



# Aprendizaje profundo

**Ivan Vladimir Meza Ruiz, IIMAS, UNAM**

@ivanvladimir



Escanear para acceder a las diapositivas

Link permanente

[https://docs.google.com/presentation/d/1Jy69MOJltDGwMyFbjCvxKz\\_X46ewBTASk-Fr6wAlpwg/edit?usp=sharing](https://docs.google.com/presentation/d/1Jy69MOJltDGwMyFbjCvxKz_X46ewBTASk-Fr6wAlpwg/edit?usp=sharing)

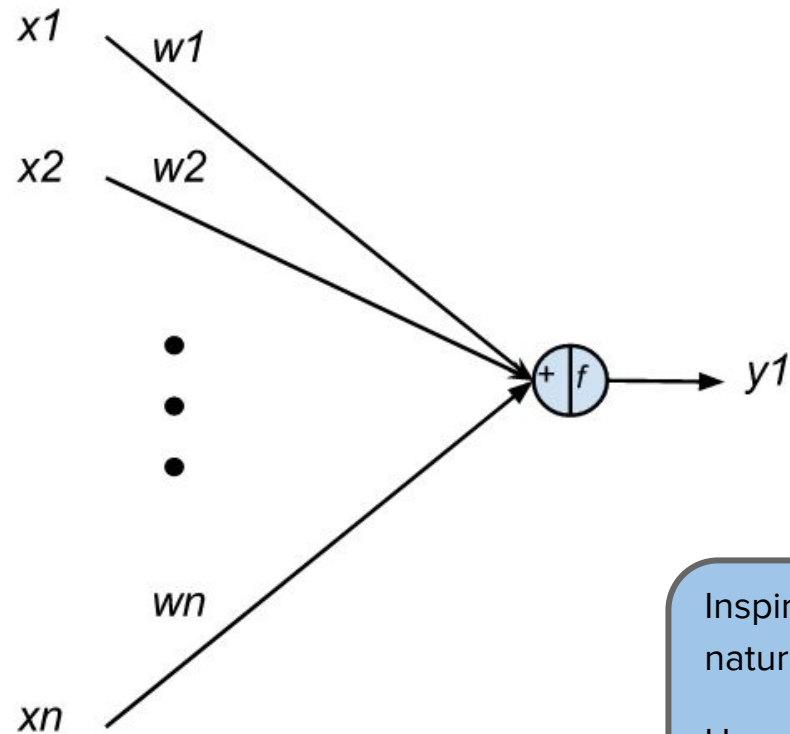
**Nuestra primera  
red no es una red**

# Ejemplo

Registros médicos, determinar si está sano o no: miles, decenas de miles, cientos de miles, millones, decenas de millones

$x_1$	$x_2$	$x_3$	...	$x_n$	$y$
3.4	1.0	2.1	...	12.0	0
6.0	1.0	22.0	...	8.9	1
...	...	...	...	...	...
8.5	0.0	11.9	...	3.2	0

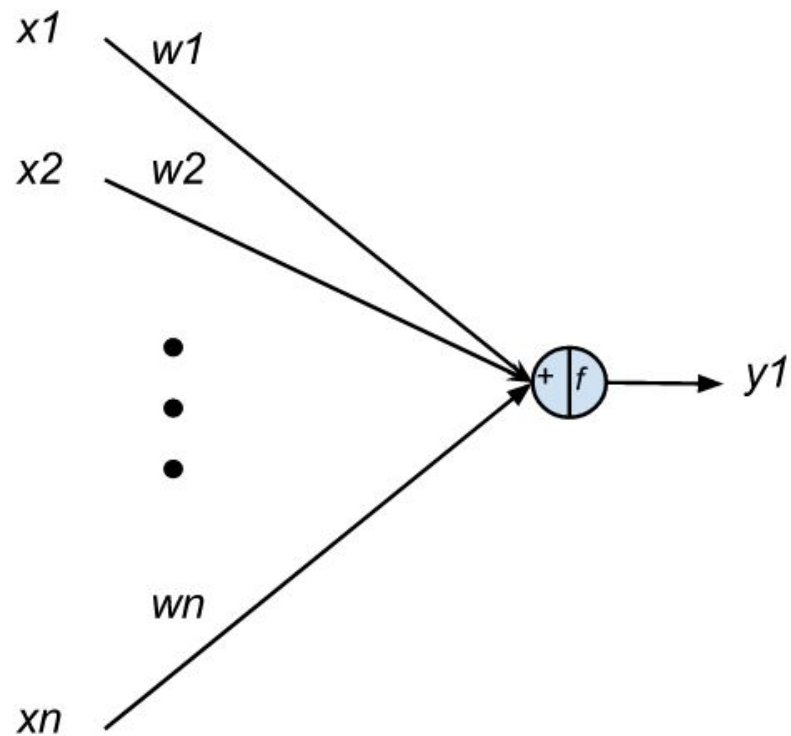
# **Algunos experimentos mentales**



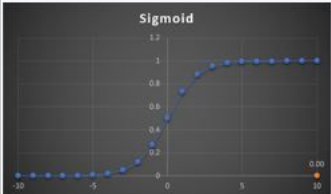
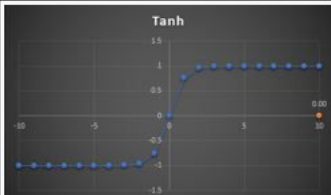
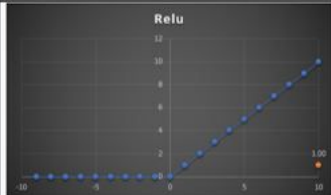
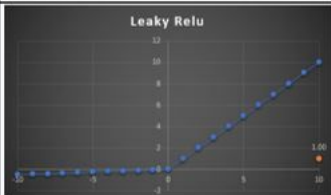
Inspirada en la redes neuronales naturales

Una neurona recibe señales de entrada


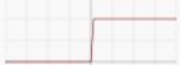


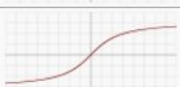




Que la activa o inhibe el nodo



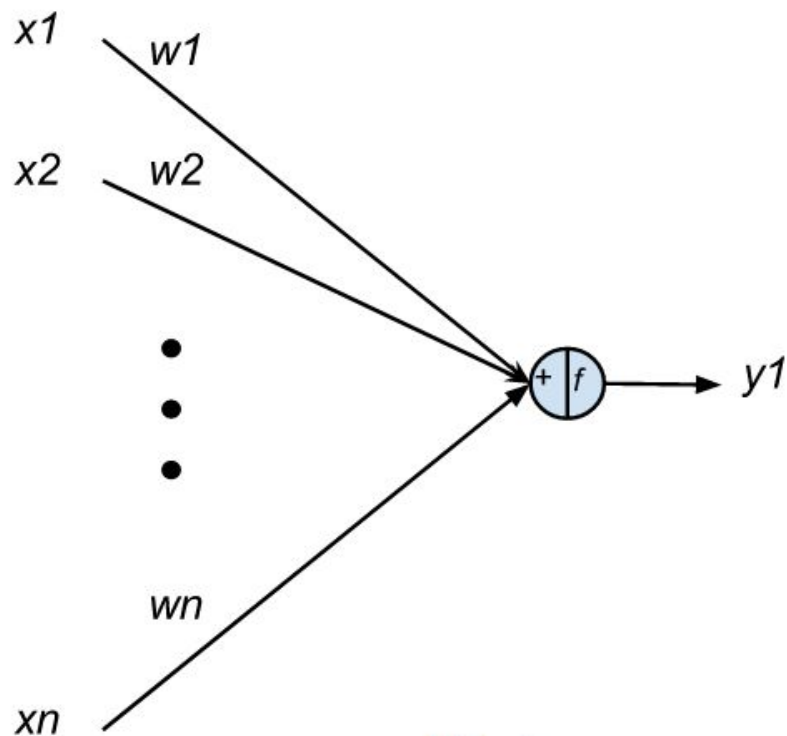
$$y = f(w_1 x_1 + w_2 x_2 + \dots + w_n x_n)$$

Name	Plot	Equation	Derivative
Sigmoid		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
Tanh		$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$f'(x) = 1 - f(x)^2$
Rectified Linear Unit (relu)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Leaky Rectified Linear Unit (Leaky relu)		$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$



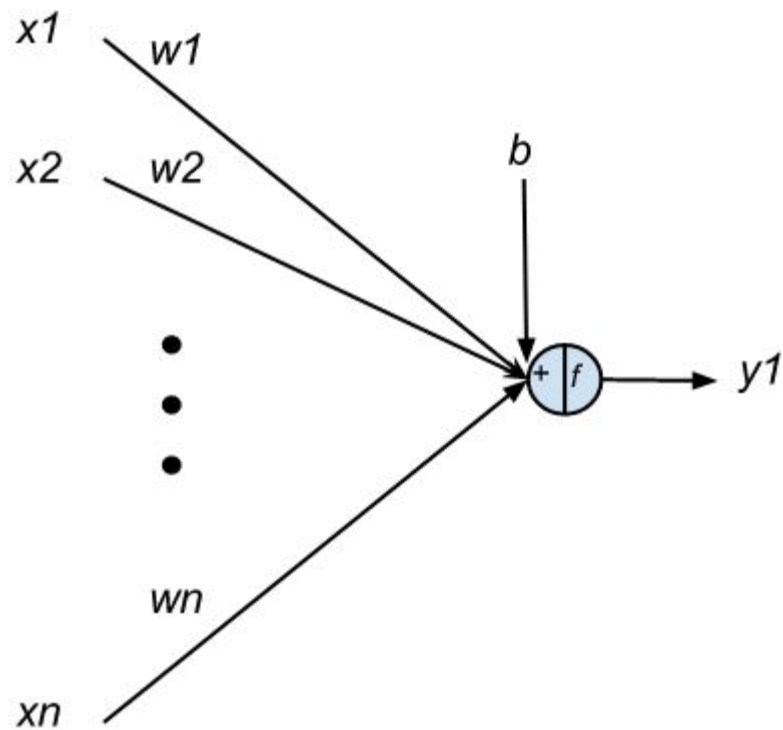
Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) <sup>[2]</sup>		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) <sup>[3]</sup>		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Si sigmoide, regresión logística



$$y = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + \dots + w_n x_n)}}$$

Con bias (sesgo)



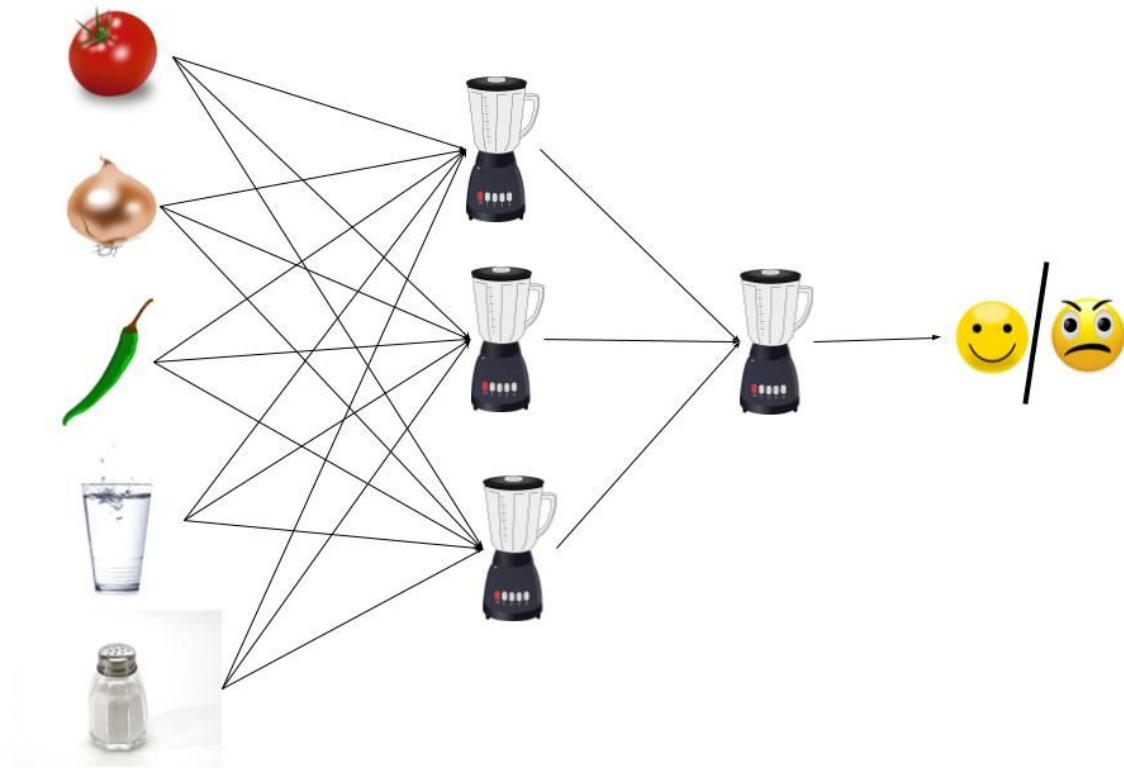
$$y = f(w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b)$$

