PERIYAR MANIAMMAI
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University)
Established Under Sec. 3 of UGC Act, 1956 • NAAC Accredited
think • innovate • transform

DATA SCIENCE

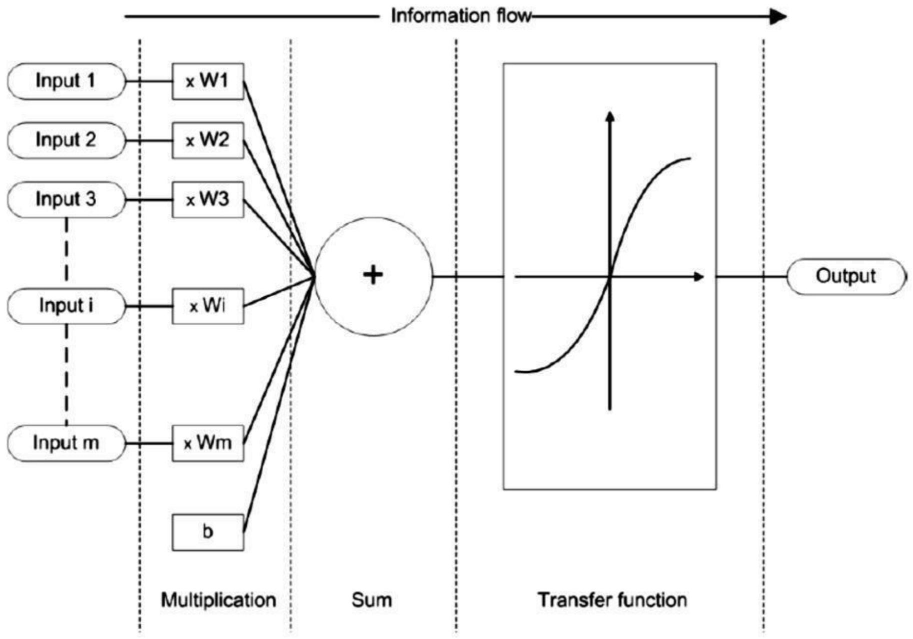
ASSIGNMENT - 1

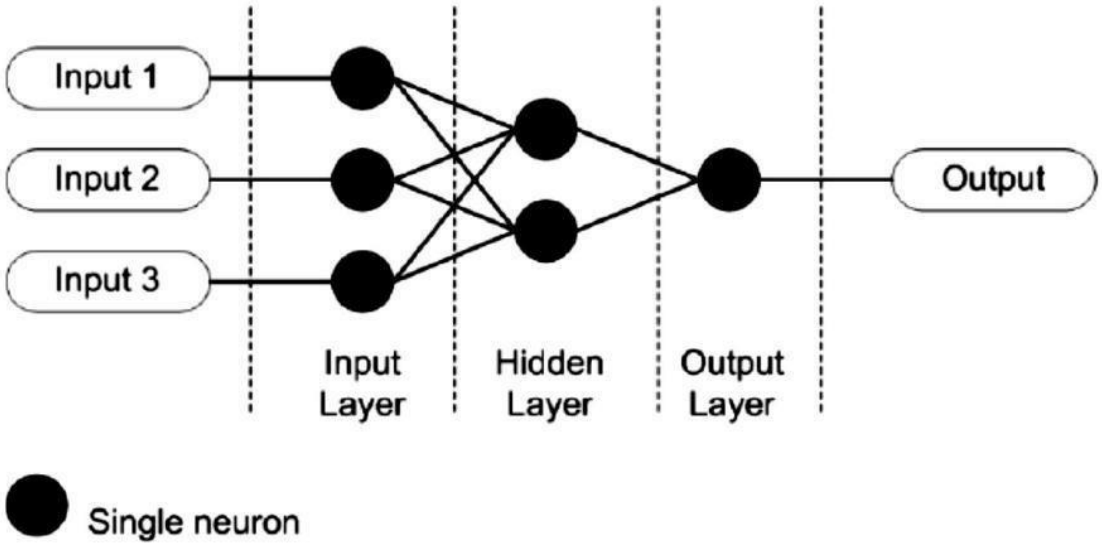
NAME	: SIVAPRIYAN.K
REGISTER NO	: 121012012768
DEGREE	: B.TECH
BRANCH	: CSE –3rd YEAR
SEMESTER	: VI
SPECIALIZATION	: DATA SCIENCE
SUBJECT	: DEEP LEARNING
SUB.CODE	: XCSHD04

NEURAL NETWORKS

WHAT IS ARTIFICIAL NEURAL NETWORK?

