Activation Functions

quantra

Table of Contents

Let us discuss a few activation functions that are already built in Keras to train the models.

Softmax

The mathematical equation of the softmax function is:

$$p(y=j \mid x) = \frac{e^{x^T w_j}}{\sum_{k=1}^{K} e^{x^T w_k}}$$

Let us understand the application of softmax function with the help of an example.

If we consider an input x = [1, 2, 3, 4, 1, 2, 3], the softmax of 'x' will be:

x'= [0.024, 0.064, 0.175, 0.475, 0.024, 0.064, 0.175].

We achieve this by calculating the exponential values of all the initial numbers in the list that is

x1=[e^1,e^2,e^3,e^4,e^1,e^2,e^3]

After this, we sum up all these exponent values

x2= sum(x1)

and then divide the individual exponent values with this sum:

x' = x1/x2

x'= [0.024, 0.064, 0.175, 0.475, 0.024, 0.064, 0.175].

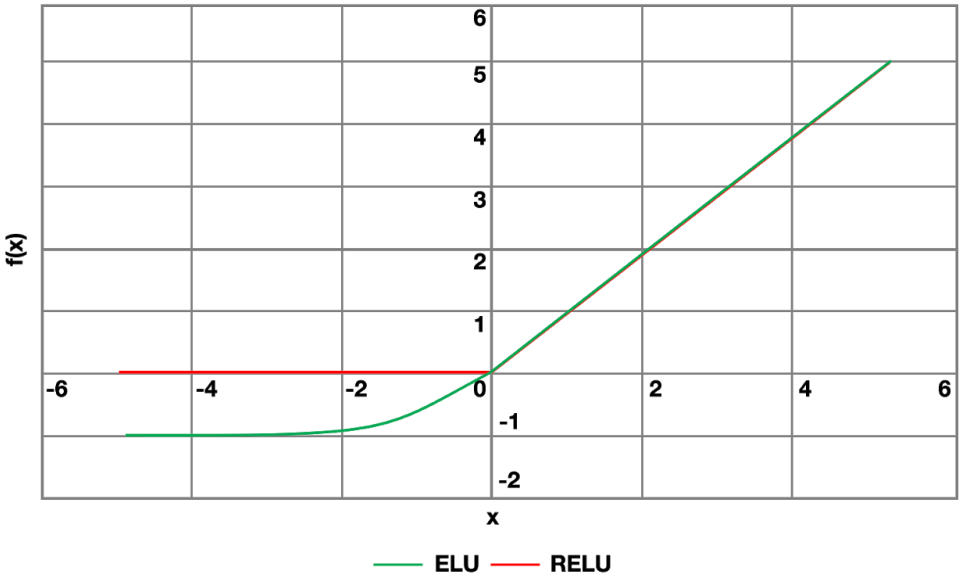
The output has most of its weight where the '4' was in the original input. This is what the function is normally used for: to highlight the largest values and suppress values which are significantly below the maximum value.

Note: softmax is not scale invariant, so if the input were [0.1, 0.2, 0.3, 0.4, 0.1, 0.2, 0.3] (which sums to 1.6) the softmax would be [0.125, 0.138, 0.153, 0.169, 0.125, 0.138, 0.153]. This shows that for values between 0 and 1 softmax in fact de-emphasizes the maximum value (note that 0.169 is not only less than 0.475, but also less than the initial value of 0.4).

Softmax is useful given outlier data, which we wish to include in the dataset while still preserving the significance of data within a standard deviation of the mean.

ELU (Exponential linear unit)

ELU is very similar to RELU except in the case of negative inputs. They are both in identity function form for non-negative inputs. On the other hand, ELU becomes smooth slowly until its output equal to -α whereas RELU sharply smoothens. Notice that α is equal to +1 in the following illustration.



ELU and RELU functions are very similar.

Derivative of the activation function is fed to the backpropagation algorithm during the training. That's why, both the function and its derivative should have low computation cost.

$f(x) = x$ if $x \geq 0$ (identity function)

$f(x) = \alpha.(e^x - 1)$ if $x < 0$

As seen, ELU consists of two different equations. That's why, their derivatives should be calculated separately.

