Activation Function in deep learning are the functions that decide the output from node or hidden layers from given set of inputs in neural network.

There are many activation function used in deep learning but among all of them Rectified Linear Unit (ReLU) is the widely used activation function in almost all deep learning neural network.

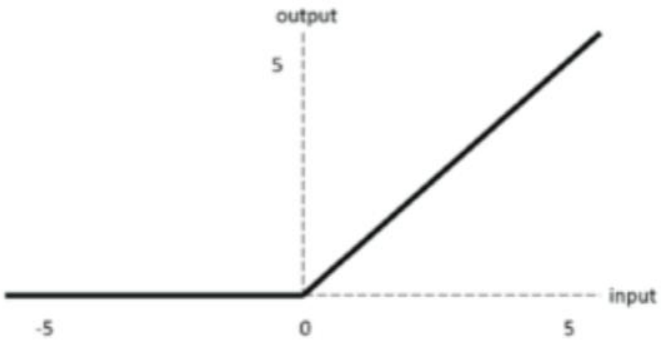
ReLU Activation Function

The full form of the term ReLU is Rectified Linear Unit. ReLU is the activation function that is most widely used in neural network architectures.

The equation for the ReLU activation function is:

> $F(x) = max(0,x)$

It tells that id input x is less than 0. Then the output from function will be 0. & if the input given to the ReLu function is greater than 0 then the output from function will be same as input.

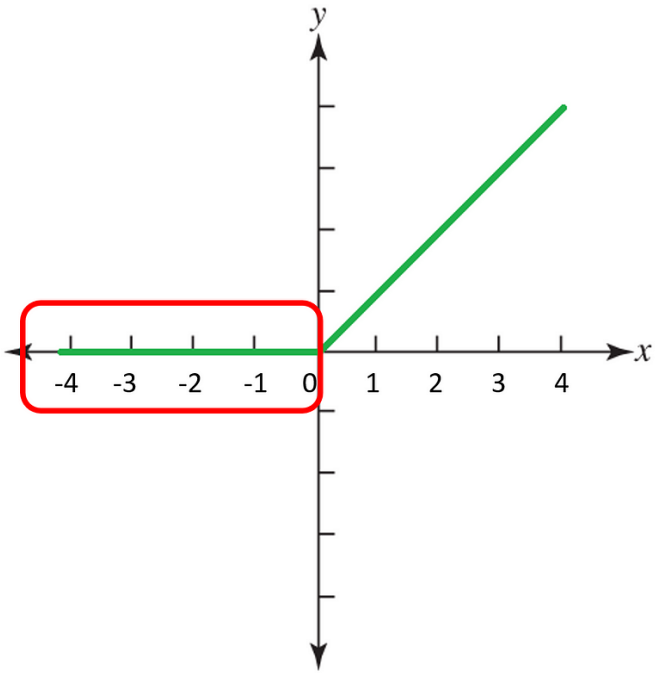


Dying ReLU Problem

Due to the formula of ReLU activation function, all the values less than 0 will be 0, after passing through ReLU function.

If the Neural network is having a lot of negative values in the dataset then they all will be converted to 0 after passing in ReLU activation Function & there will be 0 in the output which will make neurons inactive.

This type of Scenario is called Dying ReLU Problem in Neural Network.



In the above image, we can see that all of the negative values passing through the ReLU activation function will be converted to zero and there will be a straight line in the graph of the function on the negative x-axis.

Variants of ReLU

To Solve dying ReLu Problems in Neural Network there are some other variants of ReLU activation function that is developed, which do not face dying ReLU or dying neurons problem.

1.Leaky ReLU (LReLU)

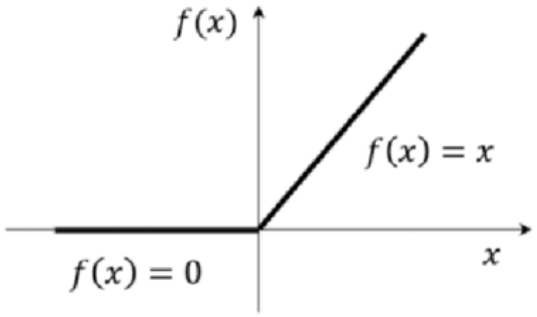
The first and easiest variant of the ReLU activation function is leaky ReLU. There is only a little change in formula of Normal Activation Function which solves the problem of Dying ReLU.

