# Algoritmo de propagación hacia atrás para el cálculo del gradiente de la función de coste en una red neuronal completamente conectada

## 1. Notación

Supongamos que tenemos una red neuronal de $L$ capas completamente conectadas. Durante este texto utilizaremos la siguiente notación:

- $n_l$ es el número de neuronas de la capa $l$. Por convención consideramos $n_0$ la dimensión de los datos de entrada.
- $X \in \mathcal{M}_{n_0 \times m}(\mathbb{R})$ es la matriz de datos de entrada, donde cada columna representa un ejemplo y cada fila representa un atributo.
- $W^{[l]} \in \mathcal{M}_{n_l \times n_{l-1}}(\mathbb{R})$ denota la matriz de pesos que conecta la capa $l-1$ con la capa $l$. Más concretamente, el elemento de $W^{[l]}$ correspondiente a la fila $j$, columna $k$, denotado $w^{[l]}_{jk}$, es un escalar que representa el peso de la conexión entre la neurona $j$ de la capa $l$ y la neurona $k$ de la capa $l-1$.
- $b^{[l]} \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$ es un vector que denota el bias de la capa $l$. El bias correspondiente a la neurona $j$ de la capa $l$ lo denotamos $b^{[l]}_j$.
- $z^{[l]} \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$ denota la combinación lineal de la entrada a la capa $l$ con los parámetros $W^{[l]}$ y $b^{[l]}$.
- $g : \mathbb{R} \to \mathbb{R}$ es una función no lineal (como relu o sigmoid). Si $M \in \mathcal{M}_{c \times d}(\mathbb{R})$ es una matriz (o un vector), denotaremos $g(M) \in \mathcal{M}_{c \times d}(\mathbb{R})$ la matriz (o el vector) que se obtiene aplicando la función $g$ a cada coordenada de $M$.
- $a^{[l]} \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$ denota el vector salida de la capa $l$ de la red neuronal. En particular $a^{[L]}$ denota la salida de la red neuronal. Por convención, denotaremos $a^{[0]}$ el vector de atributos de entrada a la red neuronal.
- Denotaremos el producto de matrices con el símbolo $*$, el producto de escalares con el símbolo $\cdot$ y el producto componente a componente con el símbolo $\odot$.

## 2. Caso particular con un único ejemplo

### 2.1. Propagación hacia delante

Si tenemos un único ejemplo entonces $m = 1$ y $X$ es un vector columna de longitud $n_0$, el número de atributos, que coincide con la dimensión de la entrada a la red neuronal. Entonces, según la convención que estamos utilizando, $a^{[0]} = X$.

En este caso, las ecuaciones para la propagación hacia delante son:

$$z_j^{[l]} = \sum_{k=1}^{n_{l-1}} w_{jk}^{[l]} \cdot a_k^{[l-1]} + b_j^{[l]} \tag{1}$$

$$a_j^{[l]} = g(z_j^{[l]}) \tag{2}$$

Para poder calcular todas las componentes a la vez es posible escribir las fórmulas anteriores en versión matricial de la siguiente forma:

$$z^{[l]} = W^{[l]} * a^{[l-1]} + b^{[l]} \tag{3}$$

$$a^{[l]} = g(z^{[l]}) \tag{4}$$

Una vez hemos aplicado las fórmulas anteriores a todas las capas obtenemos la salida de la red, $a^{[L]}$. Para saber si la salida de la red es adecuada podemos definir una función que mida qué error comete la red neuronal en la salida, $\mathcal{L}(y, a^{[L]})$, donde $y$ es la etiqueta correcta asociada al ejemplo con el que trabajamos. Si la etiqueta es binaria, es decir siempre vale 0 o 1, entonces la salida de la red neuronal está formada por un único valor (esto es, $n_L = 1$) y una posible función de error es el log-loss:

$$\mathcal{L}(y, a^{[L]}) = -(y \cdot \log(a^{[L]}) + (1 - y) \cdot \log(1 - a^{[L]})) \tag{5}$$

Es importante notar que hemos multiplicado por $-1$ la expresión dentro de los paréntesis para que el error sea positivo y minimizar el error se corresponda con minimizar la función de coste.

