Tema 2:

Elementos de procesamiento y aprendizaje. Redes neuronales básicas

Redes Neuronales Artificiales:

Tema 2.1. Conceptos básicos de procesamiento y aprendizaje neuronal

- Utilizan como inspiración dos principios fundamentales observados en las redes neuronales biológicas:
  - Procesamiento distribuido en unidades (neuronas)
  - Aprendizaje vía modificación de las conexiones

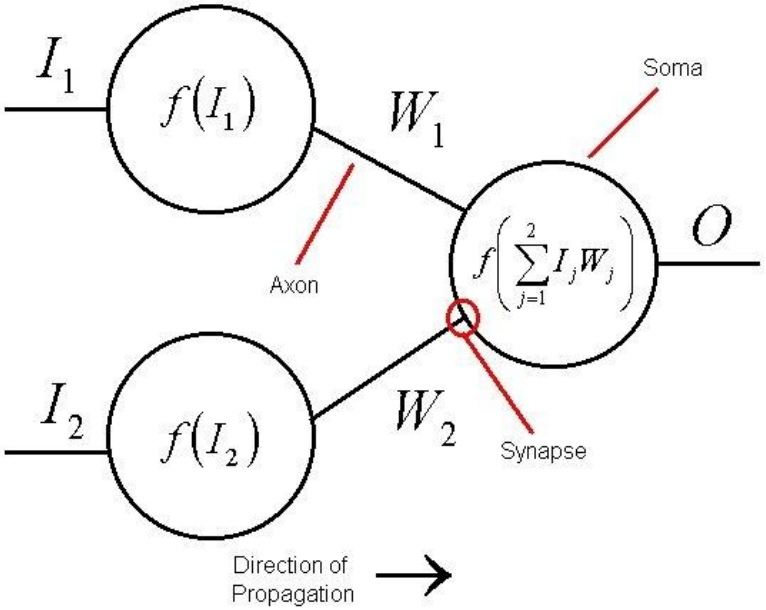
- Contemplan muchas simplificaciones para facilitar su uso y demostrar que pueden aprender a realizar tareas complejas:
  - Unidades simples y normalmente indistinguibles
  - Aprendizaje sináptico sencillo

- El conocimiento se almacena en la red de una forma distribuida, fundamentalmente en los pesos de las conexiones.

Definición

- Una red neuronal artificial es un sistema de procesamiento de información bioinspirado en el que

1. el procesamiento tiene lugar mediante muchos elementos simples que se llaman neuronas;
2. las neuronas se intercambian señales mediante conexiones;
3. cada conexión tiene asociado un peso que normalmente se multiplica por la señal transmitida;
4. cada neurona aplica una función de activación (normalmente no lineal) al conjunto de sus entradas (la suma pesada de todas las señales de entrada) para determinar su señal de salida.



- cada conexión tiene asociado un peso W_j que normalmente se multiplica por la señal transmitida.

- cada neurona aplica una función de activación *f* (normalmente no lineal) al conjunto de sus entradas (la suma pesada de todas las señales de entrada I_j) para determinar su señal de salida *O*.

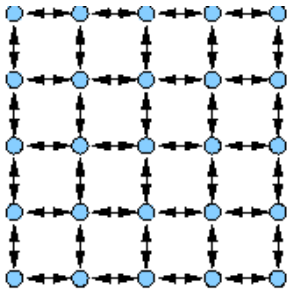
Caracterización de una red neuronal

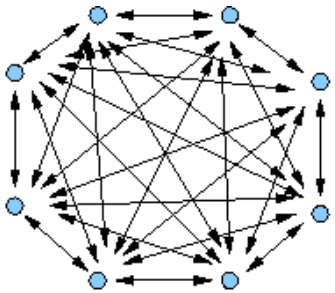
- Una red neuronal artificial se caracteriza por:

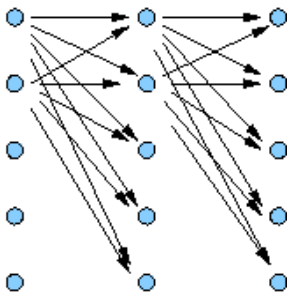
  1. Su patrón de conexiones (arquitectura de conexiones).

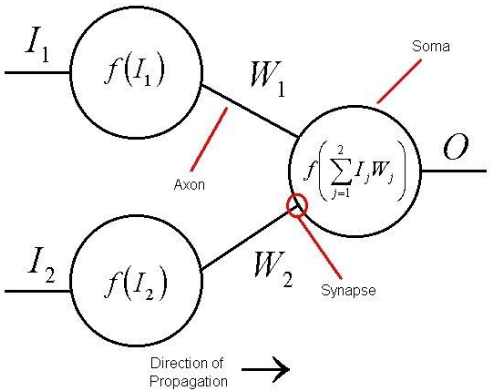
  2. El método para determinar el valor de los pesos de las conexiones (método o algoritmo de entrenamiento o aprendizaje).

  3. La función de activación de sus neuronas









¿Qué tipo de tareas pueden realizar las redes neuronales?

- Tareas de asociación o reconocimiento sin reglas fijas preestablecidas, conocidas o expresables en algoritmos:

  - Almacenamiento y recuperación de datos o patrones
  - Clasificación de patrones
  - Mapeo de patrones de entrada a patrones de salida
  - Agrupación o *clusterización* de patrones por similitud
  - Solución de problemas de optimización
  - Predicción basada en comportamiento anterior

- Recordatorio: el conocimiento se almacena en la red de una forma distribuida.

Ventajas de las redes neuronales artificiales

- Aprendizaje adaptativo: aprenden mediante entrenamiento (supervisado o no supervisado).
- Auto-organización: representación propia de la información.
- Tolerancia a fallos: robustez frente a posibles errores (en el entrenamiento y en el funcionamiento).
- Facilidad para la implementación en hardware y en arquitecturas paralelas.
- Integración modular dentro de tecnologías existentes.

Conceptos generales de RNS

- Arquitectura de red (conectividad)
- Entrenamiento (establecimiento de los valores de los pesos de las conexiones)
- Funciones de transferencia o activación
- Ejemplos

Nomenclatura de las arquitecturas de red (I)

- En general, las redes se organizan en **capas**.
- A la capa que recibe los estímulos externos se les llama **capa de entrada**. A la capa que genera la salida de la red se le llama **capa de salida**.
- Dentro de una misma capa las neuronas se suelen comportar igual: misma función de activación y mismo tipo de conectividad a otras neuronas.
- Dentro de una misma capa las neuronas suelen estar o totalmente conectadas o no conectadas en absoluto.
- A las capas que no constituyen la entrada de la red o la salida de la red se las llama **capas ocultas**.

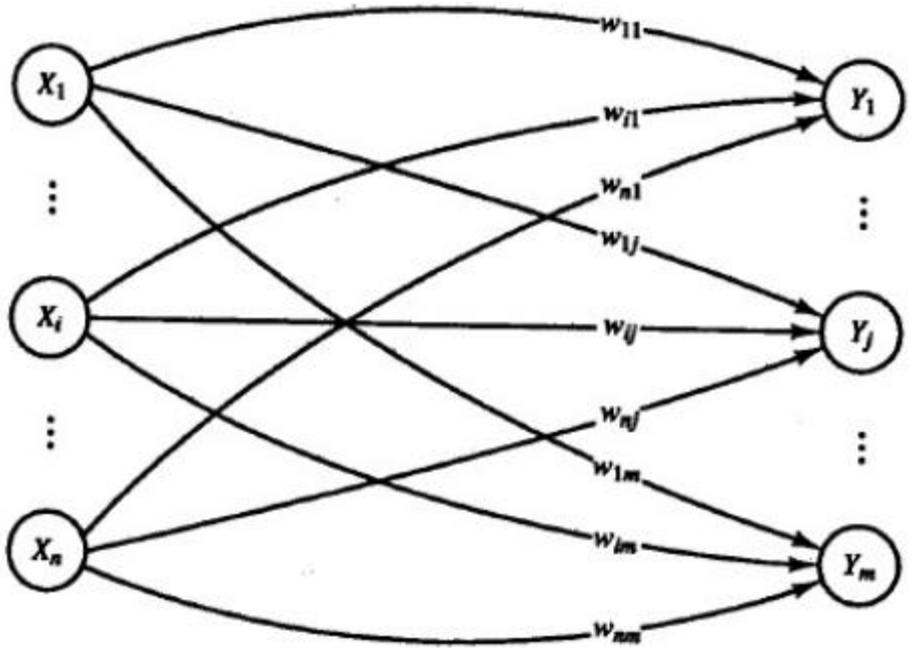
Nomenclatura de las arquitecturas de red (II)

- Las redes se suelen clasificar en **monocapa** o **multicapa**

  Normalmente a la capa de entrada no se la cuenta para determinar el número de capas ya que en ella no se realiza ninguna tarea computacional. Esto es equivalente a contar el número de “capas” de pesos.

