Redes Neuronales

Carlos Malanche

21 de mayo de 2019

Las redes neuronales son una de las técnicas del aprendizaje estadístico más prometedoras. Esto, veremos poco más adelante, debido a su poder de *aprender* las funciones de transformación de los vectores independientes a espacios linealmente separables.

1. El antecesor de la red

El concepto de una red neuronal no es algo nuevo, desde hace varias décadas se tenía ya el concepto de interacciones entre elementos no lineales que derivaban en resultados sencillos.

2. Derivando el algoritmo de propagación hacia atrás

Supondremos de manera temporal que la capa ℓ consta de α neuronas, mientras que la capa $\ell+1$ consta de β neuronas. Con esto, no debiera ser muy complicado notar que el vector saliente de las neuronas de la capa ℓ vive en $\mathbb{R}^{\alpha\times 1}$. Siguiendo esa lógica, el vector saliente de la capa $\ell+1$ debe vivir en $\mathbb{R}^{\beta\times 1}$. Con ello, y por pura convención, definimos la matriz de pesos que convierte al vector $\boldsymbol{x}^{(\ell)}$ en el vector $\boldsymbol{x}^{(\ell+1)}$ como la matriz $\boldsymbol{W}^{(\ell+1)}\in\mathbb{R}^{\alpha\times\beta}$, y el vector de sesgos $\boldsymbol{b}^{(\ell+1)}\in\mathbb{R}^{\beta\times 1}$ de modo que

$$\boldsymbol{x}^{(\ell+1)}=\phi((\boldsymbol{W}^{(\ell+1)})^T\boldsymbol{x}^{(\ell)}+\boldsymbol{b}^{(\ell+1)})$$

Donde la función de activación ϕ ya se considera como una función vectorizada que se aplica a todas las entradas del tensor al que se aplique.

Definiremos dos variables auxiliares que nos ayudarán a calcular el gradiente por capa, primero:

$$\boldsymbol{z}^{(\ell+1)}\coloneqq(\boldsymbol{W}^{(\ell+1)})^T\boldsymbol{x}^{(\ell)}+\boldsymbol{b}^{(\ell+1)} \tag{1}$$

Que no es más que meter una variable en medio de las $\boldsymbol{x}^{(\ell)}$, es decir, $\boldsymbol{x}^{(\ell)}=\phi(\boldsymbol{z}^{(\ell)})$

La segunda variable que vamos a introducir es un poco más compleja, y es la derivada de la función de costo respecto a esta misma $\boldsymbol{z}^{(\ell)}$ para una capa en específico. Vamos a llamar a esta variable *delta*.

$$\boldsymbol{\delta}^{(\ell)}\coloneqq\frac{\partial\mathcal{L}_n}{\partial\boldsymbol{z}^{(\ell)}} \tag{2}$$

En donde denoto por $\mathcal{L}_n$ la contribución a la función de costo correspondiente a la medición número n. La variable $\boldsymbol{z}$ hace la operación conocida como *forward pass*; Dada una $\boldsymbol{z}^{(\ell)}$, es muy fácil calcular la $\boldsymbol{z}^{(\ell+1)}$, con lo que estos valores se *propagan* hacia adelante de la red. Lo contrario pasa con la *delta*, que dada una $\boldsymbol{\delta}^{(\ell+1)}$, es sencillo calcular una $\boldsymbol{\delta}^{(\ell)}$. Esperen. Creo que eso aún no ha quedado claro.

Notemos pues que la $\boldsymbol{z}^{(\ell+1)}$ tiene una dependencia de la $\boldsymbol{z}^{(\ell)}$. Recordemos que la función de costo es una composición de funciones masomenos del estilo

$$\mathcal{L}_n=\frac{1}{2N}(y-\boldsymbol{x}^{(L+1)}\circ\boldsymbol{x}^{(L)}\circ\cdots\circ\boldsymbol{x}^{(0)})^2$$