# Como vectores

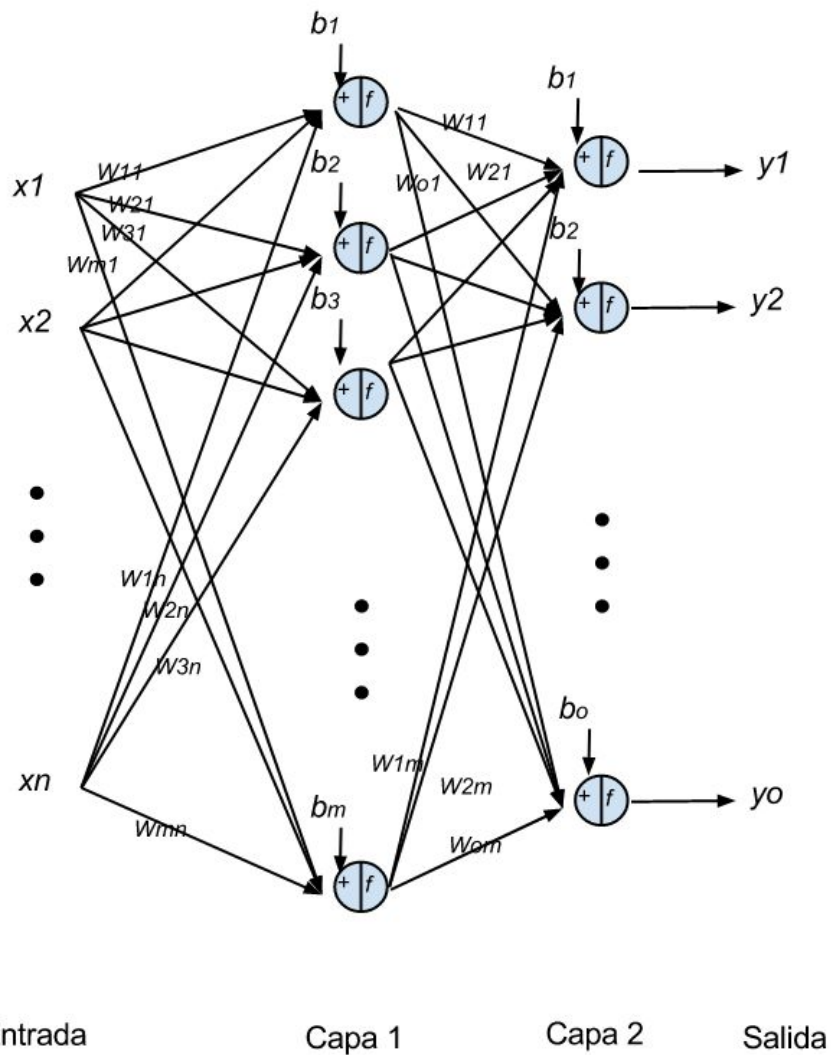
$$y = f(Wx + b)$$

$$[1 \times 1] = [1 \times m] [m \times 1] + [1 \times 1]$$

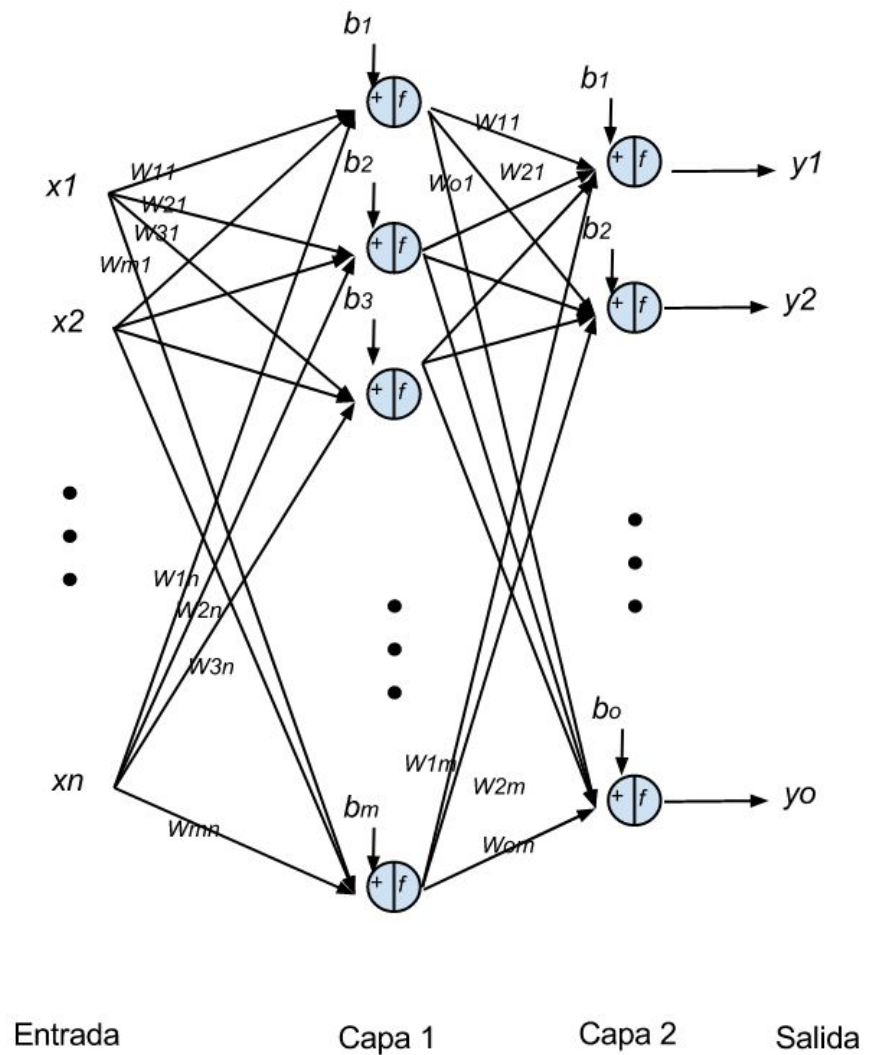
**Ahora multi capa**



# MLP



- Entrada de dimensión del tipo  $x \in \mathbb{R}^n$
- se conecta a la primera capa de  $m$  neuronas
- esta conecta a la segunda capa de  $o$  neuronas
- esta genera la salida  $y \in \mathbb{R}^o$





# Primera capa

$$h^1 = f(W^1x + b^1)$$

$$[m \times 1] = [m \times n][n \times 1] + [m \times 1]$$

## Segunda capa

$$y=f(W^2h^1+b^2)$$

$$[ox1]=[oxm][mx1]+[ox1]$$

# Toda la red

$$h^1 = f(W^1 x + b^1)$$
$$y = f(W^2 h^1 + b^2)$$

## Más de una capa

$$h^1 = f(W^1 x + b^1)$$

$$h^2 = f(W^2 h^1 + b^2)$$

$$h^3 = f(W^3 h^2 + b^3)$$

...

$$y = f(W^L h^{L-1} + b^L)$$

# Funcionamiento típico

Ejemplo diagnóstico médico

- Entrada, historia clínica como vector
- Salida, enfermo si y no

Llega un nuevo paciente

- Vectorizo al paciente
- Hago la cadena de operaciones, y veo la salida

¿Qué tengo?

¿Qué no tengo?

$$h^1 = f(W^1 \mathbf{x} + b^1)$$

$$h^2 = f(W^2 h^1 + b^2)$$

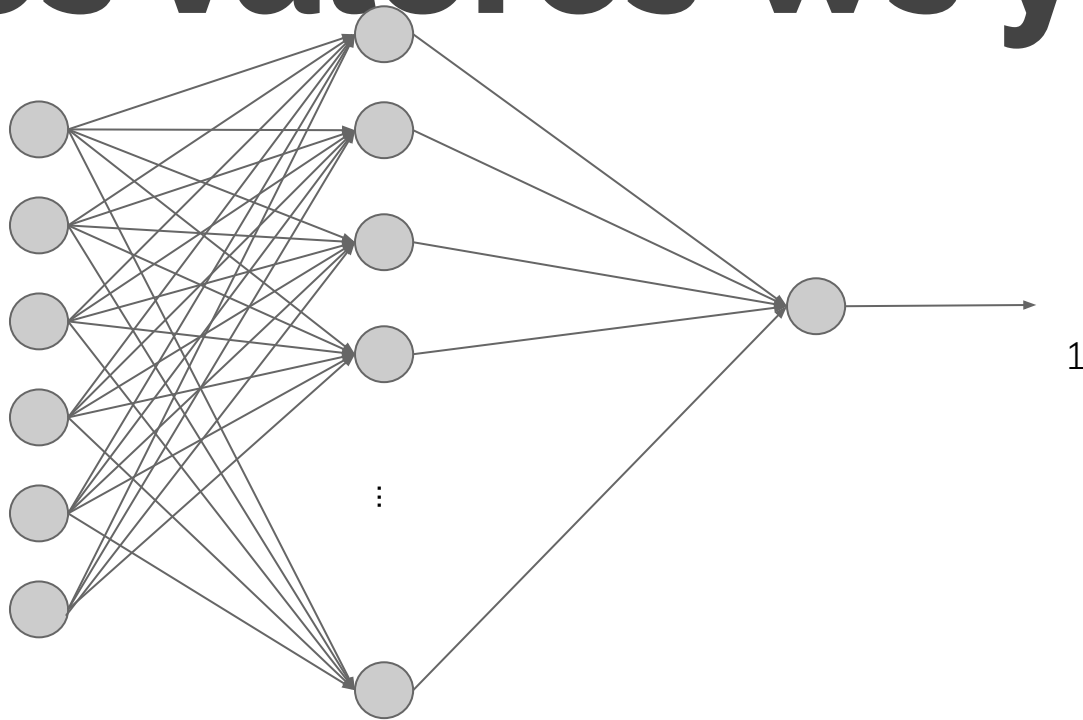
$$h^3 = f(W^3 h^2 + b^3)$$

...

$$y = f(W^L h^{L-1} + b^L)$$



# ¿De dónde salen los valores $W$ s y $b$ s?





# Calculo de $W$ s y $b$ s

Intuición de cálculo de parámetros/modelo

- Alimentar un paciente vectorizado para el cual sé el diagnóstico
- Checar la predicción
- Calcular que tan diferente es la predicción con el valor real
- Cambiar la red proporcional a la diferencia
- Hacer esto para muchos pacientes, con la esperanza que la red se estabilice

# Hasta ahora

## Intuición

- Alimentar un paciente vectorizado para el cual sé el diagnóstico ✓
- Checar la predicción ✓
- Calcular qué tan diferente es la predicción con el valor real ✗
- Cambiar la red proporcional a la diferencia ✗
- Hacer esto para muchos pacientes, con la esperanza que la red se estabilice ✗