Firstly, derivative of ELU would be one for x is greater than or equal to zero because derivative of identity function is always one.

On the other hand, what the derivative of ELU if x is less than is:

$$y = \alpha.(e^x - 1) = \alpha.e^x - \alpha$$

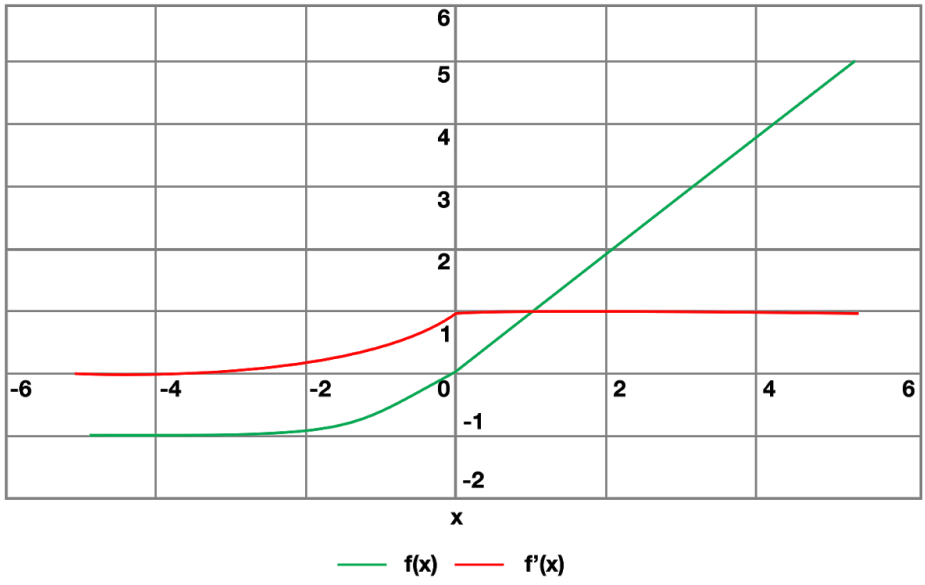
$$dy/dx = \alpha.e^x$$

That's the derivative but it can be expressed in simpler way. Adding and removing α to the derivative term wouldn't change the result. In this way, we can express the derivative in simpler way.

$$dy/dx = \alpha.e^x - \alpha + \alpha = (\alpha.e^x - \alpha) + \alpha = y + \alpha$$

$$y = \alpha.(e^x - 1)$$

$$dy/dx = y + \alpha$$



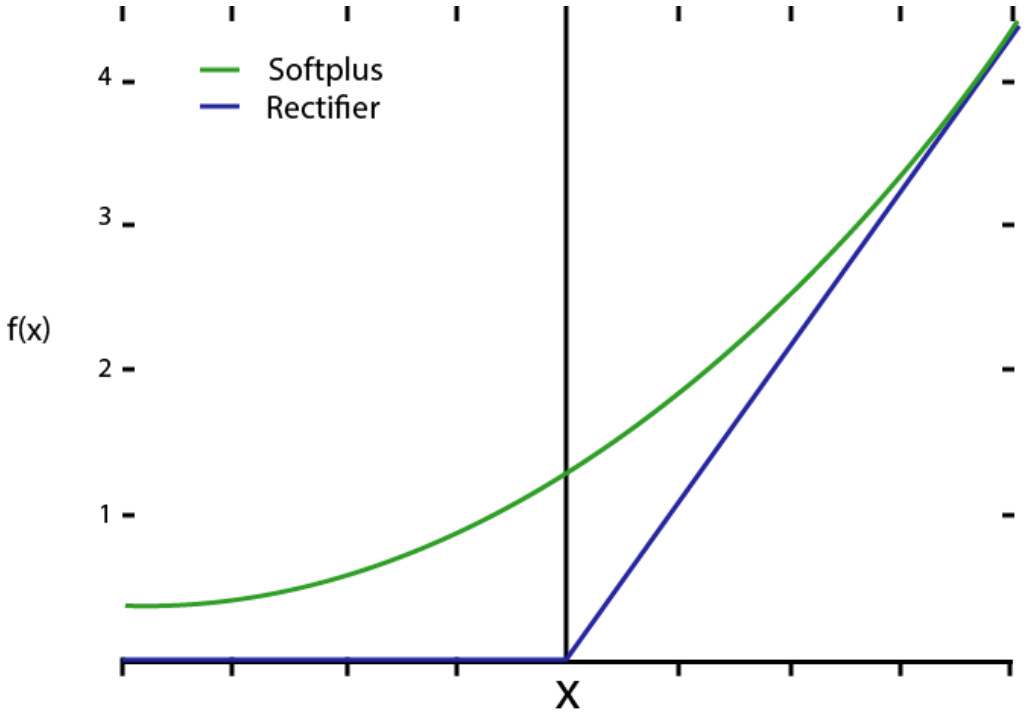
ReLU (rectified linear unit)