De aquí se sigue que si queremos derivar la función de costo respecto a una $\boldsymbol{z}^{(\ell)}$, podemos utilizar la regla de la cadena para hacer aparecer la primer función inmediata que a su vez depende de la misma $\boldsymbol{z}^{(\ell)}$:

$$\boldsymbol{\delta}^{(\ell)}\coloneqq\frac{\partial\mathcal{L}_n}{\partial\boldsymbol{z}^{(\ell)}}=\frac{\partial\boldsymbol{z}^{(\ell+1)}}{\partial\boldsymbol{z}^{(\ell)}}\frac{\partial\mathcal{L}_n}{\partial\boldsymbol{z}^{(\ell+1)}}=\frac{\partial\boldsymbol{z}^{(\ell+1)}}{\partial\boldsymbol{z}^{(\ell)}}\boldsymbol{\delta}^{(\ell+1)} \tag{3}$$

Ahora sí se ha vuelto más evidente la dependencia. La derivada que nos falta calcular no es tan complicada realmente

$$\frac{\partial \boldsymbol{z}^{(\ell+1)}}{\partial \boldsymbol{z}^{(\ell)}} = \frac{\partial}{\partial \boldsymbol{z}^{(\ell)}}((\boldsymbol{W}^{(\ell+1)})^T \phi(\boldsymbol{z}^{(\ell)}) + \boldsymbol{b}^{(\ell+1)}) = \text{diag}(\phi'(\boldsymbol{z}^{(\ell)}))\boldsymbol{W}^{(\ell+1)}$$

Con estas dos variables que hemos definido, calcular los gradientes se vuelve *casi* trivial. Primer, el vector de sesgos

$$\frac{\partial \mathcal{L}_n}{\partial \boldsymbol{b}^{(\ell)}} = \frac{\partial \boldsymbol{z}^{(\ell)}}{\partial \boldsymbol{b}^{(\ell)}} \frac{\partial \mathcal{L}_n}{\partial \boldsymbol{z}^{(\ell)}} = \boldsymbol{I}\boldsymbol{\delta}^{(\ell)} = \boldsymbol{\delta}^{(\ell)} \tag{4}$$

La derivada de la matriz de pesos la haremos entrada por entrada, pues la vida se nos puede complicar mucho si lo intentamos hacer matricialmente (aunque la expresión final será matricial).

$$\frac{\partial \mathcal{L}_n}{\partial \boldsymbol{W}_{i,j}^{(\ell+1)}} = \underbrace{\frac{\partial \boldsymbol{z}^{(\ell+1)}}{\partial \boldsymbol{W}_{i,j}^{(\ell+1)}} \frac{\partial \mathcal{L}_n}{\partial \boldsymbol{z}^{(\ell+1)}}}_{\text{producto interno}} = \sum_k (\frac{\partial}{\partial \boldsymbol{W}_{i,j}^{(\ell+1)}} \sum_s \boldsymbol{W}_{s,k}^{(\ell+1)} \boldsymbol{x}_s^{(\ell)}) \boldsymbol{\delta}_k^{(\ell+1)} = \sum_k \boldsymbol{x}_i^{(\ell)} \boldsymbol{\delta}_k^{(\ell+1)} \delta_{jk} = \boldsymbol{x}_i^{(\ell)} \boldsymbol{\delta}_j^{(\ell+1)}$$

Que escrito de forma matricial

$$\frac{\partial \mathcal{L}_n}{\partial \boldsymbol{W}^{(\ell+1)}} = \boldsymbol{x}^{(\ell)} (\boldsymbol{\delta}^{(\ell+1)})^T \tag{5}$$

Con lo que nuestro descenso de gradiente para cada capa queda escrito como:

$$_{it+1}\boldsymbol{b}^{(\ell+1)} = {}_{it}\boldsymbol{b}^{(\ell+1)} - \mu({}_{it}\boldsymbol{\delta}) \tag{6}$$

$$_{it+1}\boldsymbol{W}^{(\ell+1)} = {}_{it}\boldsymbol{W}^{(\ell+1)} - \mu \boldsymbol{x}^{(\ell)} ({}_{it}\boldsymbol{\delta}^{(\ell+1)})^T \tag{7}$$

Donde ya abusé un poco de la notación para colocar el índice de iteración como subíndice que precede.

3. Regularizador de las redes neuronales