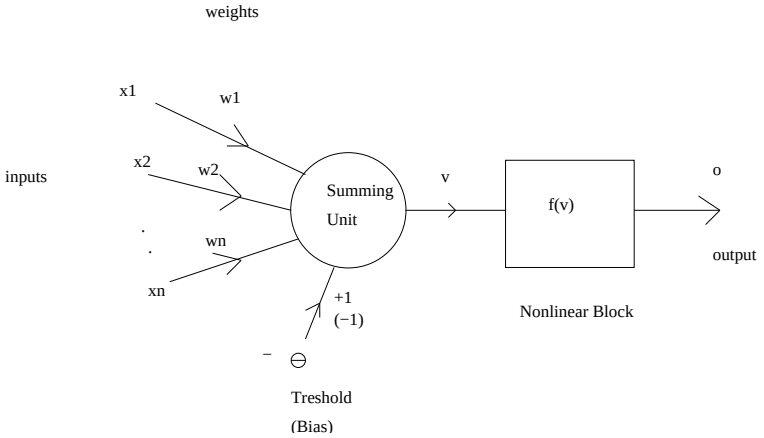
Neural Networks

Neural Network Structures

- **McCulloch-Pitts Model:** $\Longrightarrow$. 1943



- $\Sigma = w_1x_1 + w_2x_2 + \ldots + x_nx_n$, $v = \Sigma \pm \theta \rightarrow o = f(v)$.

- This model has some limitations :

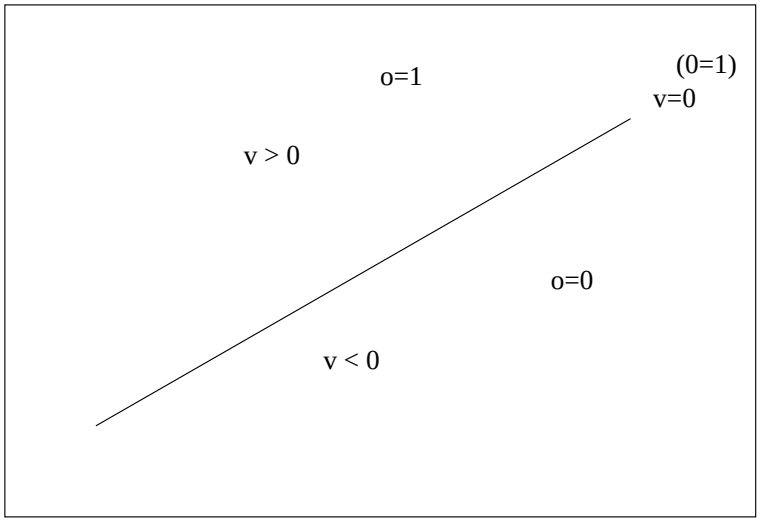
- $w_i = \pm 1$ ($+1$ for excitory synapse, -1 for inhibitory synapse)

- $x_i = 1$ or 0 (digital analogy : $+1$: signal, 0 : no signal)

- $f(v)$ has special form : **treshold function, hard limiter**

- $$f(v) = \begin{cases} +1 & \Sigma \geq \theta \\ 0 & \Sigma < \theta \end{cases}$$

• **Plane analogy:** $v = 0$ line (hyperplane for higher dimensions) separates the input space into two parts.



v=w1x1 + ...+wnxn − ⊖

Linear Separation

• **Conclusion:** If the patterns remain in one side for one class of inputs, and on the other side for the second class → separation is possible by looking at the output → **Linear Separability**

• Basic logic operations fall into this class, hence can be realized by this model → similar to ALU of a classical computer.

• **Example 1 :** Logic AND operation : $(i = 1, 2, \ldots, n)$

• $w_i = +1,\ \theta = n - 0.5 \rightarrow o = f(x_1 + x_2 + \ldots + x_n - (n - 0.5))$

• **Result :** If all $x_i = 1 \rightarrow \Sigma = n, \ \rightarrow v = 0.5 > 0 \ \rightarrow o = 1$

- If at least one $x_i = 0, \ \rightarrow \Sigma < n, \ \rightarrow v < 0 \ \rightarrow o = 0$

- Logic AND operation..

• **Example 2 :** Logic OR operation : $(i = 1, 2, \ldots, n)$

- $w_i = +1, \theta = 0.5 \rightarrow o = f(x_1 + x_2 + \ldots + x_n - 0.5)$

- **Result :** If at least one $x_i = 1 \rightarrow \Sigma \geq 1, \ \rightarrow v \geq 0.5 > 0 \ \rightarrow o = 1$

- If all $x_i = 0, \ \rightarrow \Sigma = 0, \ \rightarrow v = -0.5 < 0 \ \rightarrow o = 0$

- Logic OR operation..

• **Example 3 :** Logic NOT operation : $(i = 1)$

- $w_1 = -1, \theta = -0.5 \rightarrow \ o = f(-x_1 + 0.5)$

- **Result :** If $x_1 = 1 \ \rightarrow v = -0.5 < 0 \ \rightarrow o = 0$

• If $x_1 = 0 \ \rightarrow v = 0.5 > 0 \ \rightarrow o = 1$

- Logic NOT operation..

• **Remark 1 :** In all of the above cases, input classes are **linearly separable**

• **Remark 2 :** One can easily generate logic NAND and logic NOR operations by cascading these blocks.

- **Result :** All of the operations that can be done by classical computers can be done by neural networks → Inspired Von Neumann...

• **Basic limitation :** Linear Separability.