# Cálculo del error

Para un dataset

$$E(a^L, y) = \frac{1}{2n} \sum_{i=0}^n (a_i^L - y_i)^2$$

Donde

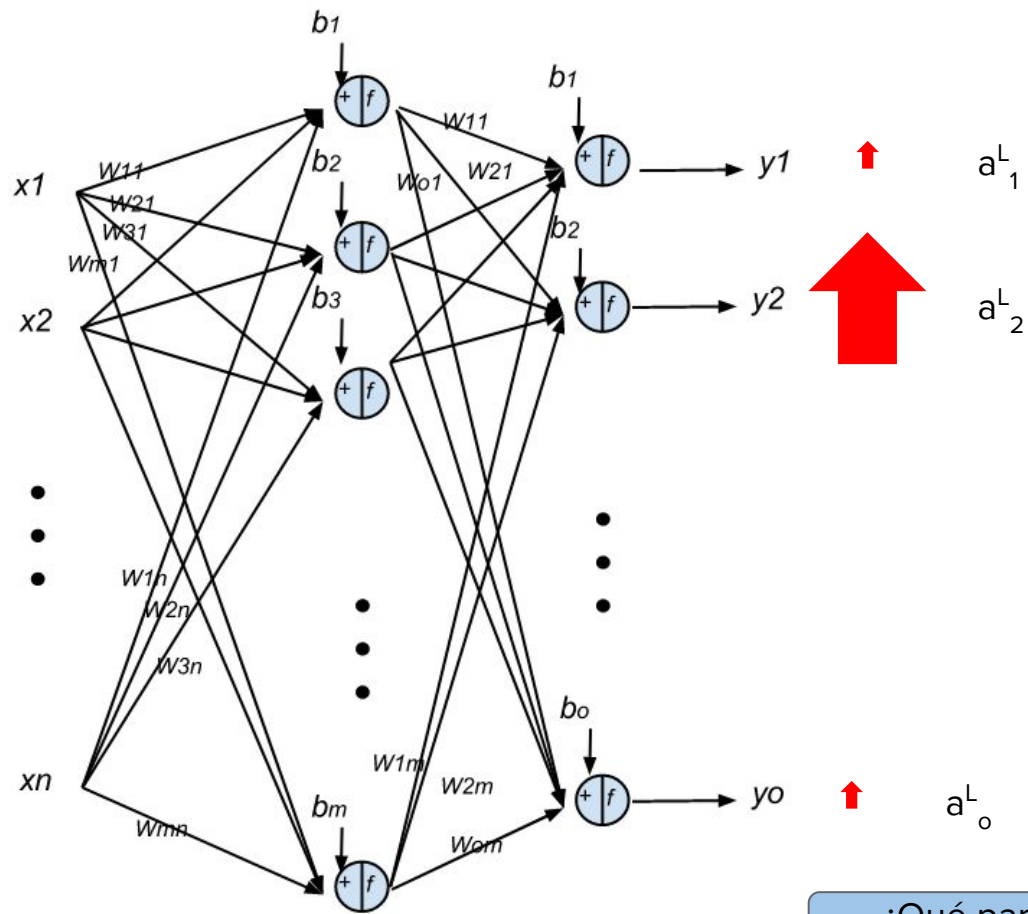
$$a^L = f(W^m h^{m-1} + b^m)$$

# Lo que queremos

Es que la cantidad  $E(a^L, y)$  se haga pequeña, si cero, significa que la red imita el comportamiento del dataset.

Lo que necesitamos es calcular una  $W$  y  $b$  nueva de forma iterativa para cada ejemplo que garantice reducir el error para ese ejemplo. Si llamamos  $\theta = (W \text{ y } b)$

# **Algunos experimentos mentales**



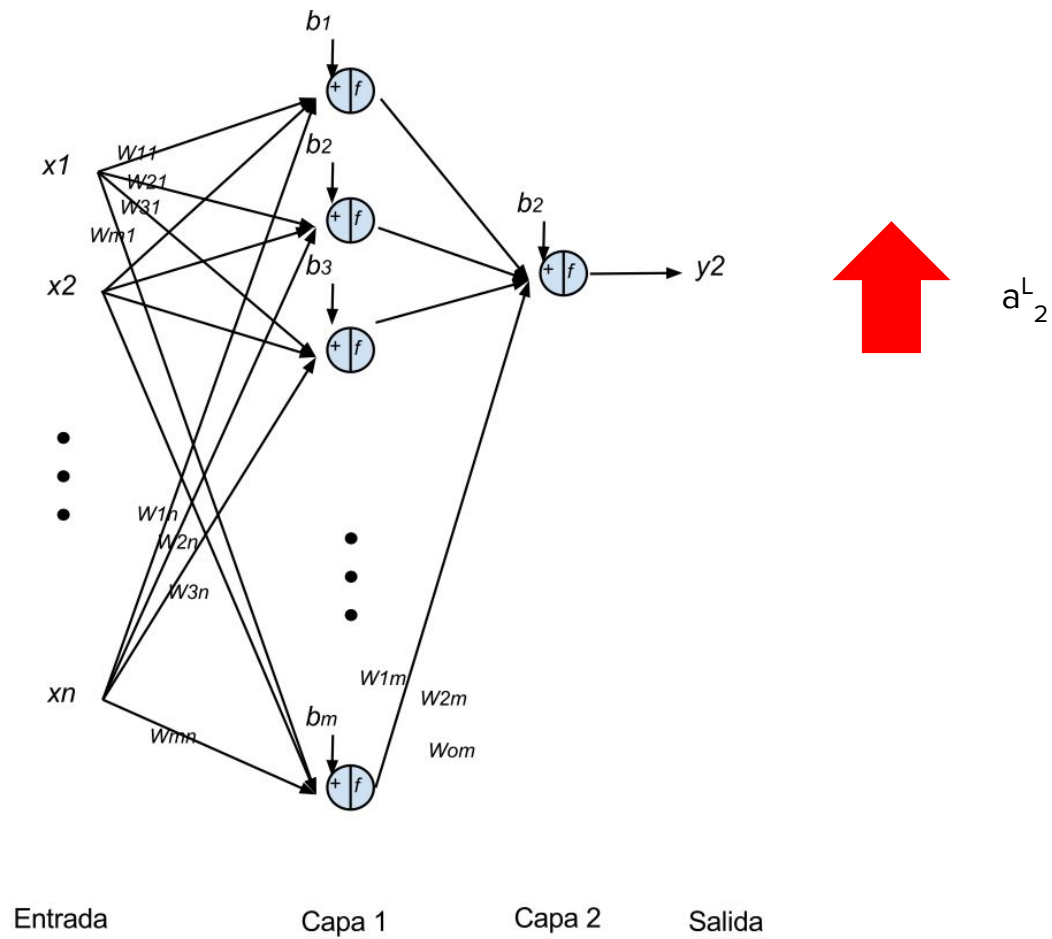
¿Qué parte es el culpable?

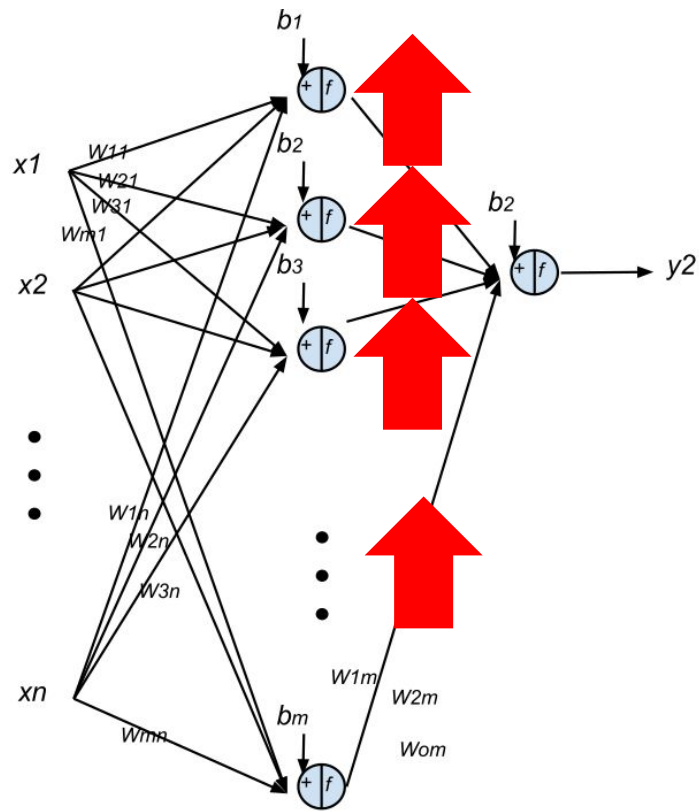
Entrada

Capa 1

Capa 2

Salida





$a_2^L$

Entrada

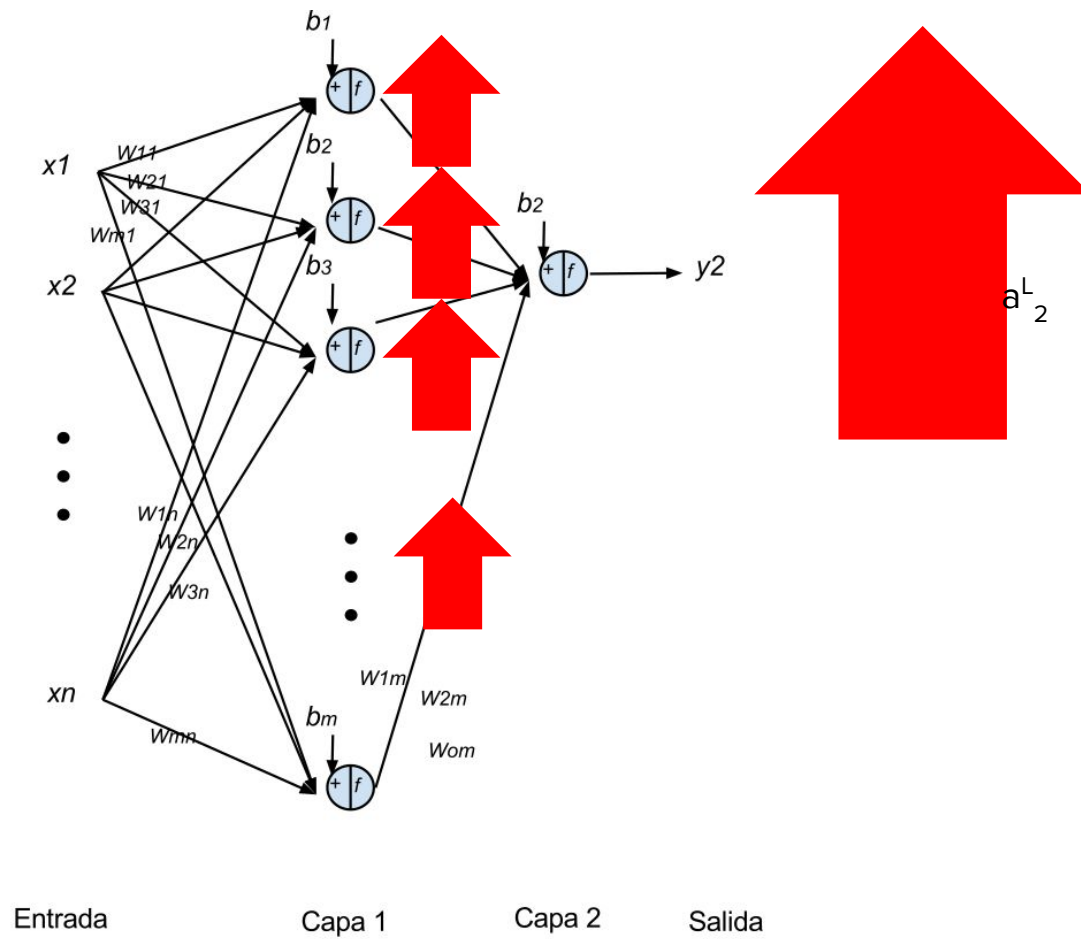
Capa 1

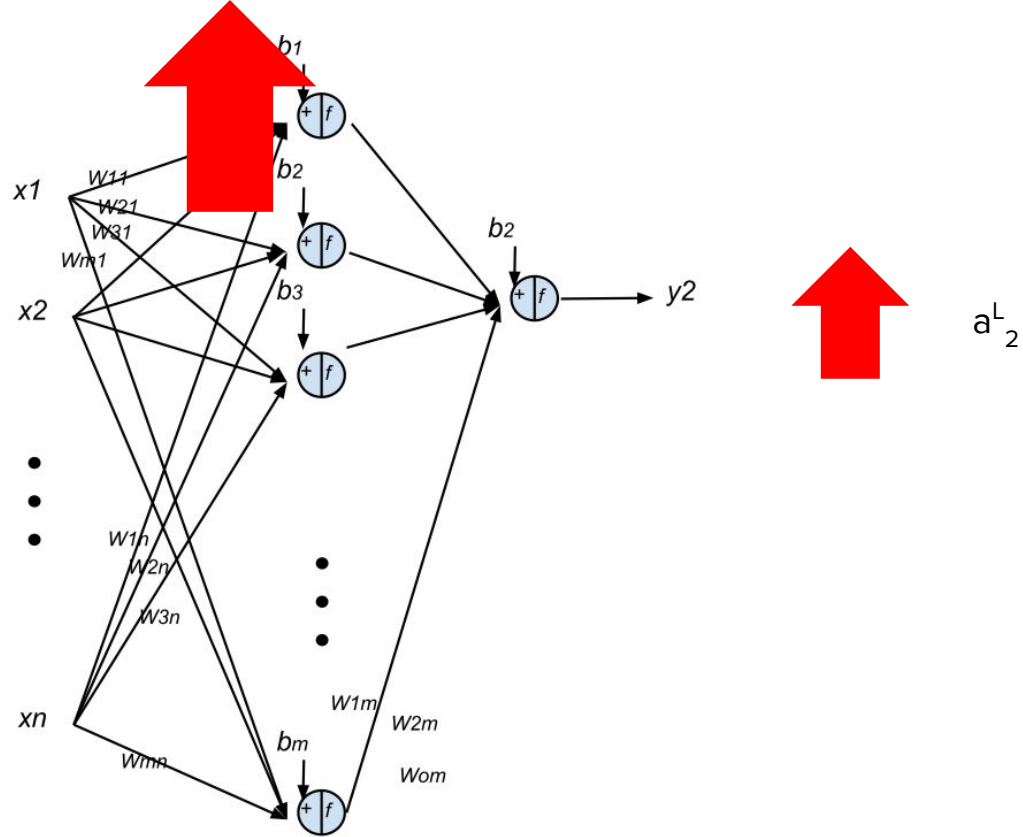
Capa 2

Salida

¿ pesos más grandes?







