Tema 2.2

Optimización II: Backpropagation

Deep Learning

Máster Oficial en Ingeniería Informática

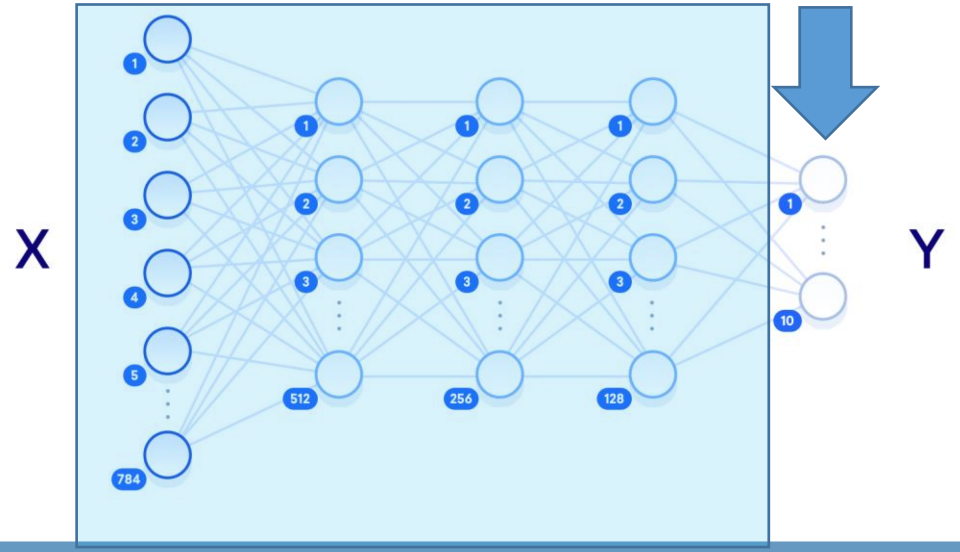
Universidad de Sevilla

Contenido

- Descenso por gradiente (cont.)
- Propagación del gradiente
- Grafo computacional

Descenso por gradiente

- Las actualizaciones se pueden aplicar directamente para actualizar pesos en modelos como regresión lineal y logística (perceptrón con función de activación sigmoide).
- ¿Cómo proceder con red neuronal multicapa?

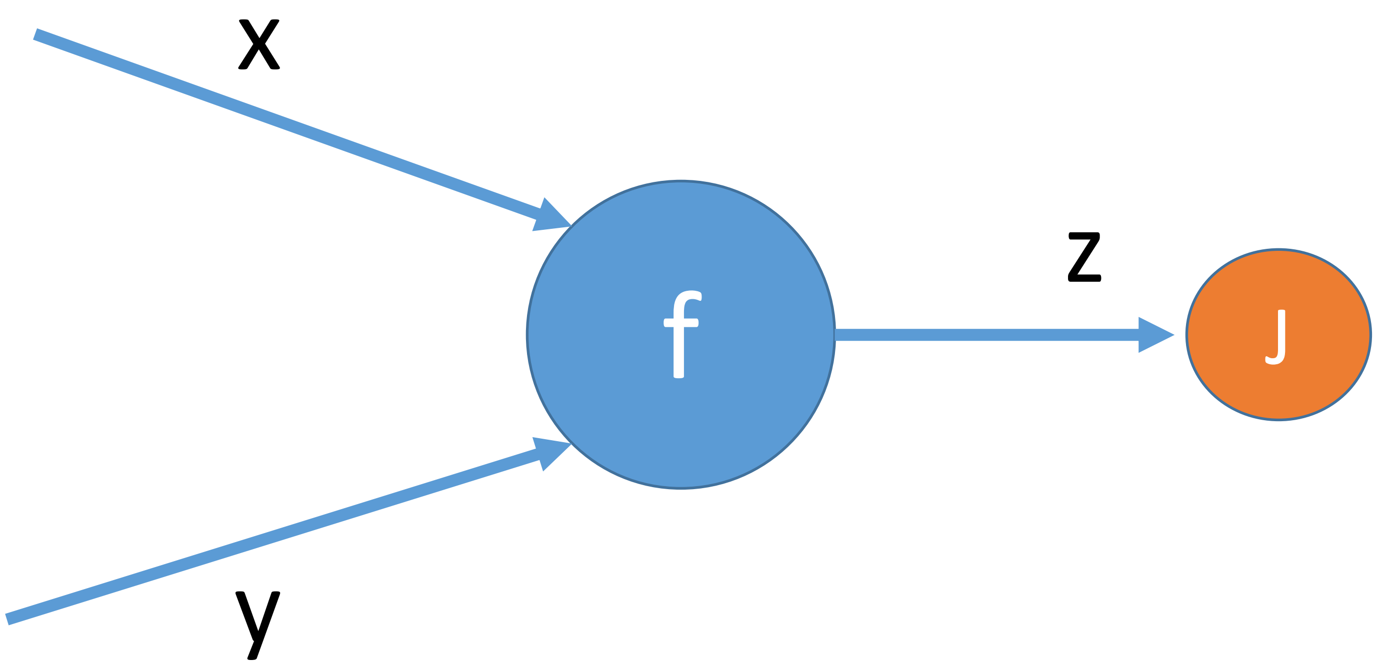


Descenso por gradiente

- Las actualizaciones se pueden aplicar directamente para actualizar pesos en modelos como regresión lineal y logística (perceptrón con función de activación sigmoide).
- ¿Cómo proceder con red neuronal multicapa?
- En la capa de salida, usar la actualización de pesos con el gradiente sobre la función de coste.
- **Problema**: en las capas ocultas desconocemos los valores esperados.
- **Solución**: propagar los gradientes, algoritmo de retropropagación (**backpropagation)**