• **Example 4 : EXOR operation** (EXclusive OR) ($n = 2,\ x_i = 1$ or 0)

• $$x_1 \oplus x_2 = \begin{cases} +1 & \text{if } x_1 \neq x_2 \\ 0 & \text{if } x_1 = x_2 \end{cases}$$

• $o = f(w_1 x_1 + w_2 x_2 - \theta)$

• $x_1 = x_2 = 0 \ \rightarrow o = f(-\theta) = 0 \ \rightarrow -\theta < 0 \ \rightarrow \theta > 0$

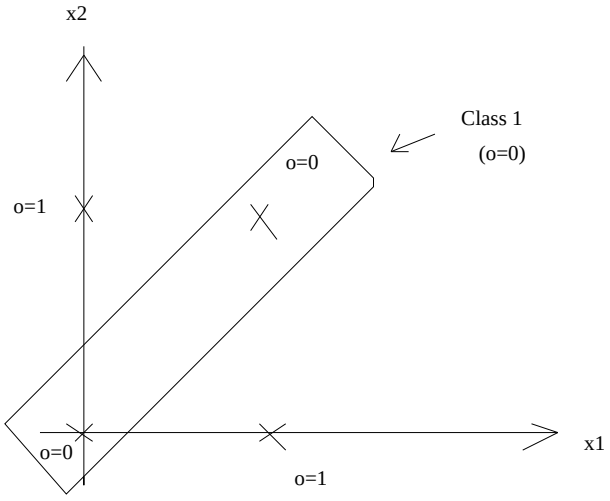
• $x_1 = x_2 = 1 \ \rightarrow o = f(w_1 + w_2 - \theta) = 0 \ \rightarrow w_1 + w_2 - \theta < 0$

• $x_1 = 1, x_2 = 0 \ \rightarrow o = f(w_1 - \theta) = 1 \ \rightarrow w_1 - \theta > 0$

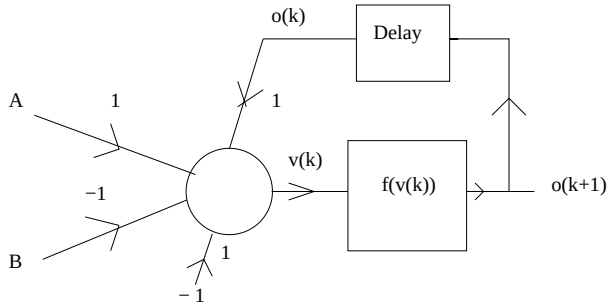
• $x_1 = 0, x_2 = 1 \ \rightarrow o = f(w_2 - \theta) = 1 \ \rightarrow w_2 - \theta > 0$

• $w_1 + w_2 - 2\theta > 0 \ \rightarrow w_1 + w_2 - \theta > \theta > 0 \ \rightarrow$ Contradiction!

• **Result :** EXOR cannot be realized by a single neuron model → Linear Separability does not hold.



• **Example 5 : Basic Memory Unit**



Basic Memory Unit

• A : excitory input (stores 1), B : inhibitory input (stores 0).

• Basic Operation : Discrete Iteration, $k = 0, 1, 2, \ldots$: represents present (discrete) time, $k + 1$: next time.

• $v(k) = o(k) + A - B - 1$, $o(k + 1) = f(v(k))$

A	B	o(k)	v(k)	o($k+1$)	Situation
0	0	0	-1	0	State is preserved
0	0	1	0	1	State is preserved
1	0	0	0	1	Excitation is preserved
1	0	1	1	1	Excitation is preserved
0	1	0	-2	0	Inhibition is preserved
0	1	1	-1	0	Inhibition is preserved
1	1	0	-1	0	State is preserved
1	1	1	0	1	State is preserved

- **Nonlinear Functions:** $\Longrightarrow$ **Unipolar case**

- **Unipolar Case** $\rightarrow 0 \leq o \leq 1$

- **Unipolar Sigmoidal Function :**

- $f(v) = \frac{1}{1+e^{-\lambda v}} \;\rightarrow\; \lambda > 0$

- $\lim_{v\rightarrow-\infty} f(v) = 0, \qquad \lim_{v\rightarrow+\infty} f(v) = 1$

- In fact any **non-decreasing** function ($v_1 > v_2 \Rightarrow f(v_1) \geq f(v_2)$)

- $\lim_{v\rightarrow-\infty} f(v) = a > -\infty, \qquad \lim_{v\rightarrow+\infty} f(v) = b < \infty \qquad (a < b)$

- is called a **sigmoidal type** function and can be used.

- **Hard Limiter :**

- $$f(v) = \begin{cases} +1 & v \geq 0 \\ 0 & v < 0 \end{cases}$$

- This is the same as **unit step** function $1(v)$..

- Note that we have $\lim_{\lambda\rightarrow\infty}$(unipolar sigmoidal) = (hard limiter)

- **Unipolar piecewise linear function :**

- $$f(v) = \begin{cases} +1 & v \geq 0.5 \\ v + 0.5 & -0.5 < v < 0.5 \\ 0 & v < -0.5 \end{cases}$$

• **Bipolar Case** $\rightarrow -1 \leq o \leq 1$

• **Bipolar Sigmoidal Function :** $\rightarrow$ Hyperbolic tangent function

• $f(v) = \frac{1-e^{-\lambda v}}{1+e^{-\lambda v}} \ \rightarrow \ \lambda > 0$

• $\lim_{v \rightarrow -\infty} f(v) = -1, \qquad \lim_{v \rightarrow +\infty} f(v) = 1$

• **Signum Function :** Threshold Logic Unit (TLU)

• $$f(v) = \begin{cases} +1 & v \geq 0 \\ -1 & v < 0 \end{cases}$$

• Note that we have $\lim_{\lambda \rightarrow \infty}$(bipolar sigmoidal) = (signum)

• **Bipolar piecewise linear function :** $\rightarrow$ Saturation Function

• $$f(v) = \begin{cases} +1 & v \geq 1 \\ v & -1 < v < 1 \\ -1 & v < -1 \end{cases}$$

• Note that this unit can be realized by operational amplifiers.