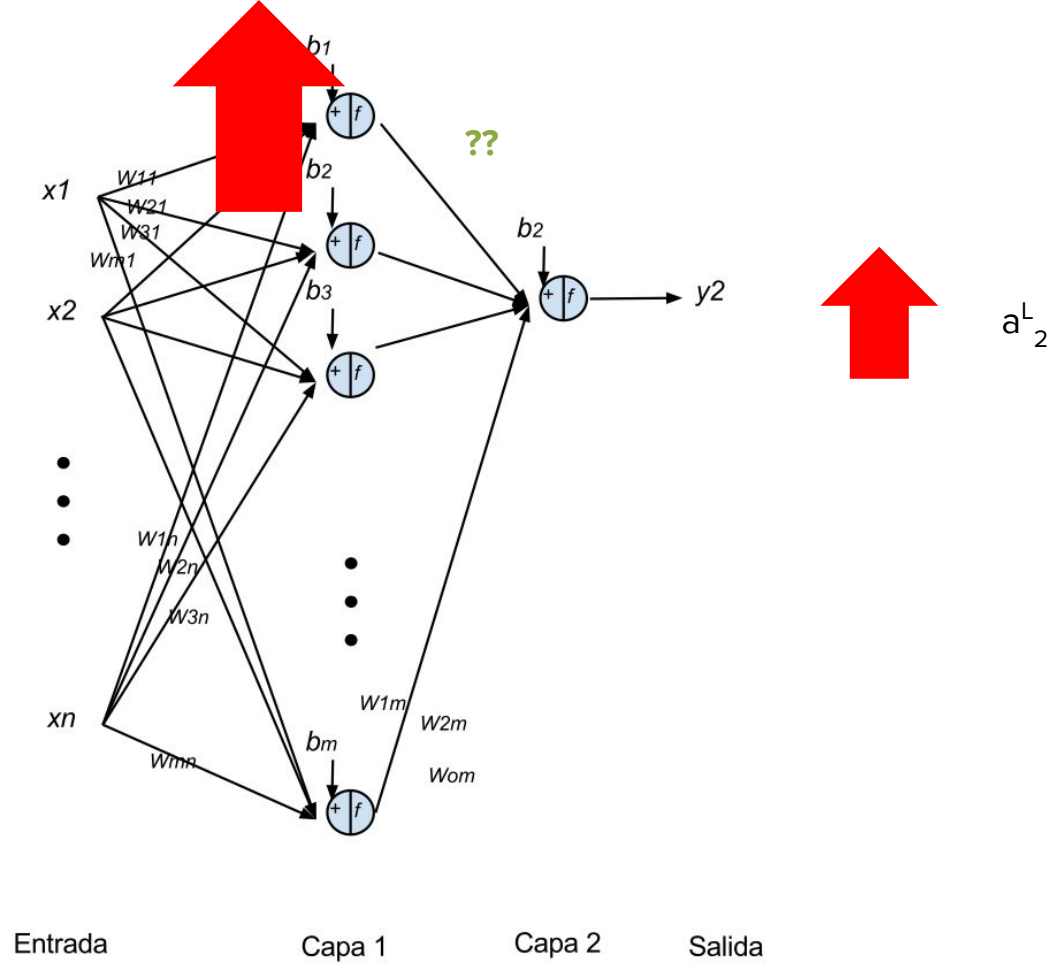
Entrada

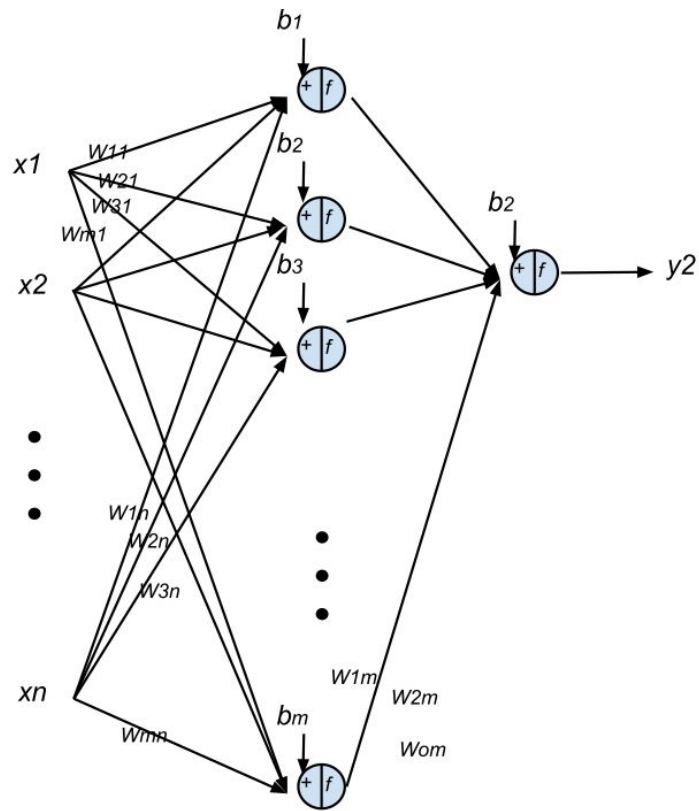
Capa 1

Capa 2

Salida

¿cuándo esa suma es grande?





$a_2^L$

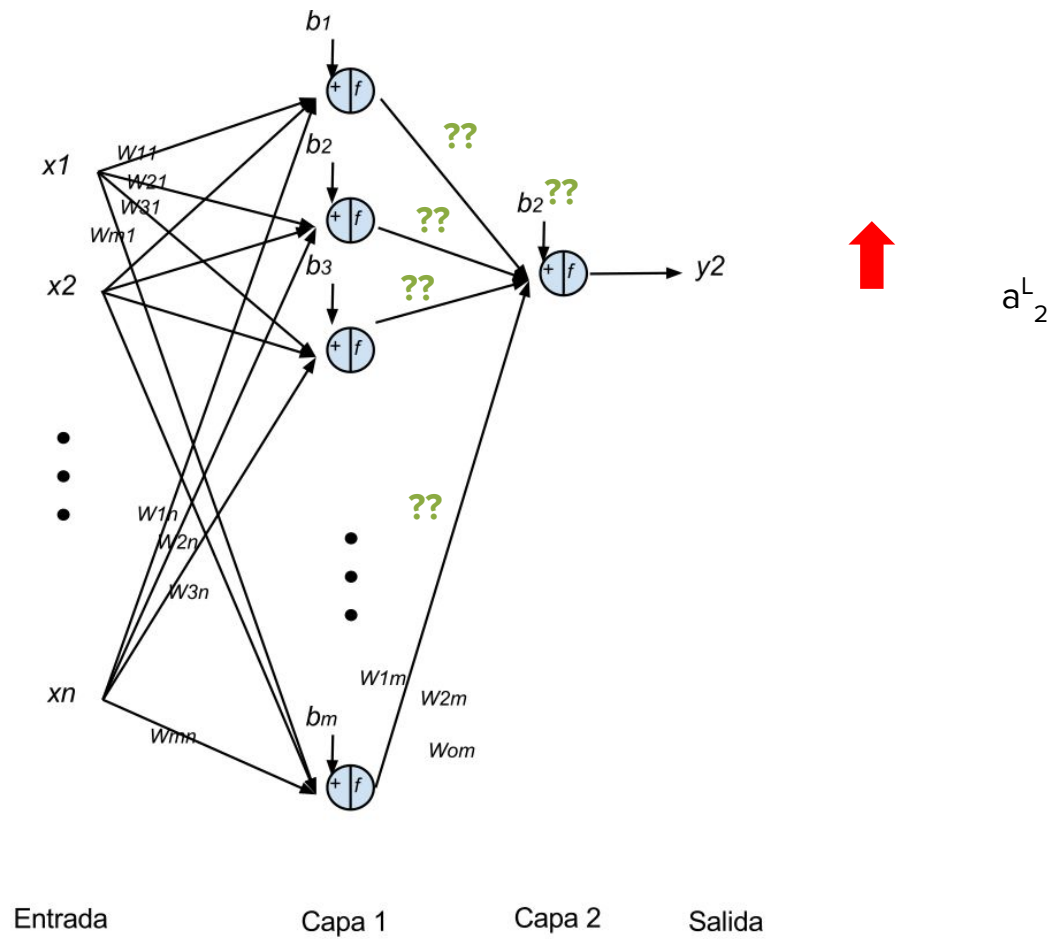
Entrada

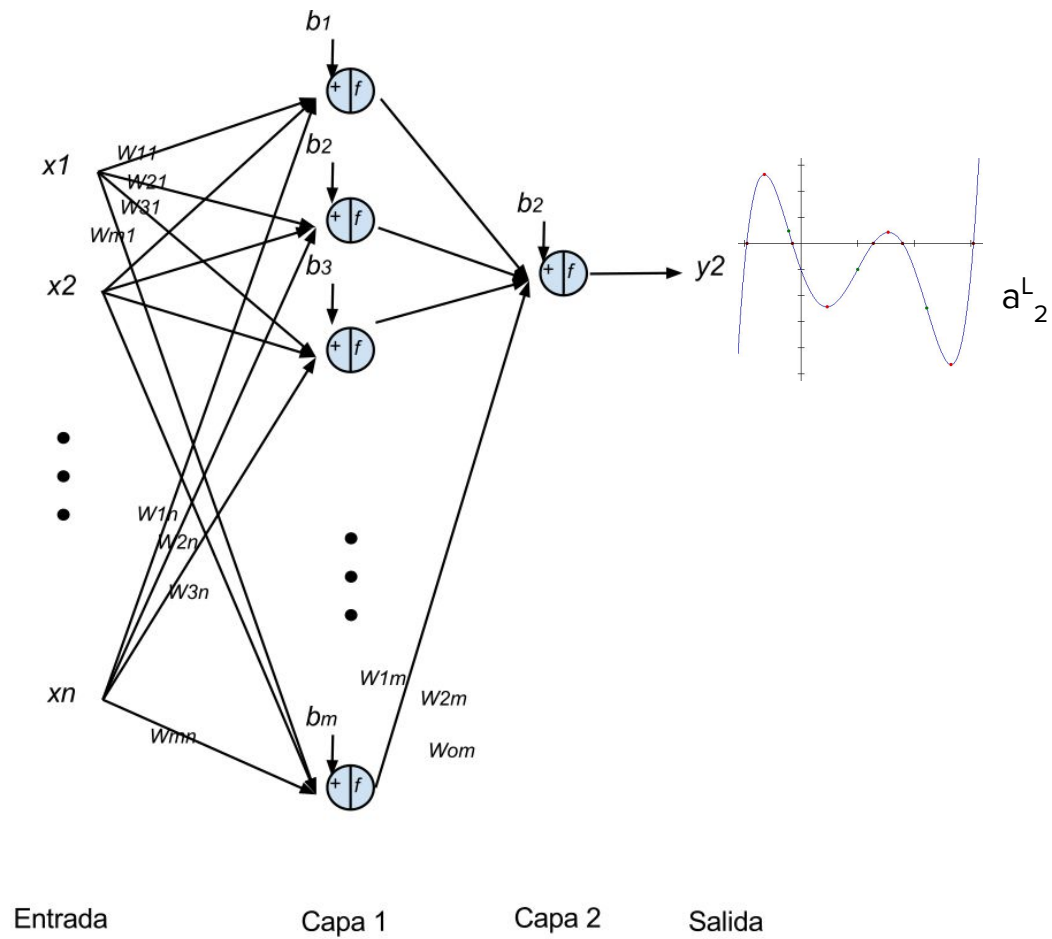
Capa 1

Capa 2

Salida

¿Qué es lo que queremos?