## 2.2. Propagación hacia atrás

Observemos que en el cálculo de $a^{[L]}$ intervienen todos los parámetros $W^{[l]}, b^{[l]}$, con $l = 1, \ldots, L$. Por lo tanto, la función de coste $\mathcal{L}(y, a^{[L]})$ también depende de los parámetros $W^{[l]}, b^{[l]}$, para todo $l = 1, \ldots, L$.

Dado que la función de coste mide la distancia entre la salida de la red y la etiqueta correcta podemos intentar minimizar esta función para acercar lo máximo posible los valores predichos a los valores correctos. Para minimizar la función de coste debemos modificar los parámetros de la red de forma adecuada, y para ello podemos calcular el gradiente de la función de coste respecto a los parámetros de la red y utilizarlo para actualizar los valores de los parámetros.

Para calcular el gradiente de la función de coste respecto a los parámetros de la red utilizaremos el algoritmo de la propagación hacia atrás. Este algoritmo se basa en aplicar repetidamente la regla de la cadena para calcular las derivadas parciales de la función de coste respecto cualquier parámetro de la red neuronal.

**Recordatorio de la regla de la cadena**

Supongamos que tenemos dos funciones de una variable

$$\begin{array}{rccc} f & : & \mathbb{R} & \to & \mathbb{R} \\ & & x & \mapsto & f(x) \end{array} \qquad \begin{array}{rccc} g & : & \mathbb{R} & \to & \mathbb{R} \\ & & y & \mapsto & g(y) \end{array}$$

Entonces, por la regla de la cadena, la derivada de la composición $g(f(x))$ respecto $x$ viene dada por el producto:

$$\frac{\partial(g \circ f)}{\partial x} = \frac{\partial g}{\partial f} \cdot \frac{\partial f}{\partial x} \tag{6}$$

Ahora supongamos que tenemos dos funciones en varias variables como las siguientes

$$\begin{array}{rccc} f & : & \mathbb{R} & \to & \mathbb{R}^d \\ & & x & \mapsto & (f_1(x), \ldots, f_d(x)) \end{array} \qquad \begin{array}{rccc} g & : & \mathbb{R}^d & \to & \mathbb{R} \\ & & (y_1, \ldots, y_d) & \mapsto & g(y_1, \ldots, y_d) \end{array}$$

Entonces, por la regla de la cadena en varias variables, la derivada de la composición $g(f(x))$ respecto $x$ viene dada por la fórmula:

$$\frac{\partial (g \circ f)}{\partial x} = \sum_{c=1}^{d} \frac{\partial g}{\partial f_c} \cdot \frac{\partial f_c}{\partial x} \tag{7}$$

Utilizando la regla de la cadena podemos calcular entonces la derivada parcial de la función de coste con respecto a cualquier parámetro de la red neuronal.

El primer paso es calcular el gradiente respecto a la salida de la red neuronal. Esto dependerá de la función de coste $\mathcal{L}$ que se utilice, pero se puede calcular de forma analítica. Por ejemplo, en el caso de la función de coste log-loss, tenemos:

$$\frac{\partial \mathcal{L}}{\partial a^{[L]}} = -\left(\frac{y}{a^{[L]}} - \frac{1-y}{1-a^{[L]}}\right) \tag{8}$$

En general, asumiremos que hemos calculado $\frac{\partial \mathcal{L}}{\partial a^{[L]}}$ y que lo podemos utilizar en la regla de la cadena. Supongamos que queremos calcular la derivada de la función de coste respecto a un peso concreto de la red neuronal $w_{jk}^{[l]}$. Para ello, asumimos que todos los demás parámetros son constantes y tenemos entonces la siguiente composición de funciones:

$$\begin{array}{ccccccc} \mathbb{R} & \to & \mathbb{R} & \to & \mathbb{R} & \to & \mathbb{R} \\ w_{jk}^{[l]} & \mapsto & z_j^{[l]} & \mapsto & a_j^{[l]} & \mapsto & \mathcal{L} \end{array}$$

