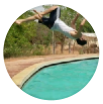
Published in Towards Data Science



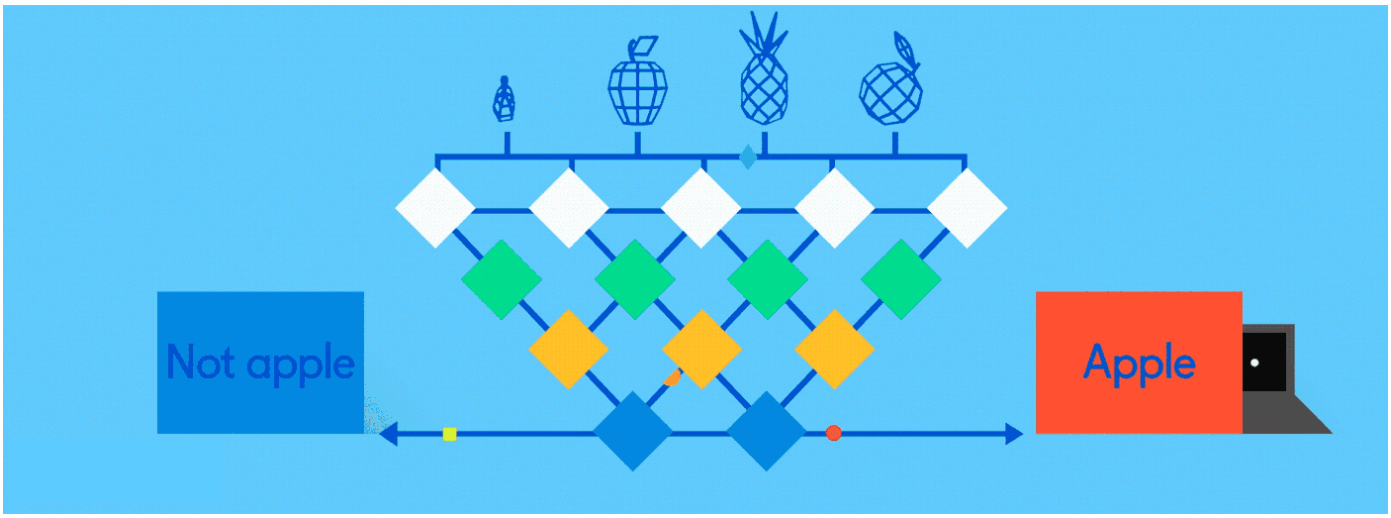
SAGAR SHARMA Follow

Sep 6, 2017 · 5 min read · Listen



# Activation Functions in Neural Networks

## Sigmoid, tanh, Softmax, ReLU, Leaky ReLU EXPLAINED !!!




### What is Activation Function?

> *It's just a thing function that you use to get the output of node. It is also known as* ***Transfer Function****.*

### Why we use Activation functions with Neural Networks?

> *It is used to determine the output of neural network like yes or no. It maps the resulting values*

1. Linear Activation Function
2. Non-linear Activation Functions

> ***FYI: The Cheat sheet is given below.***

**Linear or Identity Activation Function**

As you can see the function is a line or linear. Therefore, the output of the functions will not be confined between any range.