The formula for the Leaky ReLU activation function is

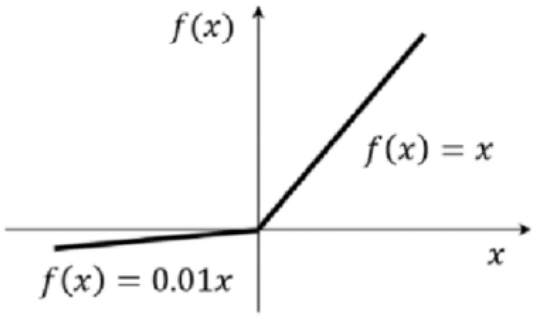
> $F(x) = max(0.01x, x)$

According to ReLU if the input passes to activation function that are less than 0, then the output value would be 0.01 times input value & if input passes to

activation function is greater than 0, then output values will be equal to input value.



ReLU activation function



LeakyReLU activation function

2.Parametric ReLU (PreLU)

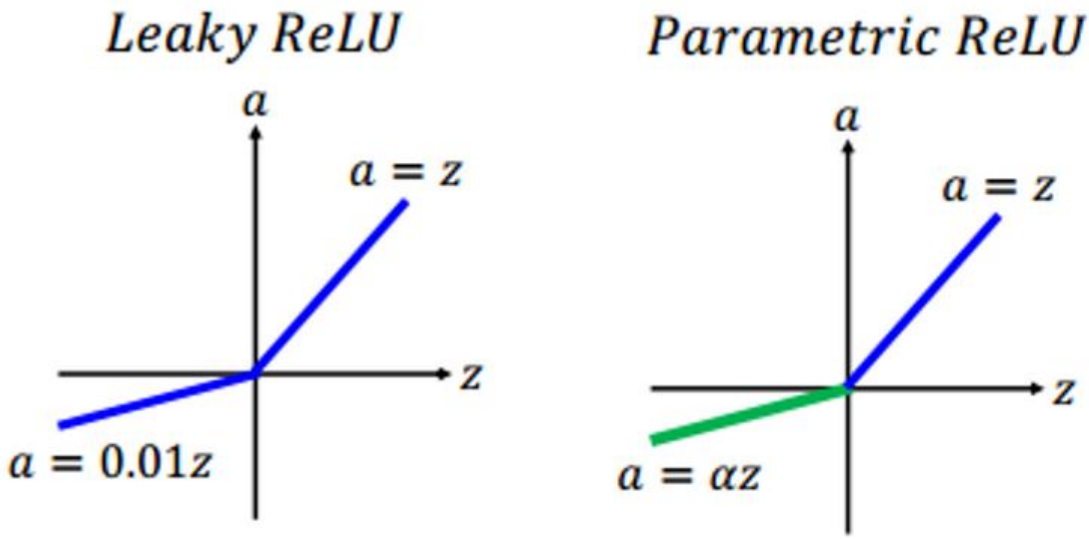
Parametric ReLU is similar to Leaky ReLU activation function, but here instead of having predetermined value for multiplying input values with 0.01, we use parameter "a" in parametric ReLU.

The Value of "a" parameter will be decided by Neural Network itself.

The formula for Parametric ReLU is

> $F(x) = max(ax, x)$

So as per Formula, if Input value given to Parametric ReLU is less than 0. Then the output values will be Equal to input values Multiplied with parameter "a" & if input value is greater than 0, then the output value will be same as input value.



3.Exponential Linear Unit (ELU)

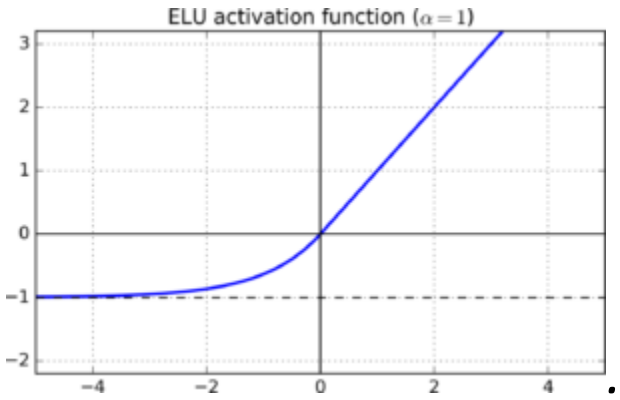
The exponential linear unit is variant of ReLU activation which fixes some problems & give some advantages compared to Normal ReLU activation function.

The formula for Exponential Linear Unit is

$$\begin{cases} x & for\ x \geqslant 0 \\ \alpha\left(e^{x}-1\right) & for\ x < 0 \end{cases}$$

ELU activation function uses a log curve and due to this it does not gives a Straight Line for Negative Values, unlike Leaky & Parametric ReLU.

ELU Activation function also solve the Dying ReLU problem and also the curve of the ELU becomes smooth slowly for negative values having high magnitude while Normal Relu becomes smooth instantly.



