CHAPTER FOUR

ARTIFICIAL NEURAL NETWORKS (ANNs)

4.1 Introduction

This chapter presents the main characteristics of the ANN, their capabilities as a powerful modeling technique for complex systems and their applications in civil engineering field.

Artificial Neural Networks (ANN_S) are a network of artificial neurons, an information processing units, are inspired by the way in which human brain performs a particular task or function of interest. Neural networks are a computational method inspired by studies of the brain and nervous systems in biological organism. Artificial Neural Networks represent highly ideological mathematical models of our present understanding of such complex systems. Artificial Neural Networks models have the ability to learn and generalize the problems even when input data contain error or incomplete.

4.2 Artificial Neural Networks Background

An ANN is a computational structure that is inspired by observed process in natural networks of biological neurons in the brain. It consists of simple computational units called neurons, which are highly interconnected. ANNs have become the focus of much attention, largely because of their wide range of applicability and the ease with which they can treat complicated problems. ANNs are parallel computational models comprised of densely interconnected adaptive processing units. These networks are fine-grained parallel implementations of nonlinear static or dynamic systems. A very important feature of these networks is

their adaptive nature, where “learning by example” replaces “programming” in solving problems. This feature makes such computational models very appealing in application domains where one has little or incomplete understanding of the problem to be solved but where training data is readily available. ANNs are now being increasingly recognized in the area of classification and prediction, where regression model and other related statistical techniques have traditionally been employed. The most widely used learning algorithm in an ANNs is the Back propagation algorithm. There are various types of ANNs like Multilayered Perceptron, “Radial” Basis Function and “Kohonen” networks.

These networks are “neural” in the sense that they may have been inspired by neuroscience but not necessarily because they are faithful models of biological neural or cognitive phenomena. In fact majority of the network are more closely related to traditional mathematical and/or statistical models such as non-parametric pattern classifiers, clustering algorithms, nonlinear filters, and statistical regression models than they are to neurobiology models.

ANNs have been used for a wide variety of applications where statistical methods are traditionally employed. The problems which were normally solved through classical statistical methods, such as discriminant analysis, logistic regression, Bayes analysis, multiple regression, and ARIMA time-series models are being tackled by ANNs. It is, therefore, time to recognize ANNs as a powerful tool for data analysis.

4.3 The Human Neural Network vs. Artificial Neural Networks

Although ANNs try to simulate some characteristics of the human nervous system, their real behavior is quite different.

A human neural network is made of billions of nerve cells, or neurons. Each neuron has dendrites (inputs), a cell body and an axon (output) as can be seen in Figure. (4.1). A neuron receives inputs from other neurons and if the 'average' input exceeds a critical level, the neuron discharges an electrical pulse that travels from the body down the axon to the next neuron or group of neurons. These three main parts of a biological neuron are reproduced in an artificial neuron. It receives inputs (dendrites), operates in reaction to them (body cell), and sends an output (axon).

The human brain contains about 10 billion neurons. Each neuron is connected to some thousand other neurons, forming a massively parallel information processing system. On the other hand, conventional computers contain a processor that executes a single series of instructions. In fact, the brain is built with very slow 'hardware', as neurons typically operate at a maximum rate of about 100 Hz (Schraudolph and Cummins, 2001), in marked contrast to a conventional CPU, which is able to carry out several hundred million operations per second.

Though a parallel processing system together with a very high number of connections offers capabilities remarkably similar to the human brain (Schraudolph and Cummins, 2001), it cannot replicate exactly the following characteristics:

- Under partial damage the performance of the brain tends to degrade gracefully. In contrast, computers usually cease to function when a small part is damaged.
- It can learn from experience (reorganizing itself).
- It can partially recover from damage if healthy neurons learn to take over the functions previously carried out by damaged areas.

- It can perform massively parallel computations extremely efficiently. For instance, researchers say that complex visual perception occurs within less than 100 ms, that is, less than 10 processing steps.