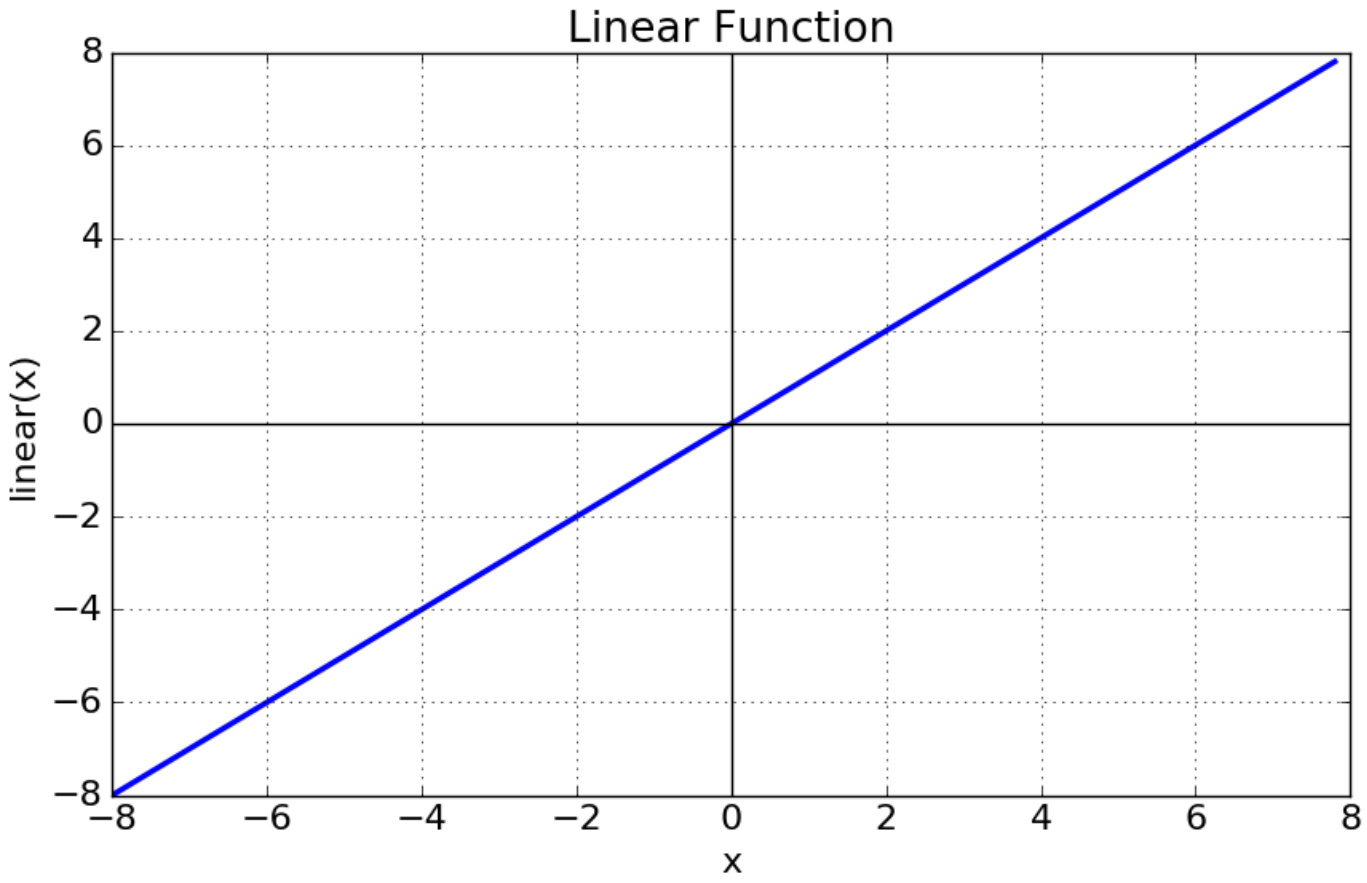



**Fig: Linear Activation Function**

**Equation :** $f(x) = x$

It doesn't help with the complexity or various parameters of usual data that is fed to the neural networks.