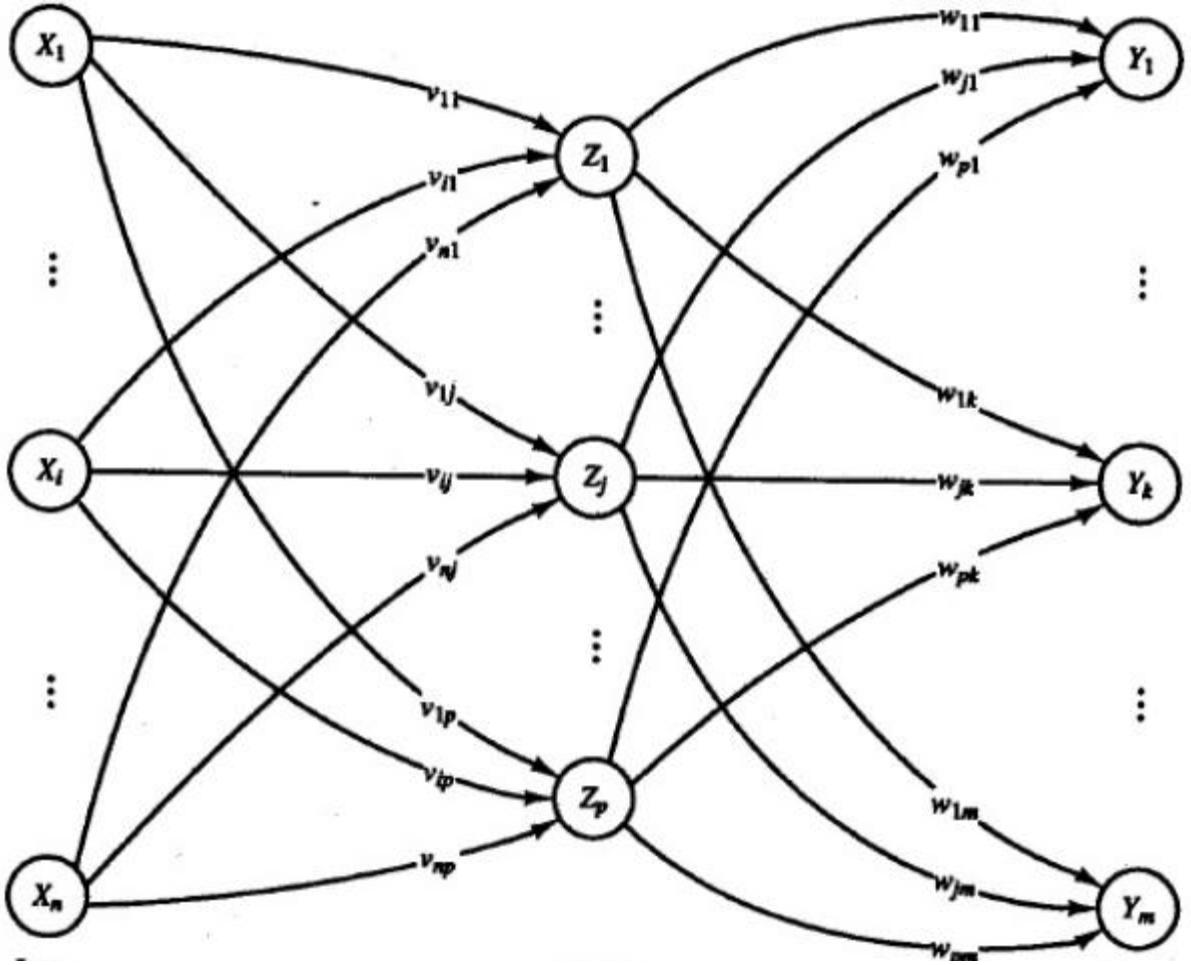
- Por la forma de fluir la información las redes pueden clasificarse como:
  - De **propagación hacia delante** (*feedforward*): desde la capa de entrada a la capa de salida
  - **Recurrentes**: redes en las que existen conexiones de vuelta a las neuronas, es decir con ciclos cerrados de propagación (la señal sale de una neurona y vuelve procesada por otras a esa neurona).

RNA *feedforward* con una sola capa



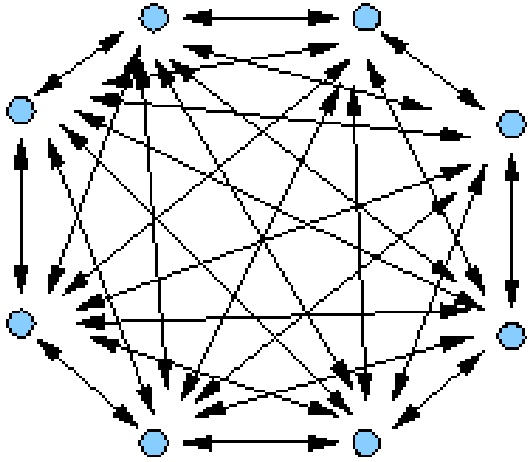
Neuronas de entrada pesos de las conexiones Neuronas de salida

RNA feedforward multicapa



Neuronas de entrada Neuronas de la capa oculta Neuronas de salida

RNA recurrente



La señal sale de una neurona y vuelve procesada por otras a esa neurona.
En este ejemplo la conectividad es todas con todas (no es un requisito en las redes recurrentes)

Este ejemplo es una red de una sola capa.

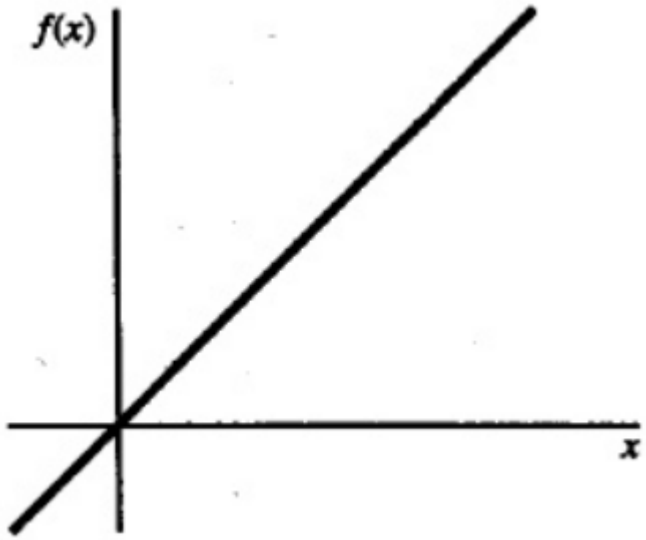
Entrenamiento = establecimiento del valor de los pesos de las conexiones

- Supervisado: el entrenamiento se consigue mediante la presentación de vectores de entrenamiento (también llamados patrones), cada uno asociado con un vector de salida. Los pesos se establecen de acuerdo al algoritmo de aprendizaje