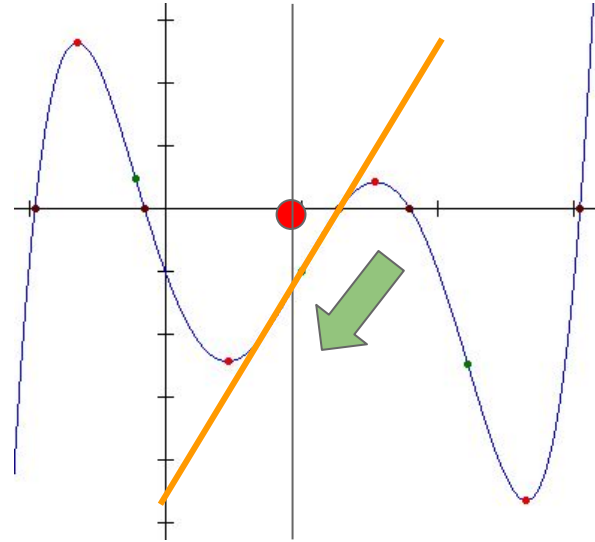




¿Si tuviera un mecanismo para saber  
cómo crece una función en un punto  
específico?

**Entra a escena la derivada**

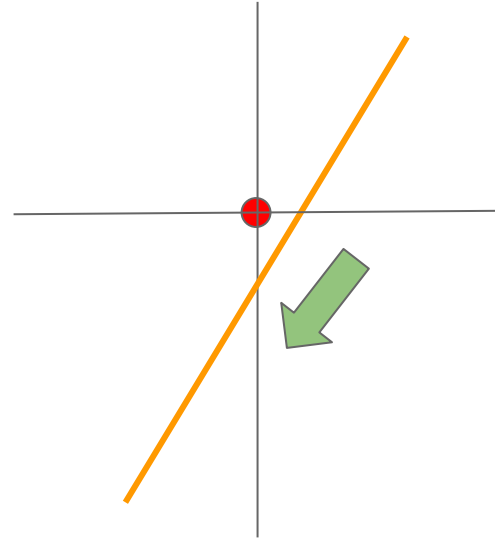
$$\frac{\partial E}{\partial w_{ij}^l}$$





**Entra a escena la derivada**

$$\frac{\partial E}{\partial w_{ij}^l}$$



# Reemplazando

$$\frac{\partial E}{\partial w_{ij}^l} = \frac{\partial \frac{1}{2}(a^L - y_i)^2}{\partial w_{ij}^l}$$

**Pero  $a^L$**

$$= \frac{\partial \frac{1}{2} (f(W^L h^{L-1} + b^L) - y_i)^2}{\partial w_{ij}^l}$$

$$\hat{\theta} = \theta - k \nabla E(a^L, y)$$

$$\nabla = \left( \frac{\partial E}{\partial w_{jk}^l}, \frac{\partial E}{\partial b_j^l} \right)$$

