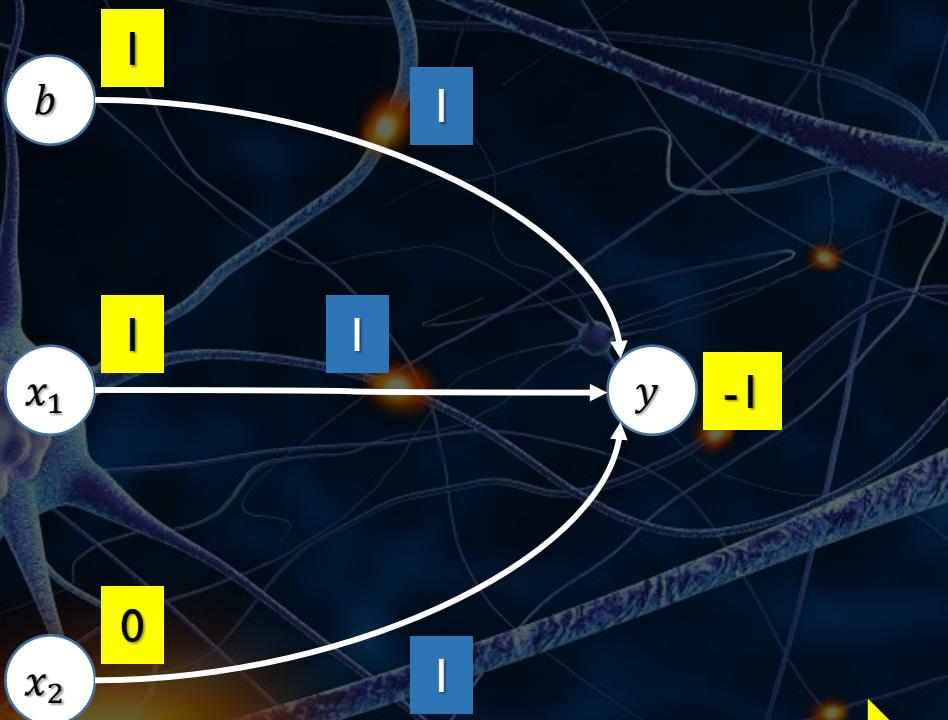
s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)

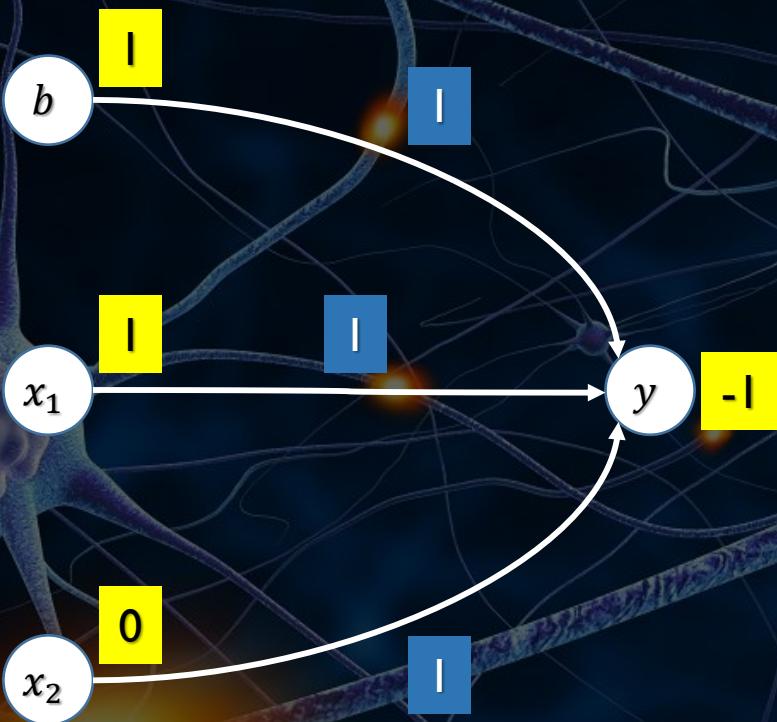


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



Ubah bobot:

$$w'_i = w_i + x_i y$$

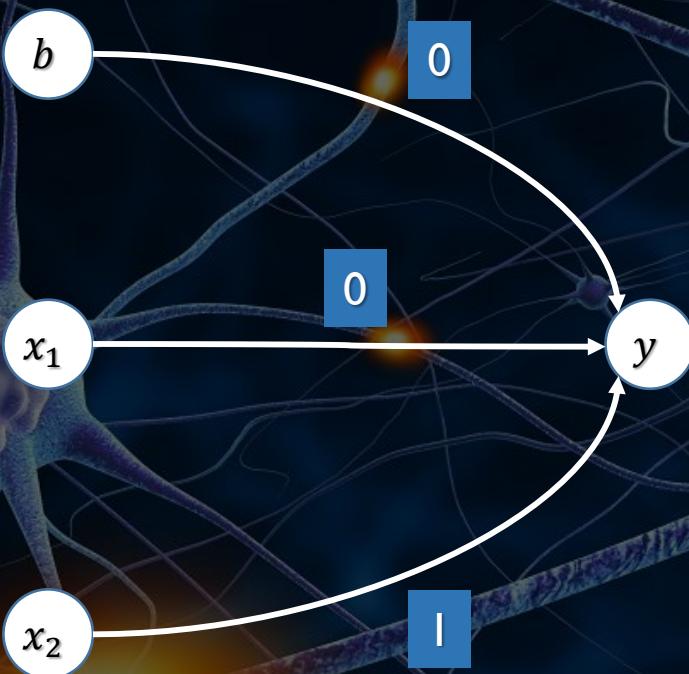
$$b' = 1 - 1 = 0$$

$$w'_1 = 1 + 1 \cdot -1 = 0$$

$$w'_2 = 1 + 0 \cdot -1 = 1$$

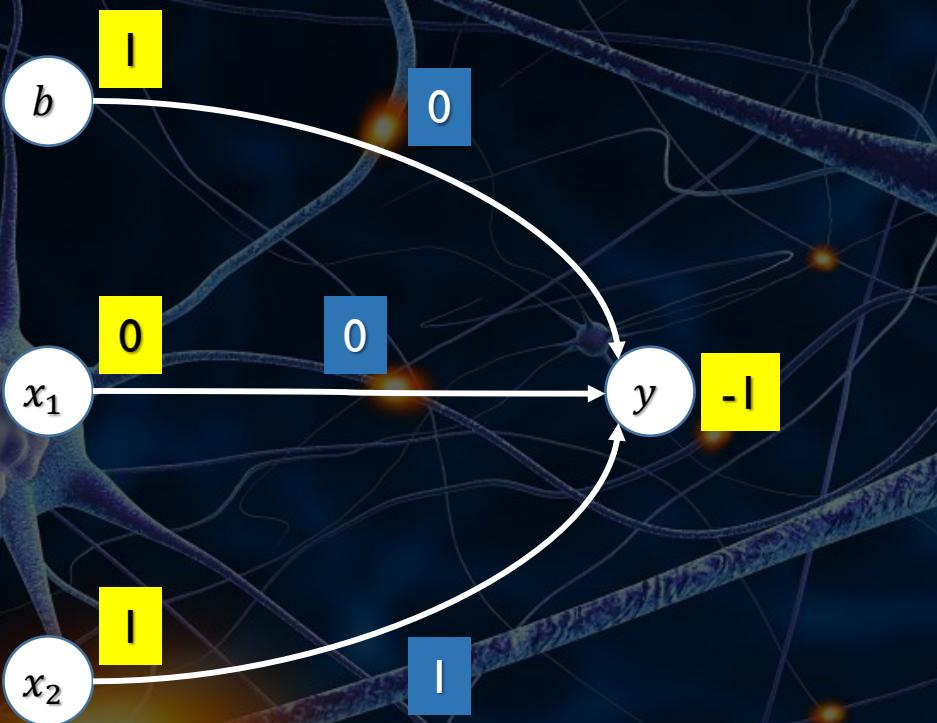
s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)

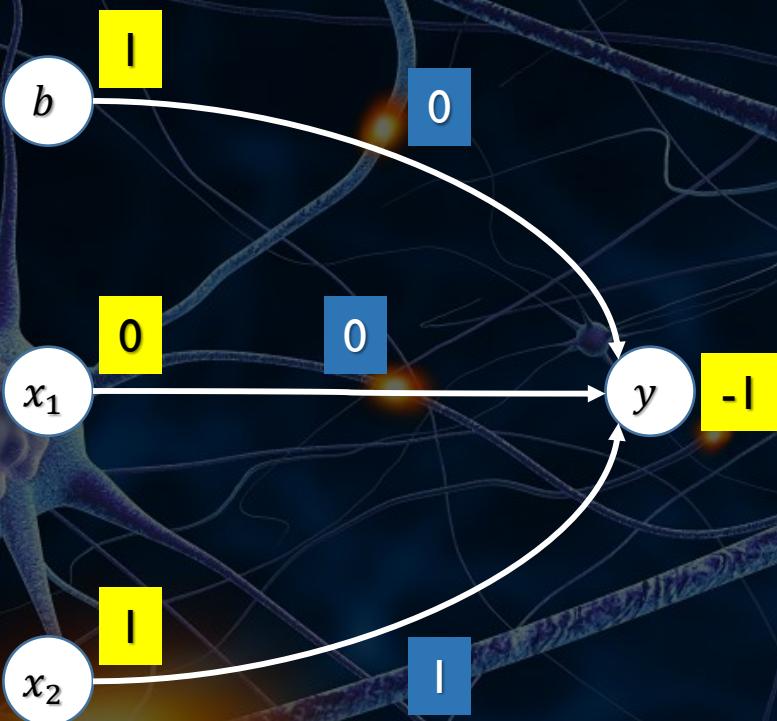


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



Ubah bobot:

$$w'_i = w_i + x_i y$$

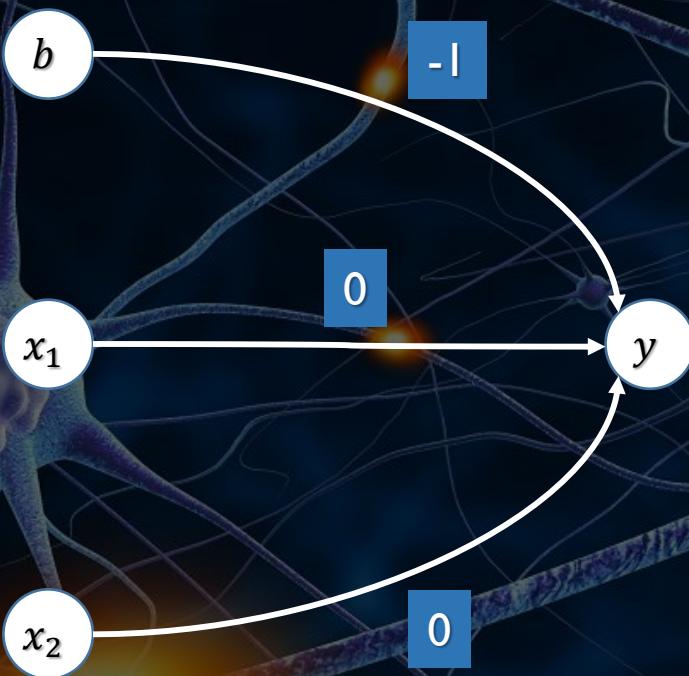
$$b' = 0 - 1 = -1$$

$$w'_1 = 0 + 0 \cdot -1 = 0$$

$$w'_2 = 1 + 1 \cdot -1 = 0$$

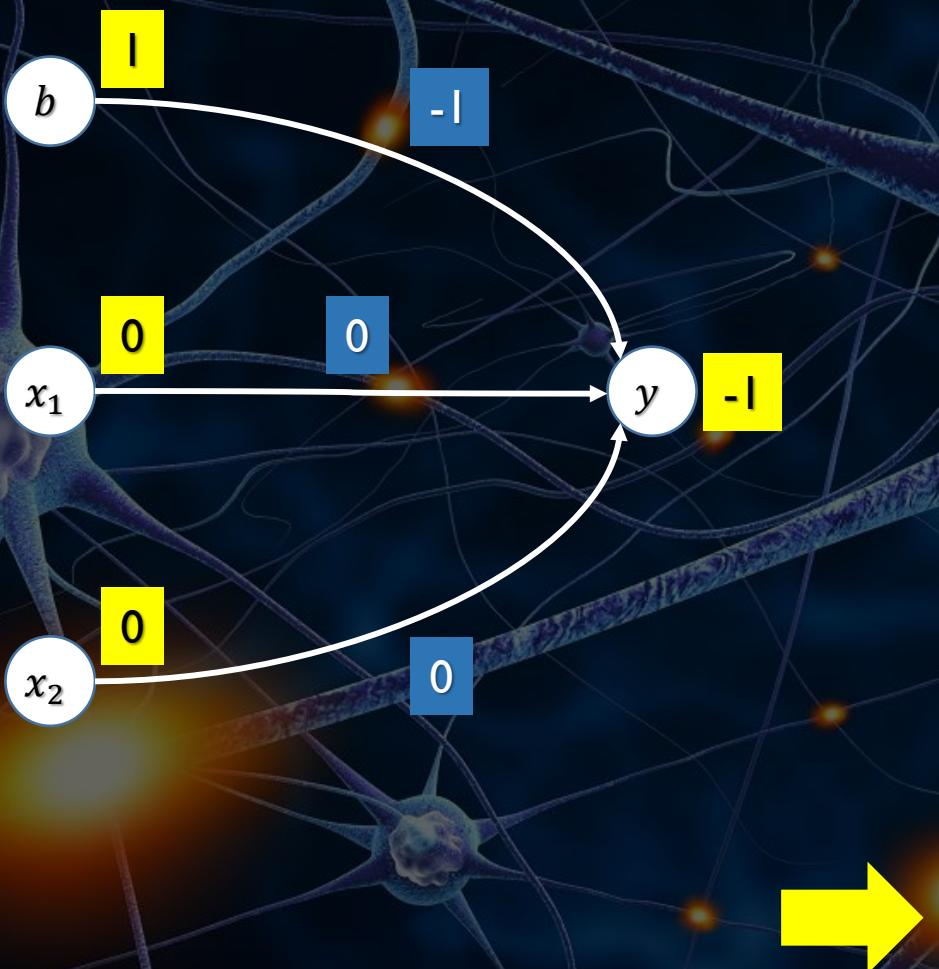
s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)

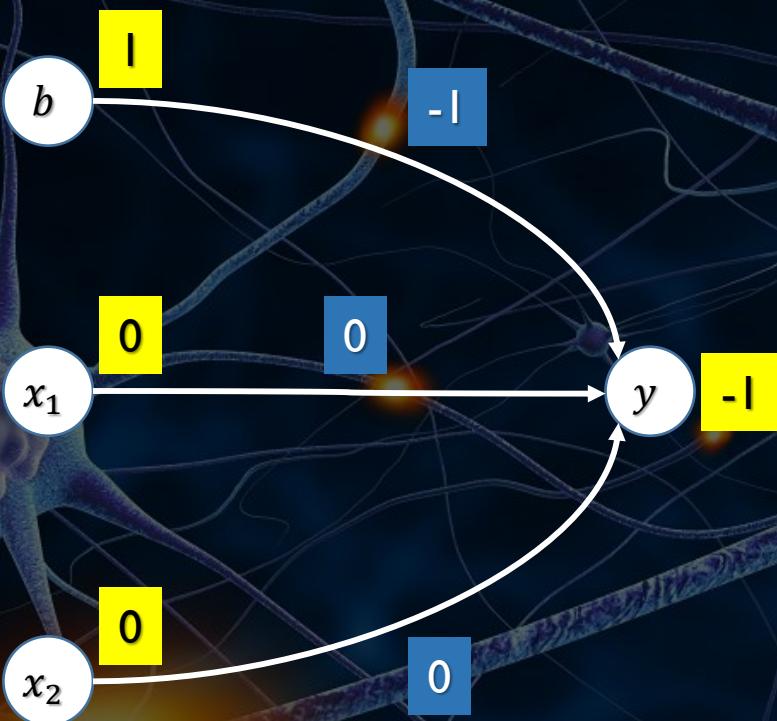


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

Training dengan Hebb Rule (2)



Ubah bobot:

$$w'_i = w_i + x_i y$$

$$b' = -1 - 1 = -2$$

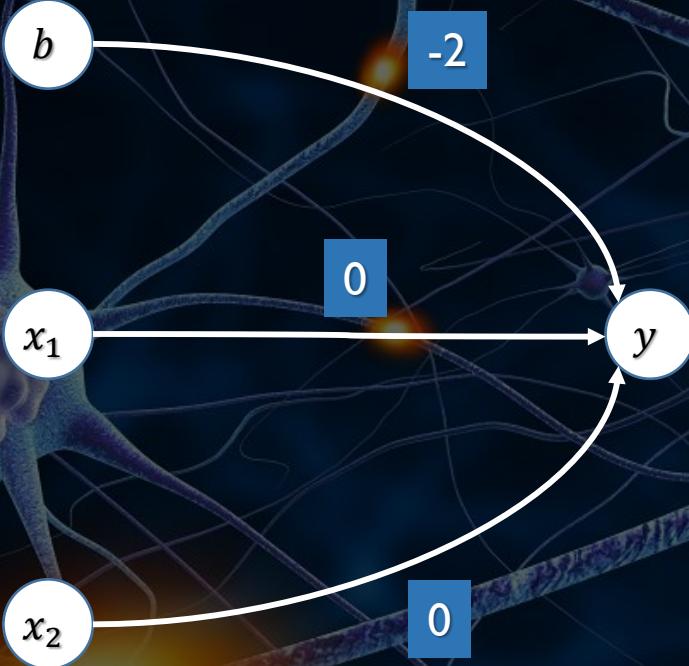
$$w'_1 = 0 + 0 \cdot -1 = 0$$

$$w'_2 = 0 + 0 \cdot -1 = 0$$

s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

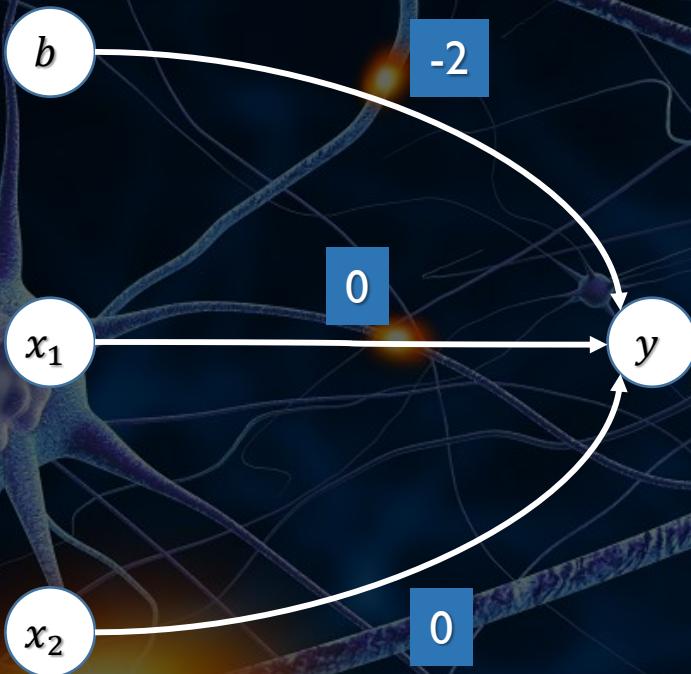
Training dengan Hebb Rule (2)

