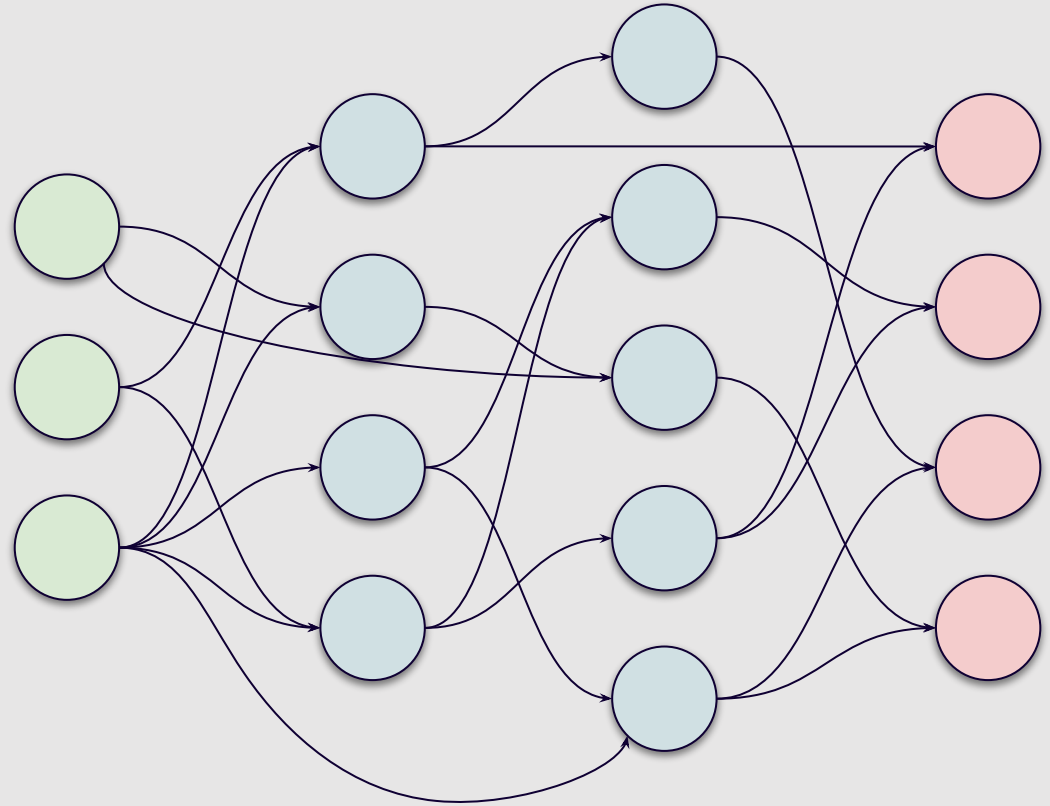


# Redes Neuronales (Modernas)

# Redes Neuronales Modernas

- Red/Grafo de computación
  - Dirigido
- Nodos
  - Entrada 
  - Interno/Oculto 
  - Salida 
- Modular / Composicional
- Diferenciable
- Optimizable



# Red de ejemplo y cálculo **forward**

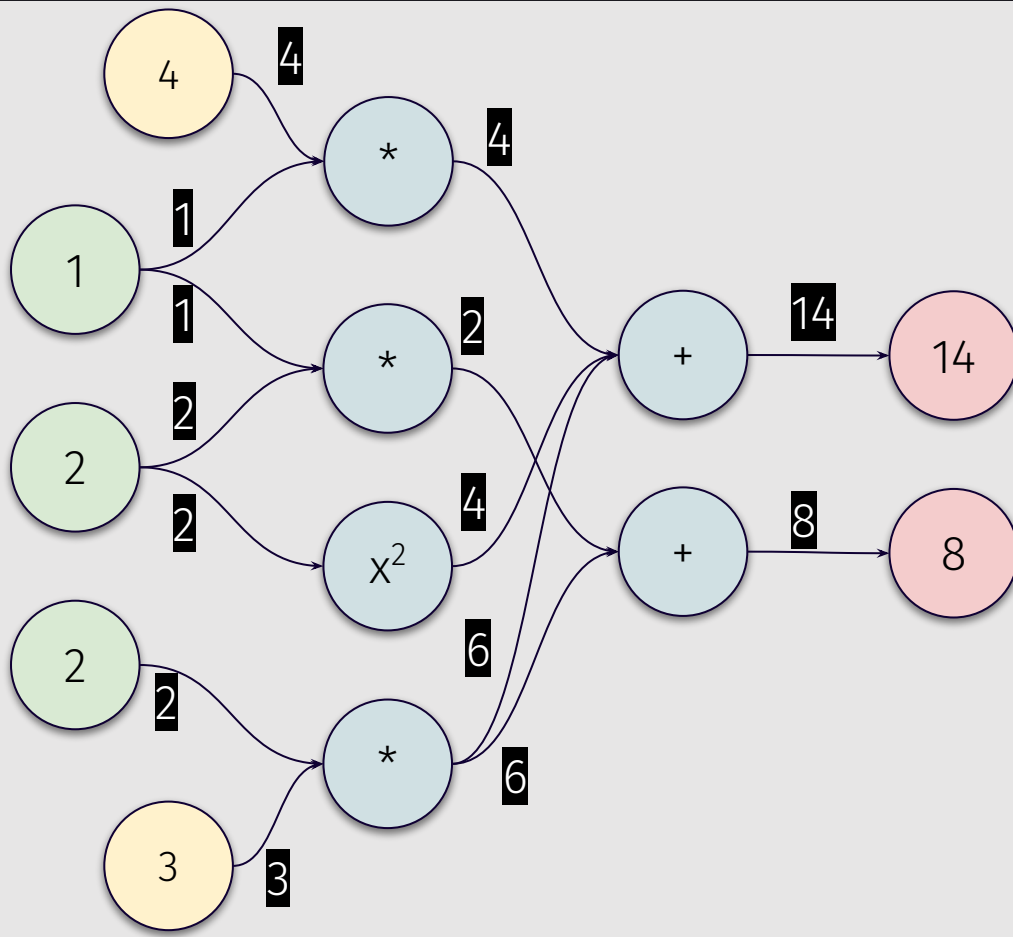
- $f(x_1, x_2, x_3) =$   
 $(4 * x_2 + x_2^2 + 3 * x_3,$   
 $x_1 * x_2 + 3 * x_3)$   
 $= (y_1, y_2)$

