Softmax

John Purcell

November 14, 2021

1 Softmax

The softmax function is often used as an activation function on the output layer only.

$$a_n = \frac{e^{z_n}}{\sum_i e^{z_i}}$$

a_n is the activation of the nth neuron
z_n is the weighted sum of the nth neuron

2 Motivation

Consider a classifier neural network, where the network has to assign input to one of several possible mutually exclusive classes or categories.

It would be useful to be able to think of the output of the network in terms of probabilities.

Example: consider a network which accepts images as input and classifies them as representing the faces of one of three possible people, A, B or C.

$$\begin{pmatrix} 0.2 \\ 0.1 \\ 0.7 \end{pmatrix}$$

This might mean the network thinks there is a probability of 0.2 (20%) that the image represents face A, a probability of 0.1 (10%) that it represents face B, and a probability 0.7 (70%) that it represents face C.

Note that these probabilities must add up to 1 (or 100%), since we're assuming that one of these possibities is definitely true, so the combined probability of one of them being true (0.2+0.1+0.7) is 1.0.

The softmax function takes the positive and negative weighted sums of the final layer and turns them into something that looks like a series of probabilities: all values are between 0.0 and 1.0 and the values add up to 1.0.

3 Example

Suppose the output layer has the following weighted sums when a particular input is presented to the network.

$$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} -0.7 \\ 1.3 \\ -2.3 \end{pmatrix}$$

The first step is to transform these values using the exponential function.

$$\begin{pmatrix} e^{z_1} \\ e^{z_2} \\ e^{z_3} \end{pmatrix} = \begin{pmatrix} e^{-0.7} \\ e^{1.3} \\ e^{-2.3} \end{pmatrix} = \begin{pmatrix} 0.50 \\ 3.67 \\ 0.10 \end{pmatrix}$$

This transforms all values so that they are positive. It also tends to increase differences between values.

Now we need to normalise these values so that they add up to 1.0. First we add up all values: $0.50 + 3.67 + 0.10 = 4.27$

Now divide all values by this total.

$$\begin{pmatrix} 0.50/4.27 \\ 3.67/4.27 \\ 0.10/4.27 \end{pmatrix} = \begin{pmatrix} 0.117 \\ 0.859 \\ 0.023 \end{pmatrix}$$

Aside from rounding errors, this adds up to 1.0 (actually 0.99, but only due to rounding errors).

We can now interpret these activations (output values) of the output layer as probabilities attached to the various possible predictions that the network makes.