An Artificial Neural Network (ANN) is a mathematical model that tries to simulate the structure and functionalities of biological neural networks. Basic building block of every artificial neural network is artificial neuron, that is, a simple mathematical model (function). Such a model has three simple sets of rules: multiplication, summation and activation. At the entrance of artificial neuron the inputs are weighted what means that every input value is multiplied with individual weight. In the middle section of artificial neuron is sum function that sums all weighted inputs and bias. At the exit of artificial neuron the sum of previously weighted inputs and bias is passing through activation function that is also called transfer function.





BRIEFLY EXPLAIN THE BASIC BUILDING BLOCKS OF ARTIFICIAL NEURAL NETWORKS.

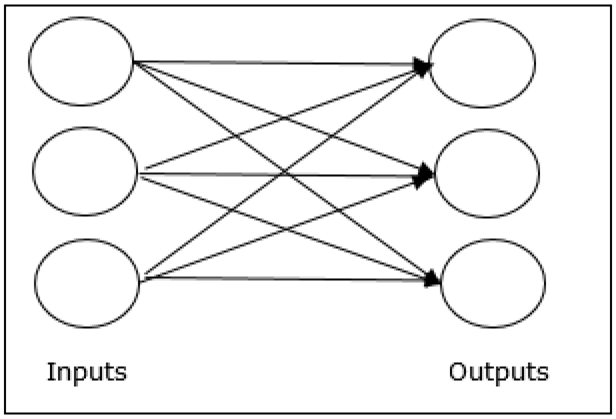
Processing of ANN depends upon the following three building blocks:

1. Network Topology
2. Adjustments of Weights or Learning
3. Activation Functions

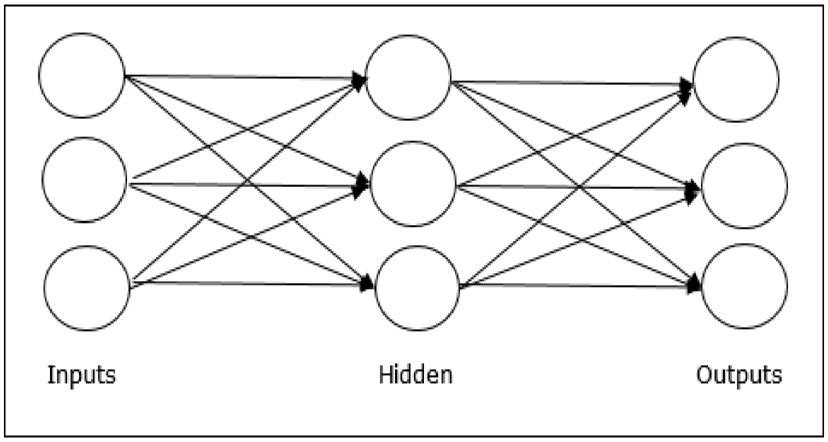
1. Network Topology: A network topology is the arrangement of a network along with its nodes and connecting lines. According to the topology, ANN can be classified as the following kinds:

A. Feed forward Network: It is a non-recurrent network having processing units/nodes in layers and all the nodes in a layer are connected with the nodes of the previous layers. The connection has different weights upon them. There is no feedback loop means the signal can only flow in one direction, from input to output. It may be divided into the following two types:

- **Single layer feed forward network:** The concept is of feed forward ANN having only one weighted layer. In other words, we can say the input layer is fully connected to the output layer.

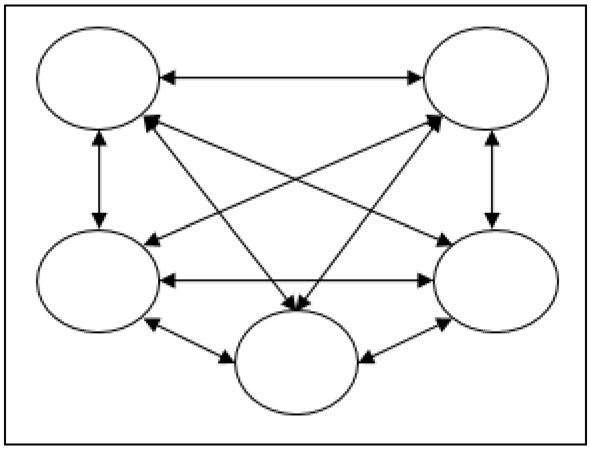


- **Multilayer feed forward network:** The concept is of feed forward ANN having more than one weighted layer. As this network has one or more layers between the input and the output layer, it is called hidden layers.