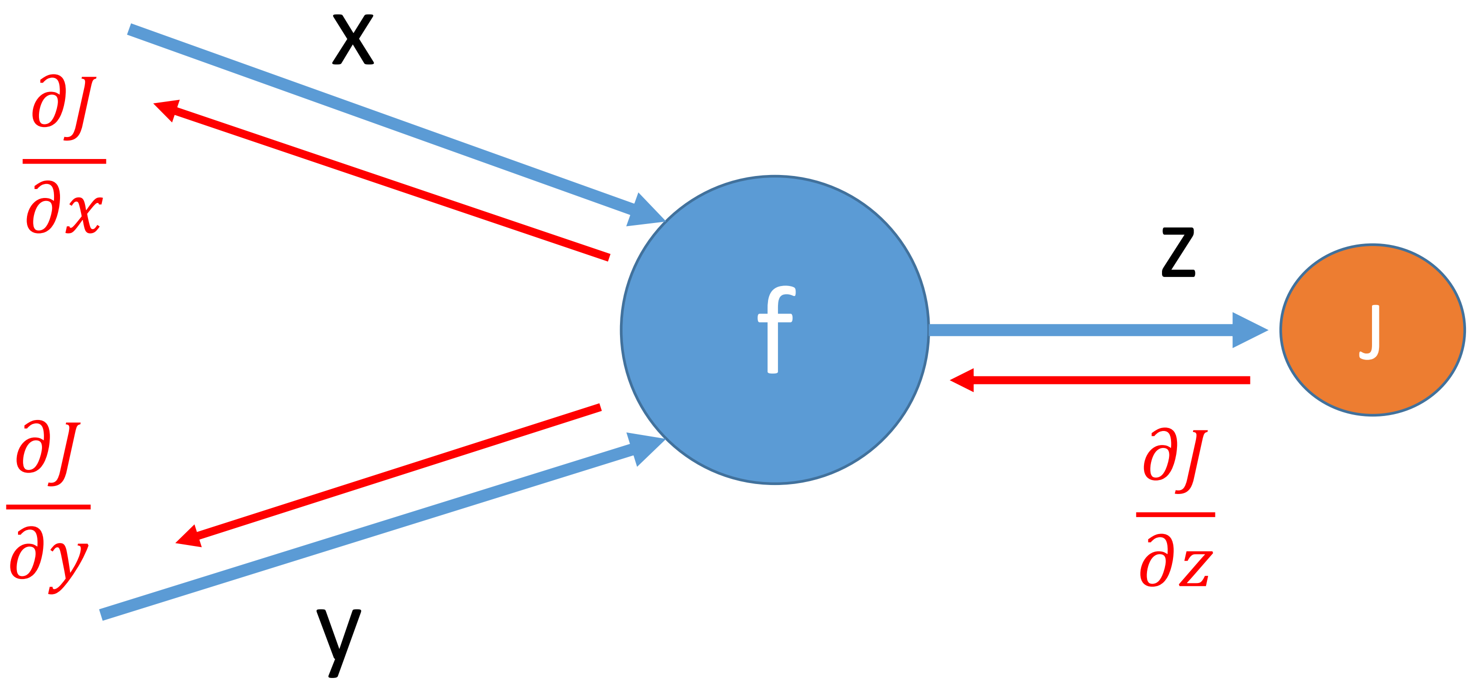
Propagación de gradiente

- Propagación hacia adelante (forward):



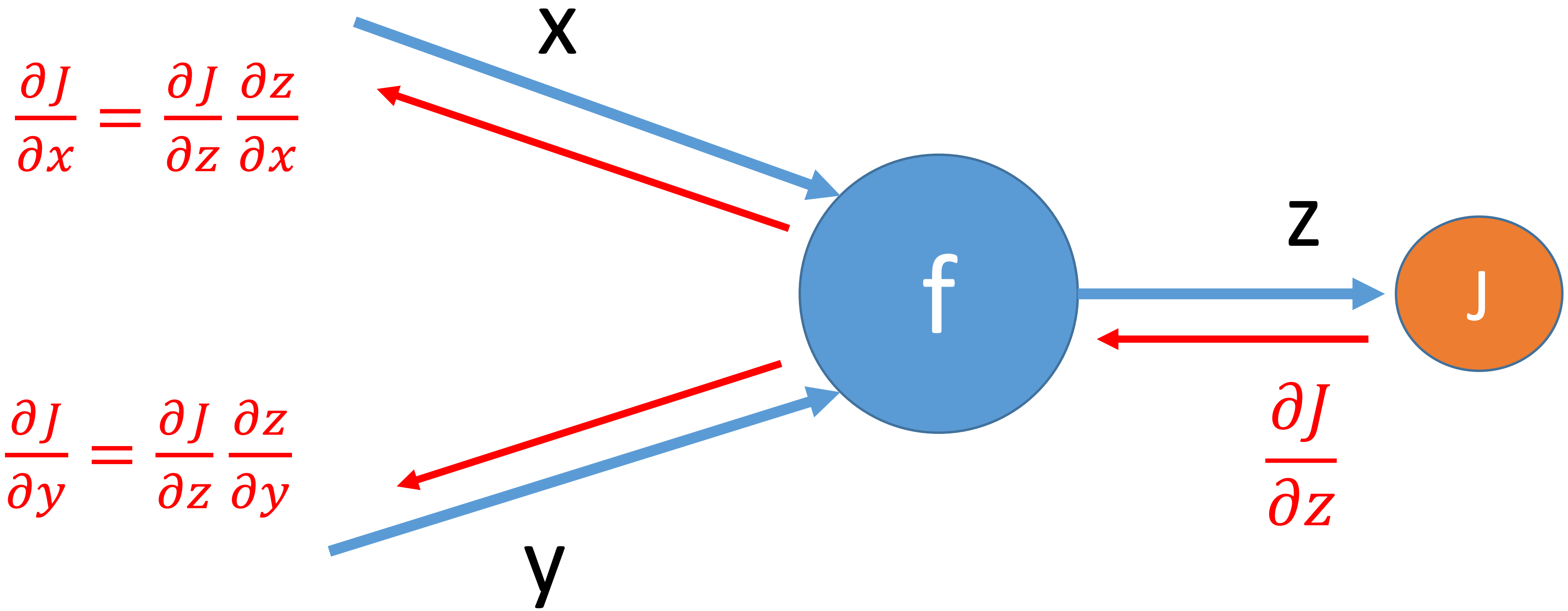
Propagación de gradiente

- Propagación hacia atrás (backward):