• Although limiter-type models are easier to analyze, for optimization we may require **differentiability**, hence we will use **sigmoidal types** for such problems (e.g. Back Propagation Algorithm)

• **Single Layer Feedforward Networks**

• Suppose that we have n inputs $(x_1, \ldots, x_n)$ and m neurons in a layer

• $v_1, \ldots, v_m \rightarrow o_1 = f(v_1), \ldots, o_m = f(v_m)$

• $v_1 = w_{11}x_1 + w_{12}x_2 + \ldots + w_{1n}x_n \rightarrow \ v_1 = w_1^T x \quad T$: Transpose

• $\quad \vdots$

• $v_i = w_{i1}x_1 + w_{i2}x_2 + \ldots + w_{in}x_n \rightarrow \ v_i = w_i^T x$

• $\quad \vdots$

• $v_m = w_{m1}x_1 + w_{m2}x_2 + \ldots + w_{mn}x_n \rightarrow \ v_m = w_{im}^T x$

• $$w_i = \begin{pmatrix} w_{i1} \\ w_{i2} \\ \vdots \\ w_{in} \end{pmatrix} \quad \text{(ith weight vector)}, \qquad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

• $$o_i = f(v_i) = f(w_i^T x), \qquad i = 1, 2, \ldots m$$

• Note that in case there is a **threshold input**, we could choose $x_n = -1$ and $w_{in} = \theta_i$: Threshold of ith neuron.

• $$v = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix} \quad \text{(vector of linear sums)}, \qquad o = \begin{pmatrix} o_1 \\ o_2 \\ \vdots \\ o_m \end{pmatrix} \quad \text{(output vector)}$$

- $$W = \begin{pmatrix} w_1^T \\ v_2^T \\ \vdots \\ v_m^T \end{pmatrix} = \begin{pmatrix} w_{11} & w_{12} & \dots & w_{1n} \\ w_{21} & w_{22} & \dots & w_{2n} \\ & & \vdots & \\ w_{m1} & w_{m2} & \dots & w_{mn} \end{pmatrix} \quad \text{(weight matrix)}$$

- $v = Wx$, $o_i = f(v_i)$

- $$o = \Gamma(v) = \begin{pmatrix} f(v_1) & 0 & \dots & 0 \\ 0 & f(v_2) & \dots & 0 \\ & & \vdots & \\ 0 & 0 & \dots & f(v_m) \end{pmatrix} \quad \text{(nonlinear vector function)}$$

- $$o = \Gamma(v) = \Gamma(Wx)$$

• **Example** A 2 input, 2 output NN is to be designed, which satisfies the following logic function.

x_1	x_2	o_1	o_2
0	0	0	0
0	1	1	0
1	0	0	1
1	1	0	0

- $v_1 = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 = w_{11}x_1 + w_{12}x_2 - \theta_1 \rightarrow \ o_1 = f(v_1)$
- $v_2 = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 = w_{21}x_1 + w_{22}x_2 - \theta_2 \rightarrow \ o_2 = f(v_2)$

- Choose $f(v)$ as hard limiter since $o = 0, 1$.

- $x_1 = x_2 = 0 \rightarrow o_1 = f(-\theta_1) = 0, o_2 = f(-\theta_2) = 0 \rightarrow \theta_1 > 0,\ \theta_2 > 0$

- $x_1 = 0,\ x_2 = 1 \rightarrow o_1 = f(w_{12} - \theta_1) = 1, o_2 = f(w_{22} - \theta_2) = 0$
- $\rightarrow w_{12} - \theta_1 \geq 0,\ w_{22} - \theta_2 < 0$

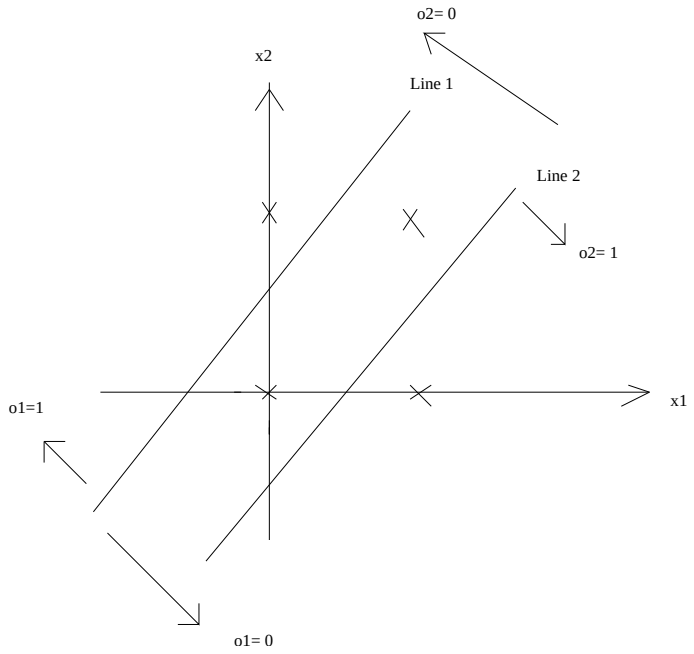
- $x_1 = 1,\ x_2 = 0 \rightarrow o_1 = f(w_{11} - \theta_1) = 0, o_2 = f(w_{21} - \theta_2) = 1$
- $\rightarrow w_{11} - \theta_1 < 0,\ w_{21} - \theta_2 \geq 0$