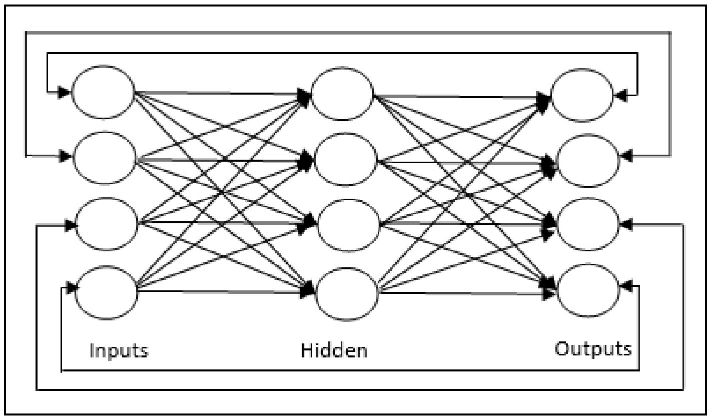


B. Feedback Network: As the name suggests, a feedback network has feedback paths, which means the signal can flow in both directions using loops. This makes it a non-linear dynamic system, which changes continuously until it reaches a state of equilibrium. It may be divided into the following types:

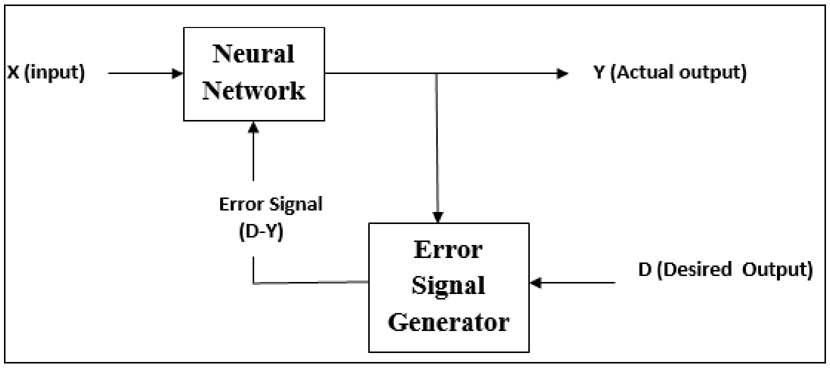
- **Recurrent networks:** They are feedback networks with closed loops. Following are the two types of recurrent networks.
- **Fully recurrent network:** It is the simplest neural network architecture because all nodes are connected to all other nodes and each node works as both input and output.



- **Jordan network** – It is a closed loop network in which the output will go to the input again as feedback as shown in the following diagram.

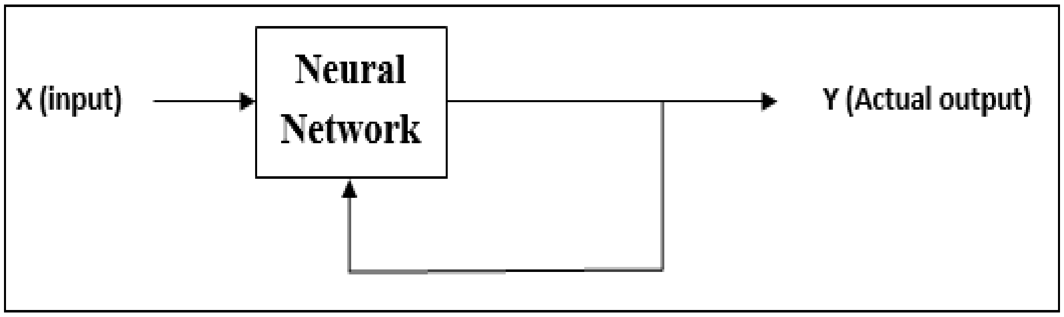


2. Adjustments of Weights or Learning: Learning, in artificial neural network, is the method of modifying the weights of connections between the neurons of a specified network. Learning in ANN can be classified into three categories namely supervised learning, unsupervised learning, and reinforcement learning.