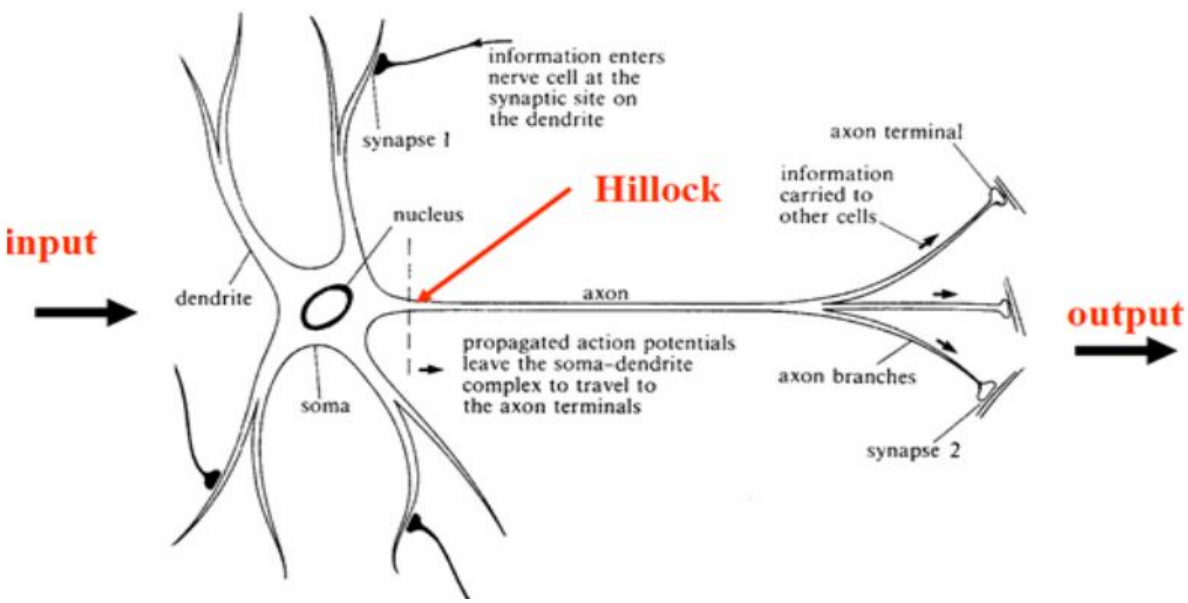

Figure (4.1): Structure of a Neural Cell in the Human Brain (Chakraborty, 2010).

The learning process is also highly simplified in Artificial Neural Networks. The brain learns online, based on experience, and normally without supervision. During this process, the strength of connections between neurons changes, and some connections are added or deleted. Learning in Artificial Neural Networks is based on adjusting the weights of previously established internal connections to satisfactorily reproduce a training pattern of data. The learning process is generally controlled by the computer user.

4.4 History of Neural Networks

Pitts (1943) proposed a model of "computing elements" called Culloch– Pitts neurons, which performs weighted sum of the inputs to these elements followed by

a threshold logic operation. Combinations of these computing elements were used to realize several logical computations. The main drawback of this model of computation is that the weights are fixed and hence the models could not learn from examples which are the main characteristic of the ANN technique which later evolved. (Rosenblatt, 1958) proposed the perceptron models, which has weight adjustable by the perceptron learning law. (Windrows and Hoff, 1960) and his group proposed an ADALINE (Adaptive Linear Element) model for computing elements and LMS (Least Mean Square) learning algorithm to adjust the weights of an ADALINE model. (Hopfield, 1982) gave energy analysis of feedback neural networks. The analysis has shown the existence of stable equilibrium states in a feedback network, provided the network has symmetrical weights. (Rumelhart, 1986) showed that it is possible to adjust the weight of a multilayer feed forward neural network in a systematic way to learn the implicit mapping in a set of input – output patterns pairs. The learning law is called generalized delta rule or error back propagation.