- Parámetros



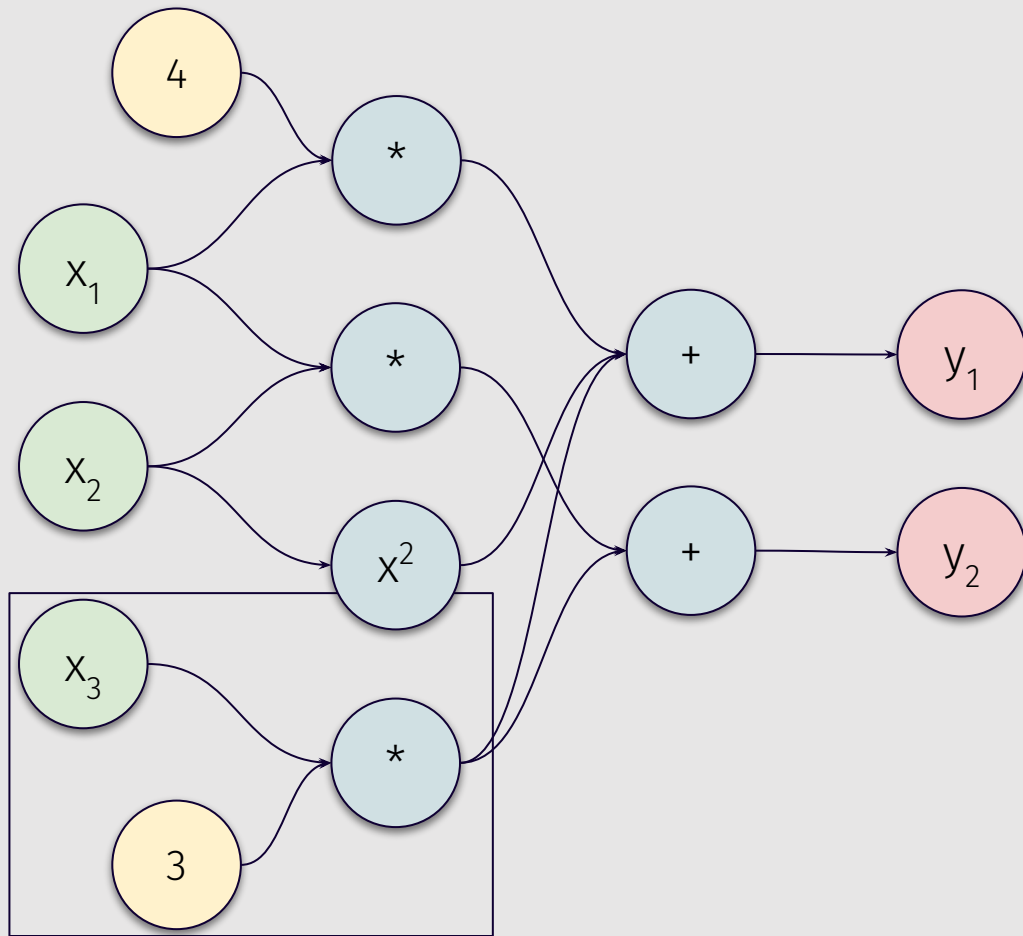
- Ejemplo

- $x_1 = 1$
- $x_2 = 2$
- $x_3 = 2$



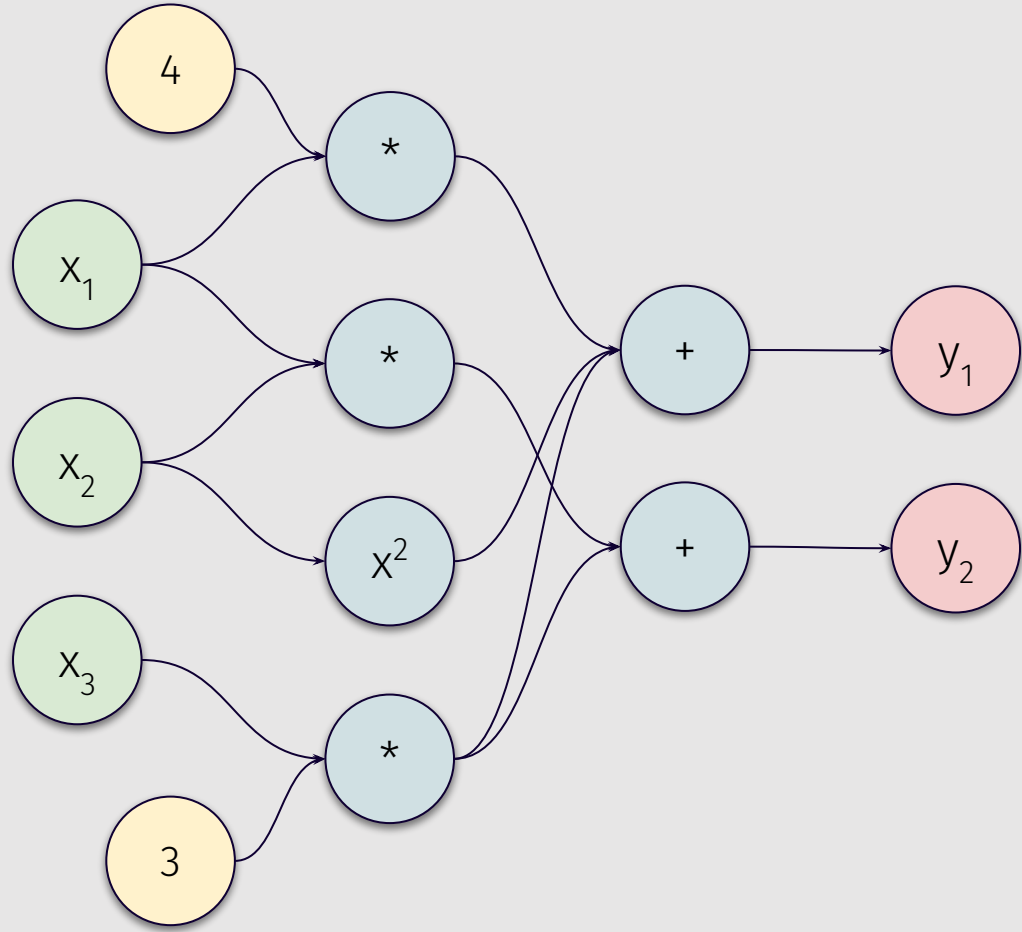
# Red de computación **modular/composicional**