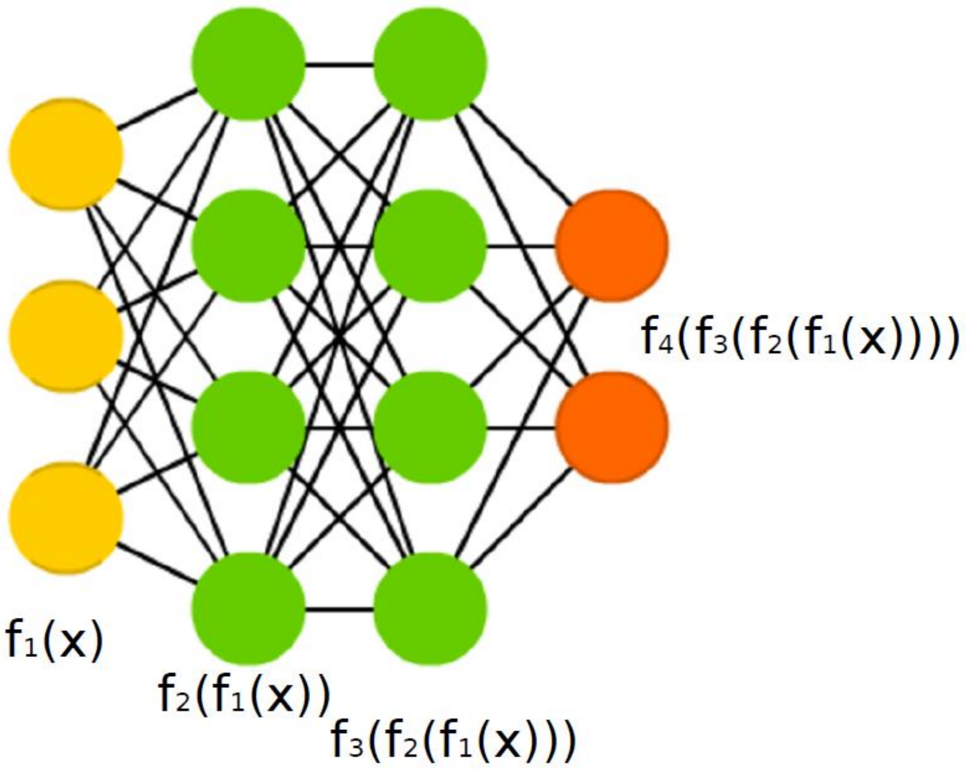
Propagación de gradiente

- Regla de la cadena:
  - Clave del algoritmo de backpropagation.



Propagación de gradiente

f(x) =



$$\frac{\partial f}{\partial x} = \frac{\partial f_4}{\partial x}$$

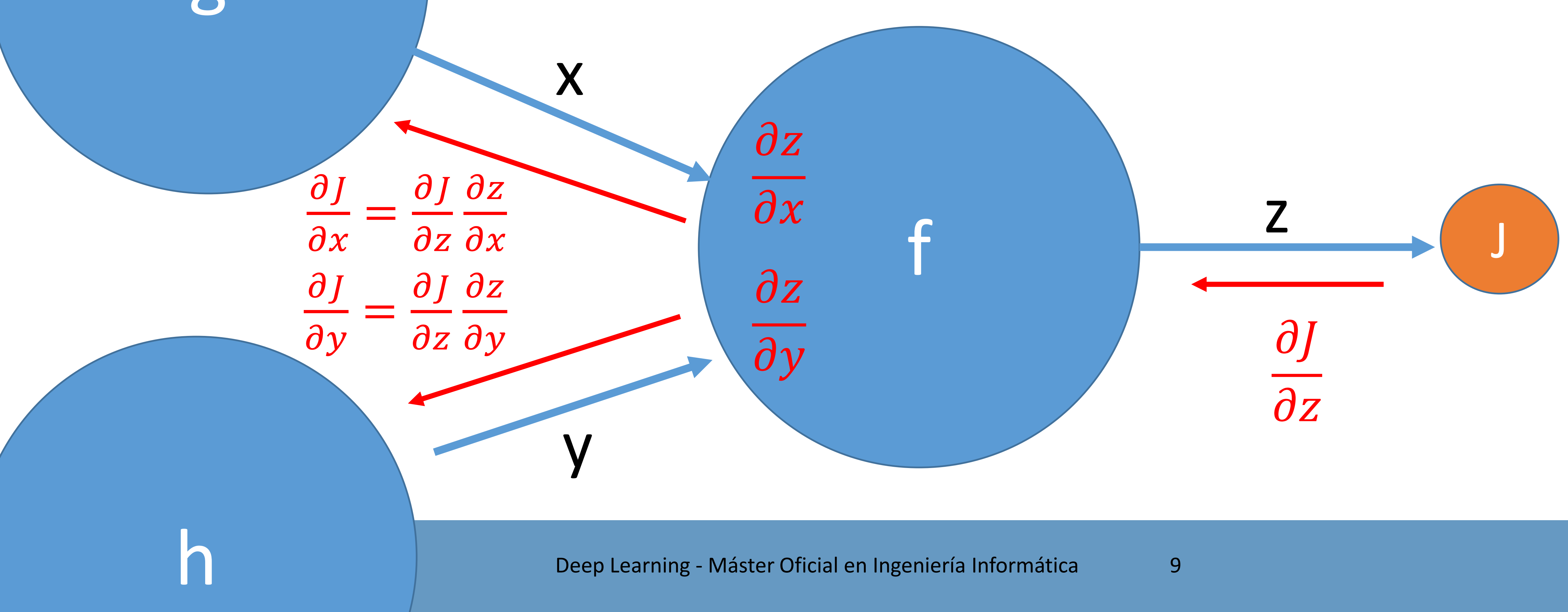
$$\frac{\partial f_4}{\partial x} = \frac{\partial f_4}{\partial f_3}\frac{\partial f_3}{\partial x}$$

$$\frac{\partial f_4}{\partial x} = \frac{\partial f_4}{\partial f_3}\frac{\partial f_3}{\partial f_2}\frac{\partial f_2}{\partial x}$$

$$\frac{\partial f_4}{\partial x} = \frac{\partial f_4}{\partial f_3}\frac{\partial f_3}{\partial f_2}\frac{\partial f_2}{\partial f_1}\frac{\partial f_1}{\partial x}$$