**Non-linear Activation Function**

The Nonlinear Activation Functions are the most used activation functions. Nonlinearity helps to makes the graph look something like this

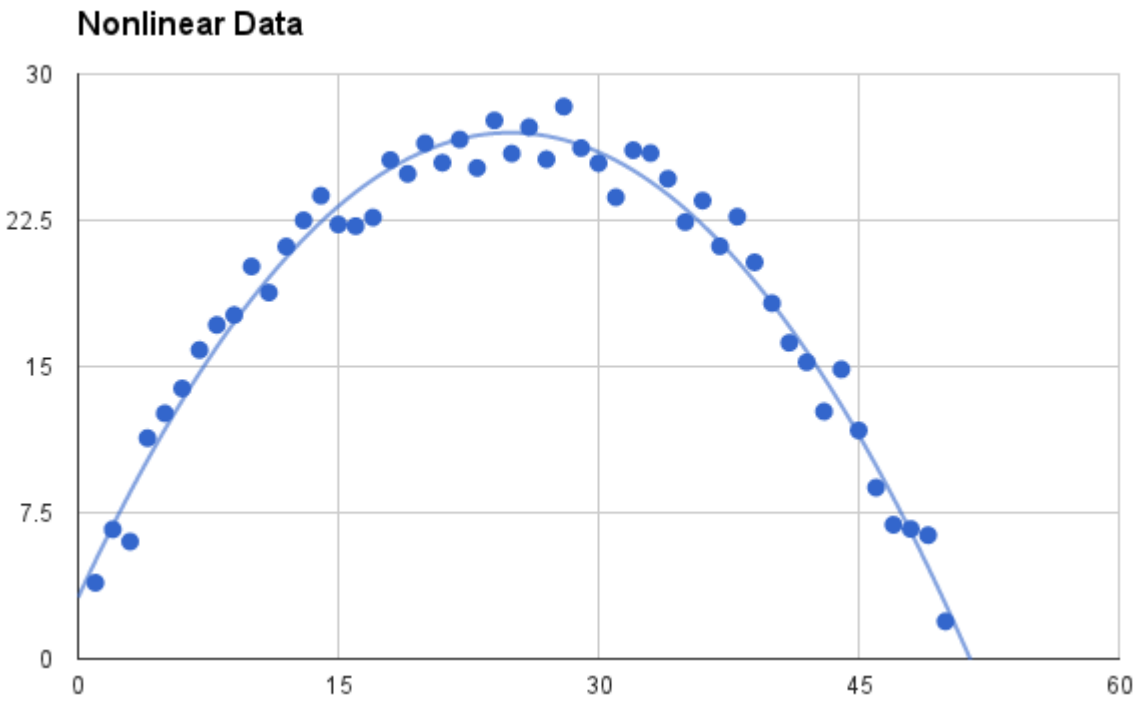



**Fig: Non-linear Activation Function**

It makes it easy for the model to generalize or adapt with variety of data and to differentiate between the output.

The main terminologies needed to understand for nonlinear functions are:

> ***Derivative or Differential:*** *Change in y-axis w.r.t. change in x-axis.It is also known as slope.*
>
> ***Monotonic function:*** *A function which is either entirely non-increasing or non-decreasing.*

The Nonlinear Activation Functions are mainly divided on the basis of their **range or**