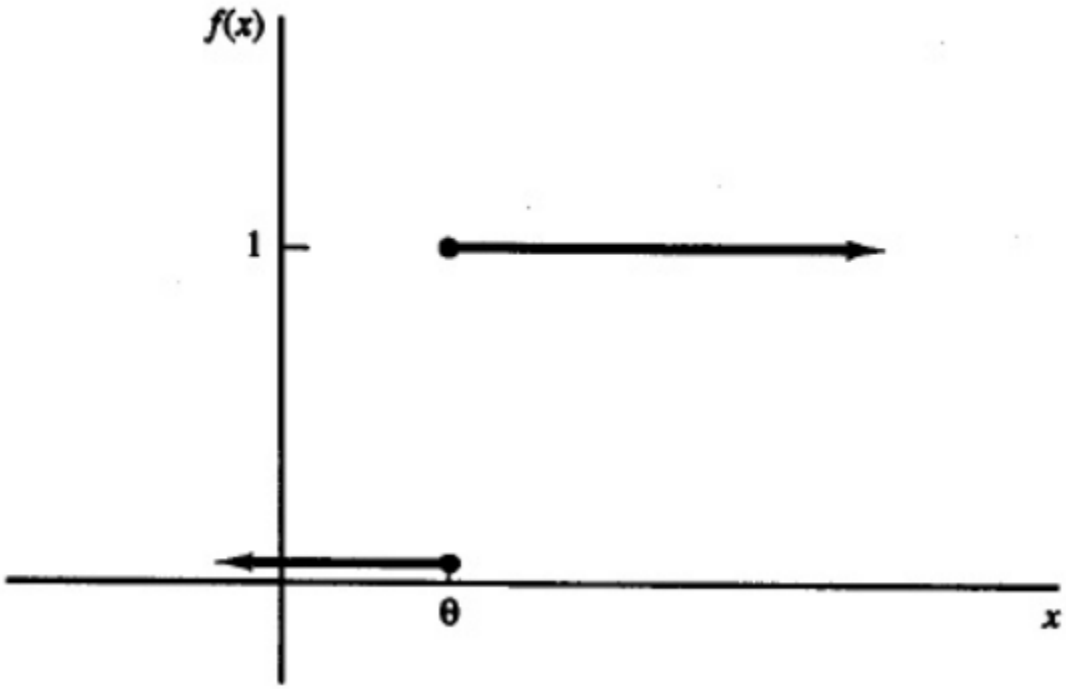
- No supervisado: el entrenamiento se establece sin vectores de entrenamiento. Se proporciona vectores de entrada pero no de salida

- Hay redes que no modifican los pesos (por ejemplo en problemas de restricciones los pesos pueden representar la restricción)

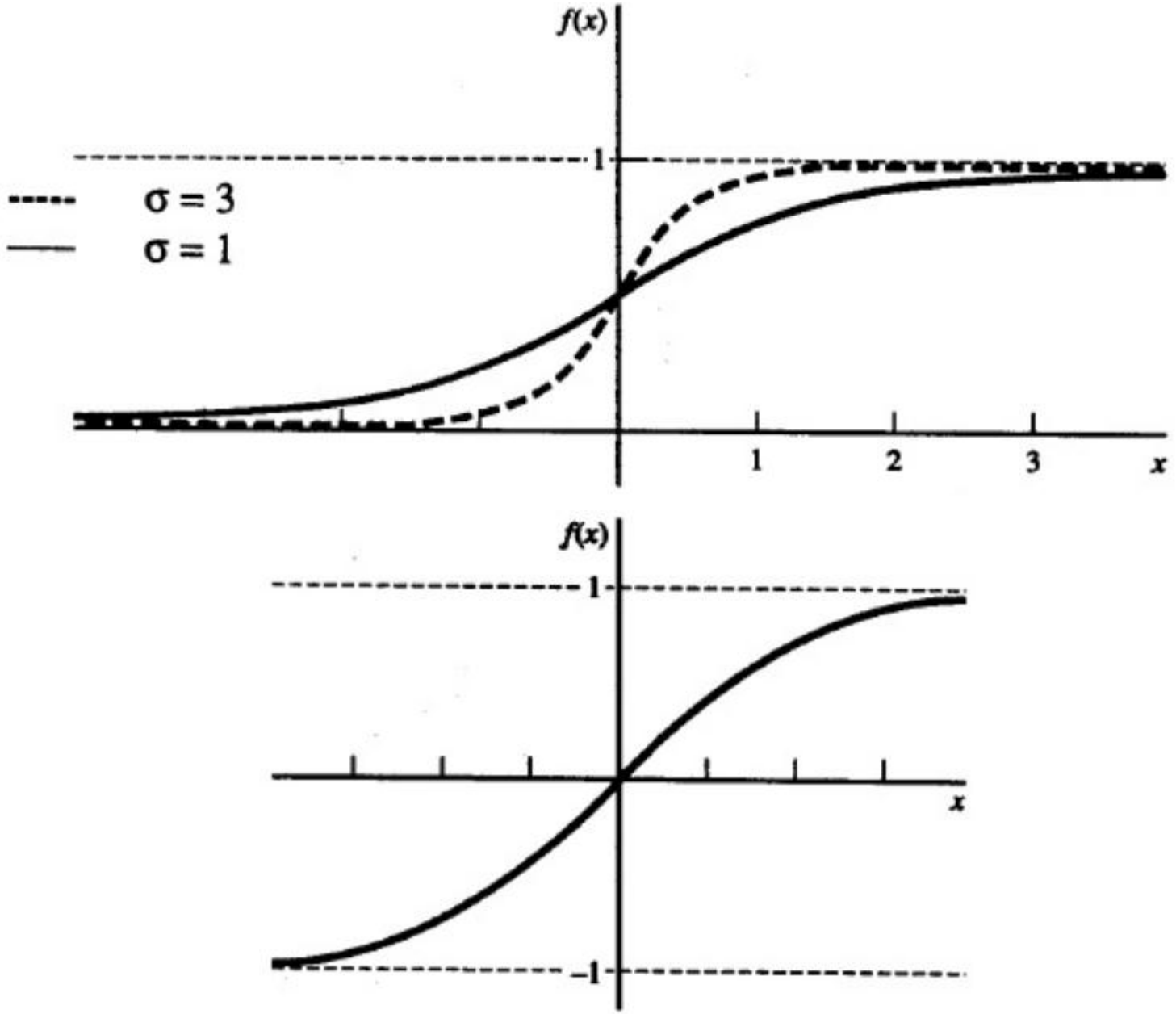
Funciones de activación típicas (I)



Funciones de activación típicas (II)



Funciones de activación típicas (II)



Notación para describir una RNA (I)

- x_i, y_i : Activaciones de las neuronas X_i , Y_j

  Para neuronas de entrada X_i : x_i = señal de entrada

  Para otras neuronas: Y_j : $y_j = f(y\_in_j)$, f función de transferencia

- w_{ij} : peso de la conexión de la neurona X_i a la neurona Y_j

- b_j : sesgo *o* bias de la neurona Y_j *(*actúa como un peso de una conexión desde una neurona que tiene una activación constante 1)

- y_in_j : input o entrada a la neurona Y_j : $y\_in_j = b_j + \sum_i x_i w_{ij}$

- W : matriz de pesos $W=\{w_{ij}\}$

- $\boldsymbol{w}_{\cdot j}$: vector de pesos $\boldsymbol{w}_{\cdot j} = (w_{1j}, w_{2j}, \ldots, w_{nj})^T$ (columna j-ésima de la matriz de pesos)

- θ_j : Umbral para la activación de la neurona Y_j (lo utilizan las funciones de transferencia)

- **s** : vector de entrenamiento de entrada: $\mathbf{s} = (s_1, \ldots., s_i, \ldots, s_n)$

- t: vector de objetivo de salida (*target*) $\mathbf{t} = (t_1, \ldots., t_i, \ldots, t_n)$

Notación para describir una RNA (II)

- **x**: vector de entrada (estímulo a la red) $\mathbf{x} = (x_1, \ldots., x_i, \ldots, x_n)$
- Δw_{ij} cambio de pesos por el aprendizaje: $\Delta w_{ij} = [w_{ij}(\text{nuevo}) - w_{ij}\ (\text{anterior})]$

recordando $y\_in_j = b_j + \sum_i x_i w_{ij}$

- α : tasa de aprendizaje (se utiliza para controlar la cantidad de ajuste de peso en cada paso del entrenamiento).

