Loss and Activation Functions and Their Derivatives

Feb. 7, 2020

Loss and Activation Functions for Output Layer

In the last layer of a neural network, for our three common settings (regression, binary classification, and multi-class classification):

- a specific loss function is always used
- a corresponding activation function is always used for the last layer (L)

Setting	Loss	Activation	Vectorized Derivative (up to constant of proportionality)
Regression $y^{(i)}$ is a number	**Mean Squared Error** $J(b,w) = \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - a_1^{(i)[L]}\right)^2$	**Linear** $a_1^{(i)[L]} = z_1^{(i)[L]}$	$\frac{dJ(b,w)}{dz^{[L]}} = \left[(a_1^{(1)[L]} - y^{(1)[L]}) \quad \cdots \quad (a_1^{(m)[L]} - y^{(m)[L]})\right]$
Binary Classification $y^{(i)}$ is 0 or 1	**Binary Cross-Entropy** $J(b,w) = \sum_{i=1}^{m} y^{(i)} \log\left(a_1^{(i)[L]}\right) + (1-y^{(i)})\log\left(1 - a_1^{(i)[L]}\right)$	**Sigmoid** $a_1^{(i)[L]} = \frac{\exp(z_1^{(i)[L]})}{1+\exp(z_1^{(i)[L]})}$	$\frac{dJ(b,w)}{dz^{[L]}} = \left[(a_1^{(1)[L]} - y^{(1)[L]}) \quad \cdots \quad (a_1^{(m)[L]} - y^{(m)[L]})\right]$
Multiclass Classification $y^{(i)} = 2$ **or** $y^{(i)} = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}$	**Categorical Cross-Entropy** $J(b,w) = \sum_{i=1}^{m} \log\left(a_{y^{(i)}}^{(i)[L]}\right)$ or $J(b,w) = \sum_{i=1}^{m}\sum_{j=1}^{K} y_j^{(i)} \log\left(a_j^{(i)[L]}\right)$	**Softmax** $\begin{bmatrix} a_1^{(i)[L]} \\ a_2^{(i)[L]} \\ \vdots \\ a_K^{(i)[L]} \end{bmatrix} = \begin{bmatrix} \frac{\exp(z_1^{(i)[L]})}{\sum_{j=1}^{K}\exp(z_j^{(i)[L]})} \\ \frac{\exp(z_2^{(i)[L]})}{\sum_{j=1}^{K}\exp(z_j^{(i)[L]})} \\ \vdots \\ \frac{\exp(z_K^{(i)[L]})}{\sum_{j=1}^{K}\exp(z_j^{(i)[L]})} \end{bmatrix}$	$\frac{dJ(b,w)}{dz^{[L]}} = \begin{bmatrix} (a_1^{(1)[L]} - y_1^{(1)[L]}) & \cdots & (a_1^{(m)[L]} - y_1^{(m)[L]}) \\ (a_2^{(1)[L]} - y_2^{(1)[L]}) & \cdots & (a_2^{(m)[L]} - y_2^{(m)[L]}) \\ \vdots & \ddots & \vdots \\ (a_K^{(1)[L]} - y_K^{(1)[L]}) & \cdots & (a_K^{(m)[L]} - y_K^{(m)[L]}) \end{bmatrix}$

Activation Functions for Hidden Layers

- A rectified linear unit is the recommended default activation function for fully connected (dense) layers.
- *tanh* is another option that was more common in the past. It can work ok.
- Sigmoid was also used in the past but is definitely not recommended.
- Lots of research about other options.

In the notation below:

- n_l is the number of units in layer l
- $\mathbb{I}_{[0,\infty)}(z) = \begin{cases} 1 \text{ if } z \in [0.\infty] \\ 0 \text{ otherwise} \end{cases}$

Activation	Vectorized Derivative (up to constant of proportionality)
Rectified Linear (ReLU)	
$\begin{bmatrix} a_1^{(i)[l]} \\ a_2^{(i)[l]} \\ \vdots \\ a_{n_l}^{(i)[l]} \end{bmatrix} = \begin{bmatrix} \max\left(0, z_1^{(i)[l]}\right) \\ \max\left(0, z_2^{(i)[l]}\right) \\ \vdots \\ \max\left(0, z_{n_l}^{(i)[l]}\right) \end{bmatrix}$	$\frac{da^{[l]}}{dz^{[l]}} = \begin{bmatrix} \mathbb{I}_{[0,\infty)}\left(z_1^{(1)[l]}\right) & \cdots & \mathbb{I}_{[0,\infty)}\left(z_1^{(m)[l]}\right) \\ \mathbb{I}_{[0,\infty)}\left(z_2^{(1)[l]}\right) & \cdots & \mathbb{I}_{[0,\infty)}\left(z_2^{(m)[l]}\right) \\ \vdots & \ddots & \vdots \\ \mathbb{I}_{[0,\infty)}\left(z_{n_l}^{(1)[l]}\right) & \cdots & \mathbb{I}_{[0,\infty)}\left(z_{n_l}^{(m)[l]}\right) \end{bmatrix}$
tanh	
$\begin{bmatrix} a_1^{(i)[l]} \\ a_2^{(i)[l]} \\ \vdots \\ a_{n_l}^{(i)[l]} \end{bmatrix} = \begin{bmatrix} tanh\left(z_1^{(i)[l]}\right) \\ tanh\left(z_2^{(i)[l]}\right) \\ \vdots \\ tanh\left(z_{n_l}^{(i)[l]}\right) \end{bmatrix}$	$\frac{da^{[l]}}{dz^{[l]}} = \begin{bmatrix} 1-\left(a_1^{(1)[l]}\right)^2 & \cdots & 1-\left(a_1^{(m)[l]}\right)^2 \\ 1-\left(a_2^{(1)[l]}\right)^2 & \cdots & 1-\left(a_2^{(m)[l]}\right)^2 \\ \vdots & \ddots & \vdots \\ 1-\left(a_{n_l}^{(1)[l]}\right)^2 & \cdots & 1-\left(a_{n_l}^{(m)[l]}\right)^2 \end{bmatrix}$