Por lo que, aplicando repetidamente la regla de la cadena en una variable, obtenemos:

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^{[l]}} = \frac{\partial \mathcal{L}}{\partial a_j^{[l]}} \cdot \frac{\partial a_j^{[l]}}{\partial z_j^{[l]}} \cdot \frac{\partial z_j^{[l]}}{\partial w_{jk}^{[l]}} \tag{9}$$

Si estamos trabajando con la última capa, entonces $l = L$ y, por lo tanto, ya tenemos calculado el valor de $\frac{\partial \mathcal{L}}{\partial a_j^{[l]}}$. Asumamos por ahora que, aunque $l$ sea menor que $L$ ya tenemos calculado el valor de $\frac{\partial \mathcal{L}}{\partial a_j^{[l]}}$, posteriormente veremos como se calcula. Entonces, utilizando las ecuaciones 1 y 2 tenemos:

$$\frac{\partial a_j^{[l]}}{\partial z_j^{[l]}} = g'(z_j^{[l]}) \tag{10}$$

$$\frac{\partial z_j^{[l]}}{\partial w_{jk}^{[l]}} = a_k^{[l-1]} \tag{11}$$

Supongamos ahora que queremos calcular la derivada $\frac{\partial \mathcal{L}}{\partial b_j^{[l]}}$. El procedimiento es análogo a lo que hemos hecho hasta ahora, tenemos en este caso la siguiente descomposición:

$$\begin{array}{ccccccc} \mathbb{R} & \to & \mathbb{R} & \to & \mathbb{R} & \to & \mathbb{R} \\ b_j^{[l]} & \mapsto & z_j^{[l]} & \mapsto & a_j^{[l]} & \mapsto & \mathcal{L} \end{array}$$

Por lo que, de nuevo aplicando la regla de la cadena en una variable, obtenemos:

$$\frac{\partial \mathcal{L}}{\partial b_j^{[l]}} = \frac{\partial \mathcal{L}}{\partial a_j^{[l]}} \cdot \frac{\partial a_j^{[l]}}{\partial z_j^{[l]}} \cdot \frac{\partial z_j^{[l]}}{\partial b_j^{[l]}} \tag{12}$$

Volviendo a asumir que tenemos calculada la derivada $\frac{\partial \mathcal{L}}{\partial a_j^{[l]}}$, podemos calcular $\frac{\partial \mathcal{L}}{\partial b_j^{[l]}}$ a partir de:

$$\frac{\partial a_j^{[l]}}{\partial z_j^{[l]}} = g'(z_j^{[l]}) \tag{13}$$

$$\frac{\partial z_j^{[l]}}{\partial b_j^{[l]}} = 1 \tag{14}$$

Si queremos calcular las derivadas de la función de coste respecto a todos las componentes de una matriz o un vector a la vez, podemos utilizar la siguiente notación para la matriz de pesos:

$$\frac{\partial \mathcal{L}}{\partial W^{[l]}} = \left( \frac{\partial \mathcal{L}}{\partial a^{[l]}} \odot g'(z^{[l]}) \right) * (a^{[l-1]})^T \tag{15}$$

Donde $\frac{\partial \mathcal{L}}{\partial a^{[l]}}, g'(z^{[l]}) \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$ y $(a^{[l-1]})^T \in \mathcal{M}_{1 \times n_{l-1}}(\mathbb{R})$, por lo que $\frac{\partial \mathcal{L}}{\partial W^{[l]}} \in \mathcal{M}_{n_l \times n_{l-1}}(\mathbb{R})$.
Y la siguiente notación para el vector de bias:

$$\frac{\partial \mathcal{L}}{\partial b^{[l]}} = \frac{\partial \mathcal{L}}{\partial a^{[l]}} \odot g'(z^{[l]}) = \frac{\partial \mathcal{L}}{\partial z^{[l]}} \tag{16}$$

Donde, en este caso, $\frac{\partial \mathcal{L}}{\partial b^{[l]}} \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$.