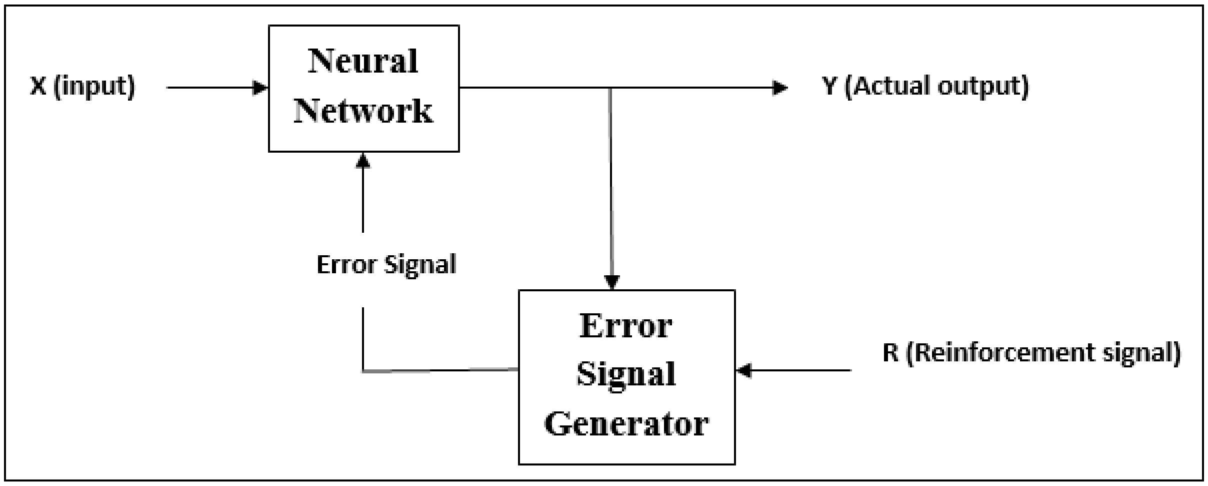


Supervised Learning: As the name suggests, this type of learning is done under the supervision of a teacher. This learning process is dependent. During the training of ANN under supervised learning, the input vector is presented to the network, which will give an output vector. This output vector is compared with the desired output vector. An error signal is generated, if there is a difference between the actual output and the desired output vector. On the basis of this error signal, the weights are adjusted until the actual output is matched with the desired output.

Unsupervised Learning: As the name suggests, this type of learning is done without the supervision of a teacher. This learning process is independent. During the training of ANN under unsupervised learning, the input vectors of similar type are combined to form clusters. When a new input pattern is applied, then the neural network gives an output response indicating the class to which the input pattern belongs. There is no feedback from the environment as to what should be the desired output and if it is correct or incorrect. Hence, in this type of learning, the network itself must discover the patterns and features from the input data, and the relation for the input data over the output.



Reinforcement Learning: As the name suggests, this type of learning is used to reinforce or strengthen the network over some critic information. This learning process is similar to supervised learning, however we might have very less information. During the training of network under reinforcement learning, the network receives some feedback from the environment. This makes it somewhat similar to supervised learning. However, the feedback obtained here is evaluative not instructive, which means there is no teacher as in supervised learning. After receiving the feedback, the network performs adjustments of the weights to get better critic information in future.



3. **Activation Functions:** An activation function is a mathematical equation that determines the output of each element (perceptron or neuron) in the neural network. It takes in the input from each neuron and transforms it into an output, usually between one and zero or between -1 and one. It may be defined as the extra force or effort applied over the input to obtain an exact output. In ANN, we can also apply activation functions over the input to get the exact output. Followings are some activation functions of interest:

i) Linear Activation Function: It is also called the identity function as it performs no input editing. It can be defined as: $F(x) = x$

ii) Sigmoid Activation Function: It is of two type as follows –

- **Binary sigmoidal function:** This activation function performs input editing between 0 and 1. It is positive in nature. It is always bounded, which means its output cannot be less than 0 and more than 1. It is also strictly increasing in nature, which means more the input higher would be the output. It can be defined as