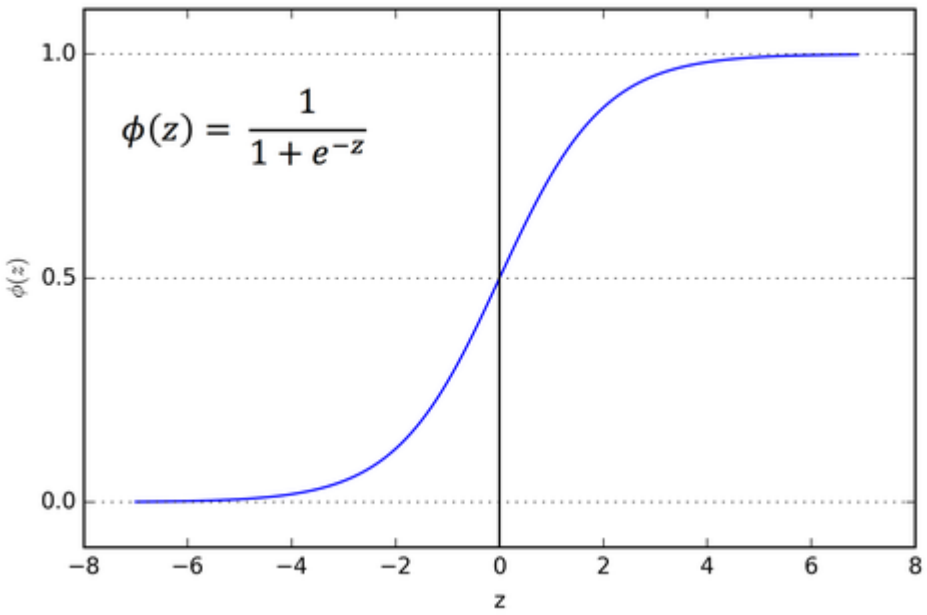



**Fig: Sigmoid Function**

The main reason why we use sigmoid function is because it exists between **(0 to 1).** Therefore, it is especially used for models where we have to **predict the probability** as an output.Since probability of anything exists only between the range of **0 and 1,** sigmoid is the right choice.

The function is **differentiable**.That means, we can find the slope of the sigmoid curve at any two points.

The function is **monotonic** but function's derivative is not.

The logistic sigmoid function can cause a neural network to get stuck at the training time.

The **softmax function** is a more generalized logistic activation function which is used for multiclass classification.

### 2. Tanh or hyperbolic tangent Activation Function

tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). tanh is also sigmoidal (s - shaped).