Gradient descent

## **Símbolo auxiliar**

Error en la neurona  $j$  de la capa /

$$\delta_j^l = \frac{\partial E}{\partial h_j^l}$$

# Última capa

$$\delta_j^L = \frac{\partial E}{\partial a_j^L} f'^L(h_j^L)$$

$$\frac{\partial E}{\partial a_j^L} = (a_j^L - y_j)$$

# Rescribiendo

$$\delta^L = (a^L - y) \odot f'^L(h^L)$$

# Capa anterior

$$\delta^l = ((W^{l+1})^T \delta^{l+1}) \odot f'^l(h^l)$$



# Error en el bias

$$\frac{\partial E}{\partial b^l} = \delta^l$$

# Error en bias

$$\frac{\partial E}{\partial w_{j,k}^l} = a_k^{l-1} \delta_j^l$$

# Back propagation

La aplicación de estas ecuaciones

**Como grafo**

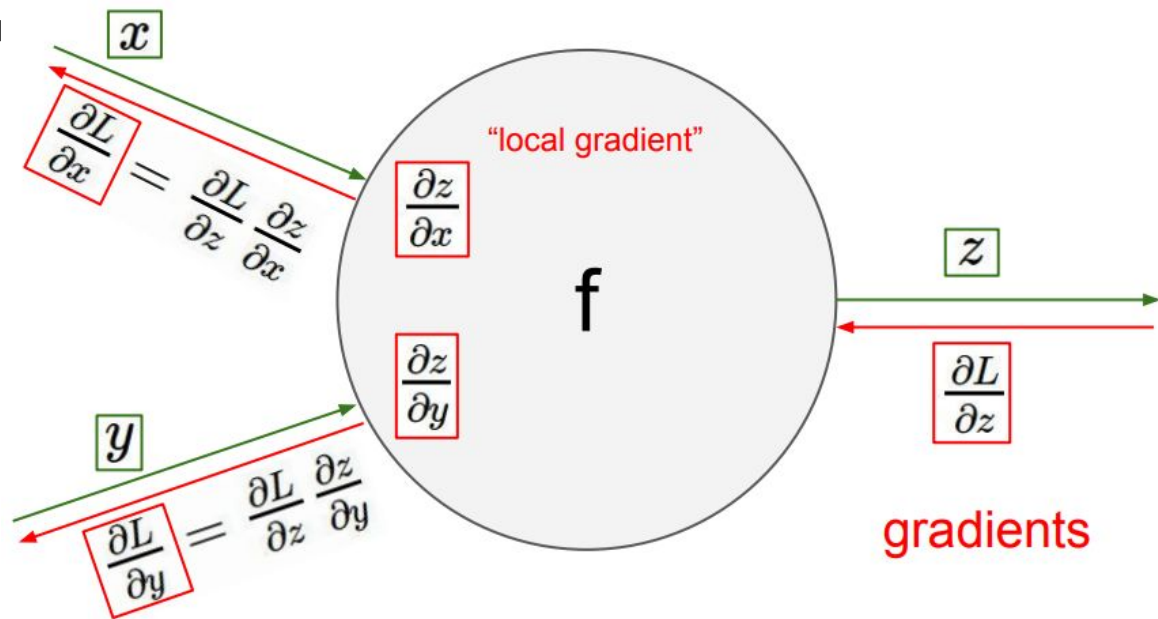
# Renombrando algunas cosas

$$C=E \text{ o } L=E$$

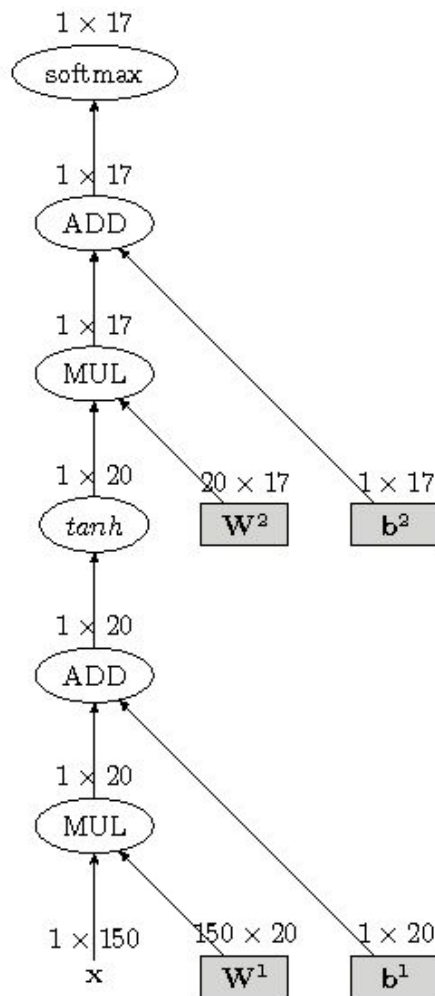
$$z=a'$$

*x, y la contribución de un peso*

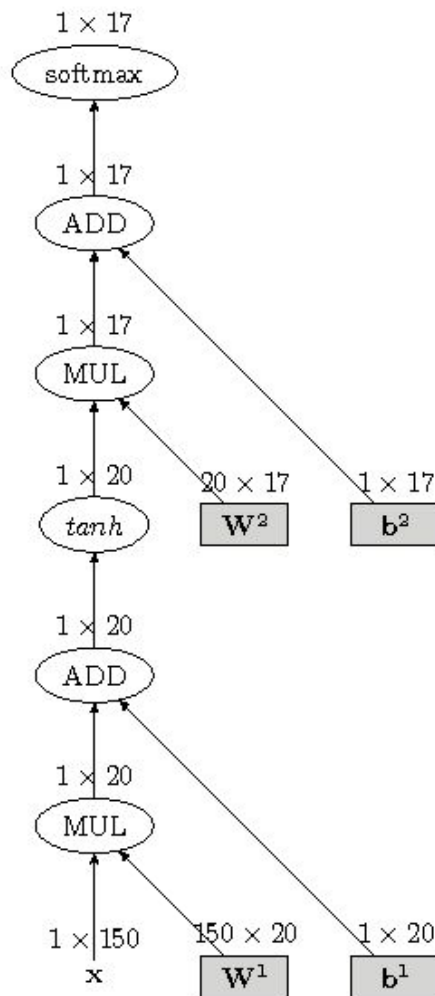
# Como funciona



# MLP como grafo de cômputo



# Forward

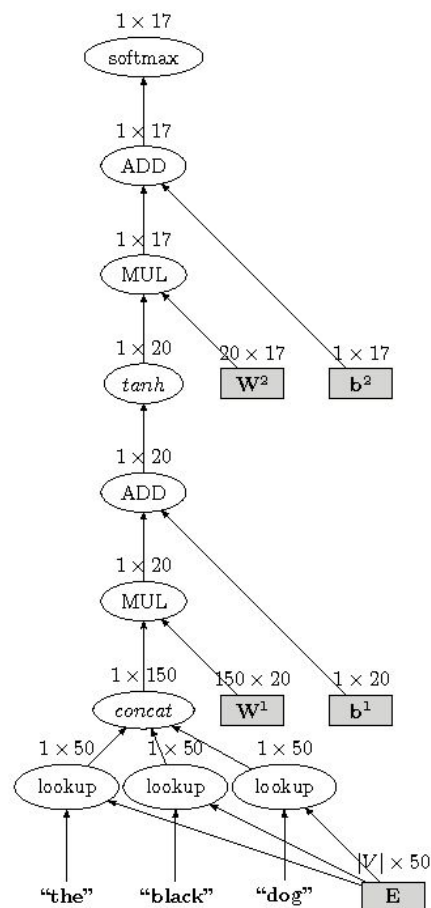


hijos

Padres



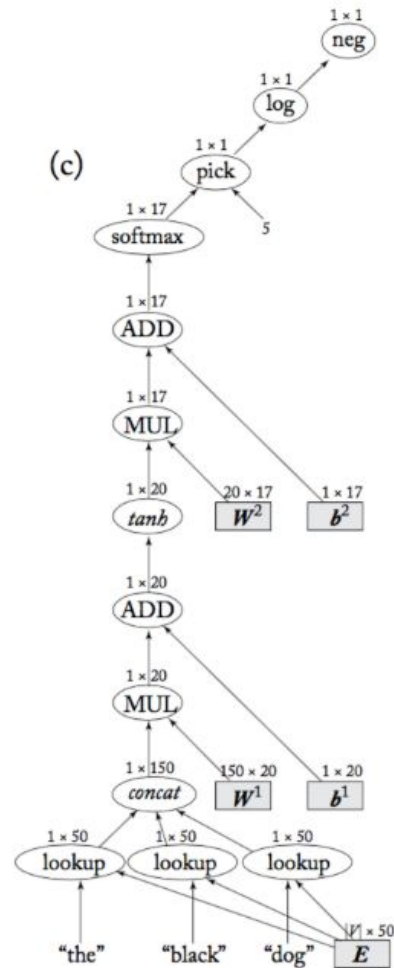
# Forward con x



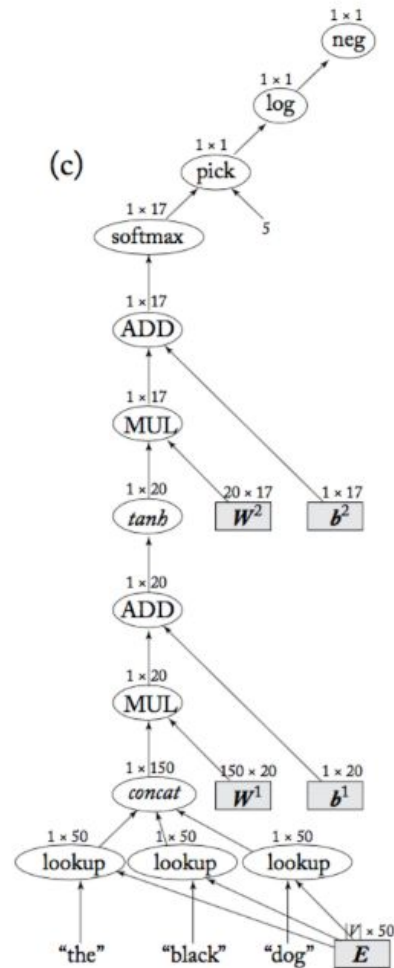
Padres

hijos

# Forward con lost

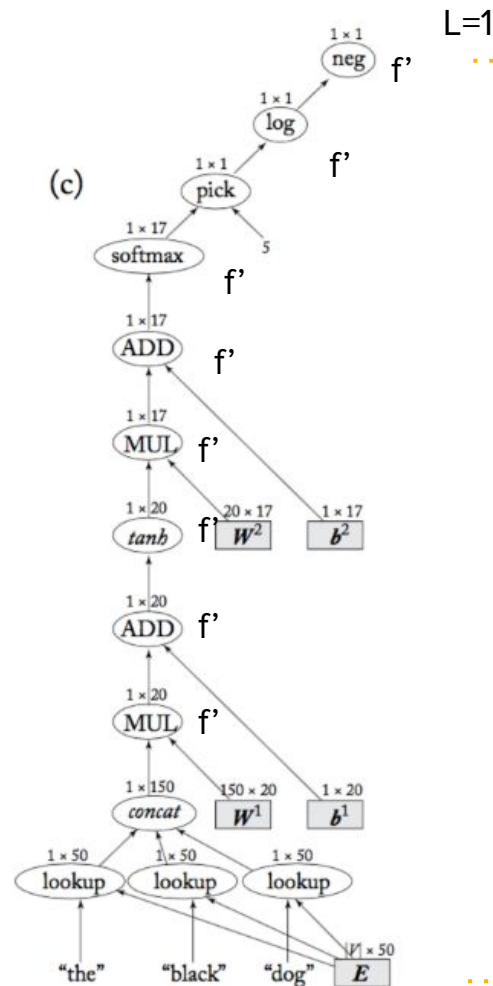


# Forward con lost



# Backward

$$\frac{\partial L}{\partial z} \frac{\partial z}{\partial w}$$



# Otros mecanismos de optimización

# Otros métodos de optimizacion

Vainilla

$$\hat{\theta} = \theta - k \nabla J(\theta, x_i, y_i)$$

# Otros métodos de optimizacion

Minibatch

$$\hat{\theta} = \theta - k \nabla J(\theta, x[i : i + n], y[i : i + n])$$

# Otros métodos de optimizacion

Momentum

$$v_t = \gamma v_{t-1} + k \nabla J(\theta)$$

$$\hat{\theta} = \theta - v_t$$



# Otros métodos de optimización

Nesterov  $\varepsilon$

$$v_t = \gamma v_{t-1} + k \nabla J(\theta - \gamma v_{t-1})$$

$$\hat{\theta} = \theta - v_t$$

# Otros métodos de optimización

Adagrad

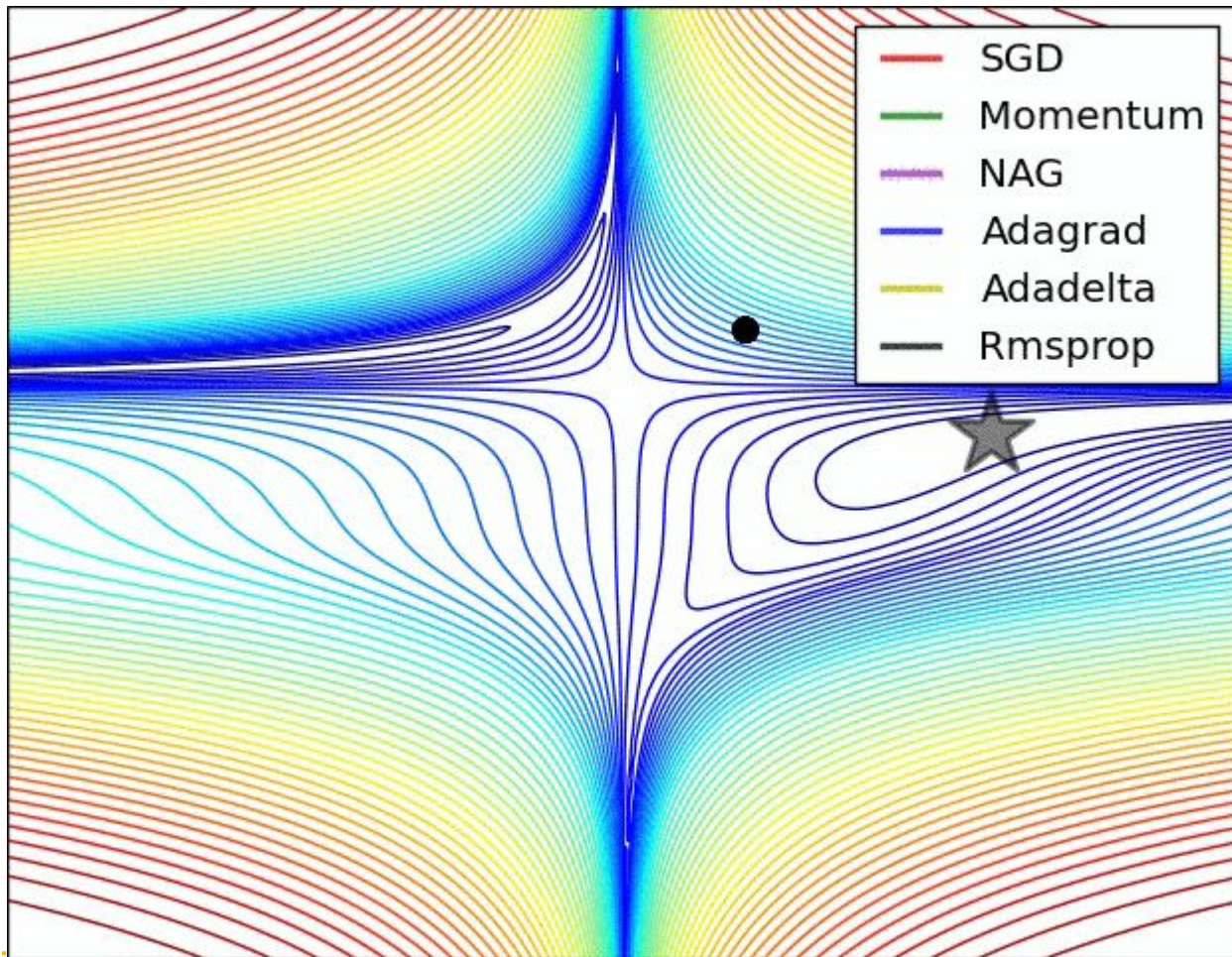
$$\theta_{t+1} = \theta_t - \frac{k}{\sqrt{G_t + \epsilon}} \odot g_t$$

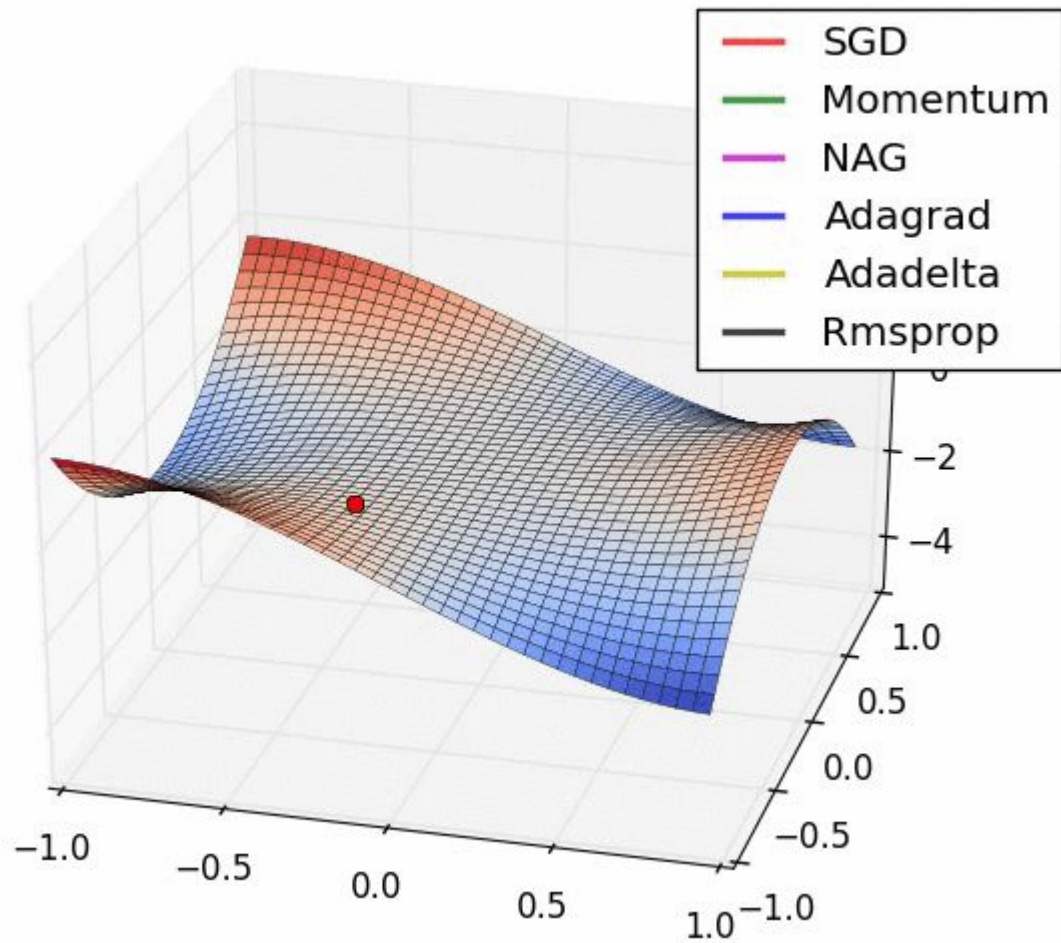
# Otros métodos de optimización

RSMPprop

$$E[g]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$$

$$\theta_{t+1} = \theta_t - \frac{k}{\sqrt{E[g^2]_t + \epsilon}} g_t$$





# Manipular

<https://losslandscape.com/explorer>

“

*Tres garantías*

# Tres garantías

1. Las redes neuronales son aproximadores universales para funciones continuas: **Funciones matemáticas**
2. Las redes neuronales recurrentes son equivalentes a máquinas de Turing: **Algoritmos**
3. El algoritmo de backpropagation va encontrar una configuración de la red que imita el comportamiento de los datos

"Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". Neural Networks. 6 (6): 861-867.

Siegelmann, H. T., & Sothg, E. D. (1992, July). On the computational power of neural nets. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 440-449).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. nature, 323(6088), 533-536.