- $x_1 = 1,\ x_2 = 1 \rightarrow o_1 = f(w_{11} + w_{12} - \theta_1) = 0, o_2 = f(w_{21} + w_{22} - \theta_2) = 0$
- $\rightarrow w_{11} + w_{12} - \theta_1 < 0,\ w_{21} + w_{22} - \theta_2 < 0$

- A set of inequalities. $\rightarrow$ Solve them. (Note that sometimes a solution may not be available $\rightarrow$ Linear programming...

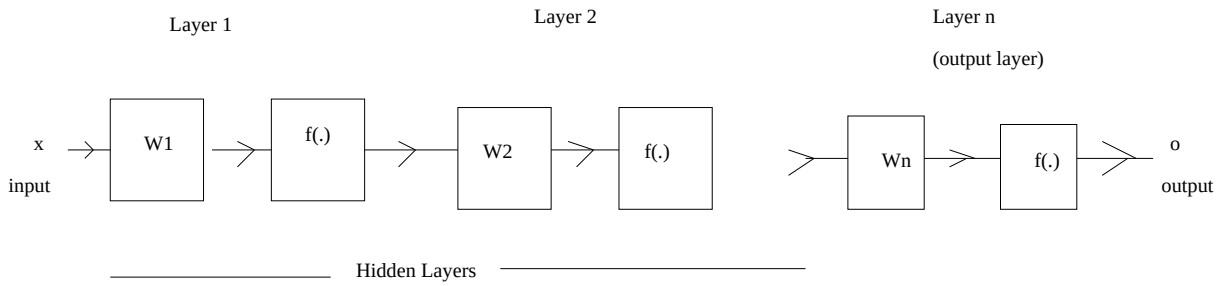
- In this case, $\theta_1 = \theta_2 = 1,\ w_{12} = w_{21} = 2,\ w_{11} = w_{22} = -2$ is **a** solution. There may be (in general) infinitely many solutions....

- **Line 1 :** $-2x_1 + 2x_2 - 1 = 0$ **Line 2 :** $2x_1 - 2x_2 - 1 = 0$



• Multilayer Structures

• If we cascade single layer structures we obtain multilayer structures...



• $o = \Gamma(W_n(\Gamma(W_{n-1}(\ldots\Gamma(W_2(\Gamma(W_1 x))\ldots)$

• **Why multilayer structures ?**

• Because some patterns which are not linearly separable by a single layer can be made linearly separable by using multilayer structures....

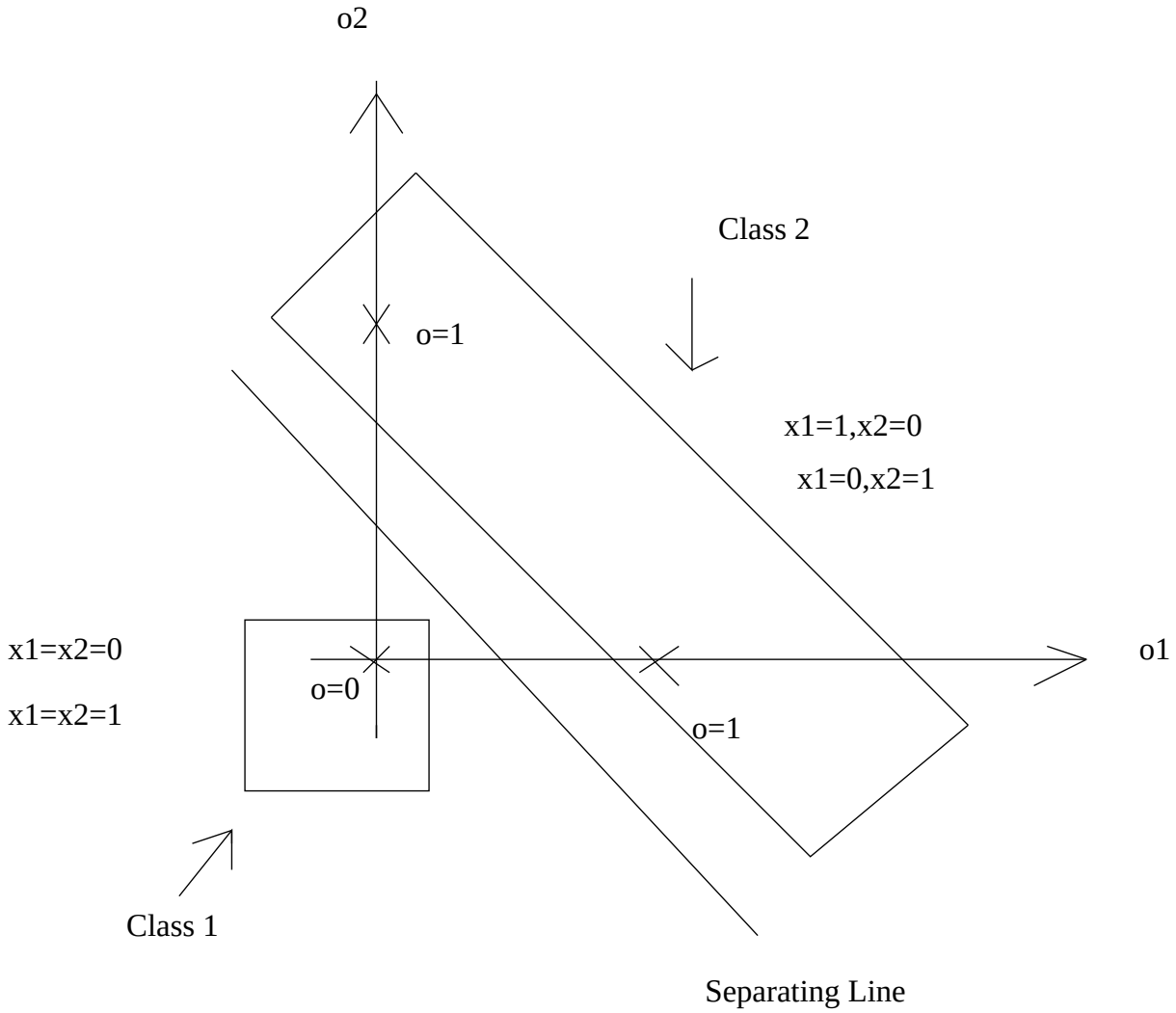
• **Example 4 : EXOR operation** (EXclusive OR) ($n = 2,\ x_i = 1$ or 0)