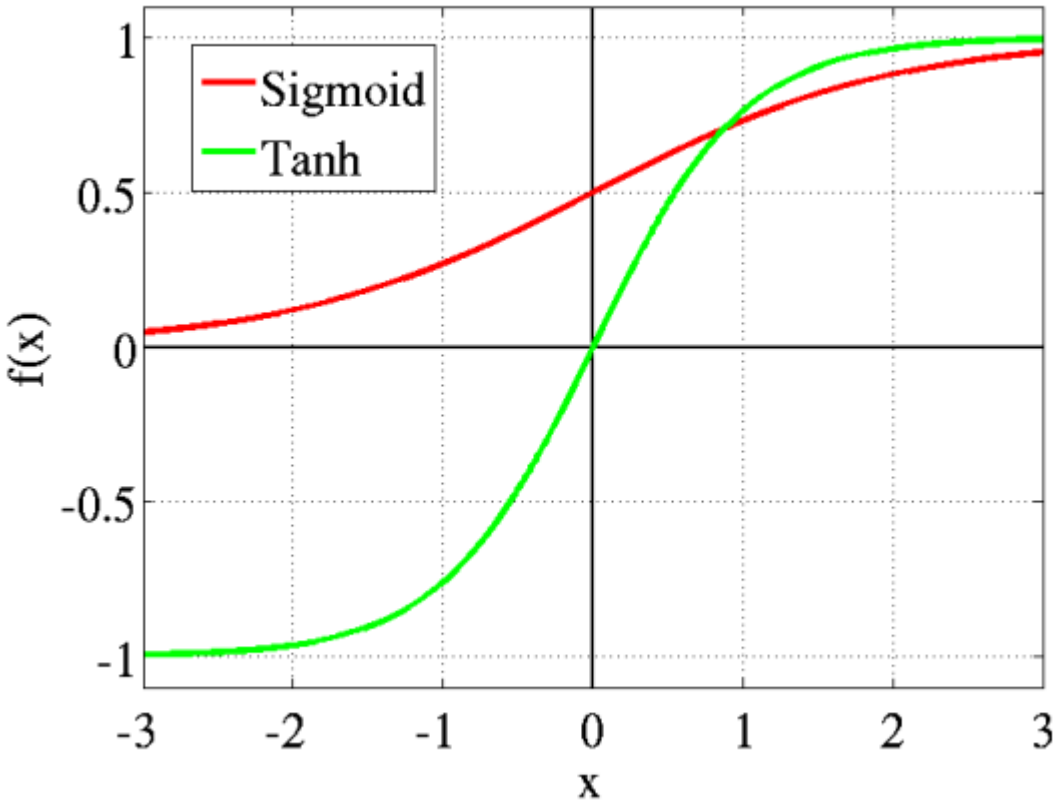



**Fig: tanh v/s Logistic Sigmoid**

The advantage is that the negative inputs will be mapped strongly negative and the zero inputs will be mapped near zero in the tanh graph.

The function is **differentiable**.

The function is **monotonic** while its **derivative is not monotonic**.

The tanh function is mainly used classification between two classes.

> *Both tanh and logistic sigmoid activation functions are used in feed-forward nets.*

### 3. ReLU (Rectified Linear Unit) Activation Function

The ReLU is the most used activation function in the world right now.Since, it is used in almost all the convolutional neural networks or deep learning.

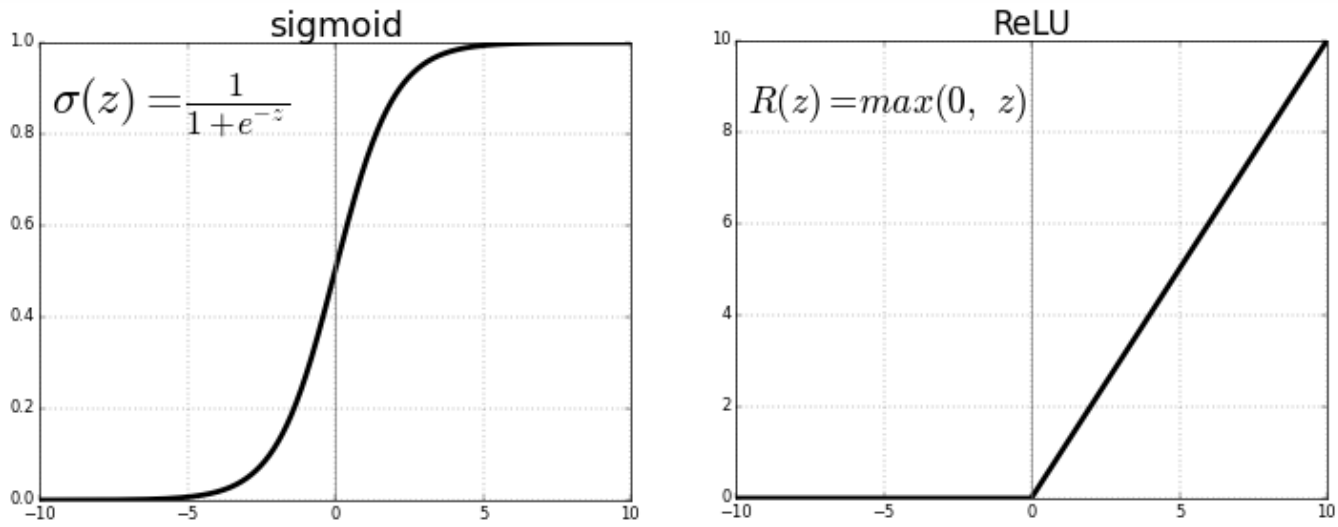



**Fig: ReLU v/s Logistic Sigmoid**

As you can see, the ReLU is half rectified (from bottom). f(z) is zero when z is less than zero and f(z) is equal to z when z is above or equal to zero.

**Range:** [ 0 to infinity)

The function and its derivative **both are monotonic**.

But the issue is that all the negative values become zero immediately which decreases the ability of the model to fit or train from the data properly. That means any negative input given to the ReLU activation function turns the value into zero immediately in the graph, which in turns affects the resulting graph by not mapping the negative values appropriately.