An excellent overview of various aspects of ANNs are provided. (Kaastra and Boyd, 1996) developed neural network model for forecasting financial and economic time series. (Dewolf, 2000) demonstrated the applicability of neural network technology for plant diseases forecasting (Zhang et al, 1998) provided the general summary of the work in ANNs forecasting, providing the guidelines for neural network modeling, general model of the ANNs especially those used for forecasting, modeling issue of ANNs in forecasting and relative performance of ANNs over traditional statistical methods. (Sanzogni et al, 2001) developed the models for predicting milk production from farm inputs using standard feed forward ANN. (Gaudart et al. 2004) compared the performance of multi-layer

perceptron MLP and that of linear regression for epidemiological data with regard to quality of prediction and robustness to deviate from underlying assumptions of normality, homoscedasticity and independence of errors. More general books on neural networks and related topics contain separate chapters/sections on neural networks, to cite a few, (Hassoun, 1995). Softwares on neural networks have also been made and are as follows:

Commercial Software: - Statistica Neural Network, TNs2Server, DataEngine, Know Man Basic Suite, Partek, Saxon, ECANSE - Environment for Computer Aided Neural Software Engineering, Neuroshell, Neurogen, Matlab:Neural Network Toolbar, Tarjan, FCM(Fuzzy Control manager).

Freeware Software: - NetII, Spider Nets Neural Network Library, NeuDC, Binary Hopfeild Net with free Java source, Neural shell, PlaNet, Valentino Computational Neuroscience Work bench, Neural Simulation language version-NSL, Brain neural network Simulator.

4.5 Development of an ANNs model

Neural Network is a vast domain of technology where one can implement “human brain decision making power” into computer programs on the basis of error and approximation. Also, lot of research and development has been made in the field of Artificial Intelligence with the help of Neural Network, Fuzzy Logic and Genetic Algorithm. Usually Neural Network comprises of three types of layer viz. Input Layer, Hidden Layer and Output layer.

(a) Input Layer

The Input layer is a layer which communicates with the external environment that presents a pattern to the neural network. Once a pattern is presented to the input layer, the output layer will produce another pattern. In essence, this is all the neural network does. The input layer should represent the condition for which were training the neural network. Every input neuron should represent some independent variable that has an influence over the output of the neural network.

(b) Output Layer

The output layer of the neural network is what actually presents a pattern to the external environment. The pattern presented by the output layer can be directly traced back to the input layer. The number of output neurons should be directly related to the type of work that the neural network is to perform. To determine the number of neurons to use in output layer, first consider the intended use of the neural network. If the neural network is used to classify items into groups, then it is often preferable to have one output neuron for each group that input items are to be assigned into. If the neural network is to perform noise reduction on a signal, then it is likely that the number of input neurons will match the number of output neurons. In this sort of neural network, the patterns to leave the neural network in the same format as they entered.

(c) Hidden Layer

The hidden layer is the collection of neurons which has activation function applied on it as well as provides an intermediate layer between the input layer and the output layer.

Much research has been carried out in order to evaluate the number of neurons in the hidden layer but still none was accurate. Another question arises that how many hidden layers has used when dealing with complex problem.

There are two functions governing the behavior of a unit in a particular layer, which normally are the same for all units within the whole ANNs, i.e.

- The input function.
- The output/activation function.

Input into a node is a weighted sum of outputs from nodes connected to it. The input function is normally given by equation (4.1) as follows:

$$Net_i = \sum_j W_{ij} x_j + \mu_i \qquad \text{Eq. (4.1)}$$

Where Net_i describes the result of the net inputs x_i (weighted by the weights w_{ij}) impacting on unit i. Also, w_{ij} are weights connecting neuron j to neuron i, x_j is output from unit j and μ_i is a threshold for neuron i, see figure (4.2).