Cálculo de la entrada mediante el producto escalar

• Si los pesos de la red están almacenados en la matriz $\mathbf{W} = (w_{ij})$, la entrada a una neurona Y_j (sin sesgo) es el producto escalar de los vectores $\mathbf{x} = (x_1, \ldots, x_i, \ldots, x_n)$ y $\mathbf{w}_{\cdot j}$ (la columna j-ésima de la matriz de pesos):

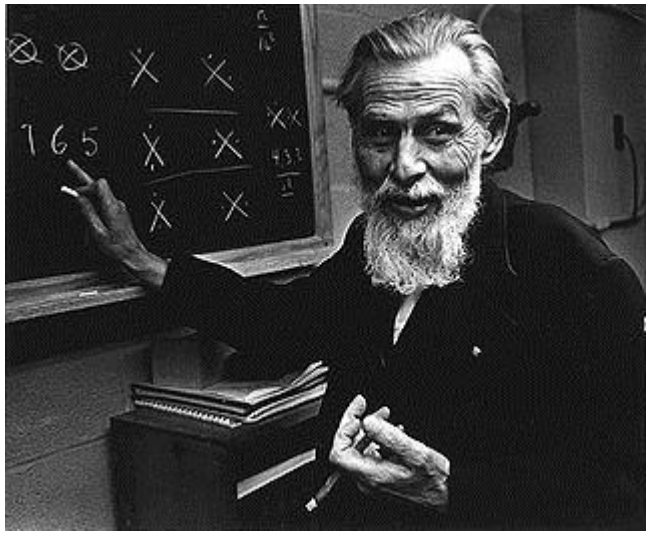
$$y\_in_j = \sum_{i=1}^{n} x_i w_{ij} = \mathbf{x} \cdot \mathbf{w}_{\cdot j}$$

• La gestión del sesgo puede hacerse incluyendo un componente $x_0=1$ al vector $\mathbf{x}$: $\mathbf{x} = (1, x_1, \ldots., x_i, \ldots, x_n)$. El sesgo se trata como cualquier otro peso, es decir $w_{0j}=b_j$. La entrada a una unidad Y_j con sesgo es:

$$y\_in_j = \sum_{i=0}^{n} x_i w_{ij} = \mathbf{x} \cdot \mathbf{w}_{\cdot j} = w_{0j} + \sum_{i=1}^{n} x_i w_{ij} = b_j + \sum_{i=1}^{n} x_i w_{ij}$$

Tema 2.2: Modelo de McCulloc-Pitts

•**Neuronas de McCulloch-Pitts**. Warren McCulloch & Walter Pitts designaron los que se considera la primera red neuronal en 1943. Las neuronas de McCulloch-Pitts implementan funciones lógicas que pueden combinarse en la red para realizar tareas complejas. Introdujeron el concepto de umbral en las neuronas.



Neurona de McCulloch-Pitts

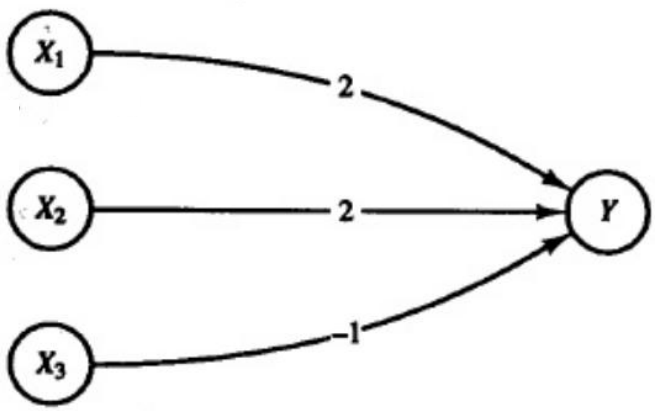
1. La salida (output) es binaria (0 -no dispara- ó 1 –dispara-)
2. Una conexión es excitadora si el peso es positivo, si no, es inhibidora. Todas las conexiones excitadoras a una neurona determinada tienen los mismos pesos.
3. Cada neurona tiene un umbral fijo de forma que si su entrada (input) es mayor que el umbral, la neurona dispara:

$$f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq \theta \\ 0 & \text{si } y\_in < \theta \end{cases}$$

4. El umbral θ se establece de forma que la inhibición es absoluta (cualquier entrada inhibitoria hace que la neurona no dispare).
5. Se requiere un paso de tiempo para que una señal pase a través de una conexión (la salida de la neurona Y en el tiempo *t* se determina por la actividad de sus inputs en el instante *t* – 1).

Neurona de McCulloch-Pitts

Ej:



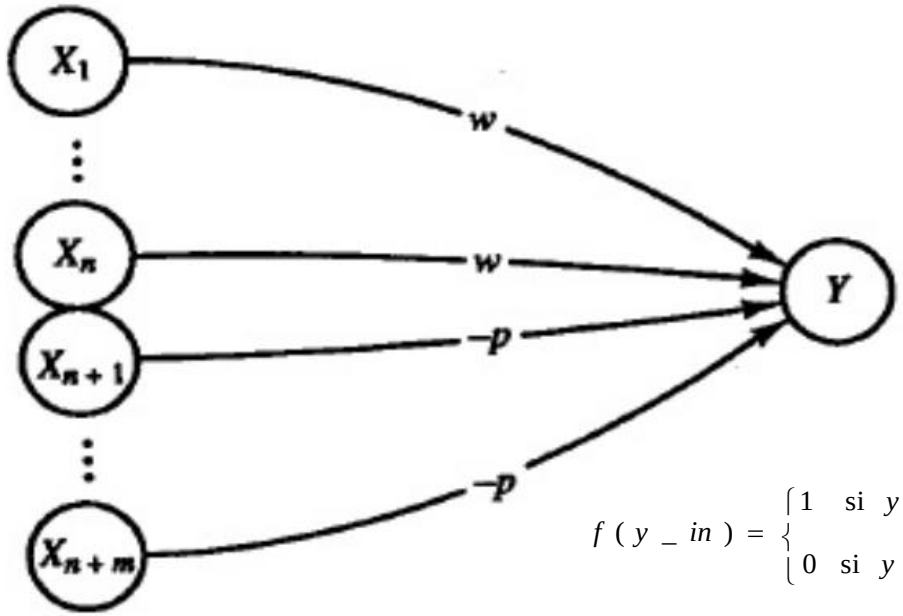
$$\theta = 4$$

$$y\_in = \sum_{i=1}^{3} x_i w_i$$

$$f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq \theta \\ 0 & \text{si } y\_in < \theta \end{cases}$$

Se requiere un paso de tiempo para que una señal pase a través de una conexión: la salida de la neurona Y en el tiempo *t* se determina por la actividad de sus inputs X_1, X_2, X_3 en el instante *t* – 1.

Neurona de McCulloch-Pitts



$$f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq \theta \\ 0 & \text{si } y\_in < \theta \end{cases}$$

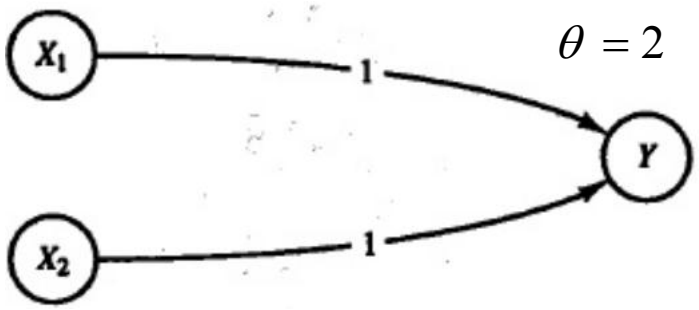
n entradas excitadoras con peso w

m entradas inhibidoras con peso $-p$

- La condición de que una inhibición sea determinante de forma absoluta para que la neurona no dispare requiere que el umbral de la función de transferencia satisfaga la desigualdad: $\theta > nw - p$
- Por tanto, Y disparará si recibe k o más inputs excitadores y ninguno inhibidor con $kw \geq \theta > (k\text{-}1)w$
- Aunque todos los pesos excitadores de conexiones que llegan a una misma neurona tienen que valer lo mismo, los pesos que llegan a una neurona Y_1 no tienen por qué ser los mismos que llegan a otra Y_2.