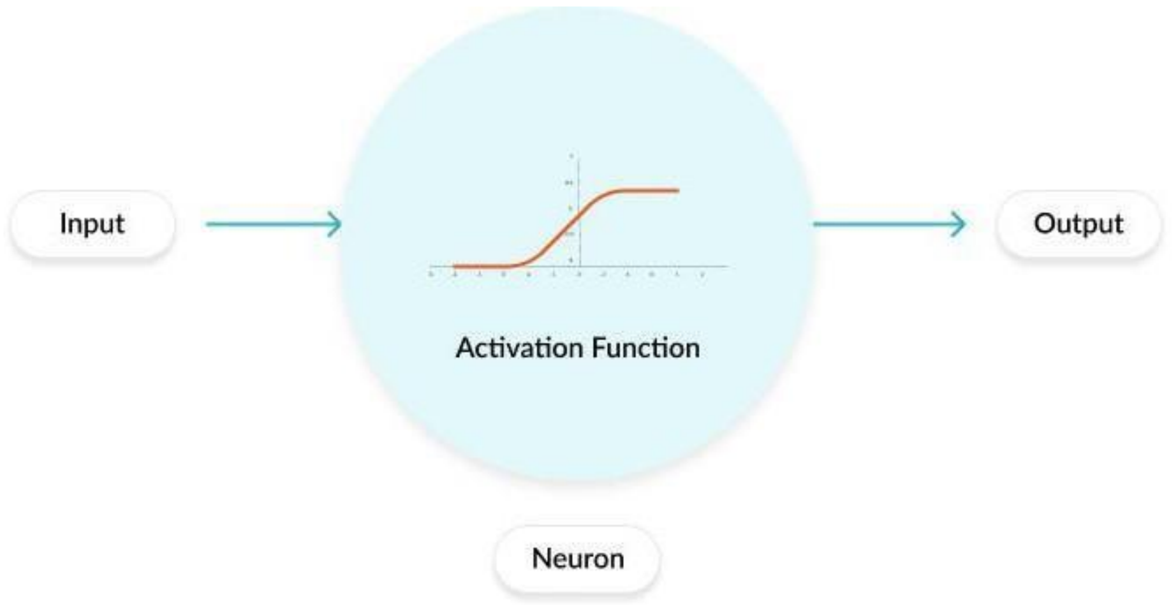
$$F(x)=sigm(x)=11+exp(-x)F(x)=sigm(x)=11+exp(-x)$$

- **Bipolar sigmoidal function:** This activation function performs input editing between -1 and 1. It can be positive or negative in nature. It is always bounded, which means its output cannot be less than -1 and more than 1. It is also strictly increasing in nature like sigmoid function. It can be defined as

$$F(x)=sigm(x)=21+exp(-x)-1=1-exp(x)1+exp(x)$$

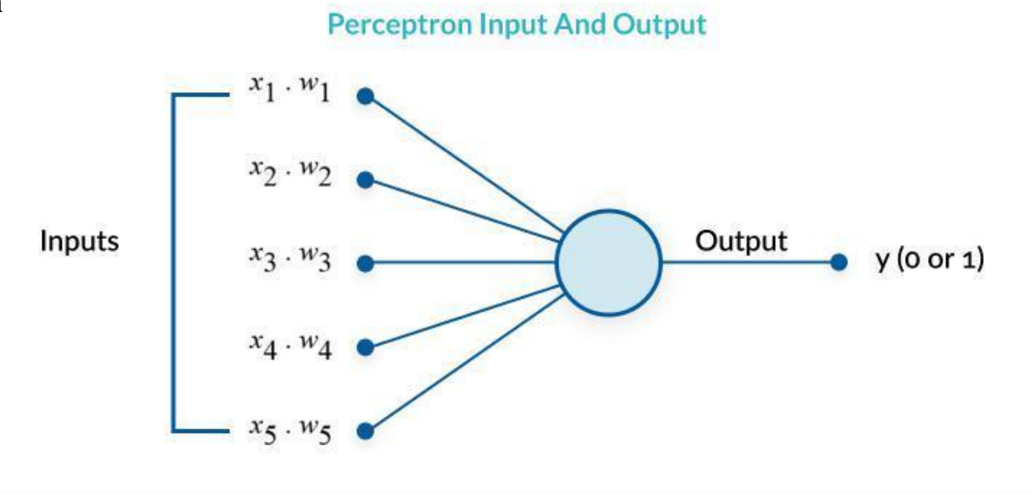
WHAT IS A NEURAL NETWORK ACTIVATION FUNCTION?