Por último, necesitamos poder calcular el valor de $\frac{\partial \mathcal{L}}{\partial a_j^{[l]}}$ para $l < L$. En este caso, la activación de la neurona $j$ en la capa $l$ afecta a todas las neuronas de la capa $l+1$, por lo que la descomposición en funciones que tenemos es la siguiente:

$$\begin{array}{ccccccc} \mathbb{R} & \to & \mathbb{R}^{n_{l+1}} & \to & \mathbb{R}^{n_{l+1}} & \to & \mathbb{R} \\ a_j^{[l]} & \mapsto & (z_1^{[l+1]}, \dots, z_{n_{l+1}}^{[l+1]}) & \mapsto & (a_1^{[l+1]}, \dots, a_{n_{l+1}}^{[l+1]}) & \mapsto & \mathcal{L} \end{array}$$

Y si aplicamos la regla de la cadena en varias variables obtenemos la siguiente fórmula:

$$\frac{\partial \mathcal{L}}{\partial a_j^{[l]}} = \sum_{c=1}^{n_{l+1}} \frac{\partial \mathcal{L}}{\partial a_c^{[l+1]}} \cdot \frac{\partial a_c^{[l+1]}}{\partial z_c^{[l+1]}} \cdot \frac{\partial z_c^{[l+1]}}{\partial a_j^{[l]}} \tag{17}$$

Ahora sí, por recursividad, podemos suponer que tenemos calculada la derivada $\frac{\partial \mathcal{L}}{\partial a_c^{[l+1]}}$ para todo $c = 1, \dots, n_{l+1}$. Por lo tanto, podemos calcular completamente la derivada $\frac{\partial \mathcal{L}}{\partial a_j^{[l]}}$ utilizando:

$$\frac{\partial a_c^{[l+1]}}{\partial z_c^{[l+1]}} = g'(z_c^{[l+1]}) \tag{18}$$

$$\frac{\partial z_c^{[l+1]}}{\partial a_j^{[l]}} = w_{cj}^{[l+1]} \tag{19}$$

El cálculo de las derivadas respecto a las activaciones de las neuronas también se puede hacer para todas las componentes a la vez utilizando la siguiente notación matricial:

$$\frac{\partial \mathcal{L}}{\partial a^{[l]}} = (W^{[l+1]})^T * \left( \frac{\partial \mathcal{L}}{\partial a^{[l+1]}} \odot g'(z^{[l+1]}) \right) \tag{20}$$

Donde $(W^{[l+1]})^T \in \mathcal{M}_{n_l \times n_{l+1}}(\mathbb{R})$ y $\frac{\partial \mathcal{L}}{\partial a^{[l+1]}}, g'(z^{[l+1]}) \in \mathcal{M}_{n_{l+1} \times 1}(\mathbb{R})$ por lo que $\frac{\partial \mathcal{L}}{\partial a^{[l]}} \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$.

Observemos que para hacer los cálculos que hemos especificado necesitamos saber los valores de $z^{[l]}$ y $a^{[l]}$ para todo $l = 1, \ldots, L$, que se han calculado anteriormente durante la propagación hacia delante.

A continuación resumimos los pasos para calcular el gradiente con un único ejemplo en el conjunto de datos.