Higher Computational powers are required while working with ELU compared to ReLU and there is a chanceof exploding gradient problems while working with ELU.

4. Scaled Exponential Linear Unit (SELU)

This activation function is one of the latest activation function in deep learning . The equation for this activation function just looks like other equations. The equation for this activation function is

$$\operatorname{selu}(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha e^x - \alpha & \text{if } x \leqslant 0 \end{cases}.$$

The equation states that, if the input value x is greater than 0, the output value becomes x multiplied by lambda term , &

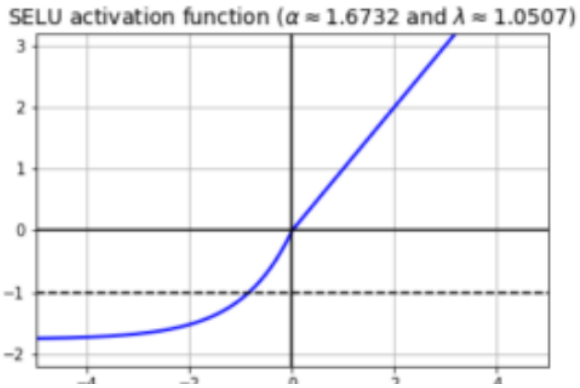
if the input value x is less than or equal to 0,

then alpha is multiplied with exponential of the x-value – alpha value,

and then we multiply by lambda value

The calculated values for alpha and lambda are

> *Alpha = 1.6732632423543772848170429916717*
> *Lambda = 1.0507009873554804934193349852946*



The Special Case in the SELU activation function is that it is self Normalizing activation function, which means that the input value will be passed through activation function and the output from function will be normalized by subtracting mean from the value anddiving it by its standard

deviation. By Doing this means Centralizing will take place and value of the mean will be zero and the S.D will be Equal to 1.

Conclusion & Key Insights

ReLu is the most widely used activation function with low computational power required for its use.

Dying ReLu is a problem associated with Normal ReLU activation function due to its nature of converting negative values to 0.

Leaky ReLu and parametric Relu are variants of Relu having Linear Negative x-Axis graph in which some value is multiplied by input value if its negative to avoid dying ReLu problem.

ELU & SELU are the Variants of normal ReLu activation function having non-linear graphs for negative x-axis value with high computation

power required and advantages like self-normalizing etc.

---

1. **ReLU (Rectified Linear Unit):**
   - ReLU is an activation function commonly used in neural networks.
   - For positive input values, it outputs the input directly.
   - However, for negative input values, it outputs zero. This can sometimes lead to "dying ReLU" problem, where neurons become inactive and stop learning.
2. **Leaky ReLU:**
   - Leaky ReLU is a variation of ReLU.
   - Instead of outputting zero for negative input values, it allows a small, positive slope for negative values.
   - This small slope helps to address the dying ReLU problem by allowing a small flow of information for negative inputs.
3. **Parametric ReLU (PReLU):**
   - Similar to Leaky ReLU, but the slope for negative values is learned during training instead of being a fixed small value.
   - This adaptability can help the network learn the most suitable slope for each neuron.
4. **ELU (Exponential Linear Unit):**
   - ELU is another variant of ReLU.
   - For negative input values, it doesn't output zero; instead, it uses an exponential function to smoothly curve down, allowing for a non-zero output for negative inputs.
   - ELU has advantages like reducing the dying ReLU problem and helping the network self-normalize.
5. **SELU (Scaled Exponential Linear Unit):**
   - SELU is an extension of ELU with a specific scaling factor.
   - It is designed to have the property of self-normalization, which means it can maintain a stable mean and variance in the activations, leading to better training performance.
   - SELU is particularly useful in deep neural networks.

In simpler terms, these variants are tweaks to the basic ReLU to handle negative inputs more effectively.

Leaky ReLU and Parametric ReLU introduce a small slope for negative values,

while ELU and SELU use more sophisticated functions to prevent the dying ReLU problem and enhance the learning capabilities of neural networks. SELU, in particular, is known for its self-normalizing properties, which can make training deep networks more stable.

Choosing the best variant often involves empirical testing. It's recommended to experiment with different activation functions and see which one performs better on your specific task and dataset. Factors such as the depth of your network, the type of data you're working with, and computational resources available can influence the choice.

In practice, Leaky ReLU or variants like Parametric ReLU are commonly used due to their simplicity and effectiveness. However, for certain scenarios, such as training very deep networks, advanced variants like ELU or SELU might be more suitable. It's always a good idea to experiment and evaluate the performance of different activation functions in your specific context.