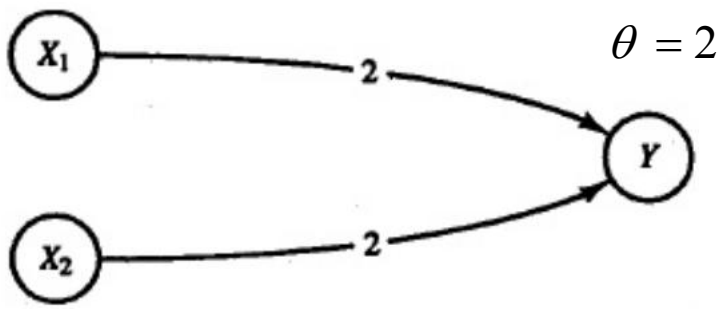
Ejemplos de redes de McCulloch-Pitts que implementan funciones lógicas

AND



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

OR



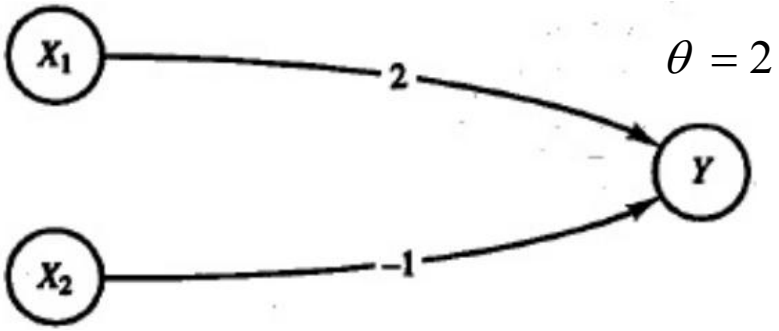
x_1	x_2	y
1	1	1
1	0	1
0	1	1
0	0	0

$$y\_in = \sum_{i=1}^{2} x_i w_i$$

$$f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq \theta \\ 0 & \text{si } y\_in < \theta \end{cases}$$

Ejemplos de redes de McCulloch-Pitts que implementan funciones lógicas

AND NOT

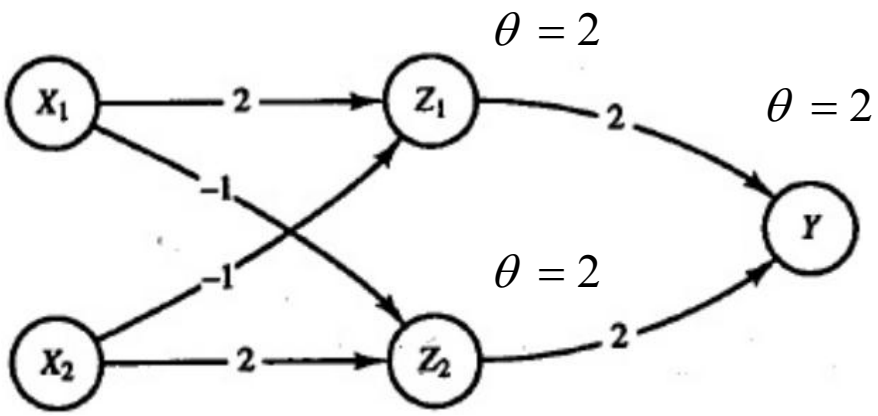


x_1	x_2	y
1	1	0
1	0	1
0	1	0
0	0	0

$$f(y\_in) = \begin{cases} 1 & \text{si } y\_in \geq \theta \\ 0 & \text{si } y\_in < \theta \end{cases}$$

Ejemplos de redes de McCulloch-Pitts que implementan funciones lógicas

XOR



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

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

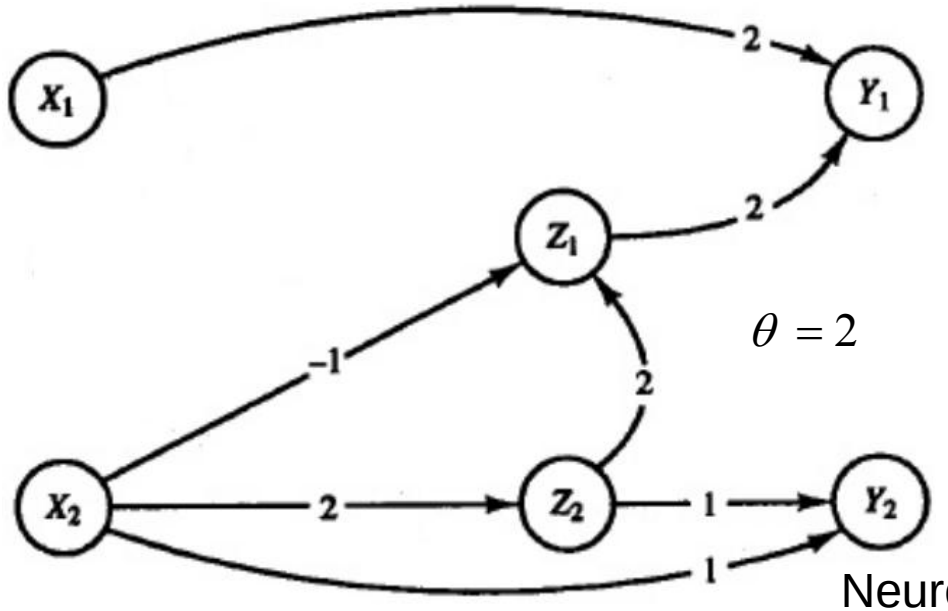
$z_1 = x_1$ AND NOT x_2
$z_2 = x_2$ AND NOT x_1

$y = z_1$ OR z_2

Ejemplo de utilización de retrasos temporales en las redes de McCulloch-Pitts

En estudios de fisiología es conocido que si se aplica un estímulo frío en la piel por un periodo muy breve se percibe como algo caliente. Si el mismo estímulo se aplica durante más tiempo se percibe como frío.

Este fenómeno se puede modelar con una red de McCulloch-Pitts que haga uso de los retrasos temporales discretos inherentes a estas redes.



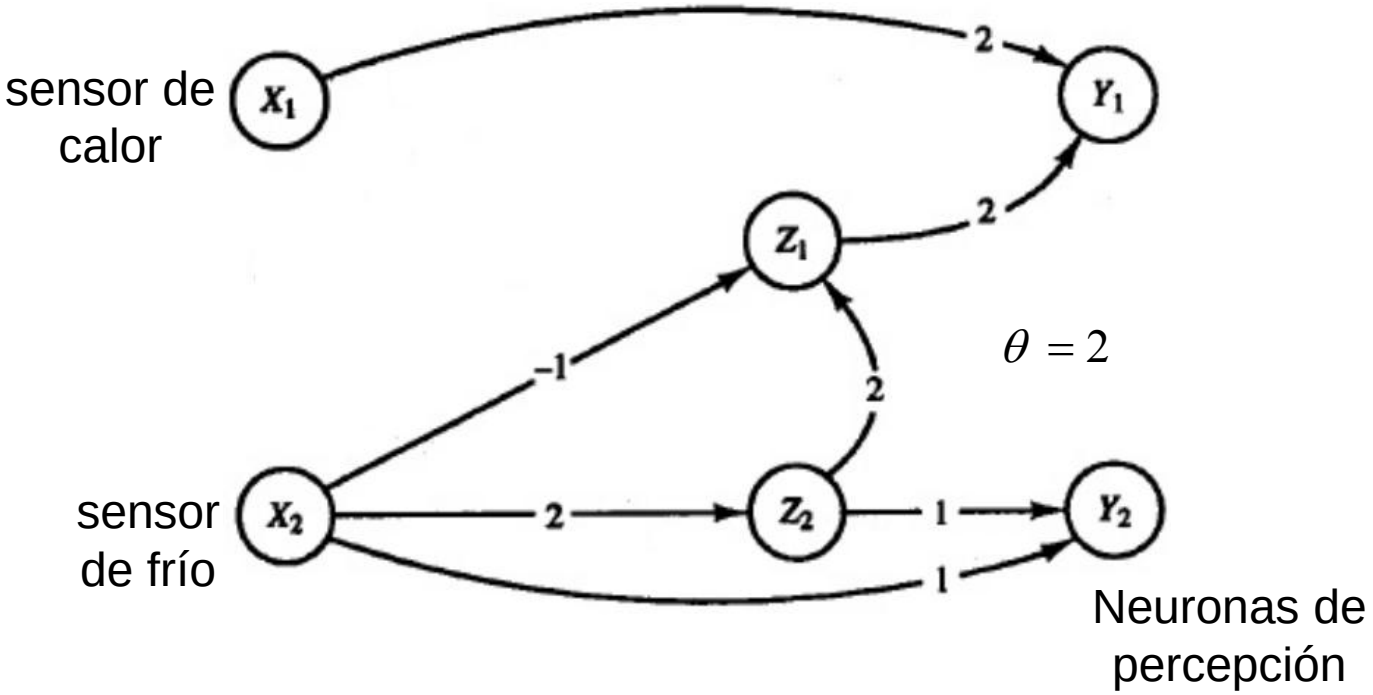
Ejemplo de utilización de retrasos temporales en las redes de McCulloch-Pitts