Proses *training* selesai



s_1	s_2	b	t
1	1	1	1
1	0	1	-1
0	1	1	-1
0	0	1	-1

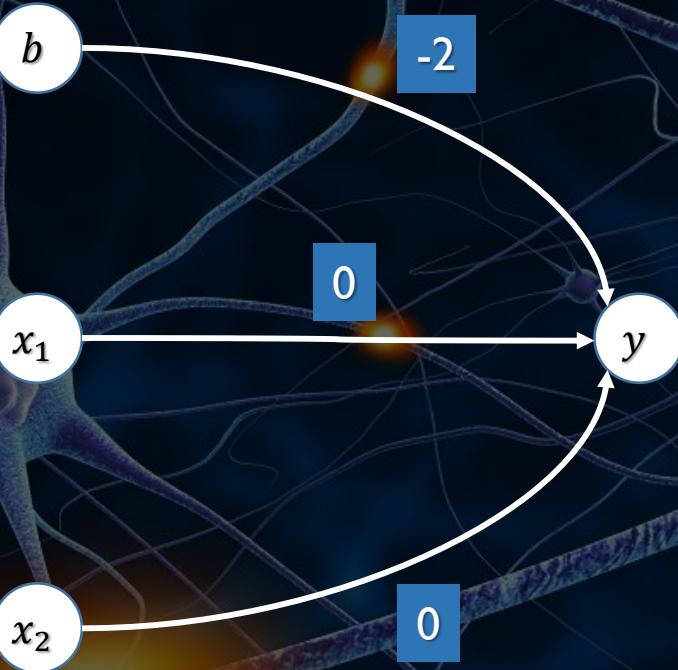
Training dengan Hebb Rule (2)



Aplikasi input logika
AND ke jaringan hasil
training

x_1	x_2	y
1	1	
1	0	
0	1	
0	0	

Training dengan Hebb Rule (2)



Training dengan data latih biner dan target bipolar juga memberikan hasil yang salah

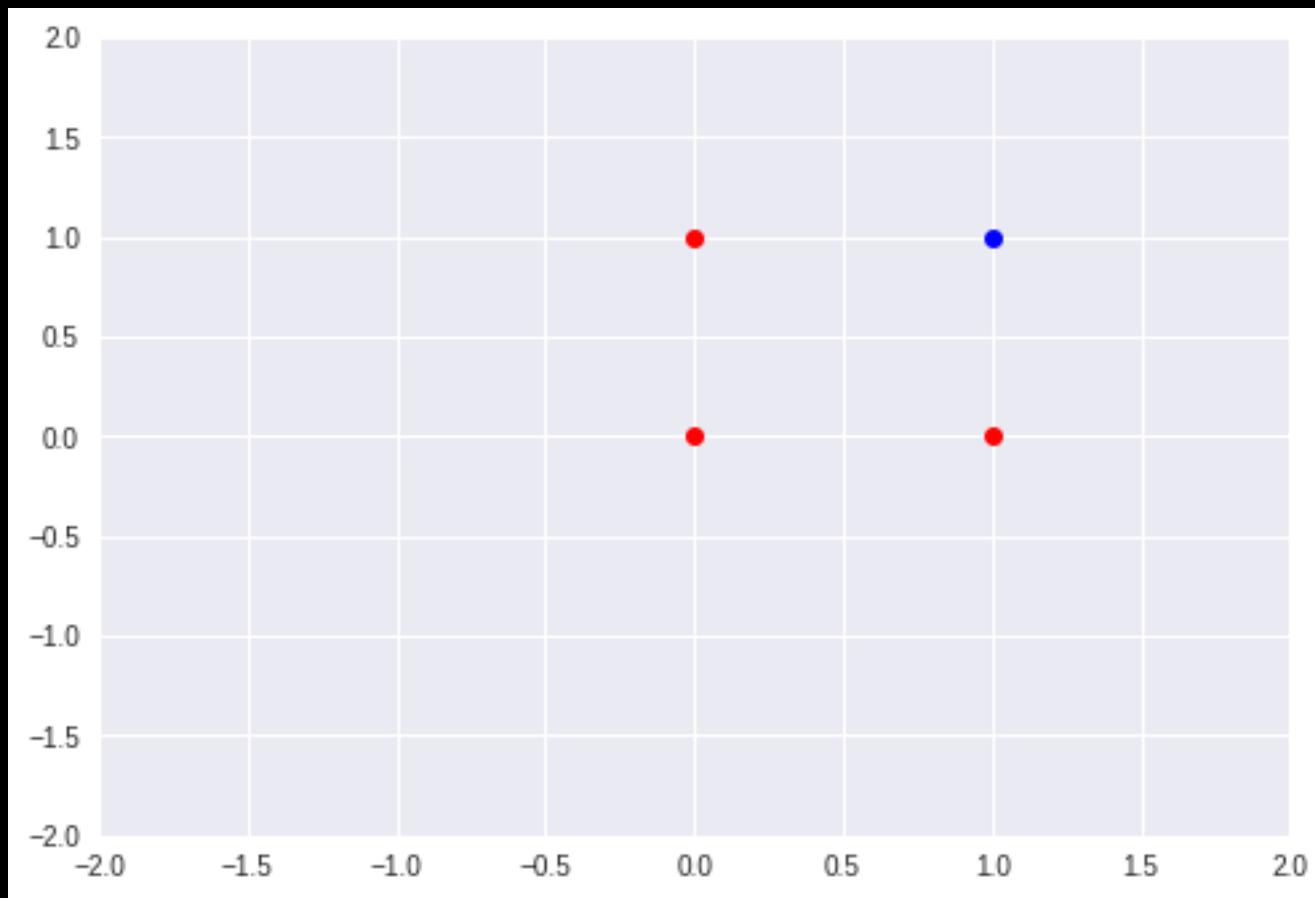
x_1	x_2	y
1	1	-1
1	0	-1
0	1	-1
0	0	-1

Training dengan Hebb Rule (2)

- Persamaan *decision boundary* yang dihasilkan:

$$\begin{aligned} b + x_1 w_1 + x_2 w_2 &= 0 \\ -2 + x_1 0 + x_2 0 &= 0 \end{aligned}$$

Training dengan Hebb Rule (2)

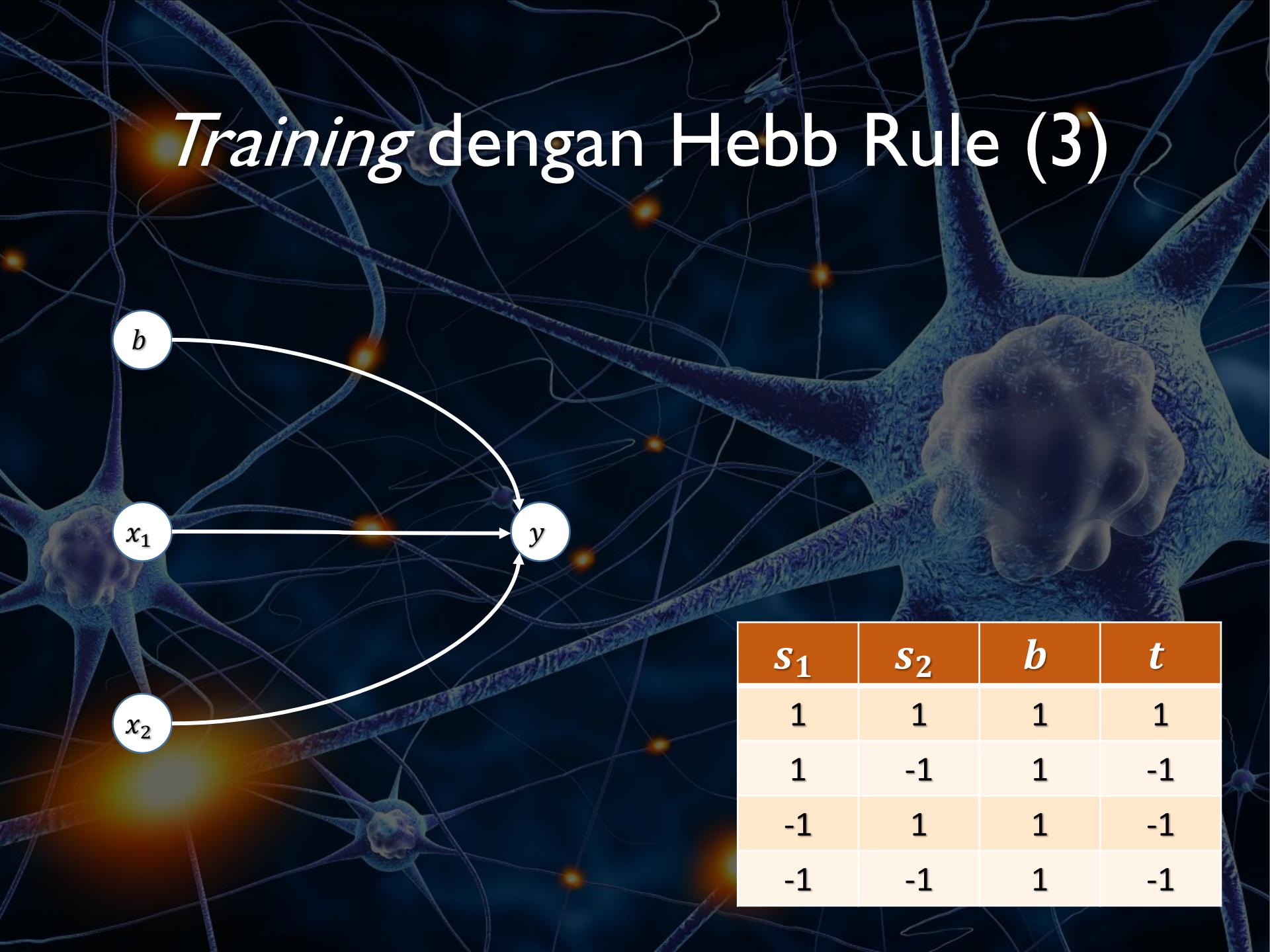


Training dengan Hebb Rule (3)

- Data *training* untuk logika AND dengan data **input dan target bipolar**:

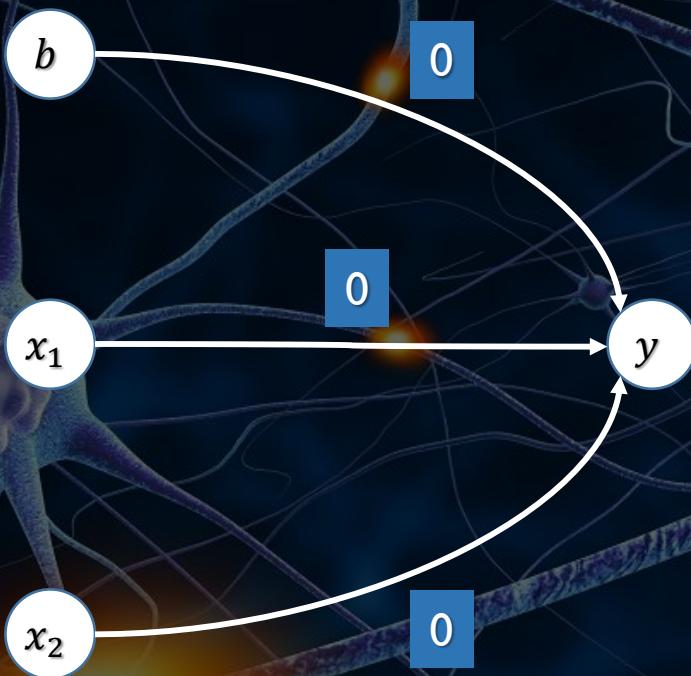
x_1	x_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

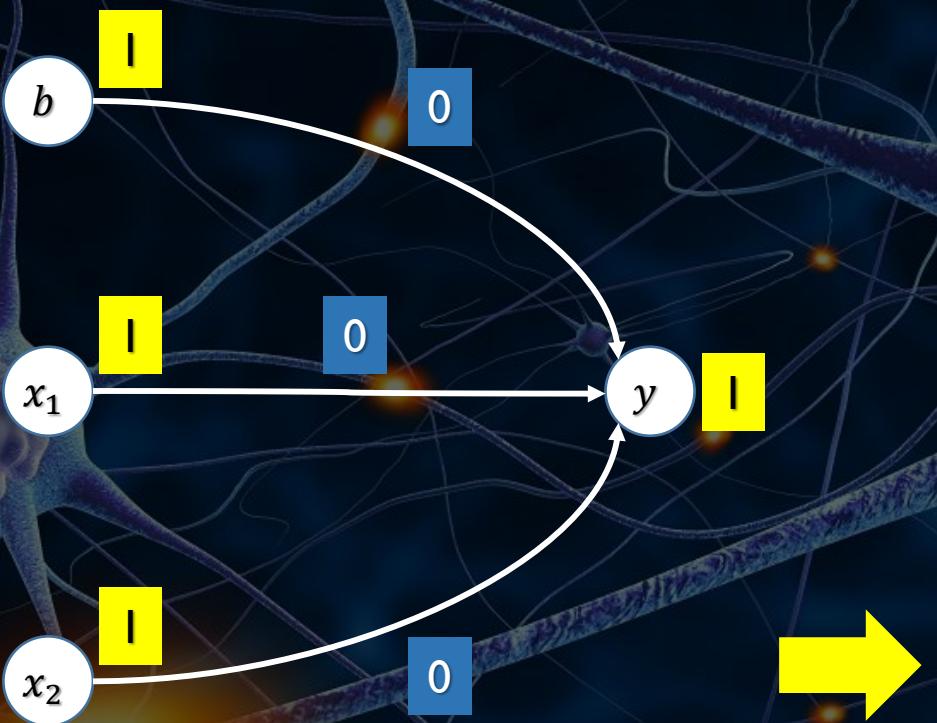
Training dengan Hebb Rule (3)



Inisialisasi semua bobot
dengan nilai 0

s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)

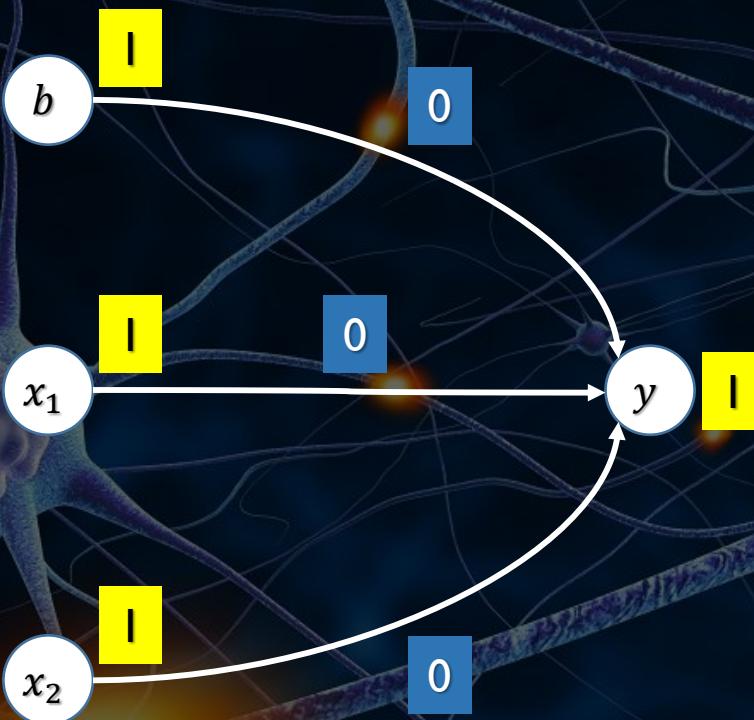


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



Ubah bobot:

$$w'_i = w_i + x_i y$$

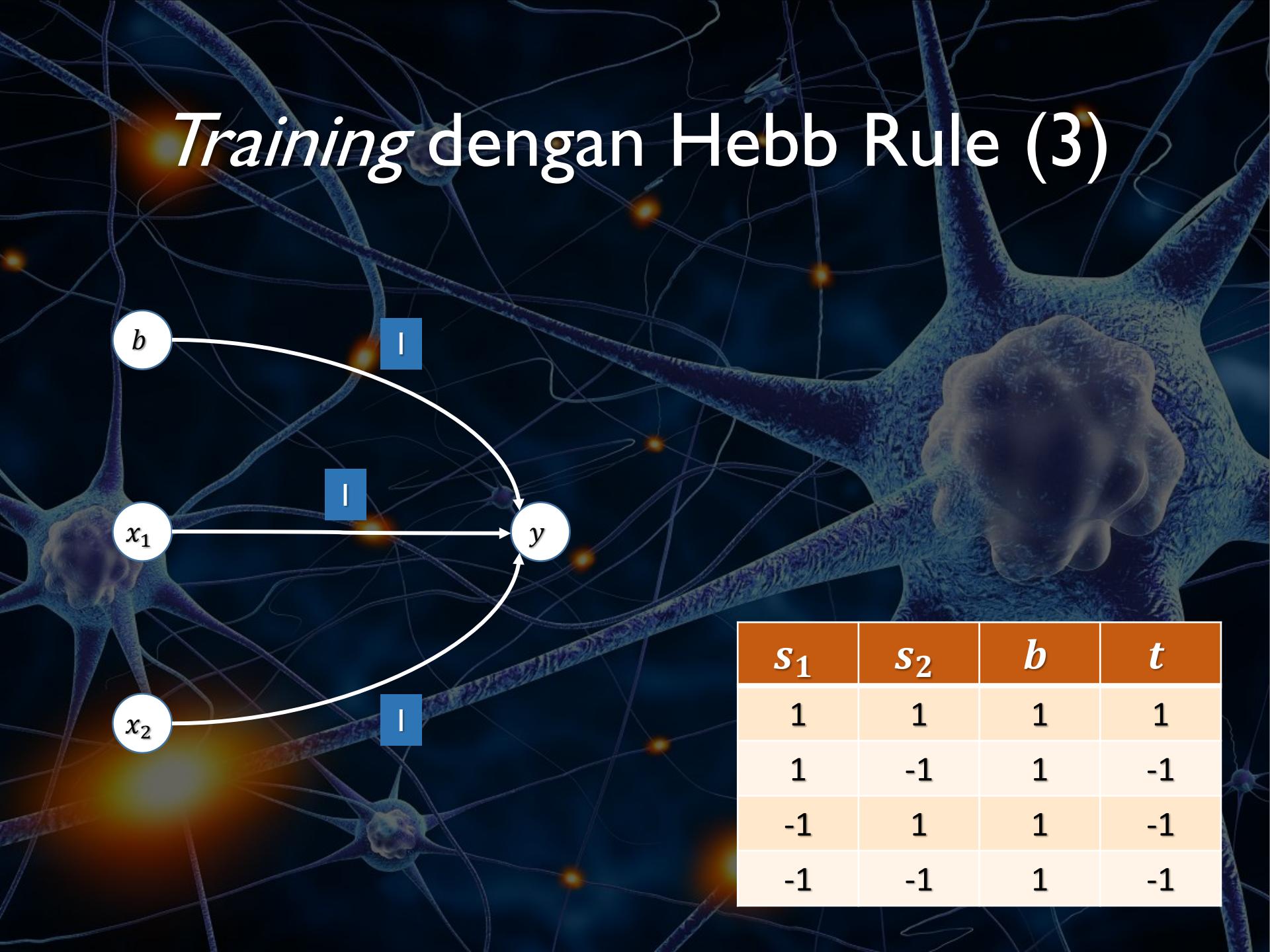
$$b' = 0 + 1 = 1$$

$$w'_1 = 0 + 1 \cdot 1 = 1$$

$$w'_2 = 0 + 1 \cdot 1 = 1$$

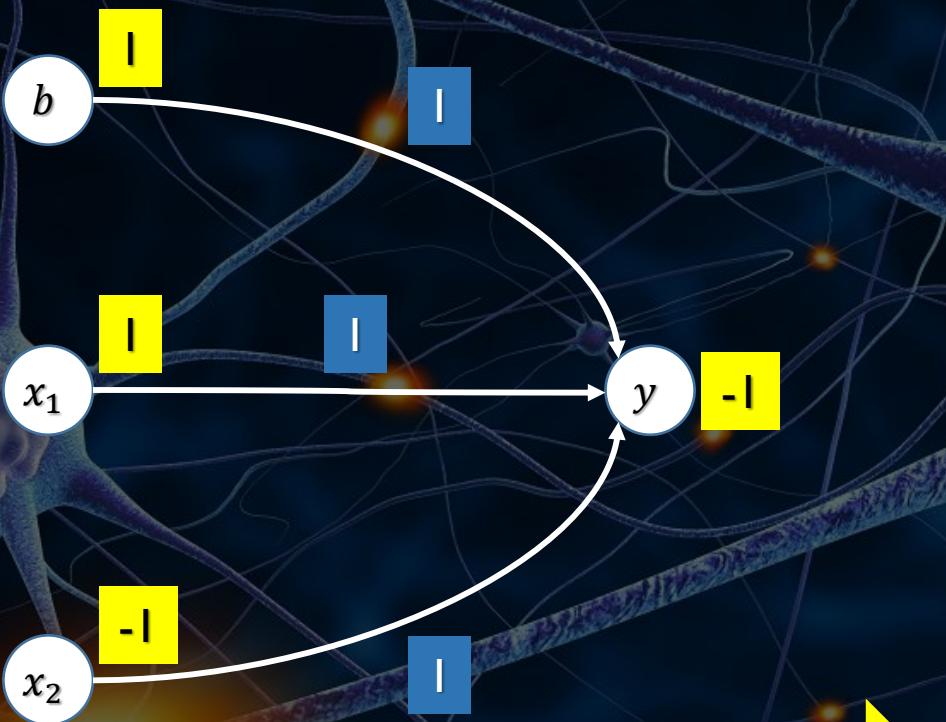
s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)

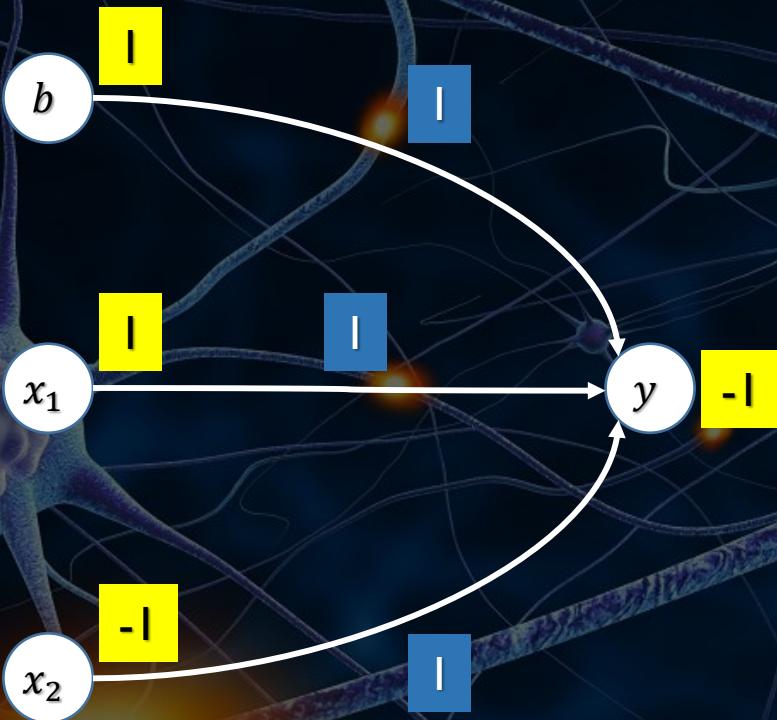


Set nilai aktivasi:

$$x_i = s_i \\ y = t$$

s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



Ubah bobot:

$$w'_i = w_i + x_i y$$

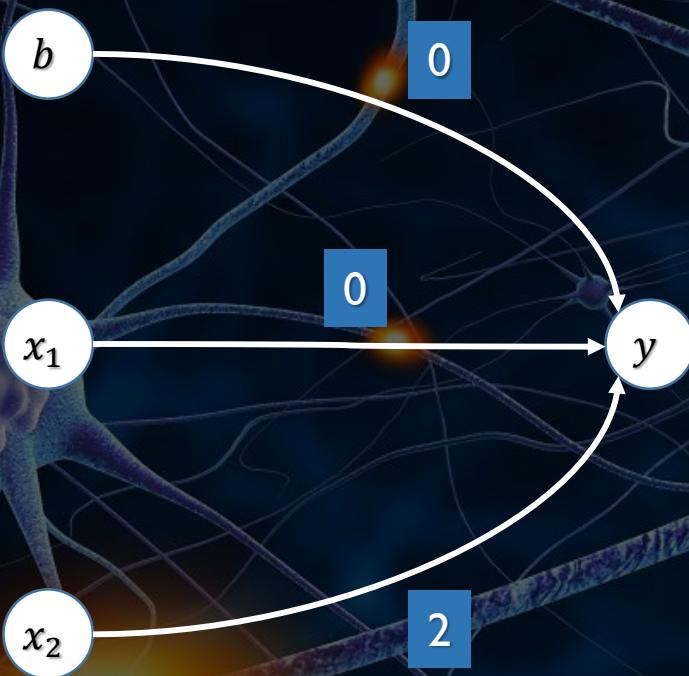
$$b' = 1 - 1 = 0$$

$$w'_1 = 1 + 1 \cdot -1 = 0$$

$$w'_2 = 1 + (-1 \cdot -1) = 2$$

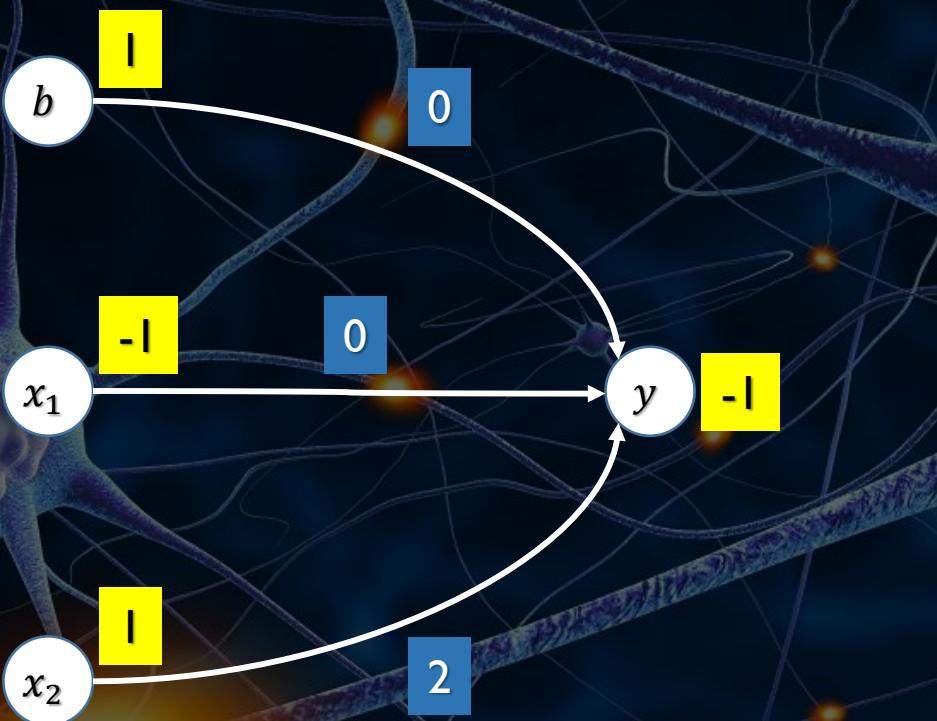
s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)

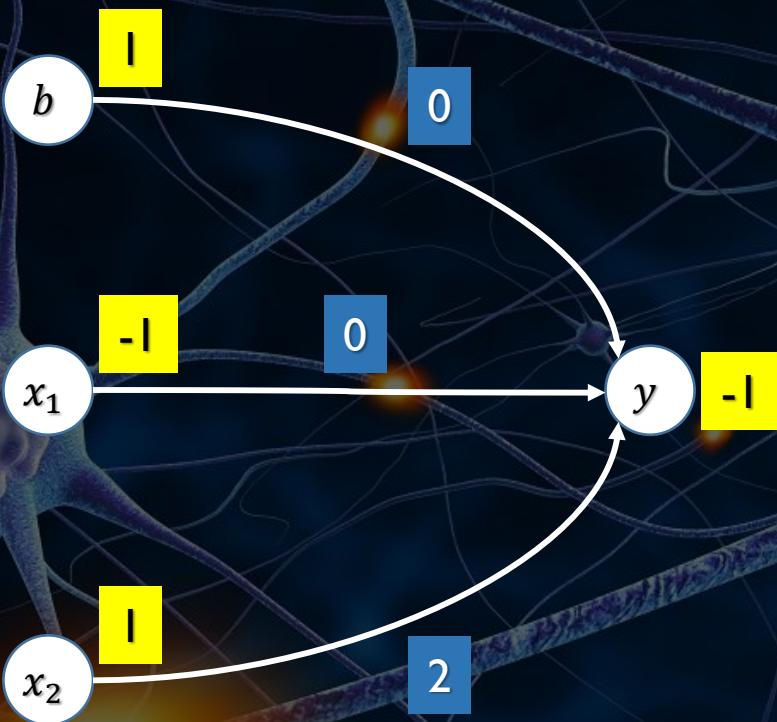


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



Ubah bobot:

$$w'_i = w_i + x_i y$$

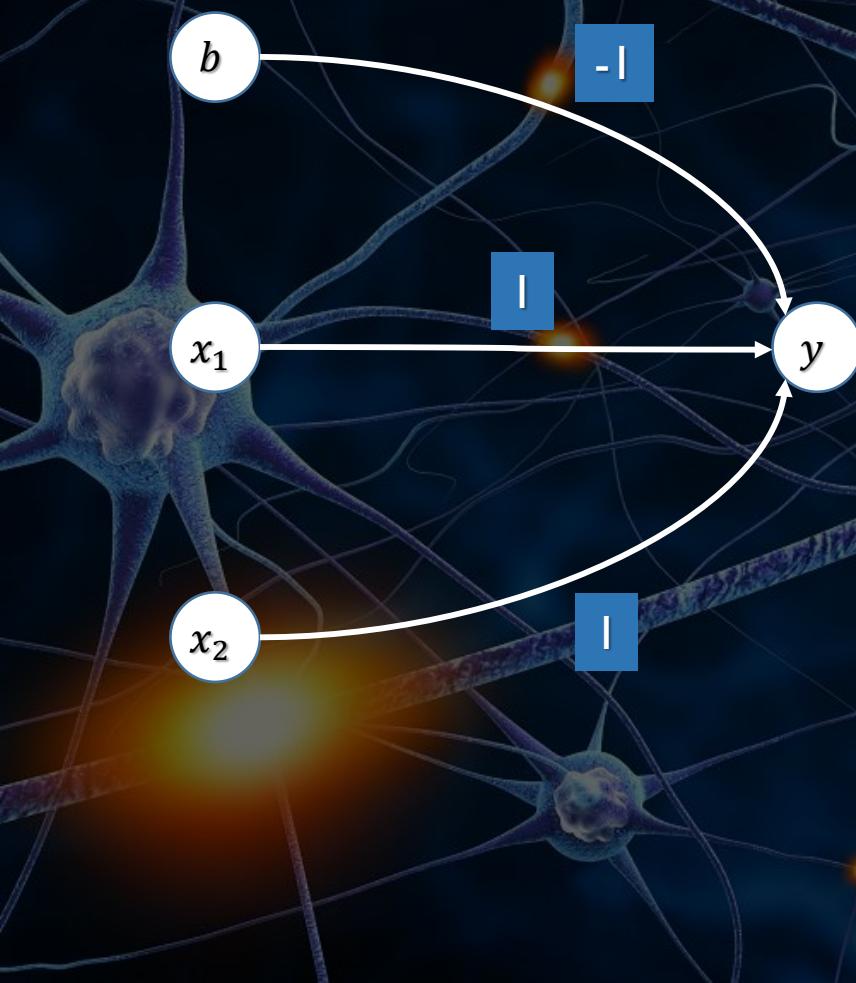
$$b' = 0 - 1 = -1$$

$$w'_1 = 0 + (-1 \cdot -1) = 1$$

$$w'_2 = 2 + 1 \cdot -1 = 1$$

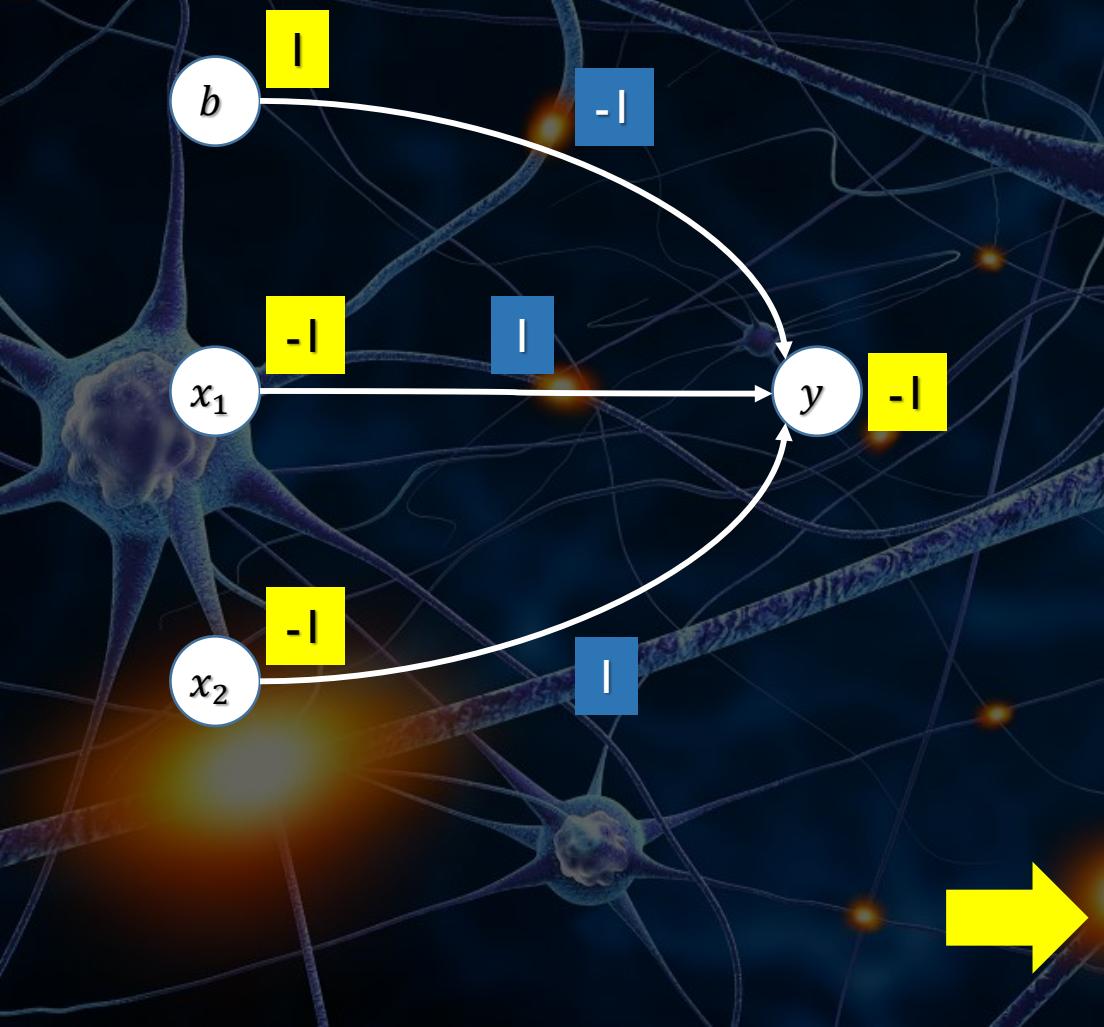
s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)

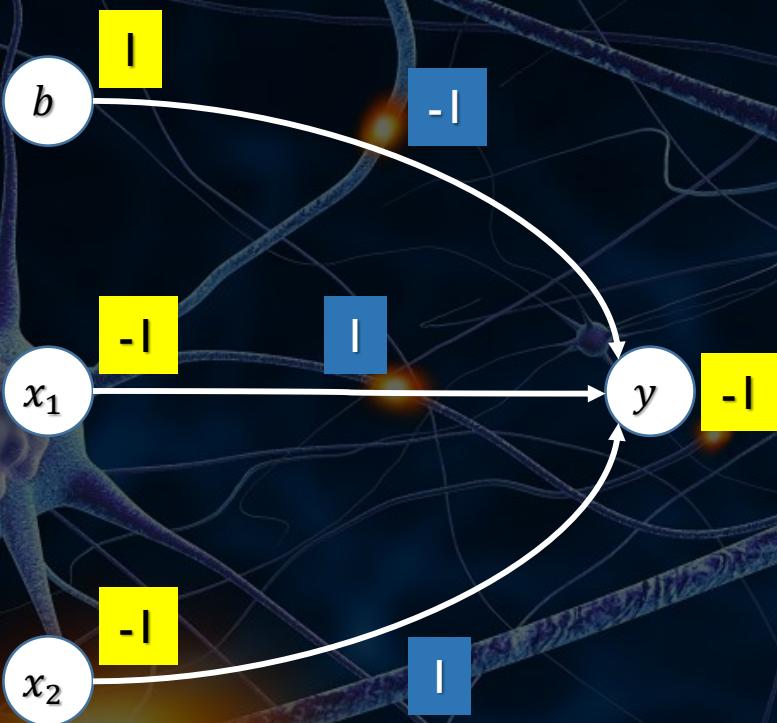


Set nilai aktivasi:

$$x_i = s_i$$
$$y = t$$

s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

Training dengan Hebb Rule (3)



Ubah bobot:

$$w'_i = w_i + x_i y$$

$$b' = -1 - 1 = -2$$

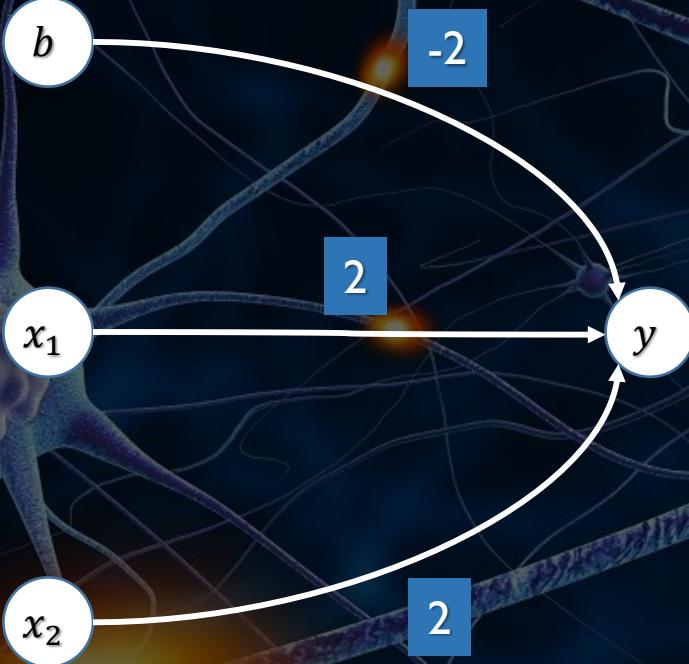
$$w'_1 = 1 + (-1 \cdot -1) = 2$$

$$w'_2 = 1 + (-1 \cdot -1) = 2$$

s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

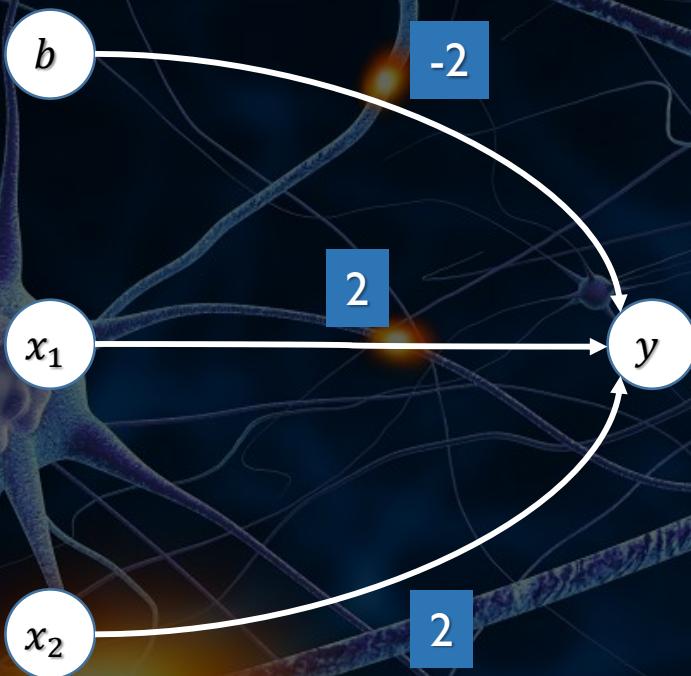
Training dengan Hebb Rule (3)

Proses *training* selesai



s_1	s_2	b	t
1	1	1	1
1	-1	1	-1
-1	1	1	-1
-1	-1	1	-1

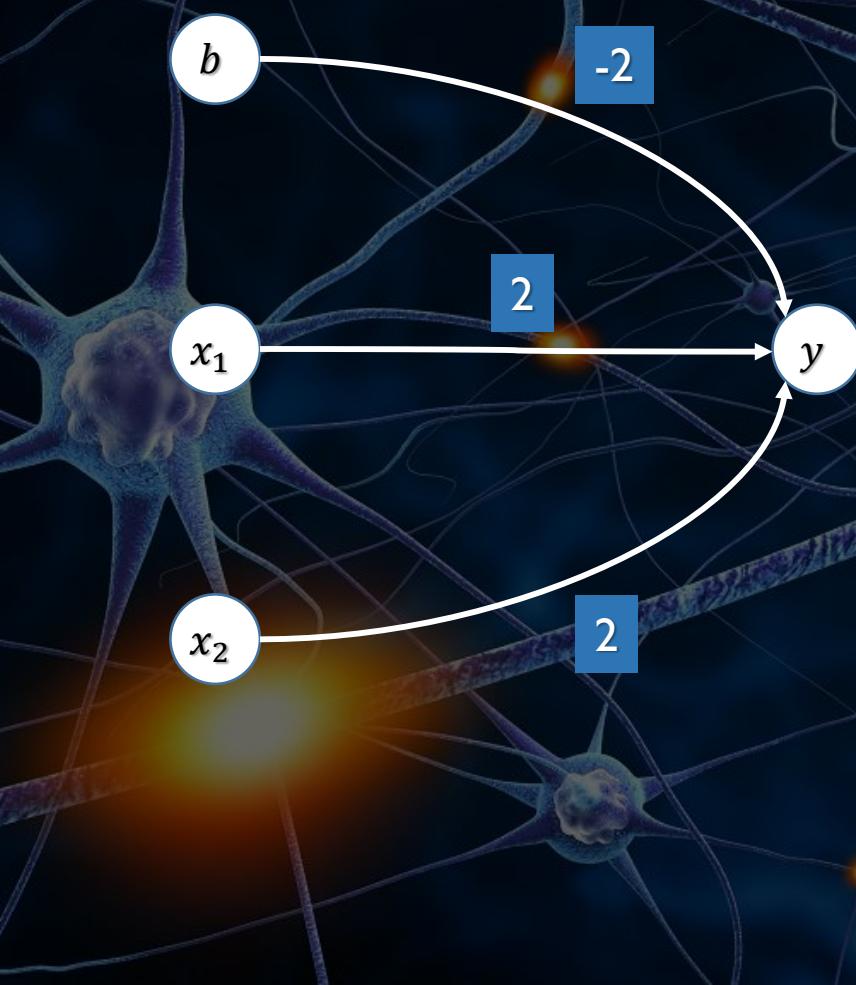
Training dengan Hebb Rule (3)



Aplikasi input logika
AND ke jaringan hasil
training

x_1	x_2	y
1	1	
1	-1	
-1	1	
-1	-1	

Training dengan Hebb Rule (3)



Training dengan input dan target bipolar memberikan hasil yang benar

x_1	x_2	y
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

Training dengan Hebb Rule (3)

- Persamaan *decision boundary* yang dihasilkan:

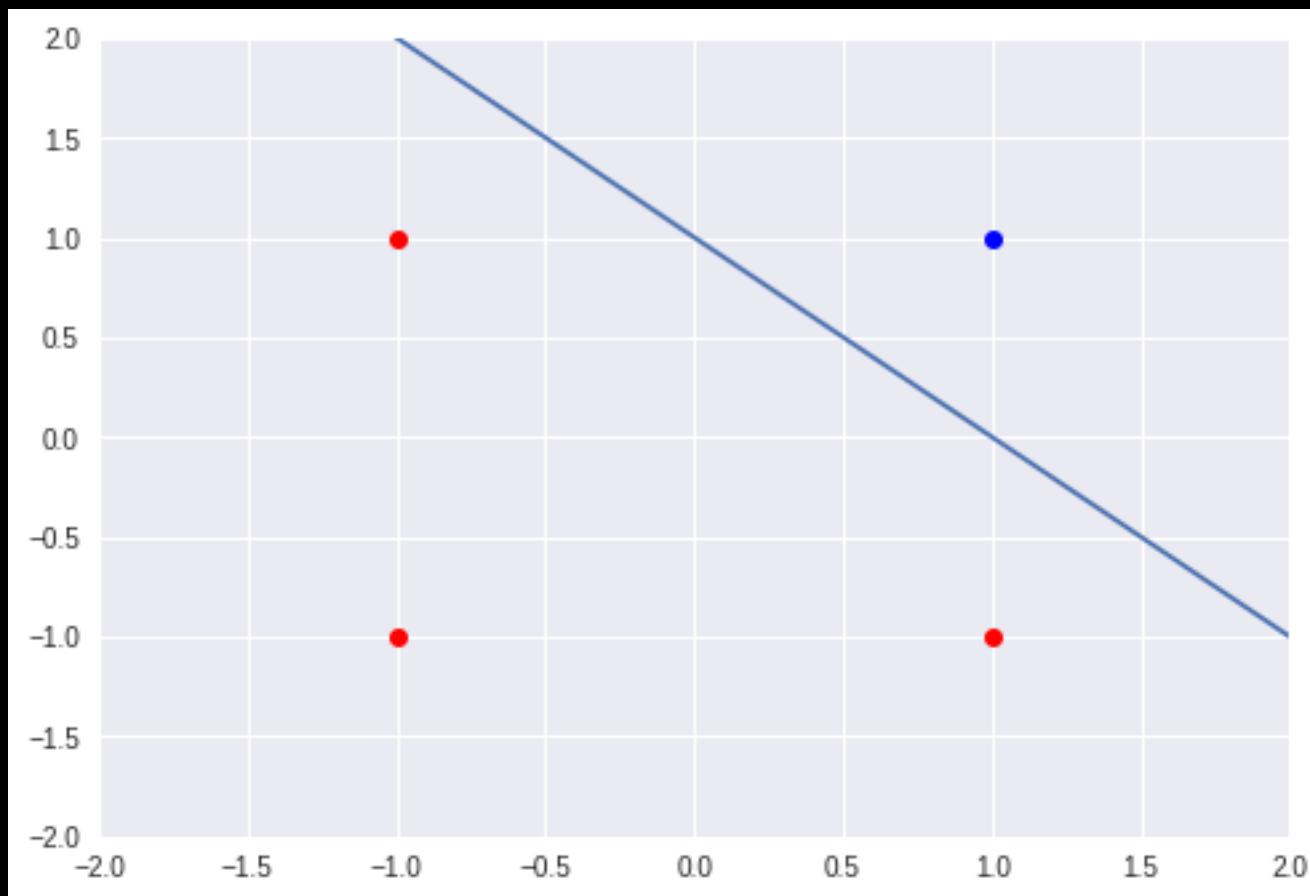
$$b + x_1 w_1 + x_2 w_2 = 0$$

$$-2 + 2x_1 + 2x_2 = 0$$

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

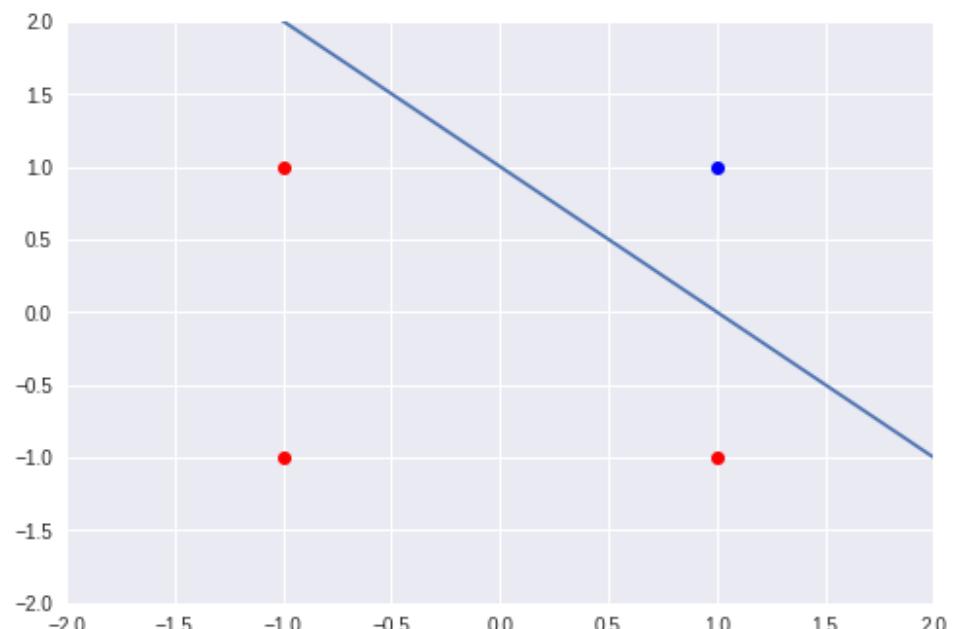
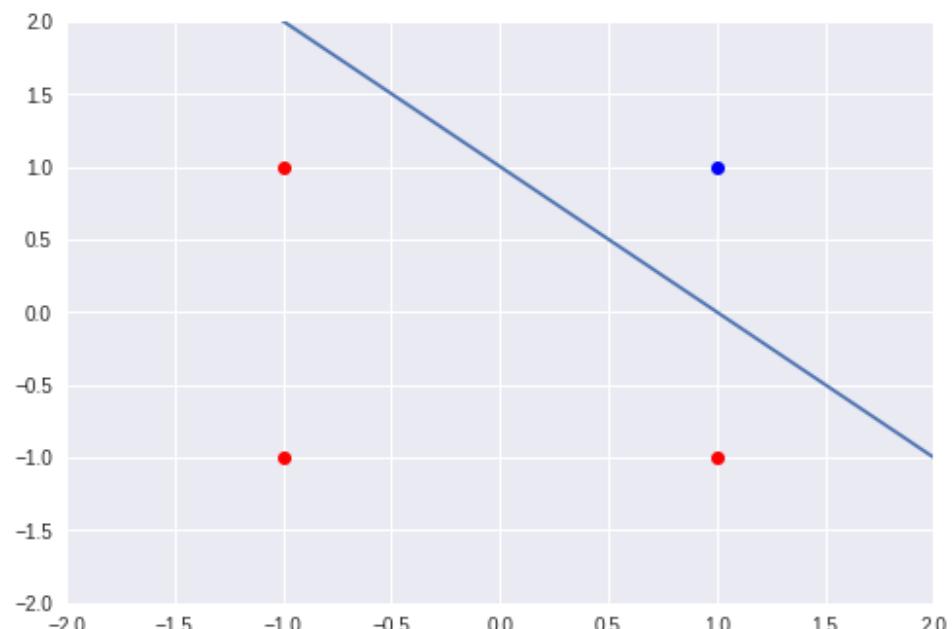
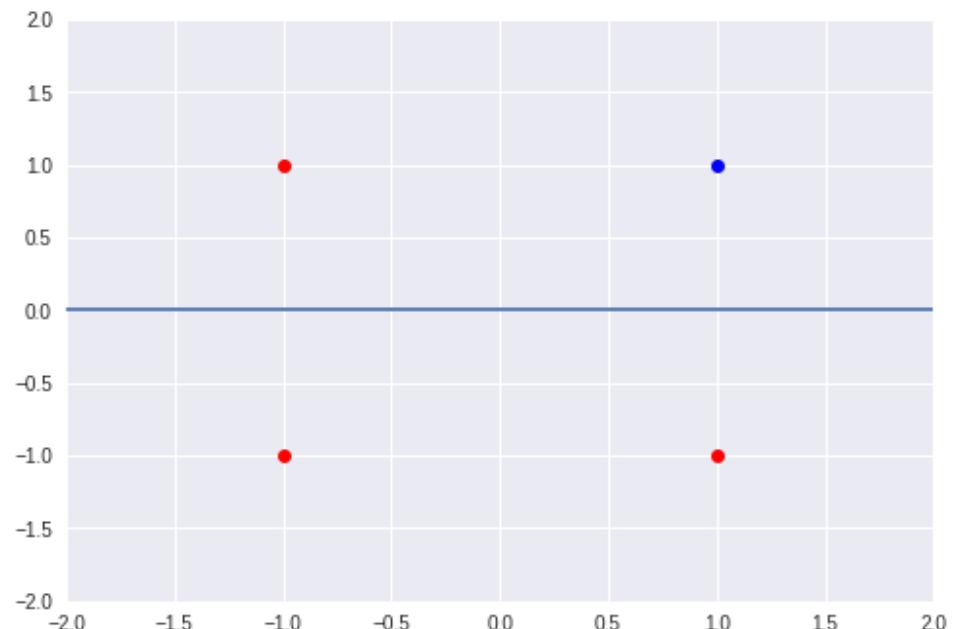
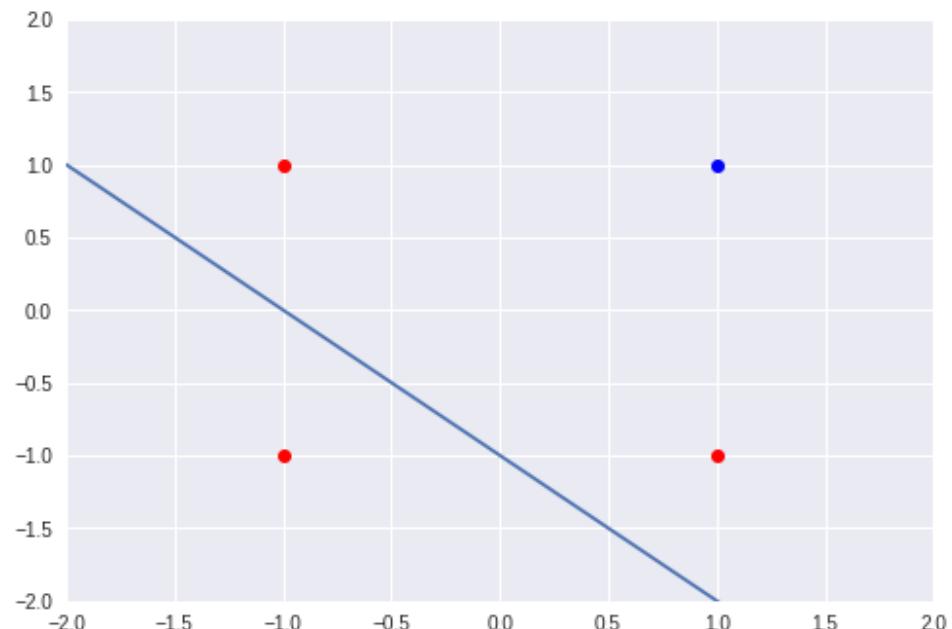
$$x_2 = -x_1 + 1$$

Training dengan Hebb Rule (3)



Training dengan Hebb Rule (3)

- Hebb rule sesuai untuk data input dan target **bipolar**
- Selama proses training, *decision boundary* akan bergeser pada saat dimasukkan data *training*



Implementasi

```
import numpy as np
import matplotlib.pyplot as plt

def bipstep(y, th=0):
    return 1 if y >= th else -1

def line(w, th=0):
    w2 = w[2] + .001 if w[2] == 0 else w[2]

    return lambda x: (th - w[1] * x - w[0]) / w2
```

Implementasi

```
def plot(f, s, t):
    x = np.arange(-2, 3)
    col = 'ro', 'bo'

    for c, v in enumerate(np.unique(t)):
        p = s[np.where(t == v)]

        plt.plot(p[:,1], p[:,2], col[c])

    plt.axis([-2, 2, -2, 2])
    plt.plot(x, f(x))
    plt.show()
```

Implementasi

```
def hebb_train(s, t, draw=False):
    w = np.zeros(len(s[0]) + 1)
    b = np.ones((len(s), 1))
    s = np.hstack((b, s))

    for r, row in enumerate(s):
        w = [w[i] + row[i] * t[r] for i in range(len(row))]

    print('Bobot: {}'.format(w))

    if draw:
        plot(line(w, 0), s, t)

return w
```

Implementasi

```
def hebb_test(x, w):  
    y_in = w[0] + np.dot(x, w[1:])  
  
    return bipstep(y_in)
```

Implementasi

```
# Logika AND
```

```
train = [[1, 1], [1, -1], [-1, 1], [-1, -1]]
```

```
target = [1, -1, -1, -1]
```

```
w = hebb_train(train, target, True)
```

```
print(hebb_test([-1, -1], w))
```

Implementasi

```
# Logika OR
```

```
train = [[1, 1], [1, -1], [-1, 1], [-1, -1]]
```

```
target = [1, 1, 1, -1]
```

```
w = hebb_train(train, target, True)
```

```
print(hebb_test([-1, -1], w))
```

Implementasi

```
# Logika XOR
```

```
train = [[1, 1], [1, -1], [-1, 1], [-1, -1]]
```

```
target = [-1, 1, 1, -1]
```

```
w = hebb_train(train, target, True)
```

```
print(hebb_test([-1, -1], w))
```



Pengenalan Karakter

Pengenalan Karakter

- Membedakan huruf X dan O

#	.	.	.	#	.	#	#	#	.
.	#	.	#	.	#	.	.	.	#
.	.	#	.	.	#	.	.	.	#
.	#	.	#	.	#	.	.	.	#
#	.	.	.	#	.	#	#	#	.

- Karakter # bernilai 1; karakter . bernilai 0

Pengenalan Karakter

- Karakter X: 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
- Karakter O: -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

#	.	.	.	#	.	#	#	#	.
.	#	.	#	.	#	.	.	.	#
.	.	#	.	.	#	.	.	.	#
.	#	.	#	.	#	.	.	.	#
#	.	.	.	#	.	#	#	#	.

Pengenalan Karakter

- Nilai target karakter X adalah 1
- Nilai target karakter O adalah -1

Pengenalan Karakter

```
# Pengenalan karakter
```

```
x = [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]
o = [-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]
train = [x, o]
target = [1, -1]
w = hebb_train(train, target)
y = hebb_test(x, w)

print(y)
```



Alhamdulillah.