**Algoritmo de propagación hacia atrás con un ejemplo**

1. Calcular $\frac{\partial \mathcal{L}}{\partial a^{[L]}}$
2. Desde $l = L$ hasta 1, repetir:
   *a)* Calcular $\frac{\partial \mathcal{L}}{\partial W^{[l]}} = \left( \frac{\partial \mathcal{L}}{\partial a^{[l]}} \odot g'(z^{[l]}) \right) * (a^{[l-1]})^T$
   *b)* Calcular $\frac{\partial \mathcal{L}}{\partial b^{[l]}} = \frac{\partial \mathcal{L}}{\partial a^{[l]}} \odot g'(z^{[l]})$
   *c)* Calcular $\frac{\partial \mathcal{L}}{\partial a^{[l-1]}} = (W^{[l]})^T * \left( \frac{\partial \mathcal{L}}{\partial a^{[l]}} \odot g'(z^{[l]}) \right)$
3. Devolver $\frac{\partial \mathcal{L}}{\partial W^{[l]}}$ y $\frac{\partial \mathcal{L}}{\partial b^{[l]}}$ para todo $l = 1, \ldots, L$.

En la descripción del algoritmo se puede ver por qué se llama "de propagación hacia atrás". En efecto, el algoritmo se basa en calcular las derivadas $\frac{\partial \mathcal{L}}{\partial a^{[l]}}$ en cada capa y propagar su valor hacia atrás para permitir el cálculo de las derivadas $\frac{\partial \mathcal{L}}{\partial W^{[l]}}$ y $\frac{\partial \mathcal{L}}{\partial b^{[l]}}$, que son los parámetros de la red neuronal que se pueden modificar.

# 3. Caso general con varios ejemplos

## 3.1. Propagación hacia delante

Asumamos ahora que tenemos varios ejemplos en la matriz de datos $X$, por lo que $m > 1$. Si nos fijamos en los diferentes valores que consideramos en el apartado de notación, veremos que los únicos que dependen de los datos (a parte de $X$), son $z^{[l]}$ y $a^{[l]}$, que son vectores columna. Podemos considerar entonces formar matrices colocando los vectores columna correspondientes a varios ejemplos uno al lado del otro. De esta forma, obtenemos:

$$Z^{[l]} = (z^{[l](1)} \quad z^{[l](2)} \quad \ldots \quad z^{[l](m)}) \tag{21}$$

$$A^{[l]} = (a^{[l](1)} \quad a^{[l](2)} \quad \ldots \quad a^{[l](m)}) \tag{22}$$

Donde $z^{[l](i)}$ y $a^{[l](i)}$ denotan los vectores $z^{[l]}$ y $a^{[l]}$ que corresponden al ejemplo $i$-ésimo, respectivamente, y hemos denotado con $A$ y $Z$ mayúsculas las matrices resultantes.

Utilizando la misma convención que anteriormente tenemos que $A^{[0]} = X$ y las ecuaciones matriciales de la propagación hacia delante se pueden escribir como:

$$Z^{[l]} = W^{[l]} * A^{[l-1]} + B^{[l]} \tag{23}$$

$$A^{[l]} = g(Z^{[l]}) \tag{24}$$

Donde $B^{[l]}$ es una matriz formada por el vector columna $b^{[l]}$ repetido $m$ veces. De esta forma, $A^{[L]}$ denota la salida de la red, donde cada columna corresponde a la salida para cada ejemplo.

Al tener varios ejemplos la función de coste global se define como la media de la función de coste para cada error, es decir:

$$J = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(y^{(i)}, a^{[L](i)}) \tag{25}$$

En este caso entonces nos interesa minimizar la función $J$ para conseguir que la salida de la red se aproxime a las etiquetas correctas de cada ejemplo.

## 3.2. Propagación hacia atrás

Dado que la acción de derivar es una transformación lineal, para calcular el gradiente de la función $J$ podemos calcular el gradiente de $\mathcal{L}$ para cada ejemplo por separado como hemos hecho en la sección anterior y posteriormente hacer la media.

Con esta información ya podríamos implementar el algoritmo de propagación hacia atrás completo con varios ejemplos. Sin embargo, dado que los procesadores actuales están diseñados para realizar cálculos en paralelo, es mucho más eficiente calcular el gradiente para todos los ejemplos a la vez utilizando matrices.