#### 4. Leaky ReLU

It is an attempt to solve the dying ReLU problem

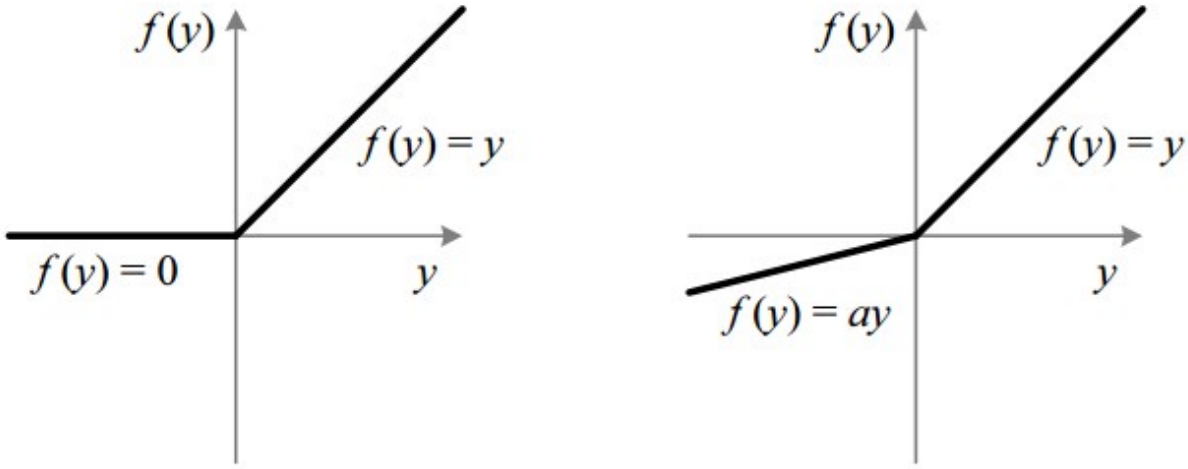



**Fig : ReLU v/s Leaky ReLU**

Can you see the Leak? 😆

The leak helps to increase the range of the ReLU function. Usually, the value of **a** is 0.01 or so.

When **a is not 0.01** then it is called **Randomized ReLU**.

Therefore the **range** of the Leaky ReLU is (-infinity to infinity).

Both Leaky and Randomized ReLU functions are monotonic in nature. Also, their derivatives also monotonic in nature.

**Why derivative/differentiation is used ?**

> When updating the curve, to know in which direction and how much to change or update the curve depending upon the slope.That is why we use differentiation in almost every part of Machine Learning and Deep Learning.



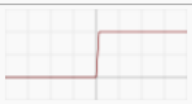
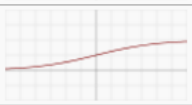
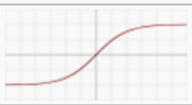
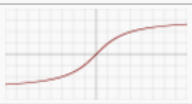
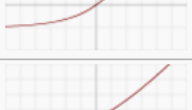

| Name | Plot | Equation | Derivative |
|---|---|---|---|
| Identity |  | $f(x) = x$ | $f'(x) = 1$ |
| Binary step |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$ |
| Logistic (a.k.a Soft step) |  | $f(x) = \frac{1}{1+e^{-x}}$ | $f'(x) = f(x)(1-f(x))$ |
| TanH |  | $f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$ | $f'(x) = 1 - f(x)^2$ |
| ArcTan |  | $f(x) = \tan^{-1}(x)$ | $f'(x) = \frac{1}{x^2+1}$ |
| Rectified Linear Unit (ReLU) |  | $f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Parameteric Rectified Linear Unit (PReLU) [2] |  | $f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| Exponential Linear Unit (ELU) [3] |  | $f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$ | $f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$ |
| SoftPlus |  | $f(x) = \log_e(1+e^x)$ | $f'(x) = \frac{1}{1+e^{-x}}$ |