- $f(x_1, x_2, x_3) =$   
 $(4 * x_2 + x_2^2 + 3 * x_3,$   
 $x_1 * x_2 + 3 * x_3)$   
 $= (y_1, y_2)$
- Modular
  - $3 * x_3$  se calcula 1 vez
  - *Subred*



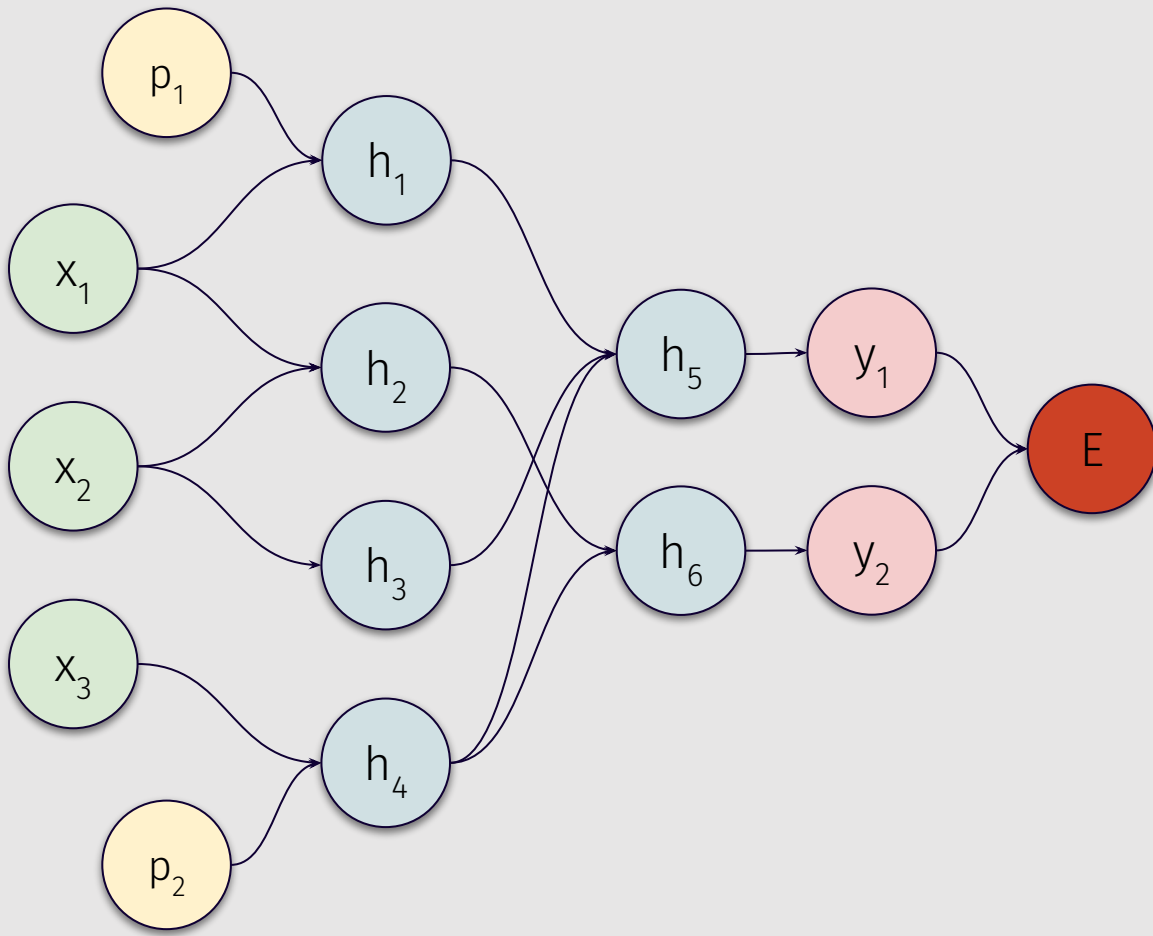
# Diferenciable

- Funciones utilizadas
  - $x^2$
  - $+$
  - $*$
- Todas derivables!
- Composición de funciones es derivable
- Red = Composición de funciones derivables