Propagación de gradiente

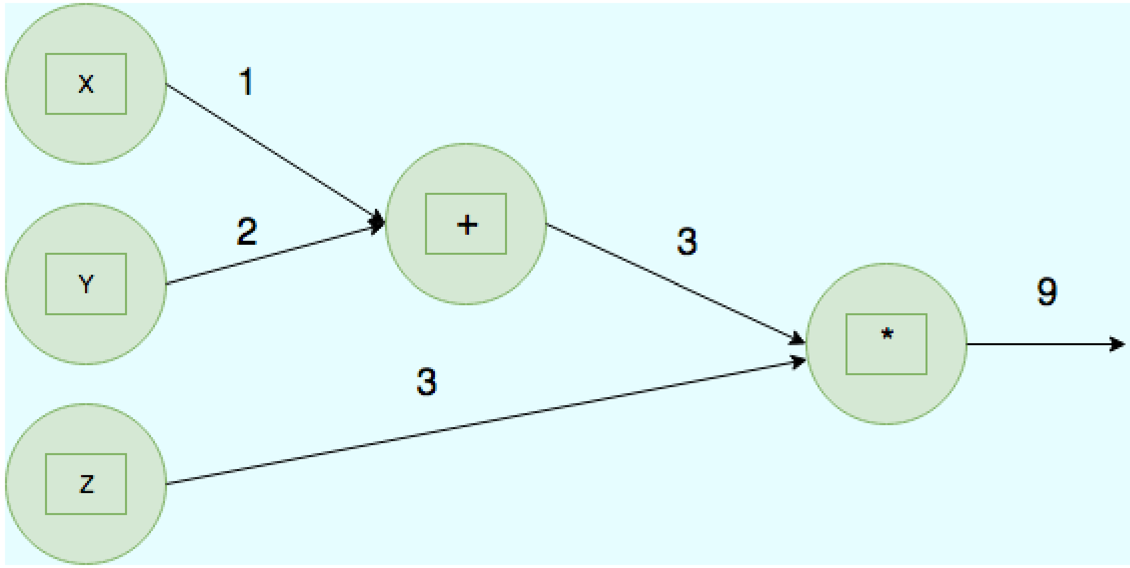
- Cálculo de gradientes locales mientras forward propagation.



Grafo computacional

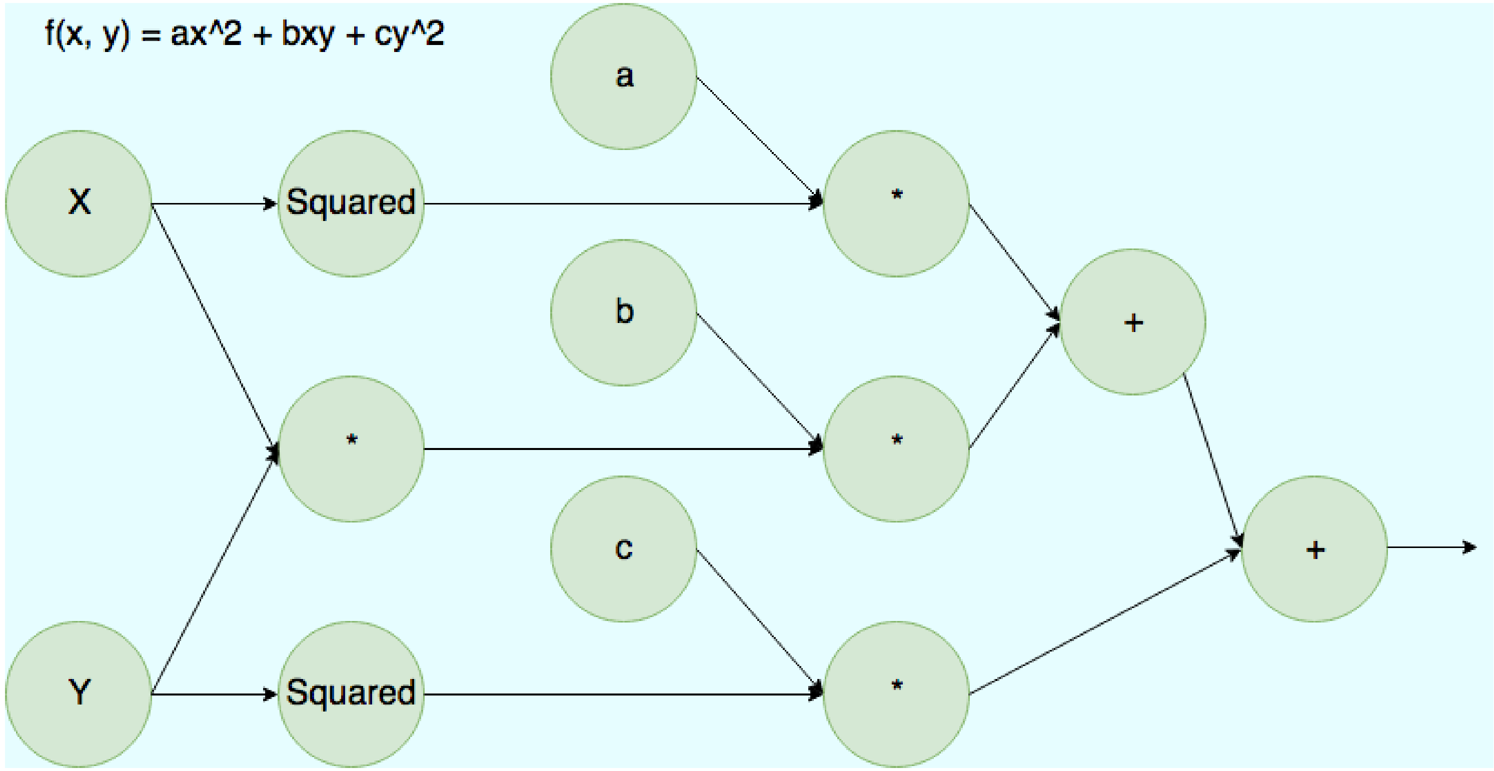
- Para calcular cómo se propaga el gradiente por una red neuronal, es muy útil representarlo como un **grafo computacional**
- Definir una red a base de las operaciones (nodos) y valores (aristas) que se van enviando.
- Por ejemplo:

$$f(x, y, z) = (x + y) * z$$



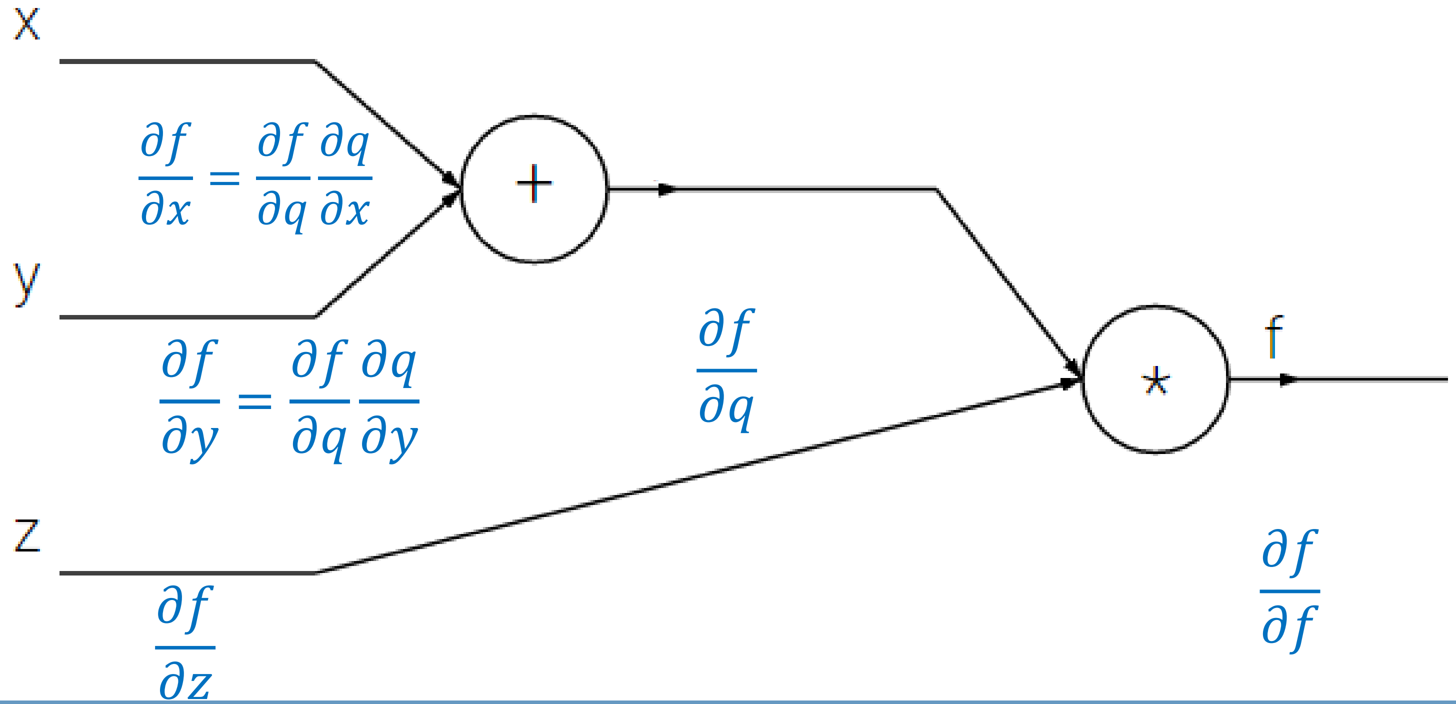
Grafo computacional

- Otro ejemplo algo más complejo:



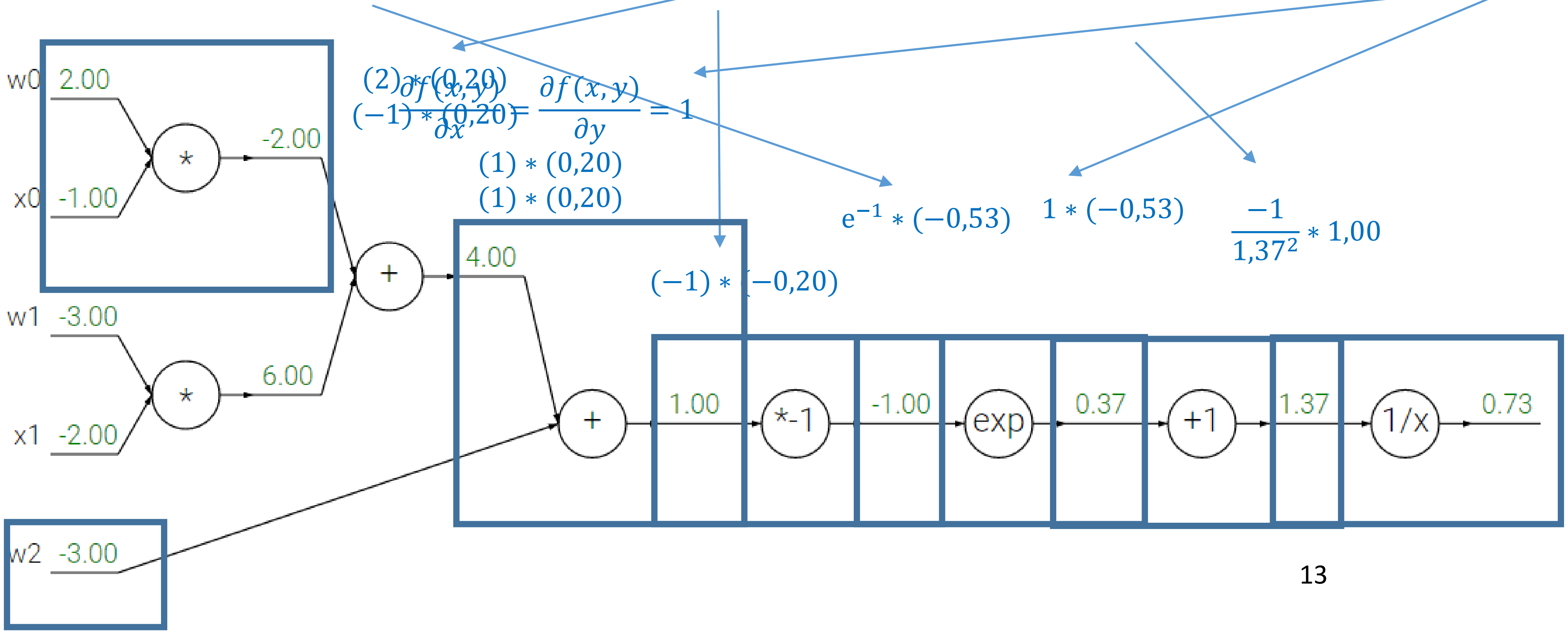
Grafo computacional: retropropagación

- Un ejemplo de propagación del gradiente: $f(x,y,z) = (x+y)z$
- Sean x=-2, y=5, z=-4
- Renombramos:
  - $q = (x+y)$
  - $f = qz$
- $\frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$
- $\frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$
- **Buscamos**: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Grafo computacional: retropropagación