ReLU is an activation function defined as the positive part of its argument:

$$f(x)=\max(0,x) \quad \text{where } x \text{ is the input to a neuron.}$$

This activation function was first introduced to a dynamical network by Hahnloser et al. in a 2000 paper in Nature with strong biological motivations and mathematical justifications.

It has been demonstrated for the first time in 2011 to enable better training of deeper networks



A smooth approximation to the rectifier is the analytic function

$$f(x)=\log(1+e^x)$$

which is also called the **softplus** function. The derivative of softplus is

$$f'(x)= e^x/(1+e^x) =1/(1+e^{-x})$$ i.e. the logistic function.

Rectified linear units find applications in computer vision, and speech recognition using deep neural nets.

Sigmoid and tanh

These two functions are very similar not only in terms of their appearance but also the mathematical presentation and usage.

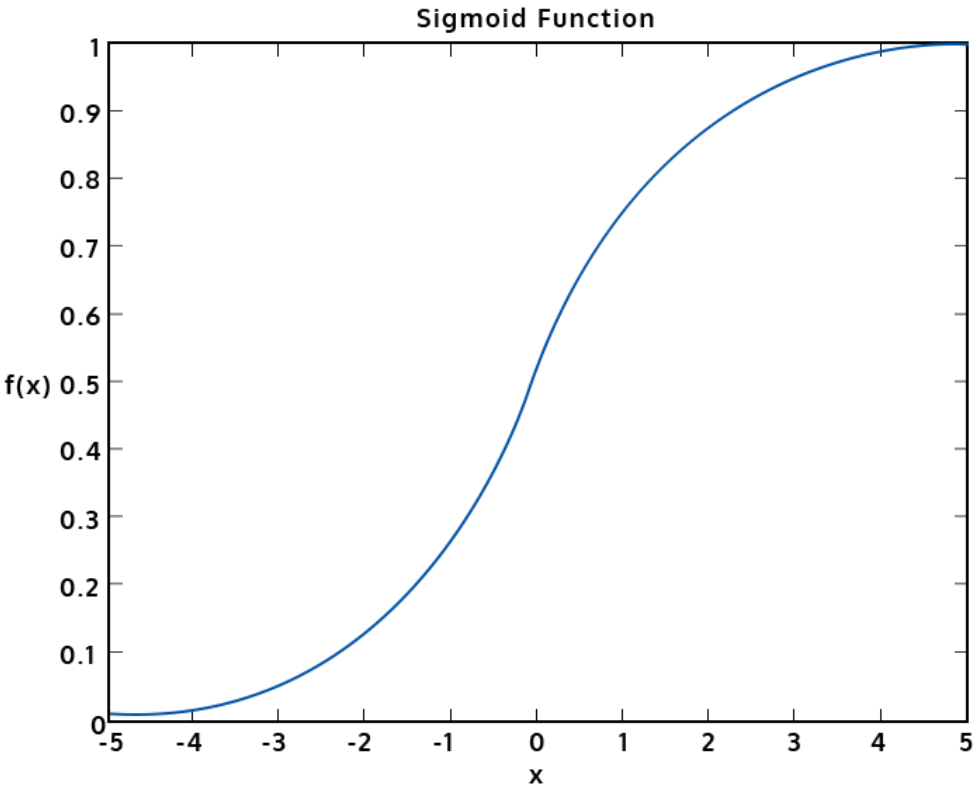
The sigmoid function is given as:

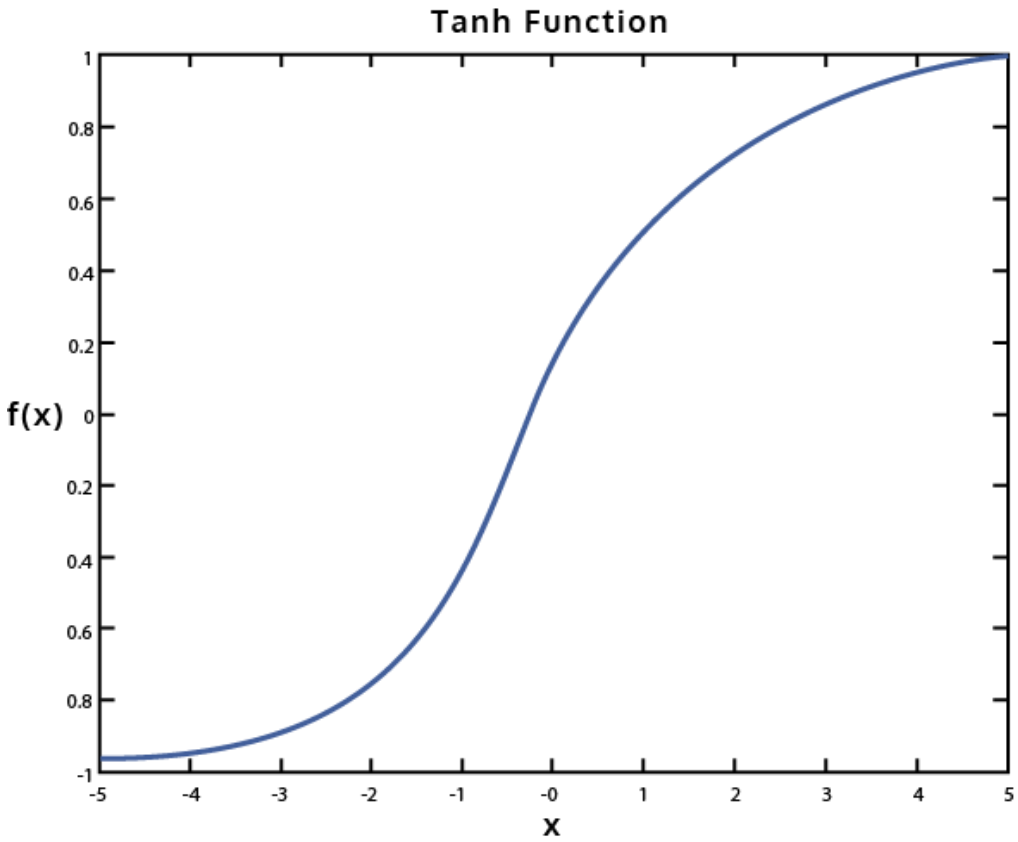
$$f(z)= \frac{1}{1+exp(-z)}$$

And the tanh function is given as:

$$f(z) = \tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

Here are plots of the sigmoid and tanh functions:





The tanh(x) function is a rescaled version of the sigmoid, and its output range is [−1, 1] instead of [0, 1].

If sigmoid function is $f(x) = 1 / (1 + e^{-x})$, then its derivative is given by $f'(x) = f(x)(1 − f(x))$.

For the tanh function, the derivative is given by $f'(x) = 1 − (f(x))^2$.)

Reasons for using tanh over sigmoid:

1. Has stronger gradients
2. Avoids bias in the gradients

For proof, please refer to the paper: http://yann.lecun.com/exdb/publis/pdf/lecun-98b.pdf