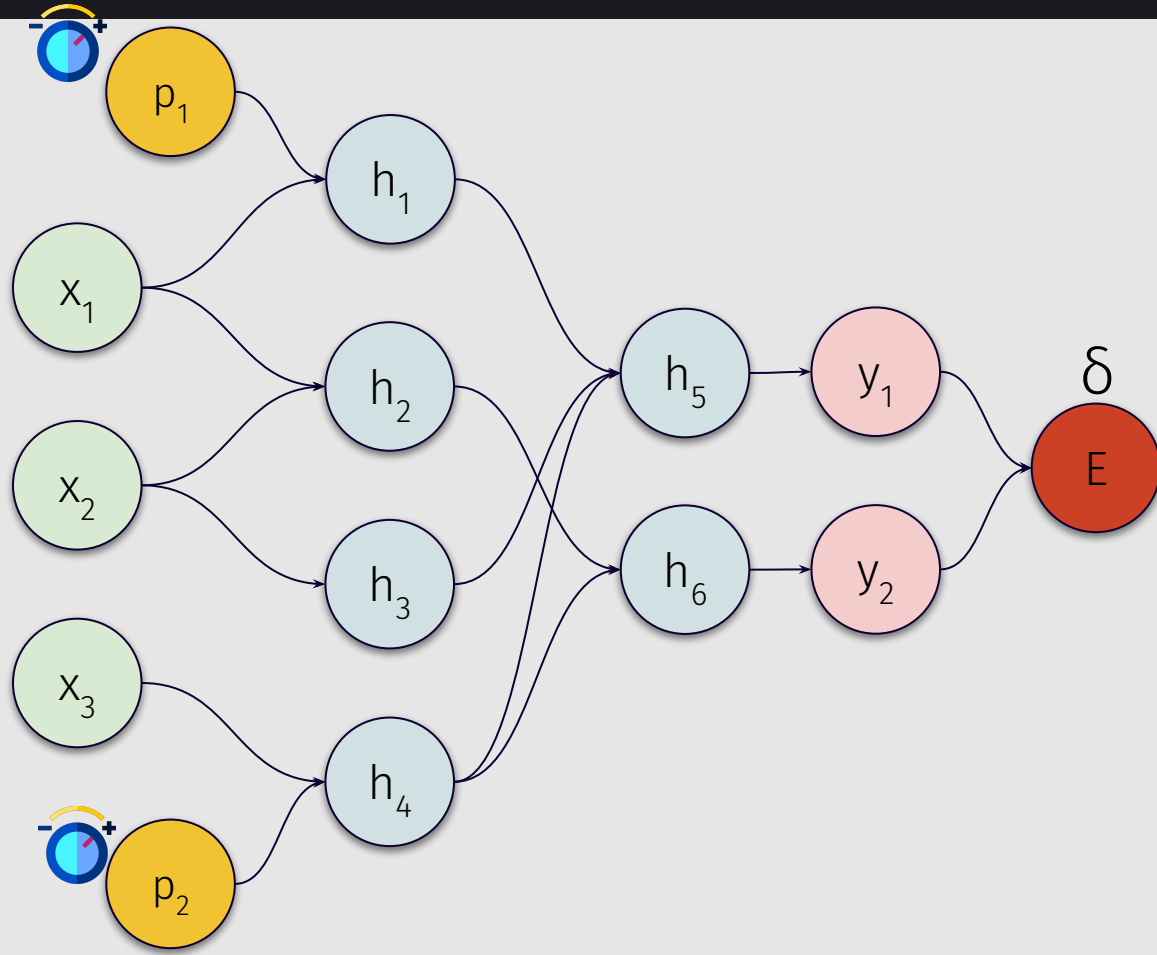
# Error

- Función de error  $E$ 
  - Derivable
  - Siempre escalar
  - **Depende de la tarea**
- Posible  $\delta E / \delta x$ 
  - Para cualquier nodo  $x$ , ej:  $\delta E / \delta p_2$



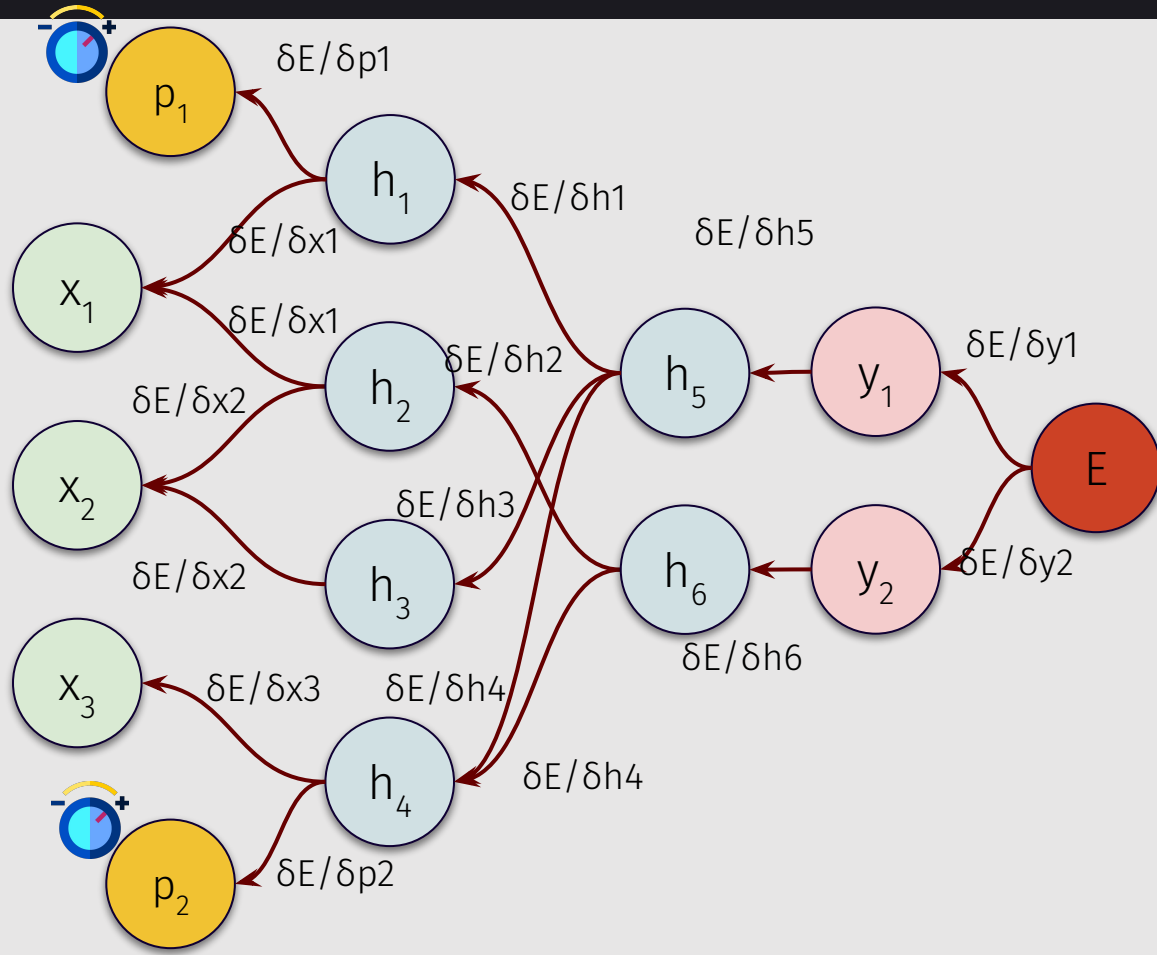
# Optimizable

- Función de error + derivada
  - Señal error
- Optimización
  - Minimizar  $E$
  - Cambiando nodos parámetro
    - $p_1, p_2$  o más