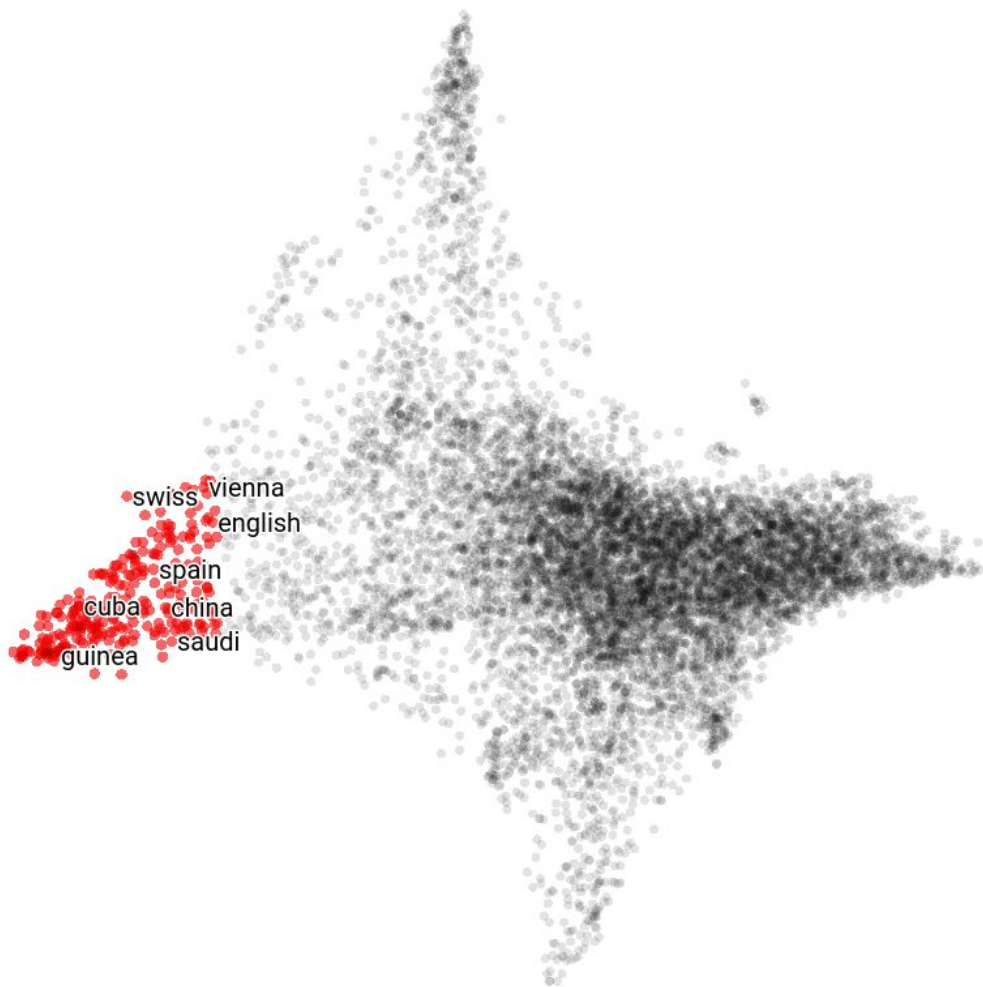
“

*Redes recorrentes*

# ***Todo es un vector***

7a metáfora





$$\sin\left(\omega t + \frac{\pi}{2}\right)$$

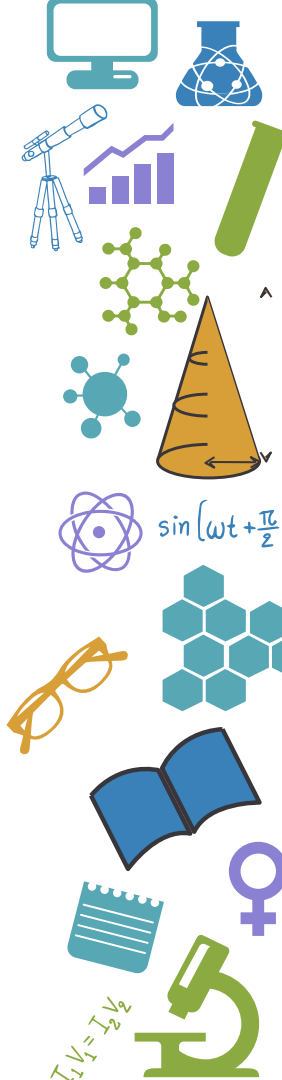


$$I_V = I_2^{1/2}$$

# Ejemplo

✓ *perros*: [0.678641579074071, 0.4726267197424602,  
0.6699618494116154, 0.6134727267440421, 0.7884295249570772,  
0.3097356074817419, 0.4108834549045387, 0.9150959196458144,  
0.04209167364154953, 0.714175609065853, 0.019089355554306242,  
0.2342050958784716, 0.9540342968538216, 0.7729056436977116,  
0.785877036975804, 0.9821768722685218, 0.7293579326768811,  
0.2014321301972668, 0.349530343157317, 0.4917367171701048,  
0.026563934882441687, 0.8914408953170919, 0.9767419151172713,  
0.4722904439150507, 0.6541608174208771, 0.8889932058447243,  
0.49292883438919444, 0.8985198063228953, 0.544002070266999,  
0.7153381641713361, 0.9459279525823148, 0.4443207032842629]

*parecen perros contra gatos* ?



$$[ ] + [ ] + [ ] + [ ] = [ ] = [s]$$

*parecen    perros    contra    gatos*

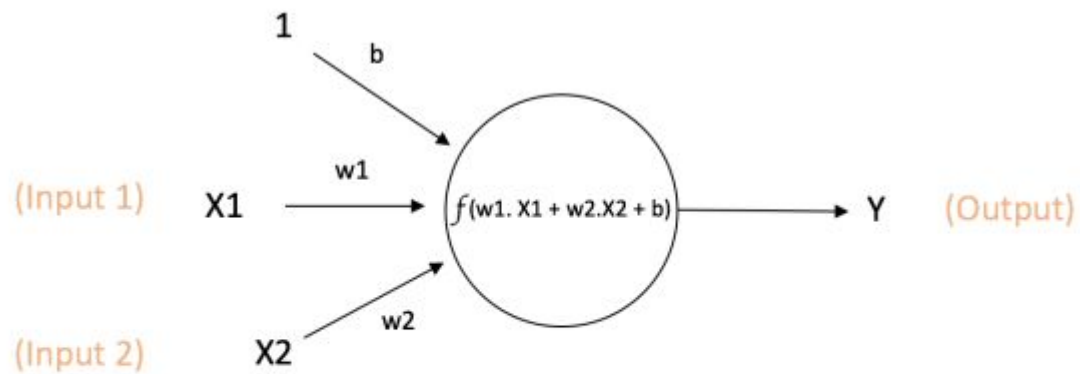


# Aprendizaje profundo

Popularizado por

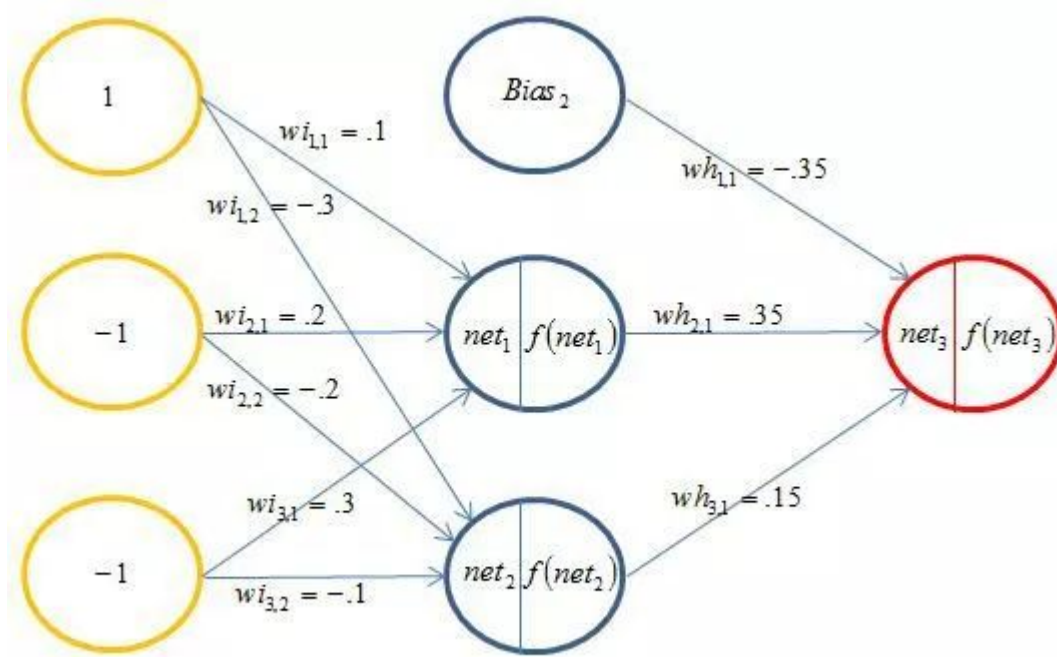
- ▶ Conducen carros
- ▶ Ganan en el juego de Go
- ▶ Reconocen objetos en imagenes
- ▶ Reconocen voz
- ▶ Traducen automáticamente