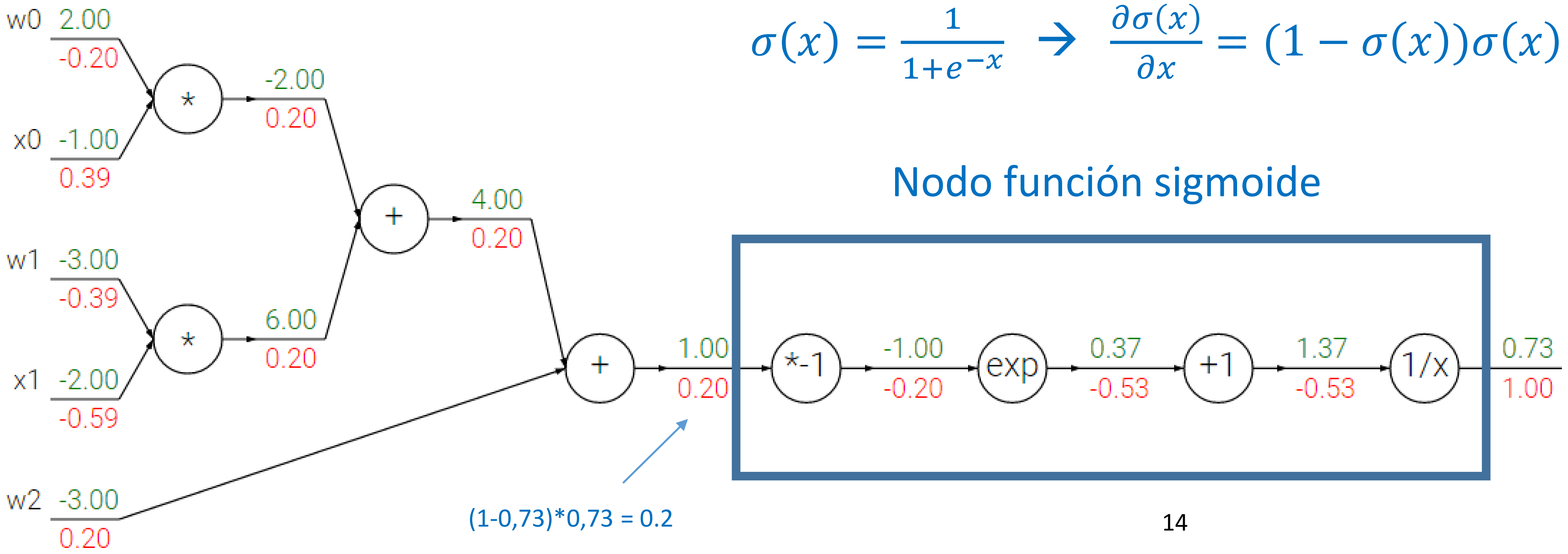
- Ej. con función sigmoide (perceptrón): $f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$
- $f(x) = e^x \rightarrow \frac{\partial f}{\partial x} = e^x$ | $f(x) = ax \rightarrow \frac{\partial f}{\partial x} = a$ | $f(x) = \frac{1}{x} \rightarrow \frac{\partial f}{\partial x} = \frac{-1}{x^2}$ | $f(x) = c + x \rightarrow \frac{\partial f}{\partial x} = 1$



Grafo computacional: retropropagación

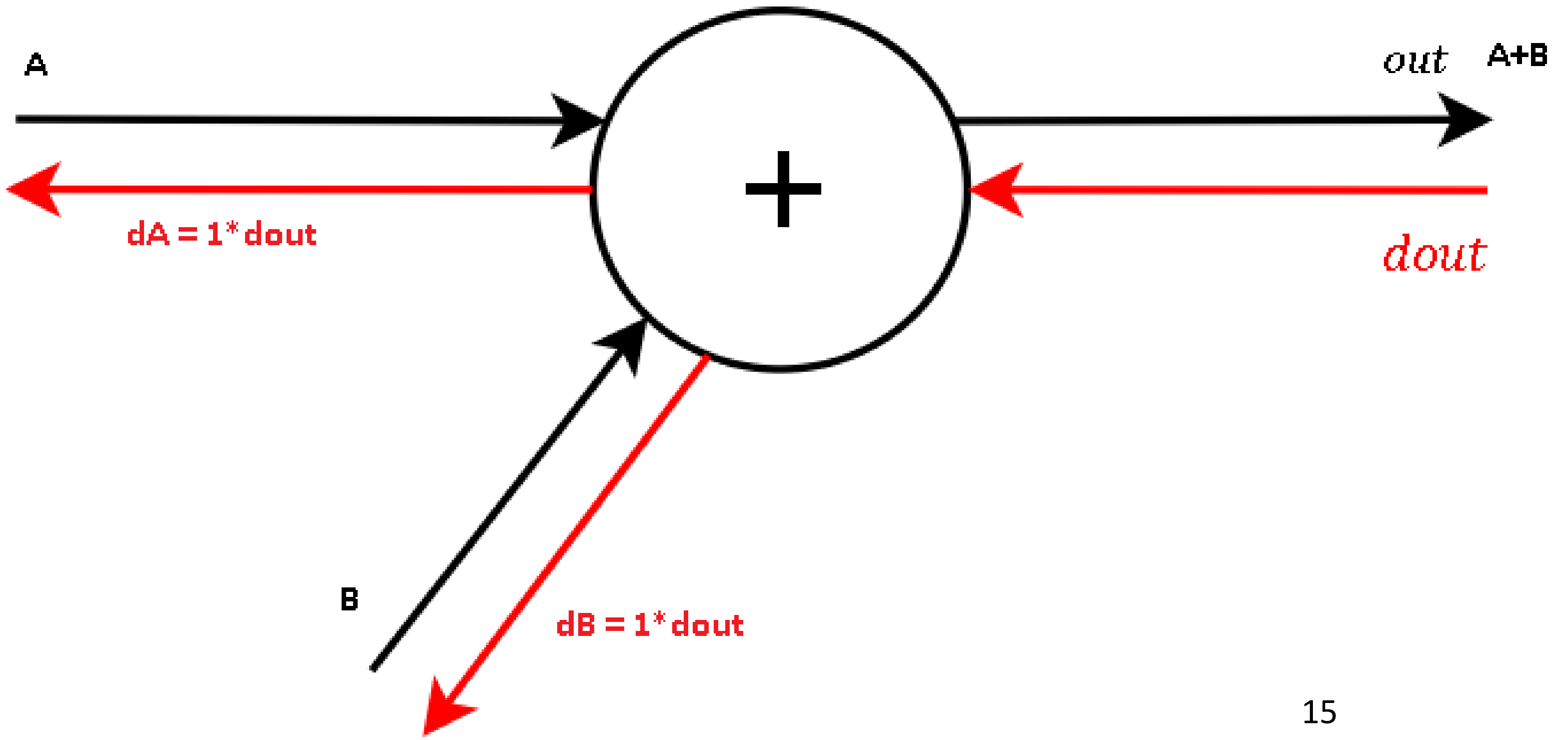
- Se pueden definir nodos para funciones conocidas para ahorrar pasos de computación.