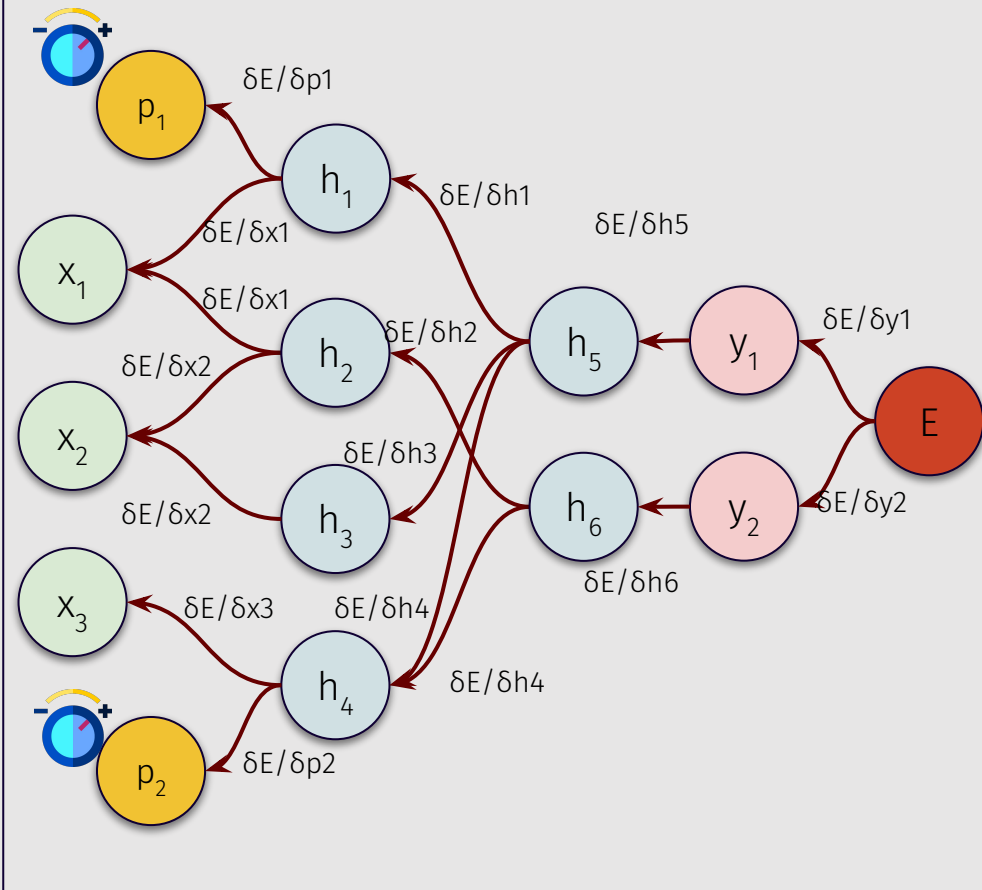
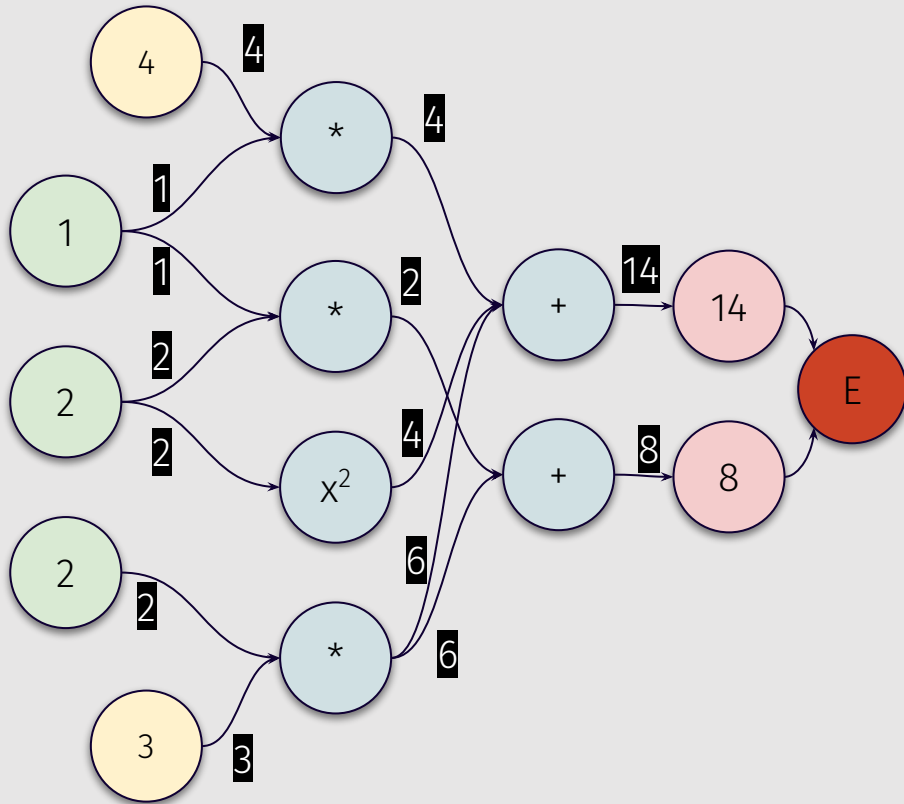
# Cálculo **backward**

- Calcula derivadas de error para todos los nodos
- $\delta E / \delta x$ 
  - Derivada del error respecto a un nodo
  - $\delta E / \delta p \rightarrow$  **más interesantes**
- Cálculo distribuido



