**DIFFERENT TYPES OF ACTIVATION FUNCTION IN DEEP LEARNING**

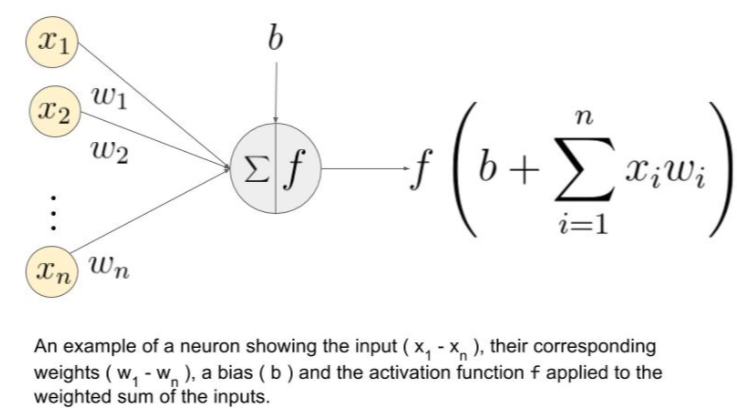
**ASSIGNMENT:** 02 **NAME:** SRIRANJANI.S **REG\_NUMBER :**122011173026

**COURSE\_NAME:** APPLIED DEEP LEARNING **COURSE\_CODE:**XAI602C

**INTRODUCTION**

* Deep learning algorithms are multi-level representation learning techniques that allow simple non-linear modules to transform representations from the raw input into the higher levels of abstract representations, with many of these trans- formations producing learned complex functions.
* The DL research was inspired by the limitations of the conventional learning algorithms especially being limited to processing data in raw form, and the human learning techniques by changing the weights of the simulated neural connections based on experiences, obtained from past data.
* The use of representation learning, which is the technique that allows machines to discover relationships from raw data, needed to perform certain tasks like classification and detection. Deep learning, a sub-field of machine learning (ML), is more recently being referred to as representation learning in some literature. The typical
* artificial neural networks (ANN) are biologically inspired computer pro- grammes, designed by the inspiration of the workings of the human brain. These ANNs are called networks because they are composed of different functions, which gathers knowledge by detecting the relationships and patterns in data using past experiences known as training examples in most literature.

The learned patterns in data are modified by an appropriate AF and presented as the output of the neuron as



**ACTIVATION FUNCTION IN NEURAL NETWORK?**

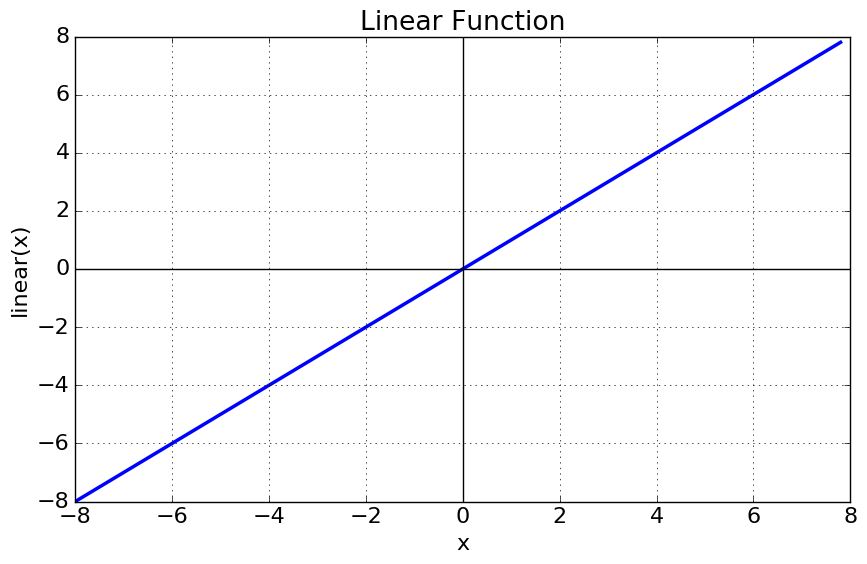
* As observed for the above figure when we do not have the activation function the weights and bias would simply do a linear transformation.
* A linear equation is simple to solve but is limited in its capacity to solve complex problems and have less power to learn complex functional mappings from data. A neural network without an activation function is just a linear regression model.
* The activation function does the non-linear transformation to the input making it capable to learn and perform more complex tasks. We would want our neural networks to work on complicated datas like *videos, audio, speech etc*. Linear transformations would never be able to perform such tasks

**WHAT CONDITION THE ACTIVATION FUNCTION SHOULD SATISFY?**

* Activation functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases. Without the differentiable non linear function, this would not be possible.
* So the functions should be differentiable and monotonic.
* **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.

**1.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*.*



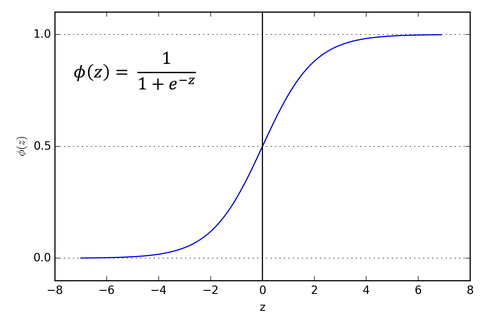
**Equation :**f(x) = x

**Range :** (-infinity to infinity)

**2.NON-LINEAR ACTIVATION FUNCTION**

* The Nonlinear Activation Functions are the most used activation functions.It makes it easy for the model to generalize or adapt with variety of data and to differentiate between the output.
* The Nonlinear Activation Functions are mainly divided on the basis of their **range or curves**-

**1. SIGMOID OR LOGISTIC ACTIVATION FUNCTION**

The Sigmoid Function curve looks like a S-shape.