$$\text{Output of neuron} = Y = f(w1.X1 + w2.X2 + b)$$









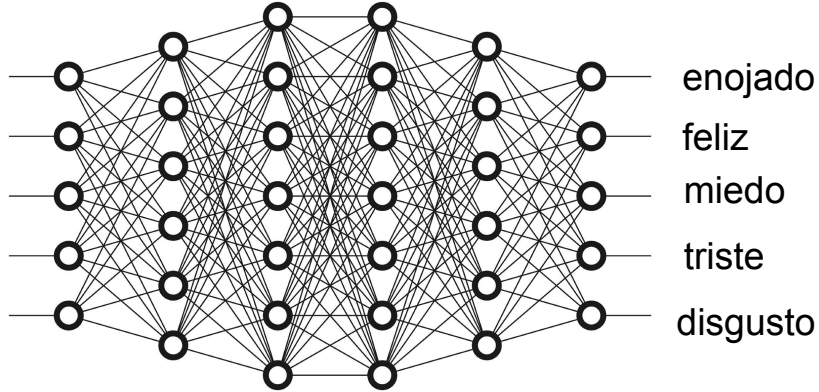
# Idea



Poner el vector resumen enfrente de la red neuronal

Ejemplo, clasificar sentimientos de un *tuit*

$[S]$



# Excelente para clasificación

- Sentimientos
- Ironía
- Detectar entidades
- ... muchas más

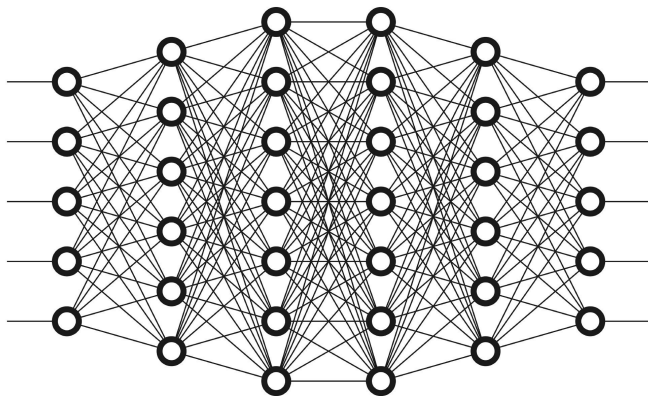


Pero teníamos una crítica de la suma

$[parecen]$



$[0]$



$[s'']$

$\updownarrow$ : concatenación de dos vectores

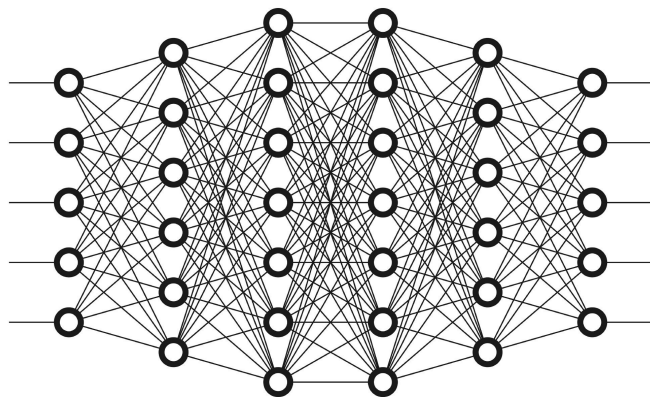


Pero teníamos una crítica de la suma

$[perros]$



$[s'']$



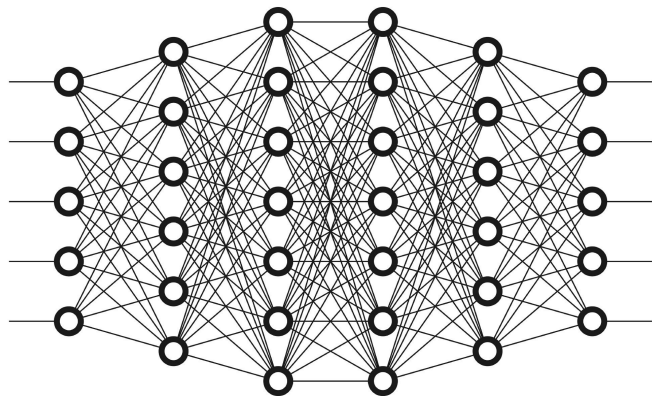
$[s''']$



$[contra]$



$[S''']$



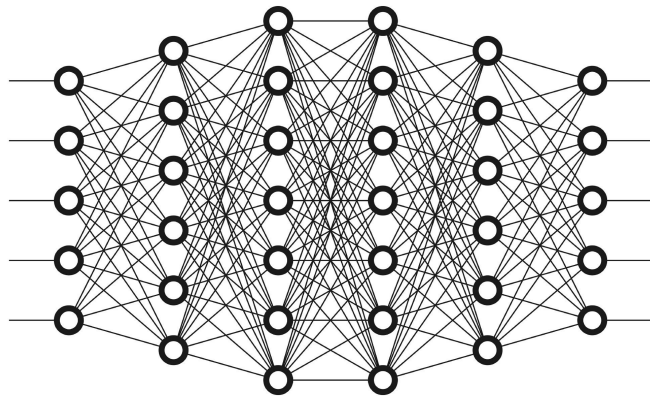
$[S''']$



$[gatos]$



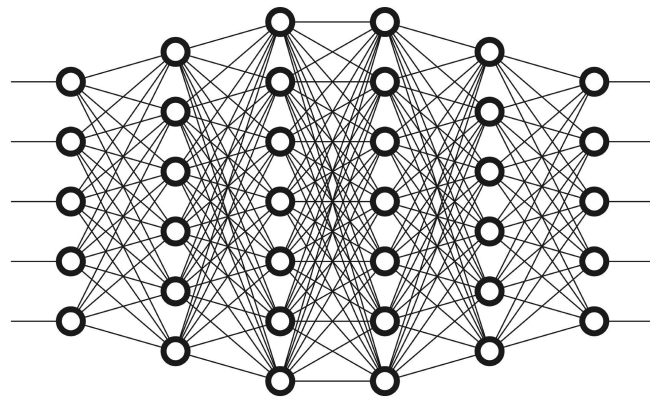
$[s''']$



$[s]$



$[S]$



enojado

feliz

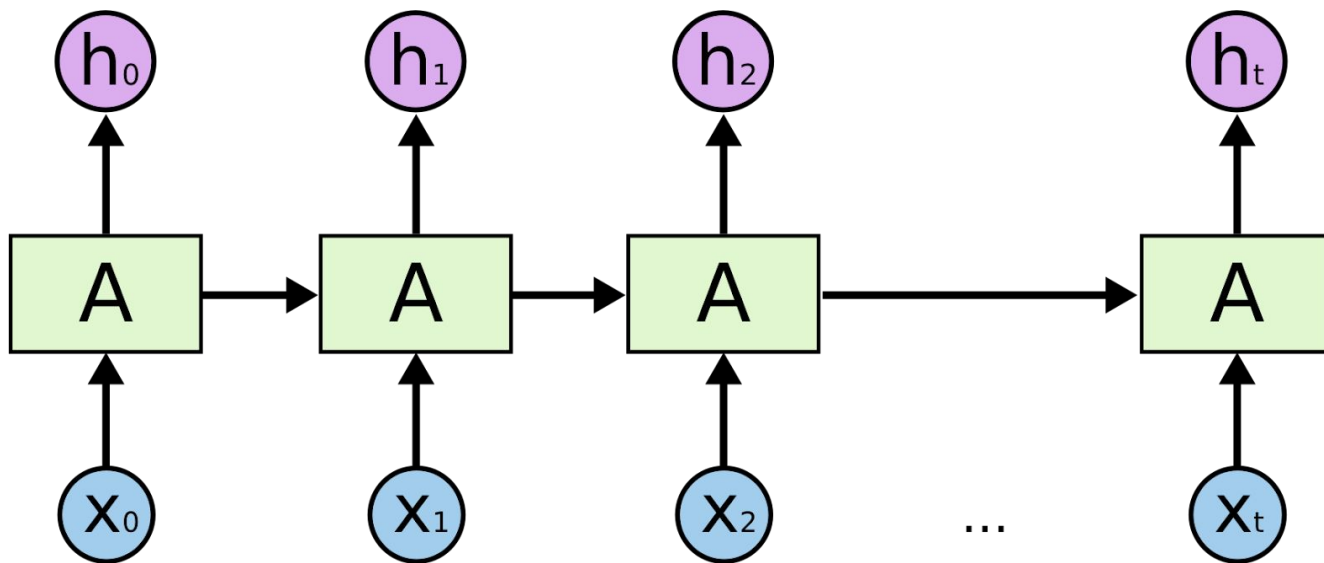
miedo

triste

disgusto



# Modelo recorrente



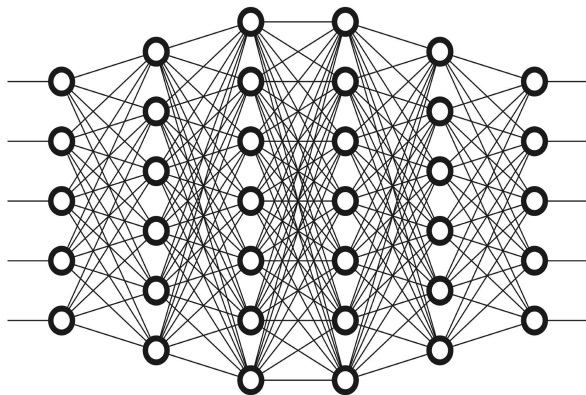


Ese nueva salida

$[contra]$



$[S''']$



$[gatos]$

$[S''']$

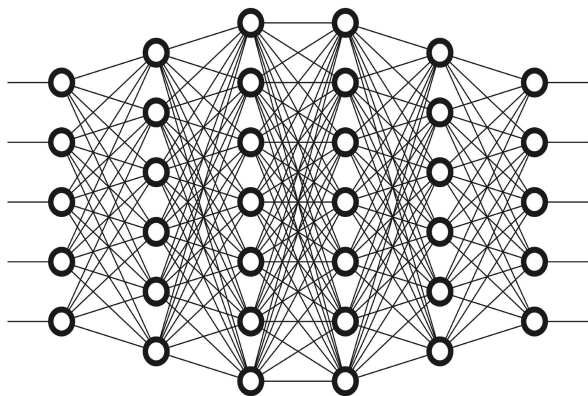


Puede predecir una palabra

$[CIMAT]$



$[0]$



$[Guanajuato]$

$[S''']$



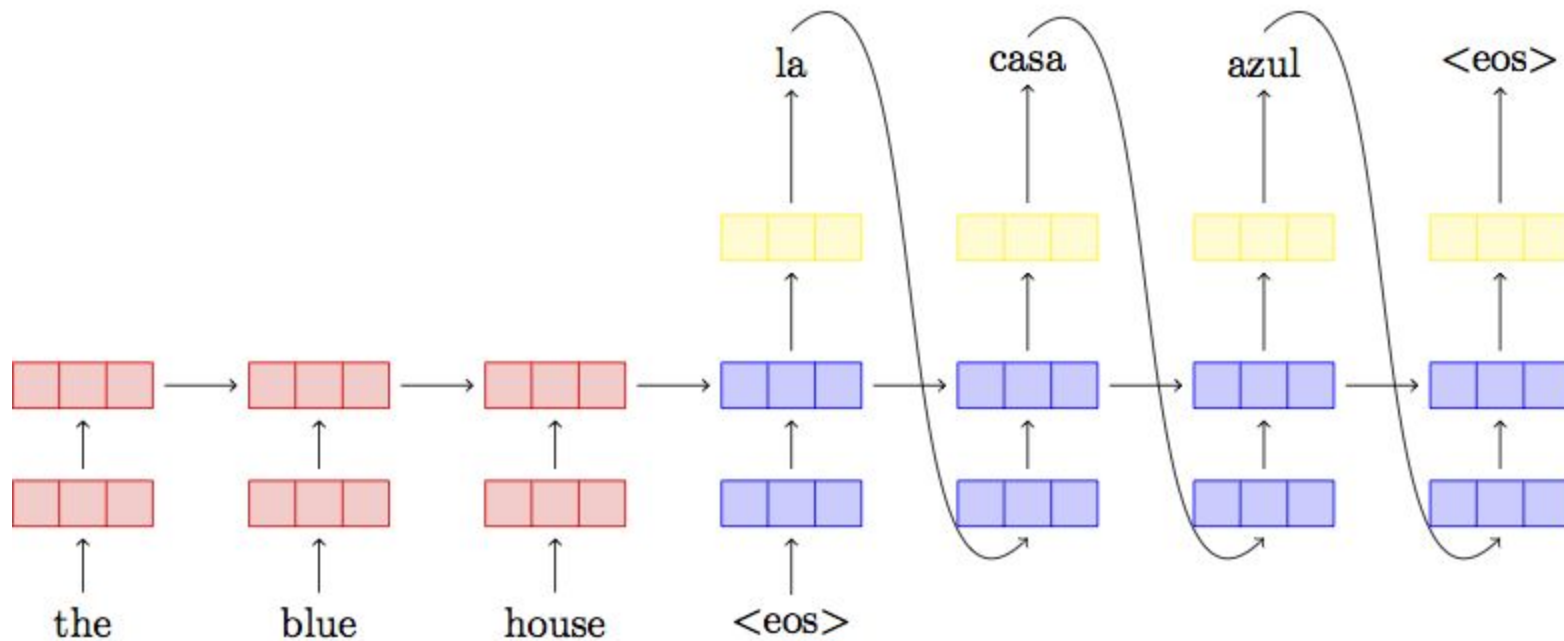


# Importante



- Tenemos algo que hace un resumen dada palabras anteriores
- Tenemos algo que predice la palabra siguiente

# Modelo seq2seq





# seq2seq



Seq2seq se usa tareas de re-escritura

- Traducción automática
- Segmentación morfológica
- Normalización de textos
- Sistemas conversacionales
- Corrección gramatical



# seq2seq



Estado del arte con Seq2seq

- Se usa con otros componentes: atención
- Arquitecturas transformers
- Herramientas múltiples: OpenNMT

“

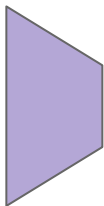
***En la práctica***

*1 acceder datos*

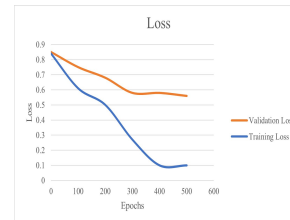
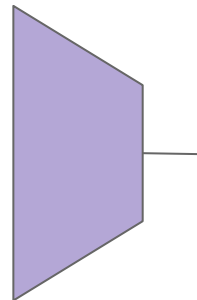
**Datos == Dataset**

**Desarrollo**  
**Prueba**

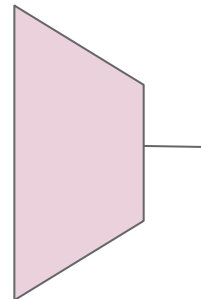
*2 diseñar red*



*3 entrenar/monitear*



*4 evaluar*



Acc  
Pres  
Rec  
F1



# iGracias!

Ivan Vladimir Meza Ruiz, IIMAS/UNAM

[ivanvladimir@turing.iimas.unam.mx](mailto:ivanvladimir@turing.iimas.unam.mx)

[@ivanvladimir](#)