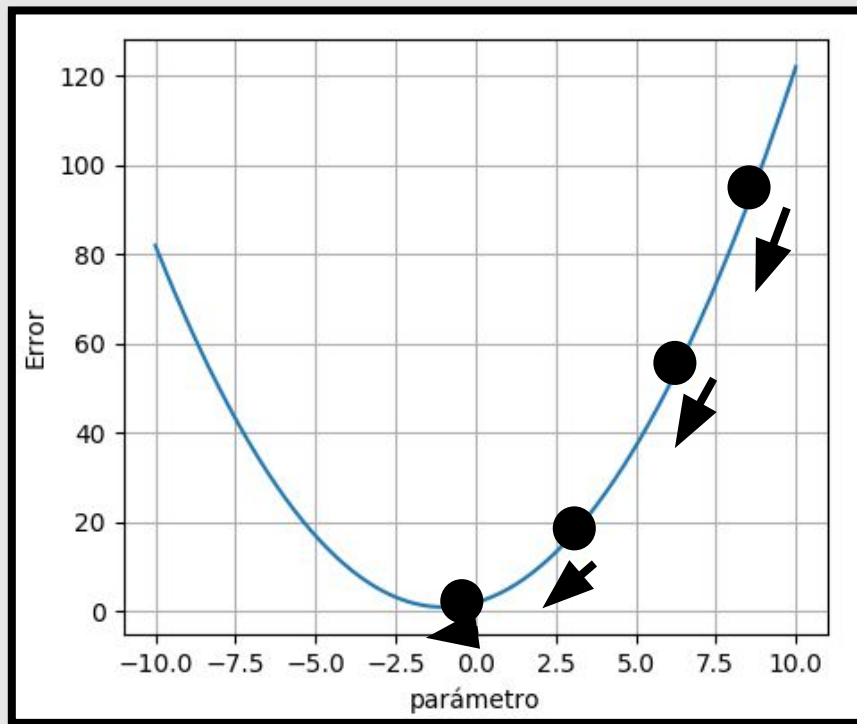


# Backpropagation = Forward + Backward



# Descenso de gradiente

- Backpropagation
  - Calcula derivadas
  - No optimiza
- **Descenso de gradiente**
  - Usa backprop
    - Flecha = derivada
  - Simple y efectivo
  - Para redes con 10 o  $10^{10}$  parámetros



# Entrenamiento y uso de redes

Ejemplos  
(etiquetados)

Estudio	Edad	Promedio	N1	N2
2	24	4	7	7.2
5	22	3	4.5	5.2
7	25	4	6.3	6

Red

Parámetros  
Iniciales  
(aleatorios)

Algoritmo de  
Aprendizaje

- Descenso de Gradiente

Error a optimizar

- Error cuadrático
- Entropía
- Otros

Modelo

Parámetros  
(entrenados)

Ejemplos  
nuevos

Estudio	Edad	Promedio
10	19	4
11	20	3

Predicciones

P1	P2
7	7.2
4.5	5.2

# Tipos de datos

- Escalares (Tensor 0D)

- 3.54

- Vectores (Tensor 1D)

- [3.12, 6.5, 8.912]

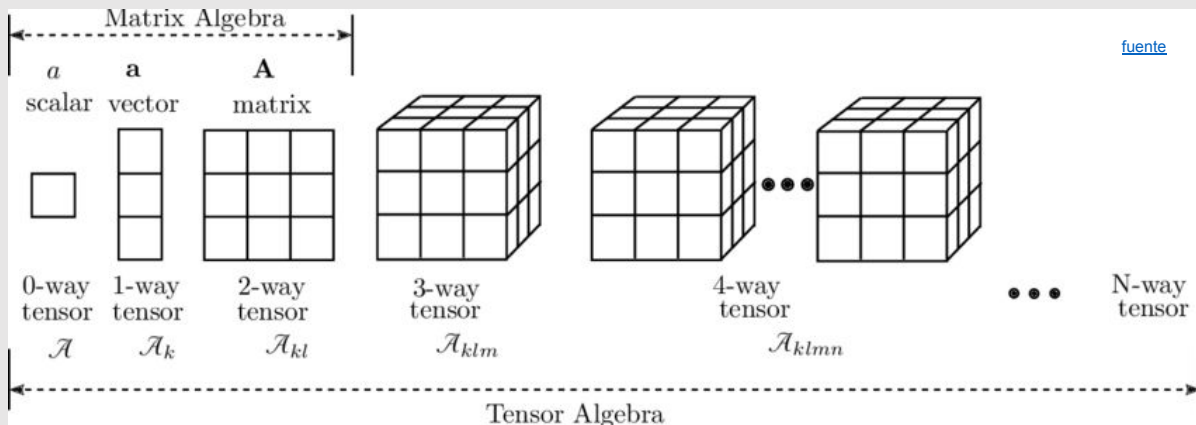
- Matrices (Tensor 2D)

- [4.3, 2.3, -5.3, 4.9]

- [12, -3.5, -8, -6]

- [-4.1, 3.2, 12, 0]

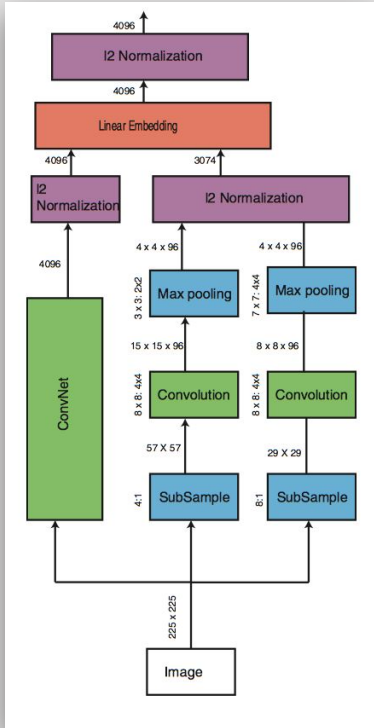
- Tensor 3D



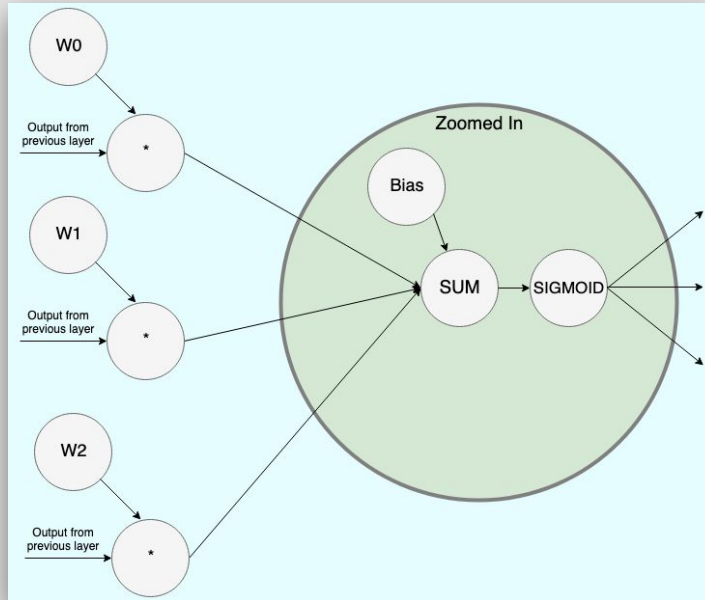
- [4.3, 2.3, -5.3, 4.9]
- [12, -3.5, -8, -6]
- [4.3, 2.3, -5.3, 4.9]
- [12, -3.5, -8, -6]
- [4.3, 2.3, -5.3, 4.9]
- [12, -3.5, -8, -6]