• $$x_1 \oplus x_2 = \begin{cases} +1 & \text{if } \ x_1 \neq x_2 \\ 0 & \text{if } \ x_1 = x_2 \end{cases}$$

• We know that this function cannot be realized by a sigle layer structure since it is not linearly separable....

x_1	x_2	o_1	o_2	$o = o_1 \vee o_2$		
0	0	0	0	0		
0	1	1	0	1	$\Rightarrow$	$o = o_1 \vee o_2 = x_1 \oplus x_2$
1	0	0	1	1		
1	1	0	0	0		

• Hence a layer realizing the previous example cascaded to an logic OR block realizes EXOR function → Multilayer structure....



• **Example 2.1** , p. 39-42.

• **Case 1 :** $f(\cdot) = sgn(\cdot)$

• $o_1 = sgn(x_1 - 1),\ o_2 = sgn(-x_1 + 2),\ o_3 = sgn(x_2),\ o_4 = sgn(-x_2 + 3)$

• $o_5 = sgn(o_1 + o_2 + o_3 + o_4 - 3.5) = o_1 \wedge o_2 \wedge o_3 \wedge o_4$

• $o_5 = \begin{cases} 1 & o_1 = o_2 = o_3 = o_4 = 1 \\ -1 & \text{otherwise} \end{cases}$

• $o_1 = \begin{cases} 1 & x_1 \geq 1 \\ -1 & x_1 < 1 \end{cases} \qquad o_2 = \begin{cases} 1 & x_1 \leq 2 \\ -1 & x_1 > 2 \end{cases}$

• $o_3 = \begin{cases} 1 & x_2 \geq 0 \\ -1 & x_2 < 0 \end{cases} \qquad o_4 = \begin{cases} 1 & x_2 \leq 3 \\ -1 & x_2 > 3 \end{cases}$

• $\Rightarrow o_5 = \begin{cases} 1 & 1 \leq x_1 \leq 2,\ 0 \leq x_2 \leq 3 \\ -1 & \text{otherwise} \end{cases}$

• Note that the first block forms the lines which separates the regions...

• Second block is an **AND** block, which determines the regions.

• Mathematically we can write the equations in matrix form as :

• **1 : Extended notation :**

• $x_e = \begin{pmatrix} x_1 \\ x_2 \\ -1 \end{pmatrix}$ $o^{1b} = \begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \end{pmatrix}$

• $W_{1e} = \begin{pmatrix} 1 & 0 & 1 \\ -1 & 0 & -2 \\ 0 & 1 & 0 \\ 0 & -1 & -3 \end{pmatrix}$ $o^{1b} = \Gamma(W_{1e}x_e)$

• $o_e^{1b} = \begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \\ -1 \end{pmatrix}$ $o = \begin{pmatrix} o_5 \end{pmatrix}$

• $W_{2e} = \begin{pmatrix} 1 & 1 & 1 & 1 & 3.5 \end{pmatrix}$ $o = \Gamma(W_{2e}o_e^{1b})$

• **1 : Threshold notation :**

• $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ $\qquad o^{1b} = \begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \end{pmatrix}$

• $W_1 = \begin{pmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{pmatrix}$ $\qquad \theta_1 = \begin{pmatrix} 1 \\ -2 \\ 0 \\ -3 \end{pmatrix}$ $\qquad o^{1b} = \Gamma(W_1 x - \theta_1)$

• $W_2 = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}$ $\qquad \theta_2 = \begin{pmatrix} 3.5 \end{pmatrix}$ $\qquad o = \Gamma(W_2 o^{1b} - \theta_2)$

• $o = \Gamma\ (W_2\ \Gamma\ (W_1 x - \theta_1)\ -\ \theta_2\ )$

• **Case 2 :** $f(\cdot)$ = unipolar sigmoidal function....