Estímulos (x_1 , x_2):

(1,0) calor
(0,1) frío

Respuesta esperada (y_1,y_2):

frío (0,1) si el estímulo frío se aplica en dos pasos de tiempo

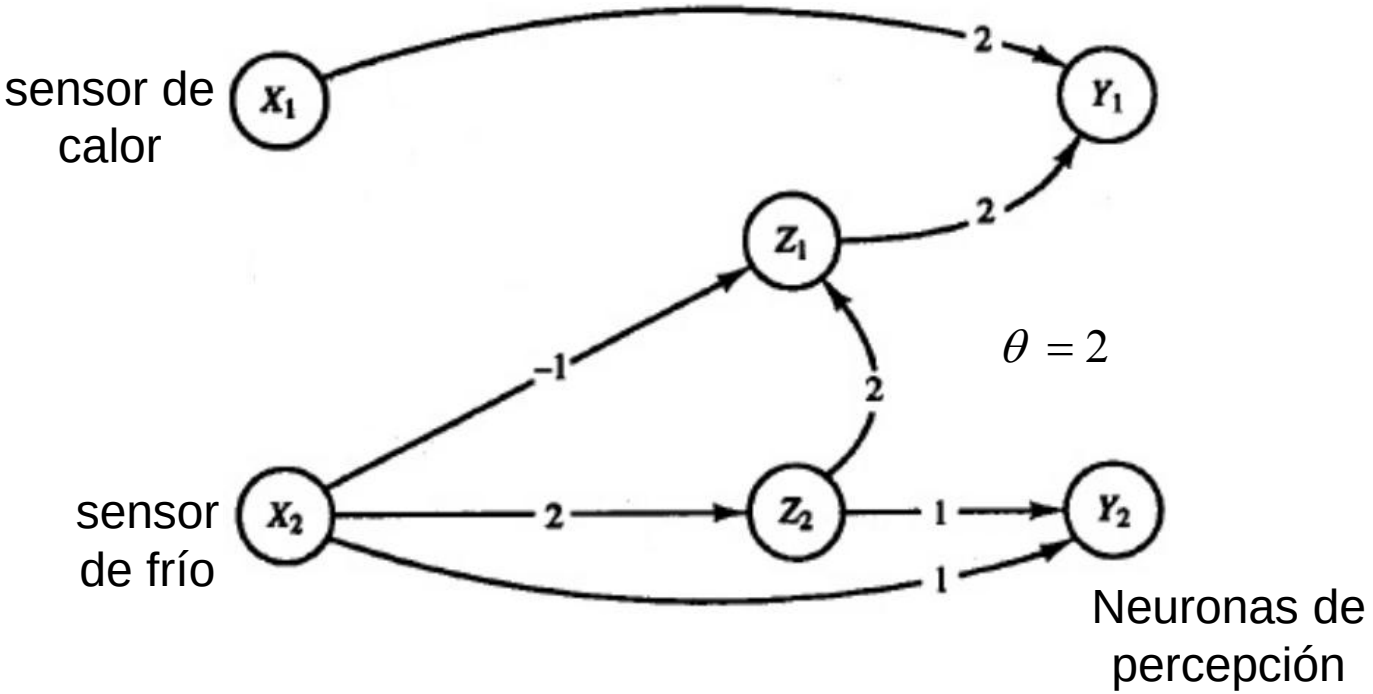


Ejemplo de utilización de retrasos temporales en las redes de McCulloch-Pitts

Respuesta esperada (y_1,y_2):
frío (0,1) si el estímulo frío se aplica en dos pasos de tiempo

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

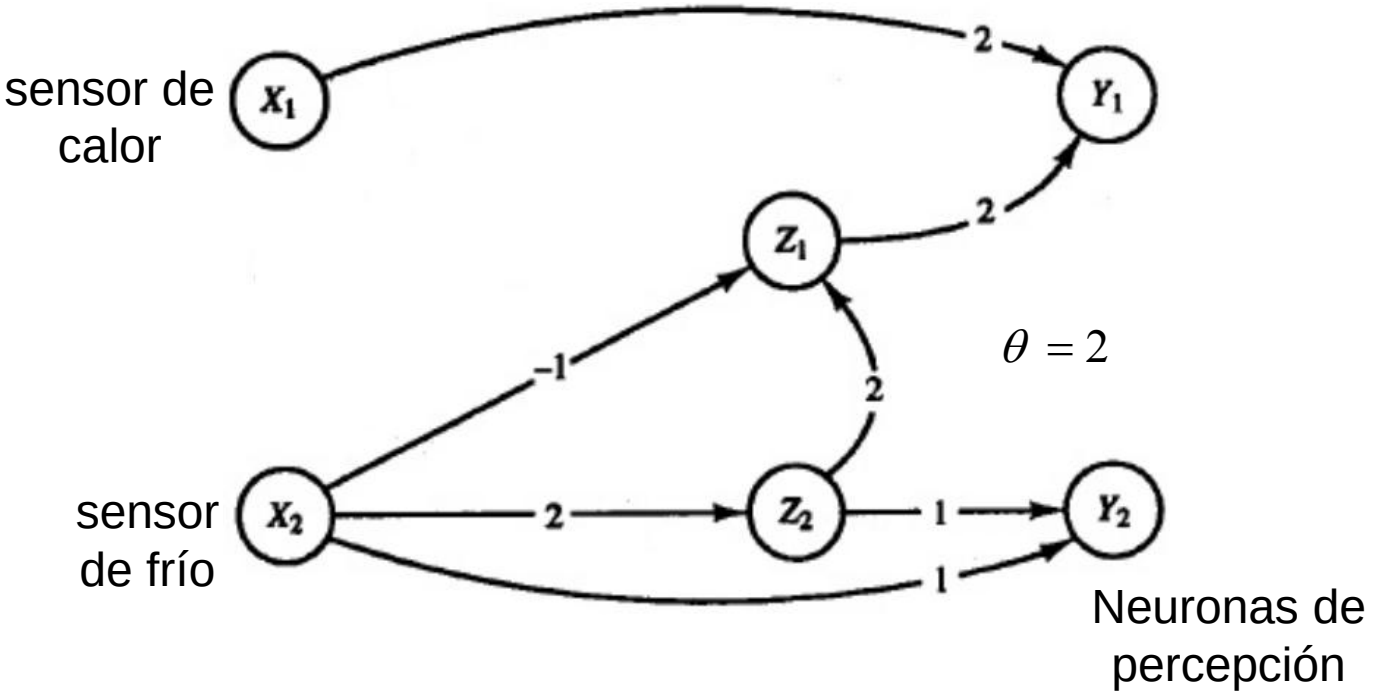
La activación de la neurona Y_2 en el instante t es $y_2(t)$
(1 si se percibe frío y 0 si no se percibe frío)



Ejemplo de utilización de retrasos temporales en las redes de McCulloch-Pitts

Respuesta esperada (y_1,y_2):
calor (1,0) si se aplica el estímulo calor y también cuando se aplica un estímulo frío en un paso de tiempo