**Fig: Activation Function Cheetsheet**

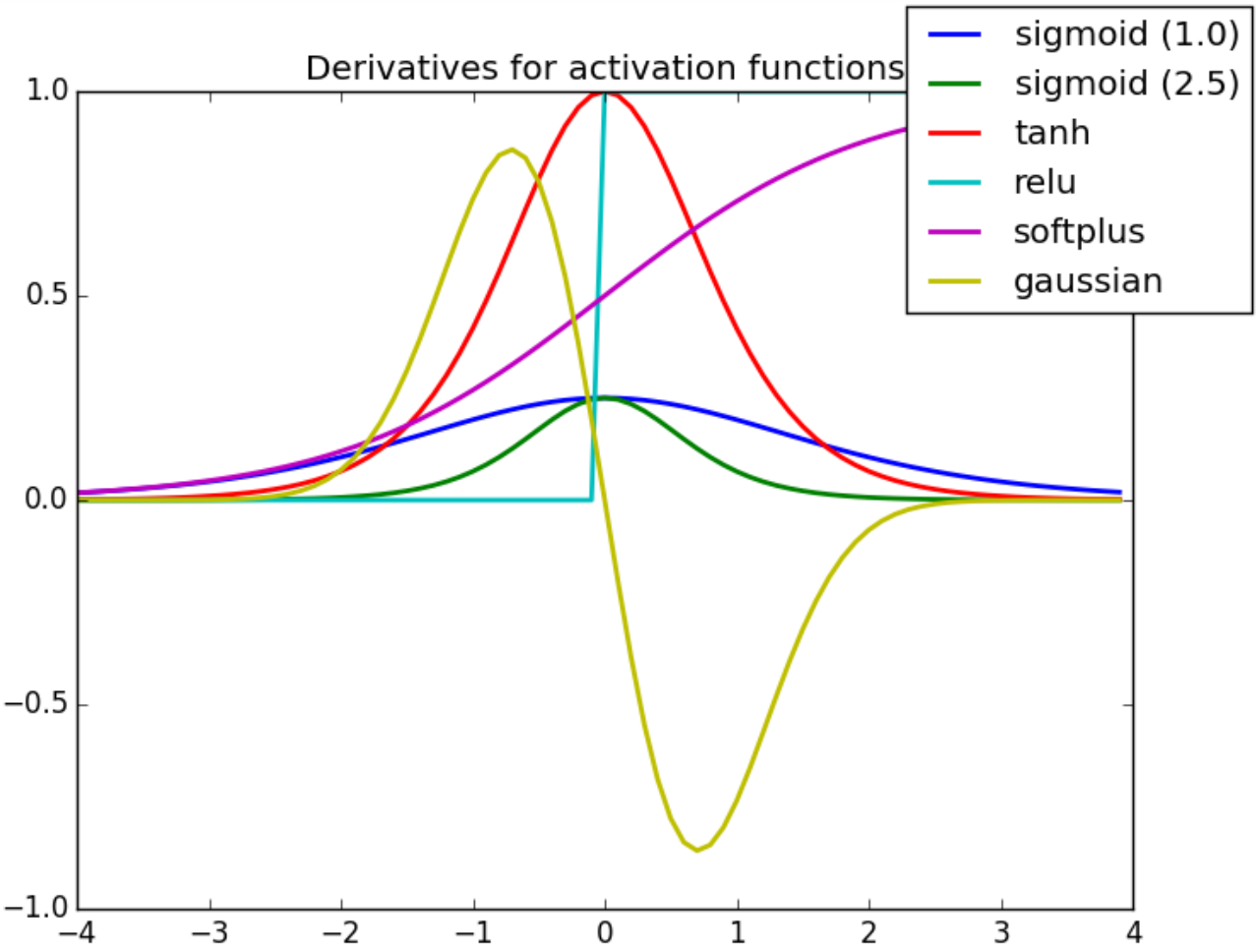



**Fig: Derivative of Activation Functions**