• $\qquad f(v) = \frac{1}{1+e^{-\lambda v}}\ \rightarrow\ \lambda > 0)$

• See Figure 2.9. If we choose large λ, we could approximate the unipolar case with **hardlimiter** function. If we use **bipolar sigmoidal** function

• ($f(v) = \frac{1-e^{-\lambda v}}{1+e^{-\lambda v}}\ \rightarrow\ \lambda > 0$),

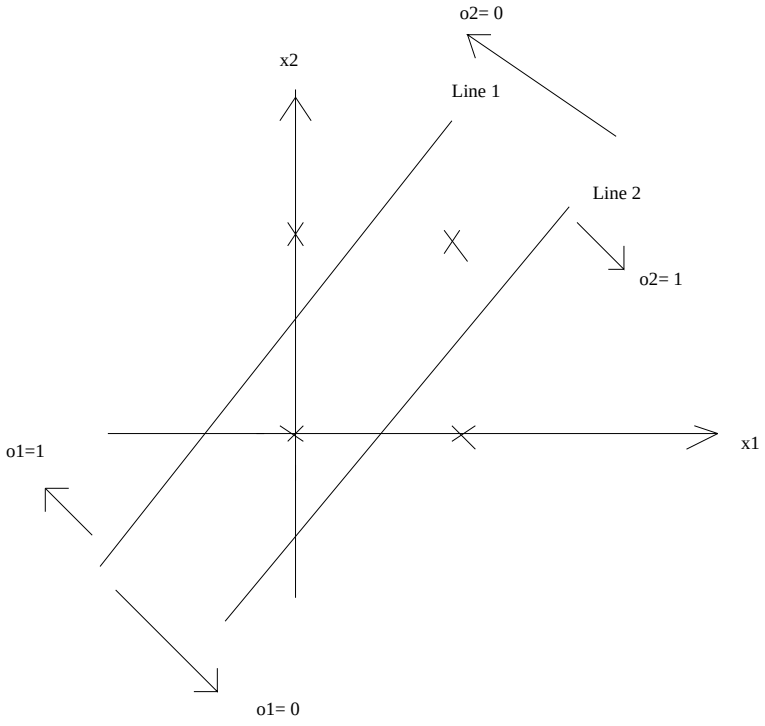
• with large λ, we could approximate the present case...

• Lets go back to the **EXOR** example once more ....

• One solution was given in previous slides as follows :

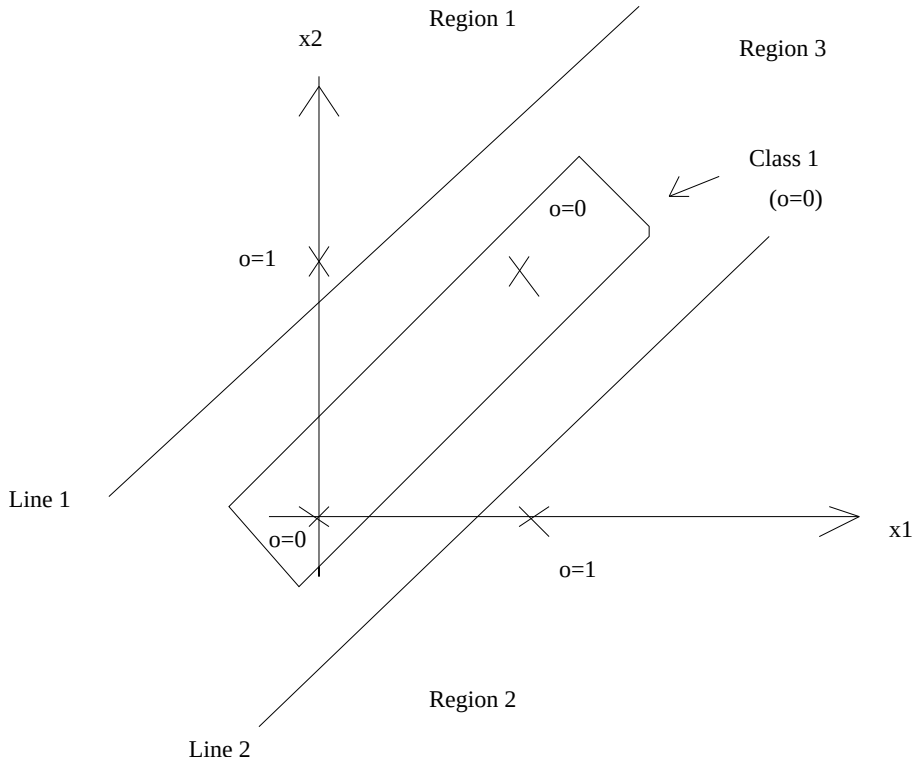
x_1	x_2	o_1	o_2	$o = o_1 \vee o_2$		
0	0	0	0	0		
0	1	1	0	1	$\Rightarrow$	$o = o_1 \vee o_2 = x_1 \oplus x_2$
1	0	0	1	1		
1	1	0	0	0		

• $o_1 = f(-2x_1 + 2x_2 - 1) = f(-x_1 + x_2 - 0.5),\ o_2 = f(2x_1 - 2x_2 - 1) = f(x_1 - x_2 - 0.5)$



• Hence one block to form the lines **separating** regions, and another block **adding** these regions (**OR Block**) is sufficient in this case...