$$\sigma(x) = \frac{1}{1+e^{-x}} \rightarrow \frac{\partial\sigma(x)}{\partial x} = (1-\sigma(x))\sigma(x)$$



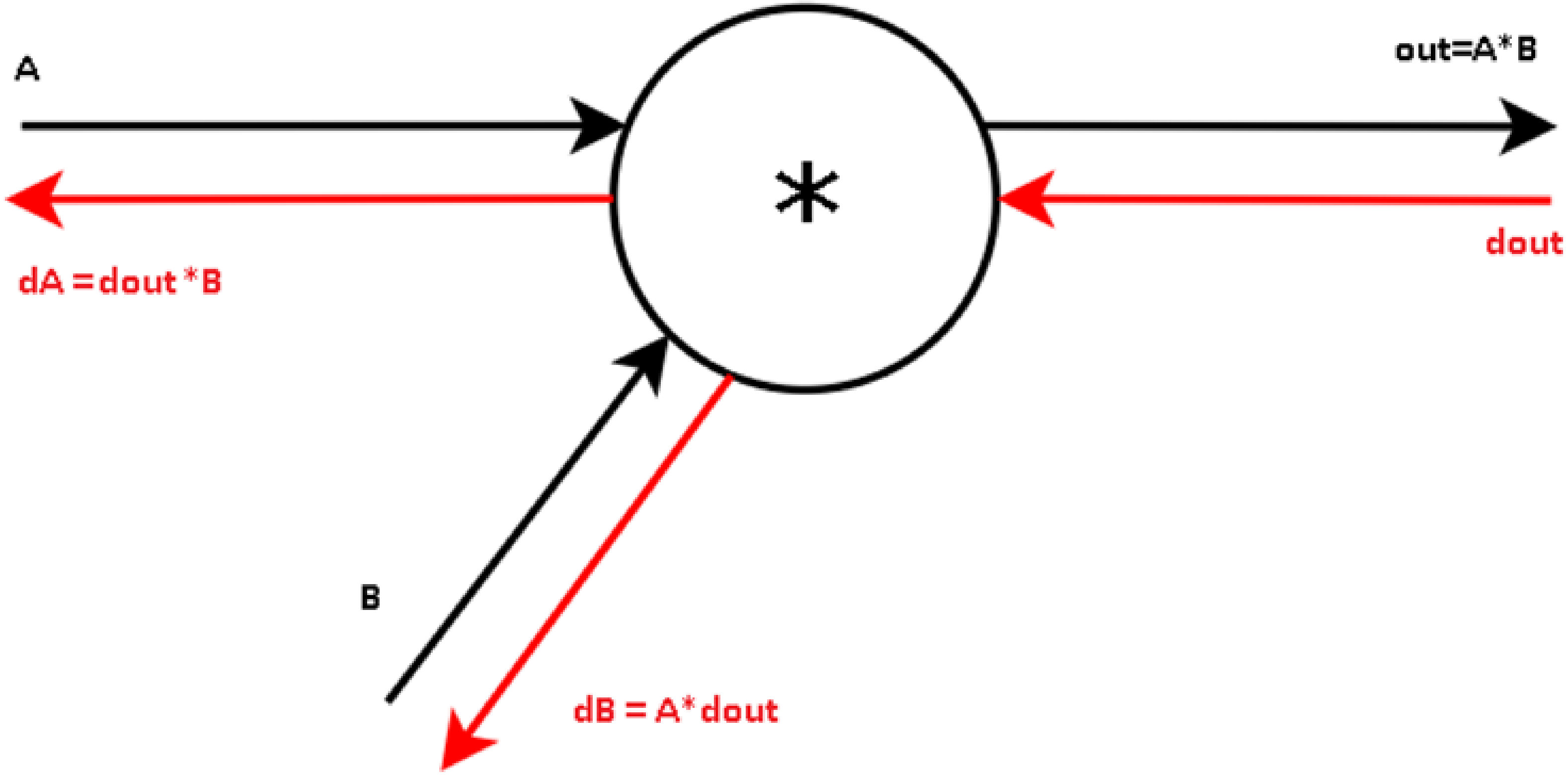
Grafo computacional: retropropagación

- Bloques básicos
  - **Suma** (distribuidor de gradiente)