Para ello, podemos utilizar las fórmulas 15, 16 y 20 y adecuarlas a las substituciones $a^{[l]} \to A^{[l]}$ y $z^{[l]} \to Z^{[l]}$.

Concretamente, para la derivada $\frac{\partial J}{\partial W^{[l]}}$ obtenemos:

$$\frac{\partial J}{\partial W^{[l]}} = \frac{1}{m}\left(\frac{\partial \mathcal{L}}{\partial A^{[l]}} \odot g'(Z^{[l]})\right) * (A^{[l-1]})^T \tag{26}$$

Donde $\frac{\partial \mathcal{L}}{\partial A^{[l]}}, g'(Z^{[l]}) \in \mathcal{M}_{n_l \times m}(\mathbb{R})$ y $(A^{[l-1]})^T \in \mathcal{M}_{m \times n_{l-1}}(\mathbb{R})$, por lo que $\frac{\partial J}{\partial W^{[l]}} \in \mathcal{M}_{n_l \times n_{l-1}}(\mathbb{R})$. Observemos que el producto de matrices provoca que en cada componente se están sumando los valores correspondientes de todos los ejemplos, por lo que simplemente dividiendo por $m$ obtenemos la media que necesitamos.

Para la derivada $\frac{\partial J}{\partial b^{[l]}}$ podemos hacer:

$$\frac{\partial J}{\partial b^{[l]}} = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial \mathcal{L}}{\partial A^{[l](i)}} \odot g'(Z^{[l](i)}) = \frac{1}{m}\sum_{i=1}^{m}\frac{\partial \mathcal{L}}{\partial Z^{[l](i)}} \tag{27}$$

Donde, en este caso, estamos obteniendo los valores de cada ejemplo separados en columnas, por lo que debemos hacer la media de las columnas para obtener el valor de $\frac{\partial J}{\partial b^{[l]}} \in \mathcal{M}_{n_l \times 1}(\mathbb{R})$.

Por último, en las fórmulas anteriores se puede ver que no es necesario calcular $\frac{\partial J}{\partial A^{[l]}}$ para obtener $\frac{\partial J}{\partial W^{[l]}}$ y $\frac{\partial J}{\partial b^{[l]}}$, pero sí necesitamos las derivadas $\frac{\partial \mathcal{L}}{\partial A^{[l]}}$. Para conseguirlas podemos adaptar directamente la fórmula 20 y considerar:

$$\frac{\partial \mathcal{L}}{\partial A^{[l]}} = (W^{[l+1]})^T * \left(\frac{\partial \mathcal{L}}{\partial A^{[l+1]}} \odot g'(Z^{[l+1]})\right) \tag{28}$$

Donde los valores correspondientes a cada ejemplo se guardan, también en este caso, separados por columnas, que es exactamente lo que nos interesa.

Finalmente, podemos resumir todo el algoritmo de propagación hacia atrás general, con cualquier número de ejemplos y en formato matricial, con el siguiente procedimiento.

**Algoritmo de propagación hacia atrás**

1. Calcular $\frac{\partial \mathcal{L}}{\partial a^{[L]}}$
2. Desde $l = L$ hasta 1, repetir:
   *a*) Calcular $\frac{\partial J}{\partial W^{[l]}} = \frac{1}{m}\left(\frac{\partial \mathcal{L}}{\partial A^{[l]}} \odot g'(Z^{[l]})\right) * (A^{[l-1]})^T$
   *b*) Calcular $\frac{\partial J}{\partial b^{[l]}} = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial \mathcal{L}}{\partial A^{[l](i)}} \odot g'(Z^{[l](i)})$
   *c*) Calcular $\frac{\partial \mathcal{L}}{\partial A^{[l]}} = (W^{[l+1]})^T * \left(\frac{\partial \mathcal{L}}{\partial A^{[l+1]}} \odot g'(Z^{[l+1]})\right)$
3. Devolver $\frac{\partial J}{\partial W^{[l]}}$ y $\frac{J}{\partial b^{[l]}}$ para todo $l = 1, \ldots, L$.