**Equation :** f(x) = 1 / 1 + exp(-x)

**Range :** (0 to 1)

**Pros:**

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

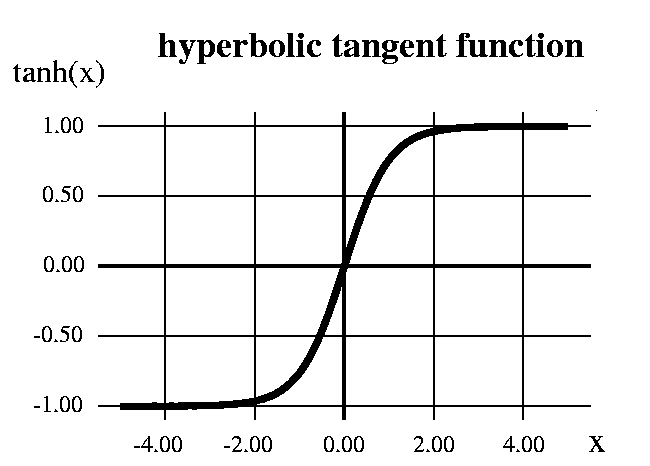
2.The function is **monotonic**but function’s derivative is not

**Cons:**

1.It gives rise to a problem of “**vanishing gradients**”, since the Y values tend to respond very less to changes in X

2.Secondly , its output isn’t zero centered. It makes the gradient updates go too far in different directions. **0 < output < 1, and it makes optimization harder.**

**2. TANH OR HYPERBOLIC TANGENT ACTIVATION FUNCTION:**



**Equation :** **f(x) = 1 — exp(-2x) / 1 + exp(-2x) or 2 \*sigmoid(2x)-1**

**Range :** (-1 to 1)

**Pros:**

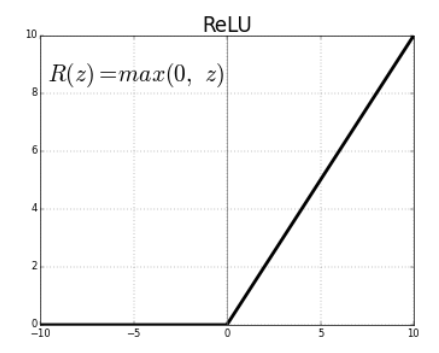
1. The function and its **derivative**both are**monotonic**
2. Output is zero centered
3. Optimization is *easier*

**Cons:**

1. It also suffers vanishing gradient problem
2. It saturate and kill gradients.

**3. RELU (RECTIFIED LINEAR UNIT) ACTIVATION FUNCTION**

The ReLU is the most used activation function in the world right now



**Equation :** **f(x) = max(0,x)**

**Range :** (0 to infinity)

**Pros:**

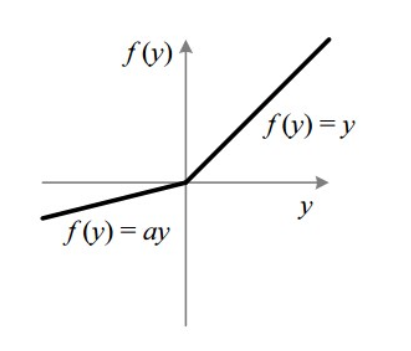
1. The function and its **derivative**both are**monotonic**.
2. Due to its functionailty it does not activate all the neuron at the same time
3. It is efficient and easy for computation.

**Cons:**

1. The outputs are not zero centered similar to the sigmoid activation function
2. When the gradient hits zero for the negative values, it does not converge towards the minima which will result in a dead neuron while back propagation.

**4. LEAKY RELU**

To solve the ReLU problem we have leaky ReLU



**Equation :** f(x) = ax for x<0 and x for x>0

**Range :** (0.01 to infinity)

**Pros:**

1. The function and its **derivative**both are**monotonic**
2. It allows negative value during back propagation
3. It is efficient and easy for computation.

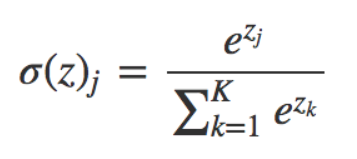
**Cons:**

1. Results are not always consistent
2. During the front propagation if the learning rate is set very high it will overshoot killing the neuron

* The idea of leaky ReLU can be extended even further. Instead of multiplying x with a constant term we can multiply it with a hyperparameter which seems to work better the leaky ReLU. This extension to leaky ReLU is known as **Parametric ReLU**.

**5. SOFTMAX**

* The softmax function is also a type of sigmoid function but it is very useful to handle classification problems having multiple classes .



* The softmax function is shown above, where z is a vector of the inputs to the output layer (if you have 10 output units, then there are 10 elements in z). And again, j indexes the output units, so j = 1, 2, …, K.
* The softmax function is ideally used in the output layer of the classifier where we are actually trying to attain the probabilities to define the class of each input.

**REFERENCE**

[Topic DL01: Activation functions and its Types in Artifical Neural network | by abhigoku10 | Medium](https://abhigoku10.medium.com/activation-functions-and-its-types-in-artifical-neural-network-14511f3080a8)