• Following this idea, we may find alternative solutions to the EXOR problem as follows :



• $$x_1 \oplus x_2 = \begin{cases} +1 \text{ if } (x_1\ x_2) \in \text{Region 1 OR Region 2} \\ 0 \qquad\qquad \text{otherwise (Region 3)} \end{cases}$$

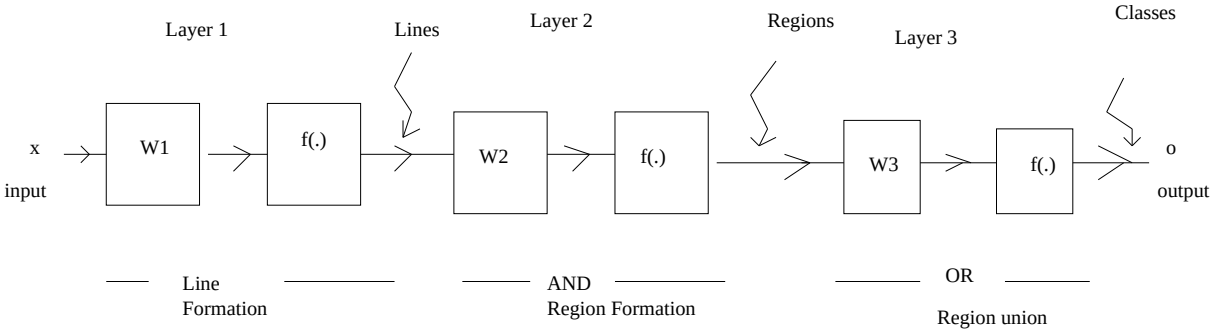
• Suppose that Line 1 $\rightarrow v_1 = w_{11}x_1 + w_{12}x_2 - \theta_1 = 0$ (which is > 0 in Region 1)

• Suppose that Line 1 $\rightarrow v_2 = w_{21}x_1 + w_{22}x_2 - \theta_2 = 0$ (which is > 0 in Region 2)

• Then a first block realizing these ($o_1 = f(v_1)$, $o_2 = f(v_2)$ and $o = o_1 + o_2 - 0.5 = o_1 \vee o_2$ realizes EXOR function. (f: hardlimiter function

• Combining these, we conclude that ....

• At most, 3 layer structure is sufficient to classify **linearly separable** patterns...

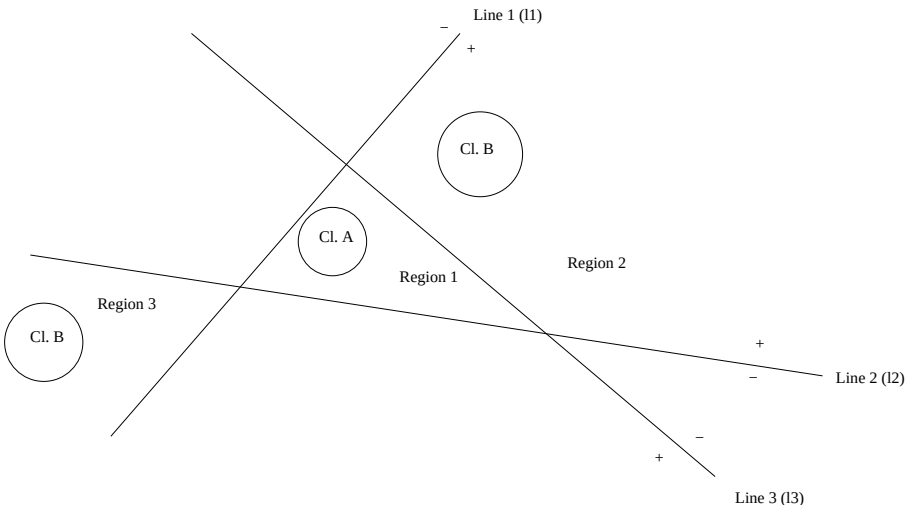


• First Layer **forms the lines** which separates the regions...

• Second Layer **forms the regions** by using **AND** operation...

• Third Layer **combines the regions** by using **OR** operation...

• **Example**



- Region 1 $\rightarrow l1 > 0,\ l_2 > 0,\ l_3 > 0 \rightarrow l_1 \wedge l_2 \wedge l_3$
- Region 2 $\rightarrow l1 > 0,\ l_2 > 0,\ l_3 < 0 \rightarrow l_1 \wedge l_2 \wedge \bar{l}_3$
- Region 3 : $\rightarrow l1 < 0,\ l_2 < 0,\ l_3 > 0 \rightarrow \bar{l}_1 \wedge \bar{l}_2 \wedge l_3$
- Class A : $\rightarrow x \in$ Region 1
- Class B : $\rightarrow x \in$ Region 2 **OR** Region 3
- $l1 : v_1 = w_{11}x_1 + w_{12}x_2 - \theta_1 = 0 \ \rightarrow o_1 = f(v_1)$
- $l2 : v_2 = w_{21}x_1 + w_{22}x_2 - \theta_2 = 0 \ \rightarrow o_2 = f(v_2)$
- $l3 : v_3 = w_{31}x_1 + w_{32}x_2 - \theta_3 = 0 \ \rightarrow o_3 = f(v_3)$
- Region 1 : $v_5 = o_1 + o_2 + o_3 - 2.5,\ o_5 = f(v_5)$
- Region 2 : $v_6 = o_1 + o_2 - o_3 - 2.5,\ o_6 = f(v_5)$
- Region 3 : $v_7 = -o_1 - o_2 + o_3 - 2.5,\ o_7 = f(v_7)$
- Region 2 **OR** Region 3 : $v_8 = o_6 + o_7 - 0.5,\ o_8 = f(v_8)$

- $x \in \begin{cases} \text{Class A} & o_5 = 1 \\ \text{not in Class A} & o_5 = 0 \end{cases}$
- $x \in \begin{cases} \text{Class B} & o_8 = 1 \\ \text{not in Class B} & o_8 = 0 \end{cases}$

• **Feedback Networks :** Hopfield Networks.