In a neural network, inputs, which are typically real values, are fed into the neurons in the network. Each neuron has a weight, and the inputs are multiplied by the weight and fed into the activation function. Each neuron's output is the input of the neurons in the next layer of the network, and so the inputs cascade through multiple activation functions until eventually, the output layer generates a prediction. Neural networks rely on nonlinear activation functions—the derivative of the activation function helps the network learn through the backpropagation process.



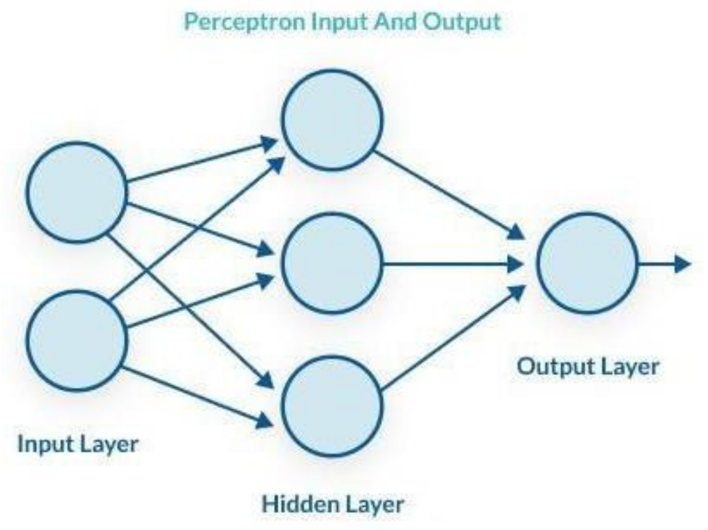
WHAT IS A PERCEPTRON?

A perceptron is a binary classification algorithm modeled after the functioning of the human brain—it was intended to emulate the neuron. The perceptron, while it has a simple structure, has the ability to learn a



What is Multilayer Perceptron?

A multilayer perceptron (MLP) is a group of perceptrons, organized in multiple layers, that can accurately answer complex questions. Each perceptron in the first layer (on the left) sends signals to all the perceptrons in the second layer, and so on. An MLP contains an input layer, at least one hidden layer, and an output layer.



The perceptron learns as follows:

1. Takes the inputs which are fed into the perceptrons in the input layer, multiplies them by their weights, and computes the sum.
2. Adds the number one, multiplied by a “bias weight”. This is a technical step that makes it possible to move the output function of each perceptron (the activation function) up, down, left and right on the number graph.
3. Feeds the sum through the activation function—in a simple perceptron system, the activation function is a step function.
4. The result of the step function is the output.

A multilayer perceptron is quite similar to a modern neural network. By adding a few ingredients, the perceptron architecture becomes a full-fledged deep learning system:

- **Activation functions and other** hyperparameters**:** a full neural network uses a variety of activation functions which output real values, not boolean values like in the classic perceptron. It is more flexible in terms of other details of the learning process, such as the number of training iterations (iterations and epochs), weight initialization schemes, regularization, and so on. All these can be tuned as hyperparameters.
- **Backpropagation**: a full neural network uses the backpropagation algorithm, to perform iterative backward passes which try to find the optimal values of perceptron weights, to generate the most accurate prediction.
- **Advanced architectures**: full neural networks can have a variety of architectures that can help solve specific problems. A few examples are Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), and Generative Adversarial Networks (GAN).

Types of Activation Functions in Neural Networks

1. Sigmoid Function

The first type of activation function in neural network is the sigmoid function. The sigmoid function is a non-linear AF used primarily in feedforward neural networks. It is a differentiable real function, defined for real input values, and containing positive derivatives everywhere with a specific degree of smoothness. The sigmoid function appears in the output layer of the deep learning models and is used for predicting probability-based outputs. The sigmoid function is represented as:

$$f(x) = \left(\frac{1}{(1 + exp^{-x})} \right) - (1.4)$$

Source

Generally, the derivatives of the sigmoid function are applied to learning algorithms. The graph of the sigmoid function is 'S' shaped.

Some of the major drawbacks of the sigmoid function include gradient saturation, slow convergence, sharp damp gradients during backpropagation from within deeper hidden layers to the input layers, and non-zero centered output that causes the gradient updates to propagate in varying directions.

