Derivative of the Categorical Cross-Entropy Function

John Purcell

May 21, 2021

Abstract

We will also use the symbol σ to indicate the softmax function.

1 Categorical Cross-Entropy

The categorical cross-entropy loss function is defined like this:

$$\lambda = -\sum_i \hat{a}_i ln(a_i) \tag{1}$$

where

$\hat{a}_i$ is the expected (desired) value for the output of neuron i in the final layer, for some particular input to the network

a_i is the actual output of the ith neuron in the final layer for the same input

λ ("lambda") is the network loss, which we seek to minimise.

We need to find the rate of change of λ with respect to each of the weighted sums z_i of the neurons in the final layer.

$$\frac{\partial \lambda}{\partial z_j} = \frac{\partial}{\partial z_j}\left(-\sum_i \hat{a}_i ln(a_i)\right) = -\sum_i \hat{a}_i \frac{\partial}{\partial z_j} ln(a_i) = -\sum_i \frac{\hat{a}_i}{a_i}\frac{\partial a_i}{\partial z_j} \tag{2}$$

We have already calculated $\frac{\partial a_i}{\partial z_j}$. Since we are using the softmax function in our network, it is the derivative of softmax with respect to the z_j.

$$\frac{\partial a_i}{\partial z_j} = \sigma(z_i)(\delta_{ij} - \sigma(z_j))$$

where σ is the softmax function.

Putting this into (2):

$$\frac{\partial \lambda}{\partial z_j} = -\sum_i \frac{\hat{a}_i}{a_i}\sigma(z_i)(\delta_{ij} - \sigma(z_j)) = -\sum_i \hat{a}_i(\delta_{ij} - \sigma(z_j))$$

We can write this as a sum of two cases: the case where $i = j$ and $\delta_{ij} = 0$ the case where $i \neq j$ and $\delta_{ij} = 1$.

$$\frac{\partial \lambda}{\partial z_j} = -\left(\sum_{i=j} \hat{a}_i(\delta_{ij} - \sigma(z_j)) + \sum_{i \neq j} \hat{a}_i(\delta_{ij} - \sigma(z_j))\right)$$

Simplifying:

$$\frac{\partial \lambda}{\partial z_j} = (\hat{a}_j(1 - \sigma(z_j))) - \sum_{i \neq j} \hat{a}_i\sigma(z_j) = (\hat{a}_j - \hat{a}_j\sigma(z_j)) - \sum_{i \neq j} \hat{a}_i\sigma(z_j)$$

Now we can easily put the term $\hat{a}_j\sigma(z_j)$ back inside the summation.

$$\frac{\partial \lambda}{\partial z_j} = \hat{a}_j - \sum_i \hat{a}_i\sigma(z_j)$$

The expression $\sigma(z_j)$ can be taken out of the summation, since it's the same for every term of the summation.

$$\frac{\partial \lambda}{\partial z_j} = \hat{a}_j - \sigma(z_j)\sum_i \hat{a}_i$$

Finally, we use the fact that the $\hat{a}_i$ sum to 1, because they are a probability distribution. In fact, for our particular purposes, since we're using one-hot vectors in this course, they're all zero apart from one of them, which will be equal to 1.

$$\frac{\partial \lambda}{\partial z_j} = \hat{a_j} - \sigma(z_j)$$

We can write this in vector form:

$$\frac{\partial \lambda}{\partial \mathbf{z}} = \hat{\mathbf{a}} - \mathbf{a}$$

Where $\hat{\mathbf{a}}$ is the expected vector of outputs of the neural network, $\mathbf{a}$ is the actual vector of outputs, and $\mathbf{z}$ are the weighted sums of the neurons in the output layer.