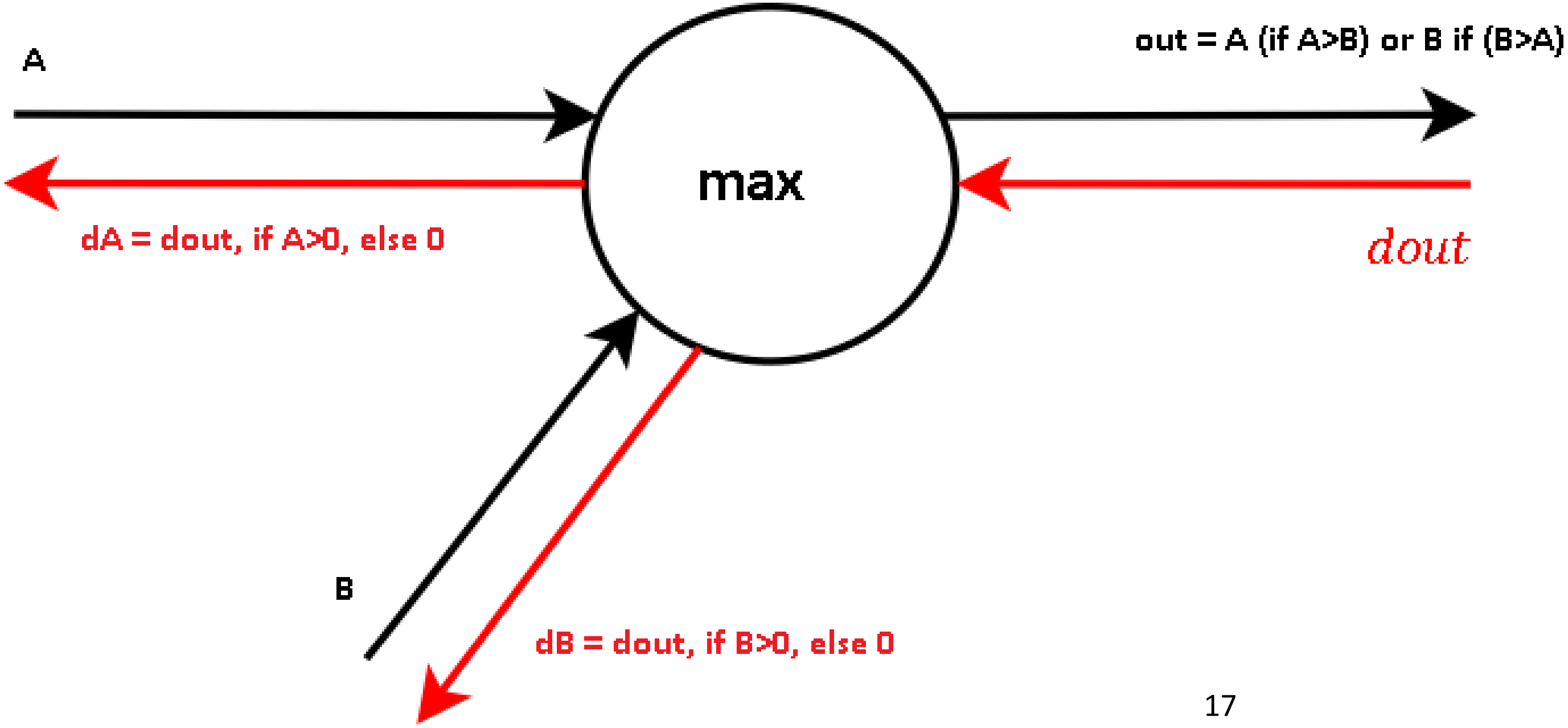
Grafo computacional: retropropagación

- Bloques básicos
  - **Multiplicación** (intercambiador de gradiente)



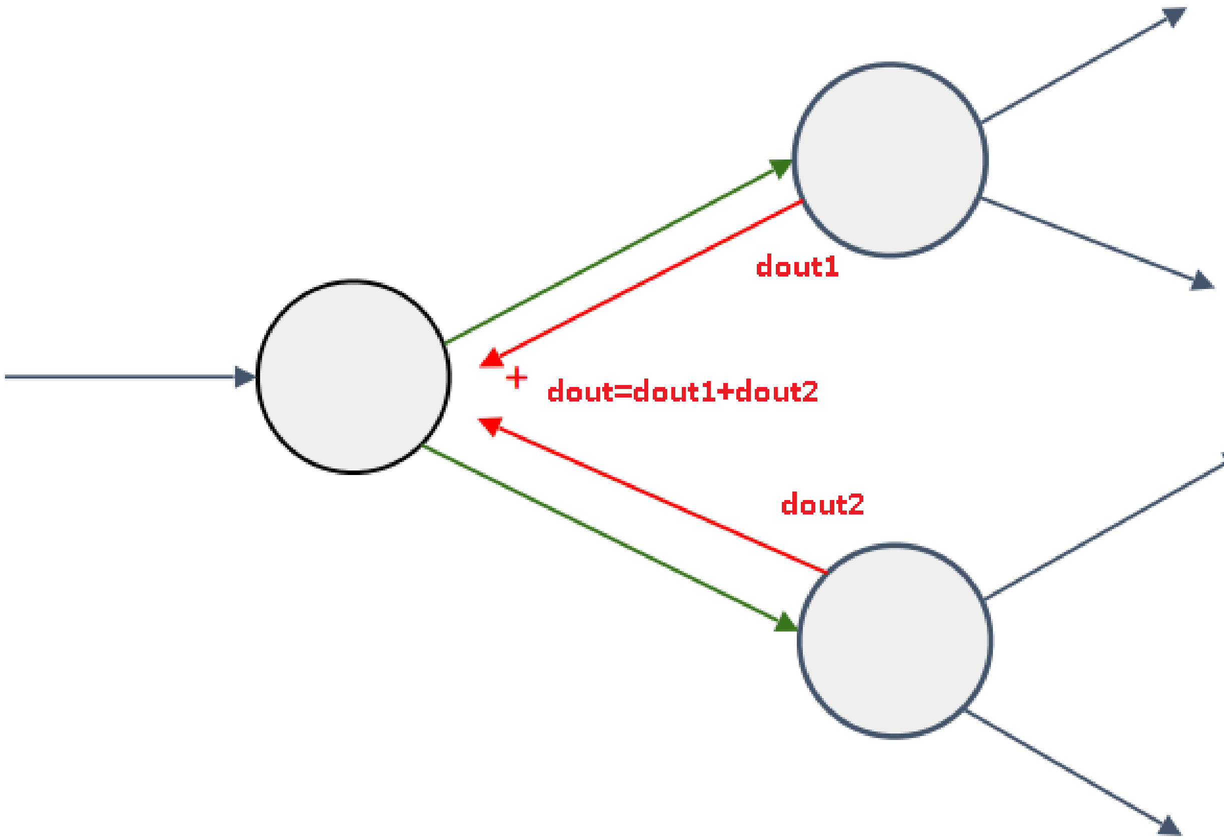
Grafo computacional: retropropagación

- Bloques básicos
  - **Máximo** (enrutador de gradiente)



Grafo computacional: retropropagación

- Bloques básicos
  - **Ramas** (sumador de gradientes)



Recapitulación

- La base del algoritmo de backpropagation es la **regla de la cadena**.
- La **propagación del gradiente** es la clave para actualizar los pesos en las **capas ocultas**.
- Si vemos la red como un **grafo computacional**, podemos entender la intuitivamente cómo funciona el algoritmo.
- Es importante entender cómo se propagan los gradientes para evitar el **problema del desvanecimiento del gradiente**.