# Redes Neuronales Generales - Escalas

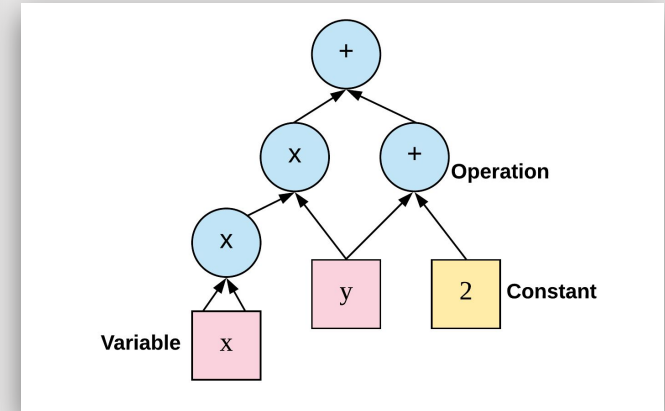
## Capas (nivel alto)



## Tensores (nivel medio)

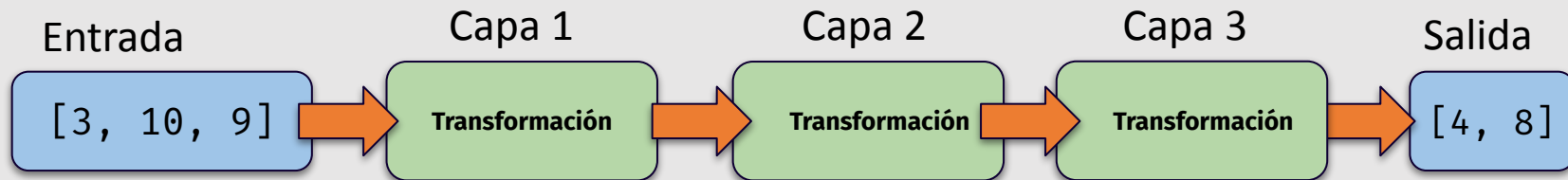


## Números (nivel bajo)



# Redes Neuronales a nivel de Capas

- Grafo de **transformaciones** derivables
  - Máximo
  - Promedio
  - Regresión Lineal
  - Regresión Logística
  - etc
- Transformación = Capa