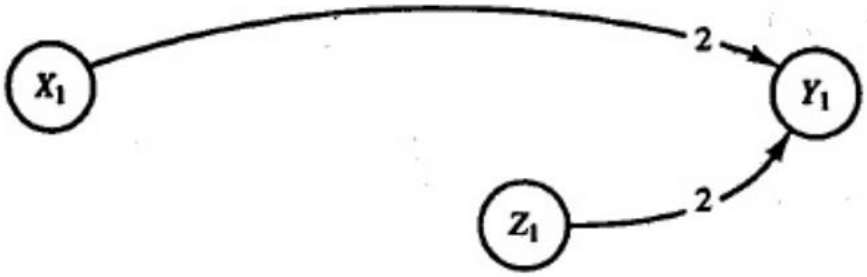
$$y_1(t) = \{x_1(t\text{-}1)\} \text{ OR } \{x_2(t\text{-}3) \text{ AND NOT } x_2(t\text{-}2)\}$$

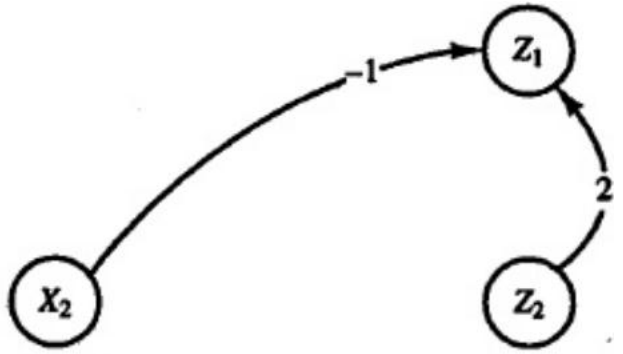


Ejemplo de utilización de retrasos temporales en las redes de McCulloch-Pitts

Respuesta esperada (y_1,y_2):
calor (1,0) si se aplica el estímulo calor y también cuando se aplica un estímulo frío en un paso de tiempo. Para Y_1:

$y_1(t) = x_1(t\text{-}1)$ OR $z_1(t\text{-}1)$ $\qquad$ $z_1(t\text{-}1) = \{z_2(t\text{-}2)\}$ AND NOT $x_2(t\text{-}2)\}$





además $\qquad z_2(t\text{-}2) = x_2(t\text{-}3)$

Por tanto: $\qquad y_1(t) = \{x_1(t\text{-}1)\}$ OR $\{x_2(t\text{-}3)$ AND NOT $x_2(t\text{-}2)\}$

Ejemplo de utilización de retrasos temporales en las redes de McCulloch-Pitts

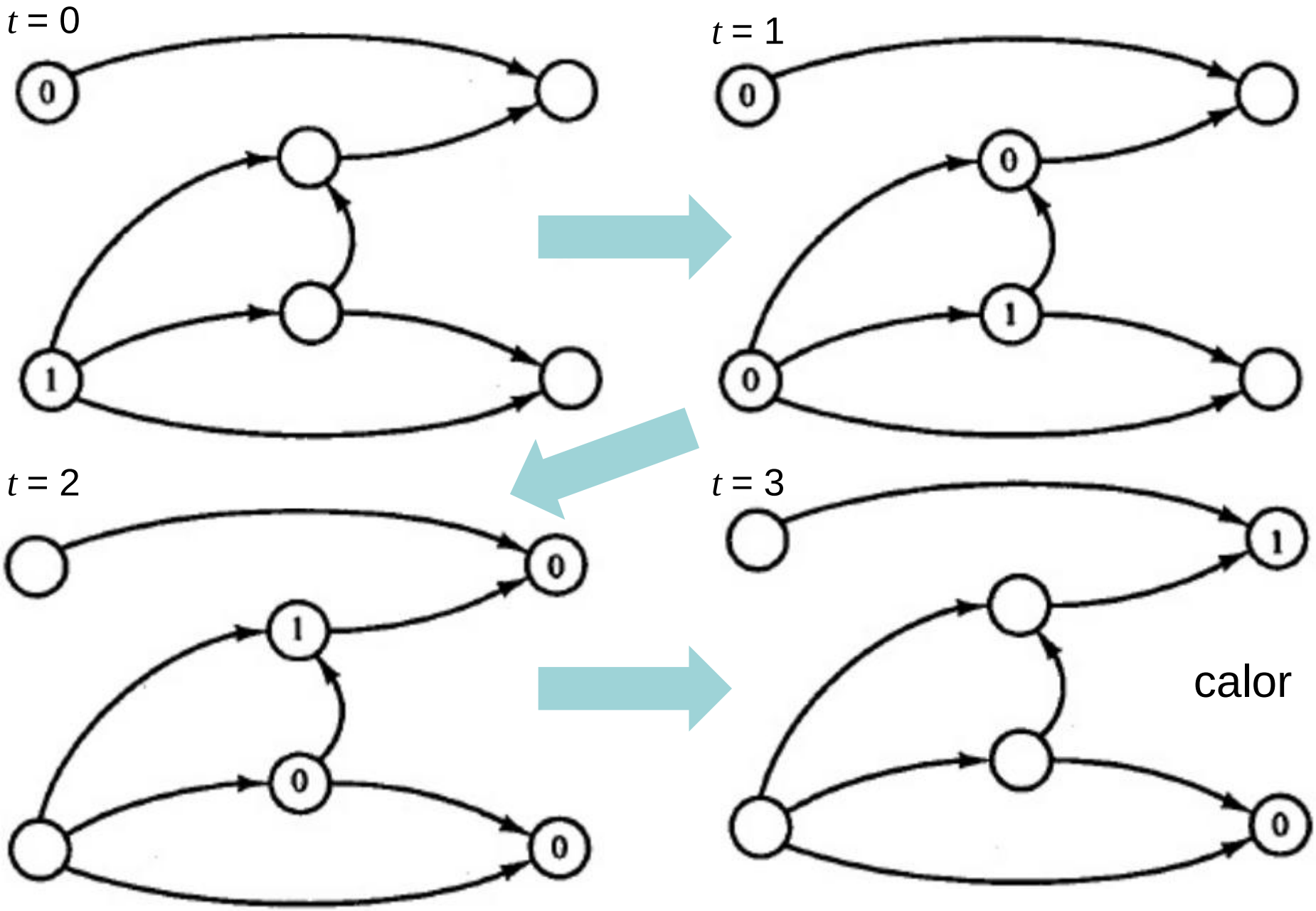
Respuesta esperada (y_1,y_2):
calor (1,0) si se aplica el estímulo calor y también cuando se aplica un estímulo frío en un paso de tiempo. Para Y_2:

$$y_2(t) = z_2(t\text{-}1) \text{ AND } x_2(t\text{-}1)$$

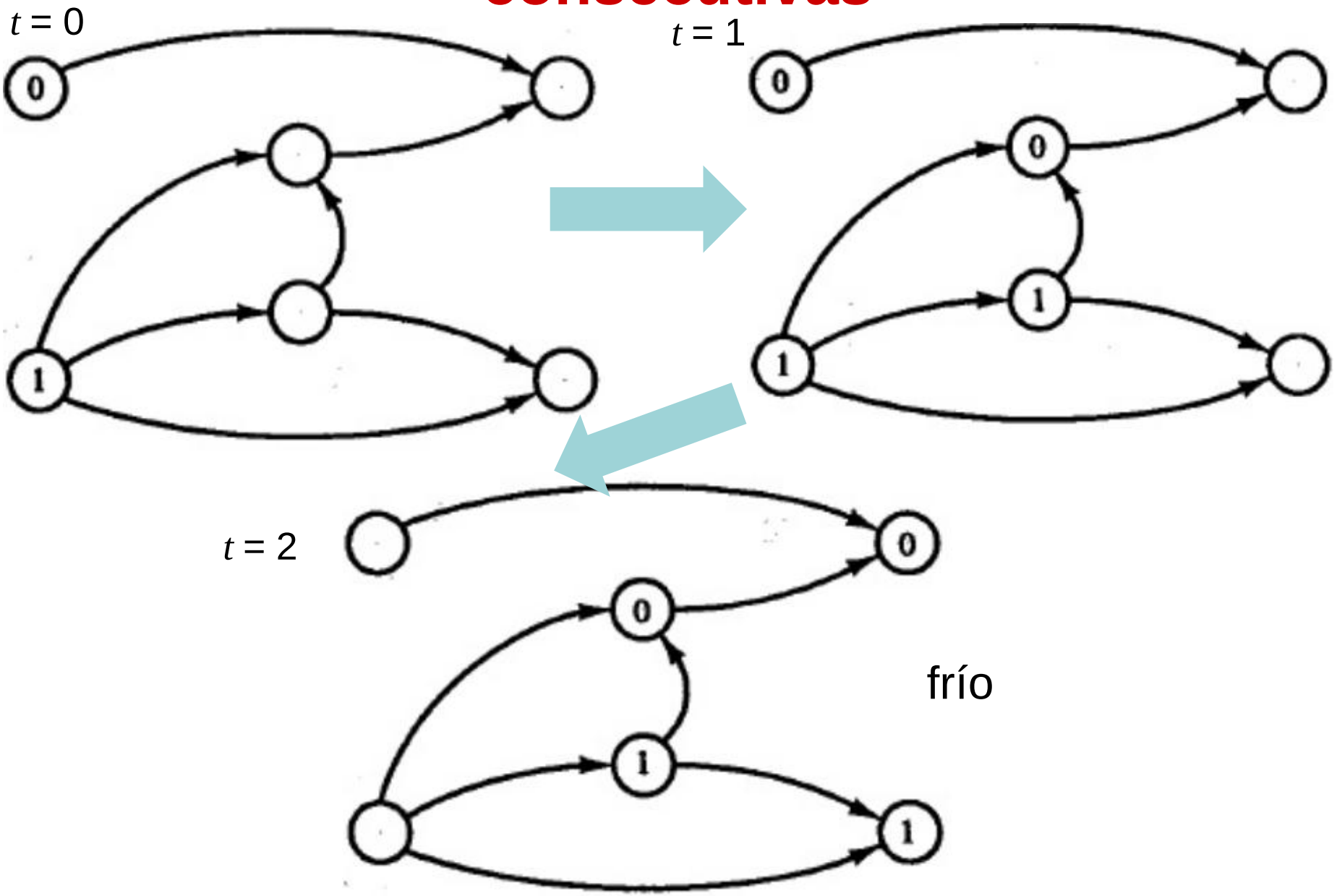
además $z_2(t\text{-}1) = x_2(t\text{-}2)$

Por tanto: $y_2(t) = x_2(t\text{-}2) \text{ AND } x_2(t\text{-}1)$

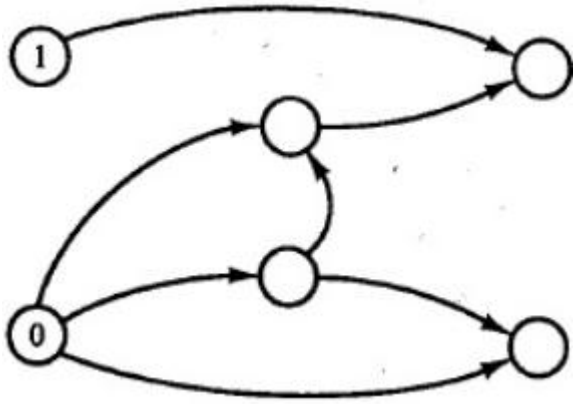
Estímulo frío aplicado una única vez

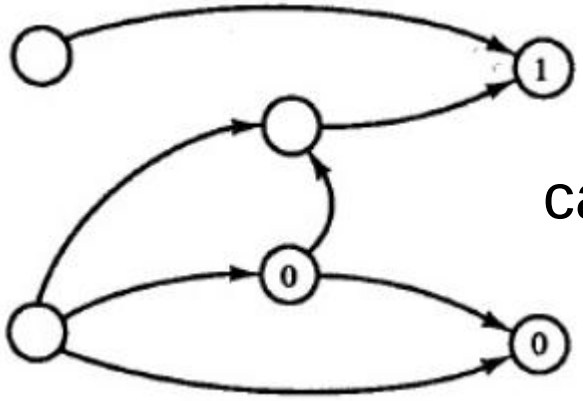


Estímulo frío aplicado dos veces consecutivas



Estímulo caliente aplicado una vez

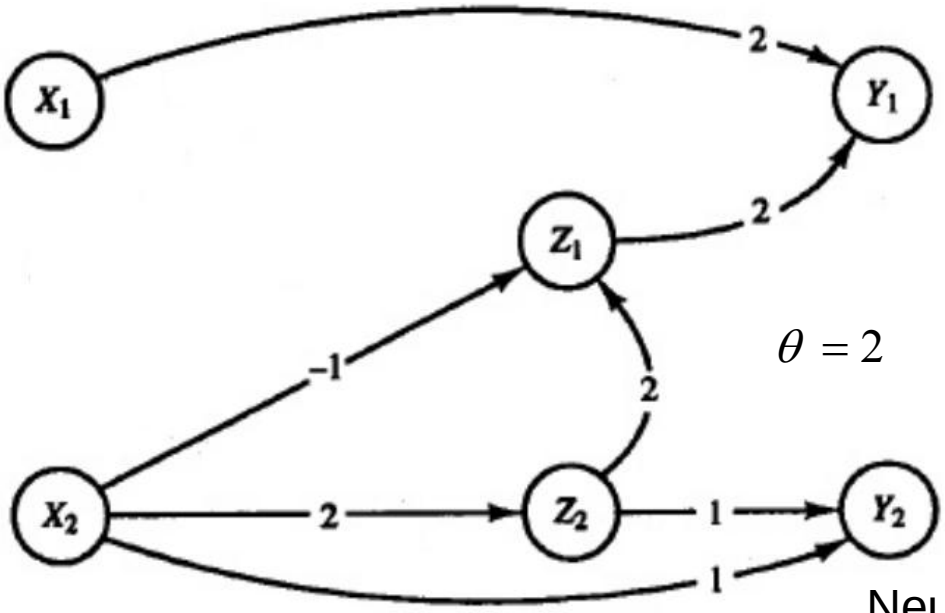




calor

sensor de calor

sensor de frío



$\theta = 2$

Neuronas de percepción

Breve historia de las redes neuronales

- **1940s:**
  - **Neuronas de McCulloch-Pitts**. Warren McCulloch & Walter Pitts designaron los que se considera la primera red neuronal en 1943. Las neuronas de McCulloch-Pitts implementan funciones lógicas que pueden combinarse en la red para realizar tareas complejas. Introdujeron el concepto de umbral en las neuronas
  - **Aprendizaje Hebbiano.** Donald Hebb, propuso la primera regla de aprendizaje para redes neuronales artificiales en 1949.
- **1950s y 1960s (primera era de oro de las redes neuronales):**
  - **Perceptrones:** Frank Rosenblatt, Block, Minsky y Papert desarrollaron el concepto de perceptrón con su correspondiente regla de aprendizaje.
  - **ADALINE:** Bernard Widrow y Marcian Hoff desarrolloaron en 1960 una regla de aprendizaje denominada regla delta que reduce la diferencia entre la entrada a la neurona de salida y el output deseado lo que conduce al menor error cuadrático medio. Este aprendizaje permite una mayor generalización. El nombre de ADALINE se refiere a ADAptive LInear Neuron.

Breve historia de las redes neuronales

• **1970s** (periodo de duda por el artículo de Minsky y Papert sobre las limitaciones del perceptrón):

  • **Redes de Kohonen**: Teuvo Kohonen desarrolló una familia de redes de memoria asociativa y mapas auto-organizativos.

  • **Anderson.** James Anderson desarrolló la red denominada Brain-State-in-a-Box que trunca la salida lineal para que no se vuelva demasiado grande en las iteraciones de la red.

  •**Grossberg y Carpenter**: Stephen Grossberg y Gail Carpenter desarrollaron las redes autoorganizativas de categorías ART (Adaptive Resonance Theory)

• **1980s y 1990s** (nuevo periodo de éxito de las redes neuronales)

  •**Backpropagation:** regla de aprendizaje para el perceptrón multicapa diseñada por David Parker yLe Cun, y mejorada por Rumelhart y otros.

  • **Redes de Hopfield:** demostró que las redes asociativas pueden aprender por analogía a la autoorganización de los sistemas físicos.

  •**Neocognitrón:** Kinihiko Fukushima desarrolló una familia de redes para el reconocimiento de caracteres.

  •**Máquinas de Boltzman**: regla de aprendizaje basada en funciones de densidad de probabilidad.

Breve historia de las redes neuronales