• **Discrete Case :** See Figure 2.10 for a representation..

• Basic Operation : Iterative Process ...

• Suppose that at step k, we have $o(k)$ as the past output (Note the delay unit)

• $v(k) = Wo(k) \rightarrow o(k+1) = \Gamma(v(k)) = \Gamma(Wo(k)) \qquad k = 0, 1, 2, \ldots,$

• $o(0)$ is the initial condition $\rightarrow$ initial pattern. It evolves according to the above iteration scheme....

• $o(1) = \Gamma(Wo(0)), \quad o(2) = \Gamma(Wo(1)), \quad o(3) = \Gamma(Wo(2))$
• $\ldots o(k+1) = \Gamma(Wo(k)) \ldots ....$

• This process is useless unless it stops (i.e. keeps iterating the same output) after certain point. Suppose this pattern is o^*. Then we must have
• $\qquad\qquad o^* = \Gamma(Wo^*)$

• Such an o^* is called a **fixed point** of the discrete iteration $\rightarrow$ memory pattern....

• **Practical application :**

• Suppose that $o(0) = o^* + \Delta$, where Δ represents a perturbation for the memory pattern o^* ..

• If we have $o(0) \rightarrow o(1) \rightarrow o(2) \ldots \rightarrow o^* \rightarrow o^* \ldots$
• Hence we recover the original pattern...(See Figures 1.7 and 1.8)

- **Example :** See Fig. 2.12

- $W = \begin{pmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{pmatrix} \qquad \rightarrow \qquad o(k+1) = \Gamma(Wo(k))$

- Assume that nonlinearity is **hard limiter**

- $o(0) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \rightarrow v(0) = Wo(0) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -3 \end{pmatrix} \rightarrow o(1) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix}$

- $o(1) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} \rightarrow v(1) = Wo(1) = \begin{pmatrix} 3 \\ 3 \\ 3 \\ -3 \end{pmatrix} \rightarrow o(2) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} = o^*$

- o^* is a stored pattern....

- $o(0) = \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix} \rightarrow v(0) = Wo(0) = \begin{pmatrix} 1 \\ 1 \\ 3 \\ -1 \end{pmatrix} \rightarrow o(1) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} = o^*$

• Can we have other memory patterns? **YES**....$o^* = \Gamma(Wo^*)$

• $o(0) = \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \rightarrow v(0) = Wo(0) = \begin{pmatrix} -3 \\ -1 \\ -1 \\ 1 \end{pmatrix} \rightarrow o(1) = \begin{pmatrix} -1 \\ -1 \\ -1 \\ 1 \end{pmatrix}$

• $o(1) = \begin{pmatrix} -1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \rightarrow v(1) = Wo(1) = \begin{pmatrix} -3 \\ -3 \\ -3 \\ 3 \end{pmatrix} \rightarrow o(2) = \begin{pmatrix} -1 \\ -1 \\ -1 \\ 1 \end{pmatrix} = o^{**}$

• o^{**} is **also** stored pattern....

• **Continuous Case :** See Figure 2.15 for a representation..

• Here, instead of a discrete iteration, we have a **differential** equation...

• $\frac{dx}{dt} = f(x)$.

• Given $x(0)$ (initial pattern), we have a solution $x(t)$. Unless $x(t) \rightarrow x^*$ (a constant), this system is not very useful.

• If this is the case $\Rightarrow \frac{dx}{dt} \rightarrow 0$

• Then we have (asymptotically) $f(x^*) = 0 \rightarrow$ **equilibrium point** (DC operating point in Electronics)

• See Figure 2.15. Define $x_1 = net_1,\ x_2 = net_2 \rightarrow o_1 = f(x_1),\ o_2 = f(x_2)$

• Nonlinearity is either bipolar sigmoidal (practical op-amp) or signum function (ideal op-amp).

• At Node 1 : $C_1 \frac{dx_1}{dt} + \frac{x_1}{R} + \frac{x_1 - o_2}{R_{12}} = 0$

• At Node 2 : $C_2 \frac{dx_2}{dt} + \frac{x_2}{R} + \frac{x_2 - o_1}{R_{21}} = 0$

• Assume that $C_1 = C_2 = C > 0,\ R < 0\ , R_{12} = R_{21} < 0$

• $C \frac{d(x_1 - x_2)}{dt} + \frac{x_1 - x_2}{R} + \frac{x_1 - x_2}{R12} + \frac{o_1 - o_2}{R_{12}} = 0$

• $x_1 = x_2 \Leftrightarrow o_1 = o_2 \rightarrow$ equilibrium....

• Assume that $x_1 > x_2 \Leftrightarrow o_1 > o_2 \rightarrow C \frac{d(x_1 - x_2)}{dt} > 0 \rightarrow x_1 - x_2 \uparrow$

• $\rightarrow o_1 - o_2 \uparrow$

• Asymptotically $o_1 - o_2 \uparrow$ to its maximum, which is $o_1 = 1,\ o_2 = -1$...Equilibrium point....

• Alternatively, assume that $x_1 < x_2 \Leftrightarrow o_1 < o_2 \rightarrow C \frac{d(x_1 - x_2)}{dt} < 0 \rightarrow x_1 - x_2 \downarrow$

• $\rightarrow o_1 - o_2 \downarrow$

• Asymptotically $o_1 - o_2 \downarrow$ to its minimum, which is $o_1 = -1,\ o_2 = 1$...Equilibrium point....