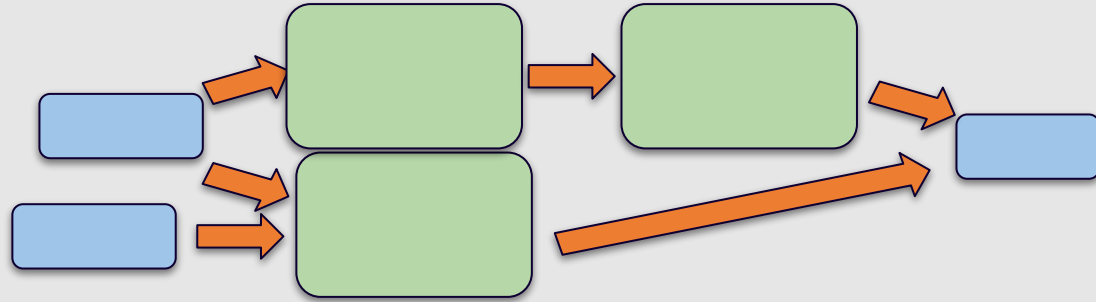


# Topologías/arquitecturas posibles de red

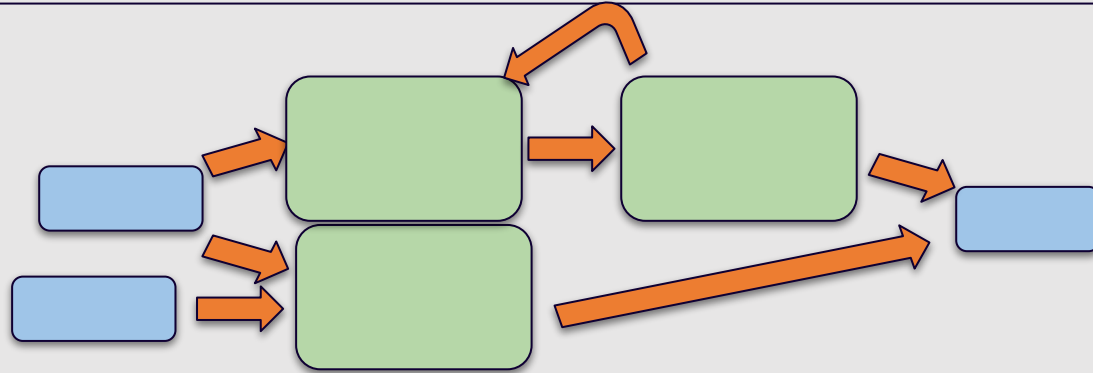
**Lineal  
(Secuencial)**



Acíclica

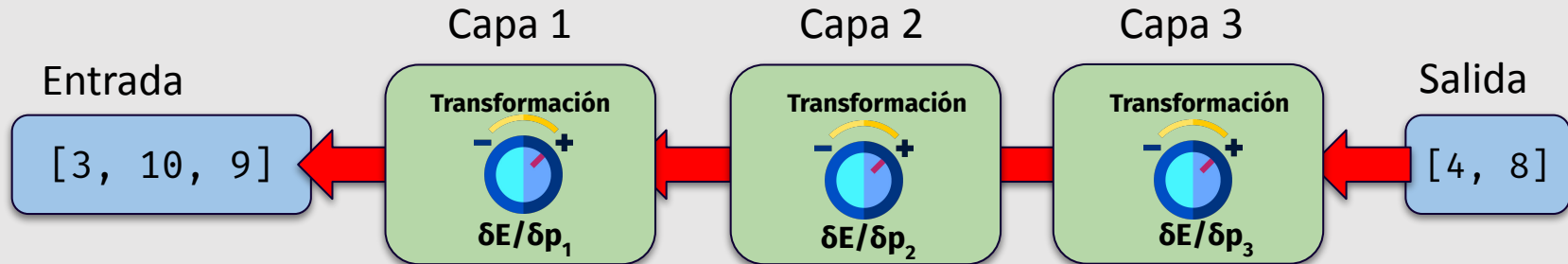
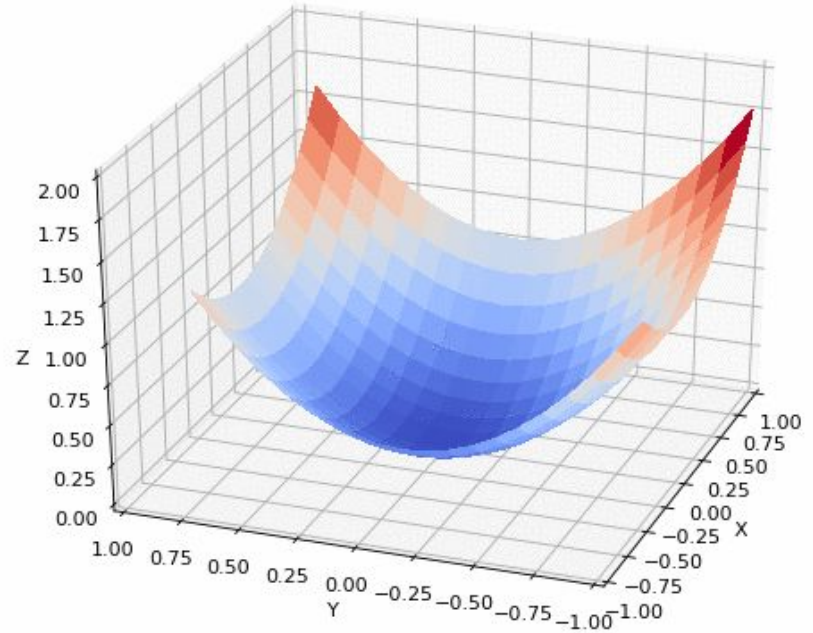


Cíclica  
(recurrente)



# Redes Neuronales

- Transformaciones *aprendidas*
  - Parámetros *ajustables*
- Aprendizaje
  - **Descenso de gradiente**
- Gradientes
  - **Algoritmo Backpropagation**





# Capas básicas

**Regresión Lineal**



**Regresión Logística**



**Convolutacional**



**Funciones de Activación**

**Batch Normalization**



**Dropout**

**Bloques Residuales**



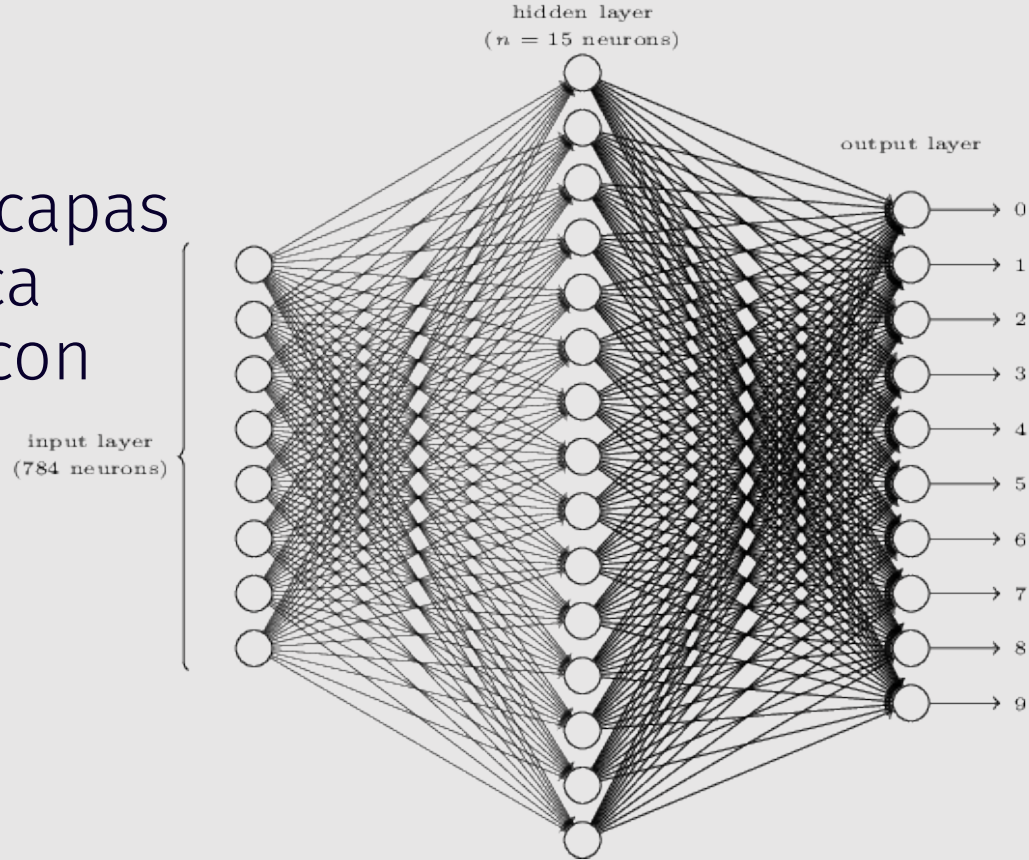
**Bloques Inception**



**Cientos más!**

# Nomenclatura tradicional

- Nomenclatura inversa
  - Nodos = valores
  - Aristas = funciones/capas
- Interpretación biológica
  - **Nodos** = Neuronas, con valor de *activación*
  - **Aristas** = Sinapsis *transformadoras*
- Se usan ambas nomenclaturas
  - Aprender ambas y a distinguir



# Redes profundas

- Una red de 2 capas puede resolver cualquier problema
  - En teoría (Cybenko 1989)
  - Resultado no constructivo
    - Pesos existe, pero ¿cuáles?
- En la práctica, poco eficiente:
  - Gran número de neuronas
  - Difícil entrenamiento
    - Es más fácil con **más capas**
- **Deep** Learning