2. Hyperbolic Tangent Function (Tanh)

The hyperbolic tangent function, a.k.a., the tanh function, is another type of AF. It is a smoother, zero-centered function having a range between -1 to 1. As a result, the output of the tanh function is represented by:

$$f(x_i) = \frac{exp\ (x_i)}{\sum_j exp\ (x_j)} \quad - (1.12)$$

Source

The tanh function is much more extensively used than the sigmoid function since it delivers better training performance for multilayer neural networks. The biggest advantage of the tanh function is that it produces a zero-centered output, thereby supporting the backpropagation process. The tanh function has been mostly used in recurrent neural networks for natural language processing and speech recognition tasks.

However, the tanh function, too, has a limitation – just like the sigmoid function, it cannot solve the vanishing gradient problem. Also, the tanh function can only attain a gradient of 1 when the input value is 0 (x is zero). As a result, the function can produce some dead neurons during the computation process.

3. Softmax Function

The softmax function is another type of AF used in neural networks to compute probability distribution from a vector of real numbers. This function generates an output that ranges between values 0 and 1 and with the sum of the probabilities being equal to 1. The softmax function is represented as follows:

$$f(x_i) = \frac{exp\ (x_i)}{\sum_j exp\ (x_j)} \quad -(1.12)$$

Source

This function is mainly used in multi-class models where it returns probabilities of each class, with the target class having the highest probability. It appears in almost all the output layers of the DL architecture where they are used. The primary difference between the sigmoid and softmax AF is that while the former is used in binary classification, the latter is used for multivariate classification.

4. Softsign Function

The softsign function is another AF that is used in neural network computing. Although it is primarily in regression computation problems, nowadays it is also used in DL based text-to-speech applications. It is a quadratic polynomial, represented by:

$$f(x) = \left(\frac{x}{|x|\ +\ 1}\right) \quad -(1.13)$$

Source

Here "x" equals the absolute value of the input.

The main difference between the softsign function and the tanh function is that unlike the tanh function that converges exponentially, the softsign function converges in a polynomial form.

5. Rectified Linear Unit (ReLU) Function

One of the most popular AFs in DL models, the rectified linear unit (ReLU) function, is a fast-learning AF that promises to deliver state-of-the-art performance with stellar results. Compared to other AFs like the sigmoid and tanh functions, the ReLU function offers much better performance and generalization in deep learning. The function is a nearly linear function that retains the properties of linear models, which makes them easy to optimize with gradient-descent methods.

The ReLU function performs a threshold operation on each input element where all values less than zero are set to zero. Thus, the ReLU is represented as:

$$f(x) = max(0, x) = \begin{cases} x_i, & if\ x_i \geq 0 \\ 0, & if\ x_i < 0 \end{cases} \quad -(1.14)$$

Source

By rectifying the values of the inputs less than zero and setting them to zero, this function eliminates the vanishing gradient problem observed in the earlier types of **activation function in soft computing** (sigmoid and tanh).

The most significant advantage of using the ReLU function in computation is that it guarantees faster computation – it does not compute exponentials and divisions, thereby boosting the overall computation speed. Another critical aspect of the ReLU function is that it introduces sparsity in the hidden units by squishing the values between zero to maximum.

6. Exponential Linear Units (ELUs) Function

The exponential linear units (ELUs) function is an AF that is also used to speed up the training of neural networks (just like ReLU function). The biggest advantage of the ELU function is that it can eliminate the vanishing gradient problem by using identity for positive values and by improving the learning characteristics of the model.

ELUs have negative values that push the mean unit activation closer to zero, thereby reducing computational complexity and improving the learning speed. The ELU is an excellent alternative to the ReLU – it decreases bias shifts by pushing mean activation towards zero during the training process.

The exponential linear unit function is represented as:

$$f(x) = \begin{pmatrix} x, & if\ x\ >\ 0 \\ \alpha\ exp(x)\ -\ 1, & if\ x \leq 0 \end{pmatrix}\ -(1.22)$$

The derivative or gradient of the ELU equation is presented as:

$$f' = \begin{pmatrix} 1, & if\ x\ >\ 0 \\ f(x)\ +\ \alpha, & if\ x\ \leq\ 0 \end{pmatrix}\ -(1.23)$$

Source

Here “α” equals the ELU hyperparameter that controls the saturation point for negative net inputs, which is usually set to 1.0. However, the ELU function has a limitation – it is not zero-centered.