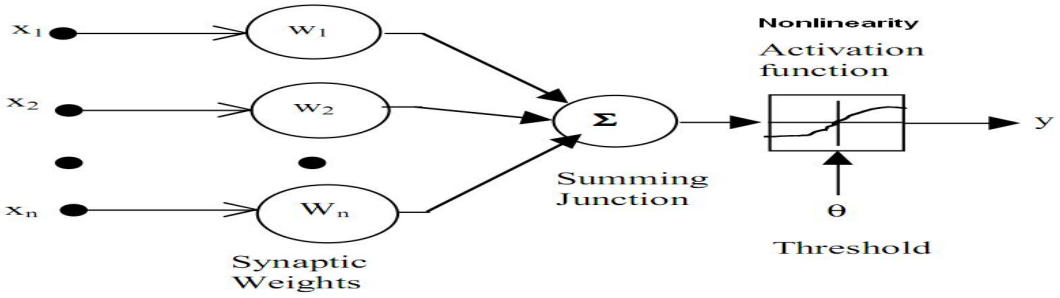

Figure (4.2): Mathematical Representation of Neural Network (Chakraborty, 2010).

Threshold term is baseline input to a node in absence of any other inputs. If a weight w_{ij} is negative, it is termed inhibitory because it decreases net input, otherwise it is called excitatory.

Each unit takes its net input and applies an activation function to it. For example, output of j^{th} unit, also called activation value of the unit, is g ($\sum w_{ji} x_i$), where xi is output of i^{th} unit connected to unit j. A number of nonlinear functions, as shown in figure (4.3), have been used in the literature as activation functions.

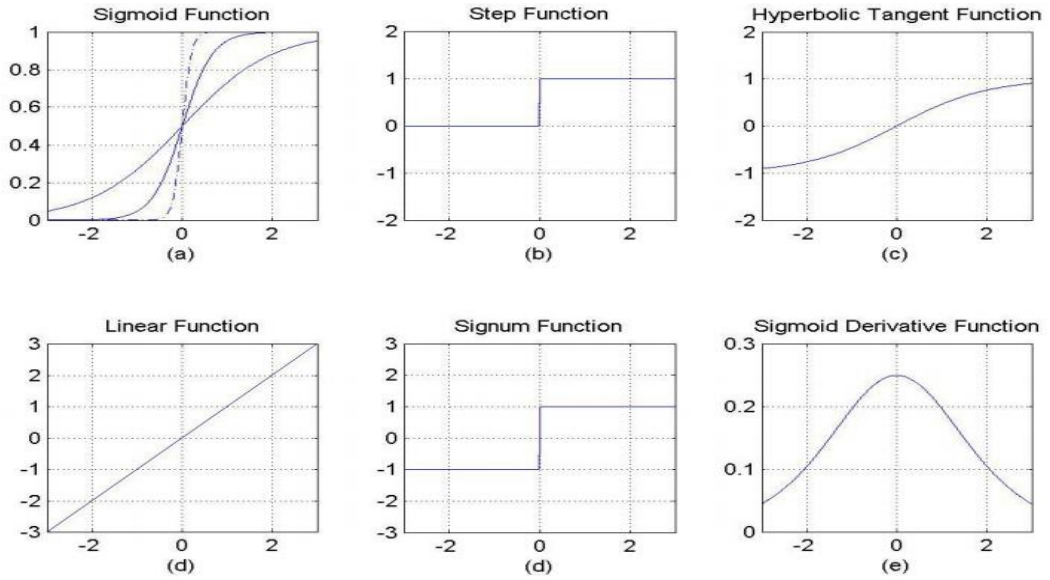

Figure (4.3): Typical Profiles of Eight Activation Functions (Chakraborty, 2010).

4.6 Model Building

Multilayer feed forward neural network or multi-layer perceptron (MLP), see figure (4.4), is very popular and is used more than other neural network type for a wide variety of tasks. Multilayer feed forward neural network learned by back propagation algorithm is based on supervised procedure, i.e., the network constructs a model based on examples of data with known output. It has to build the model up solely from the examples presented, which are together assumed to implicitly contain the information necessary to establish the relation.

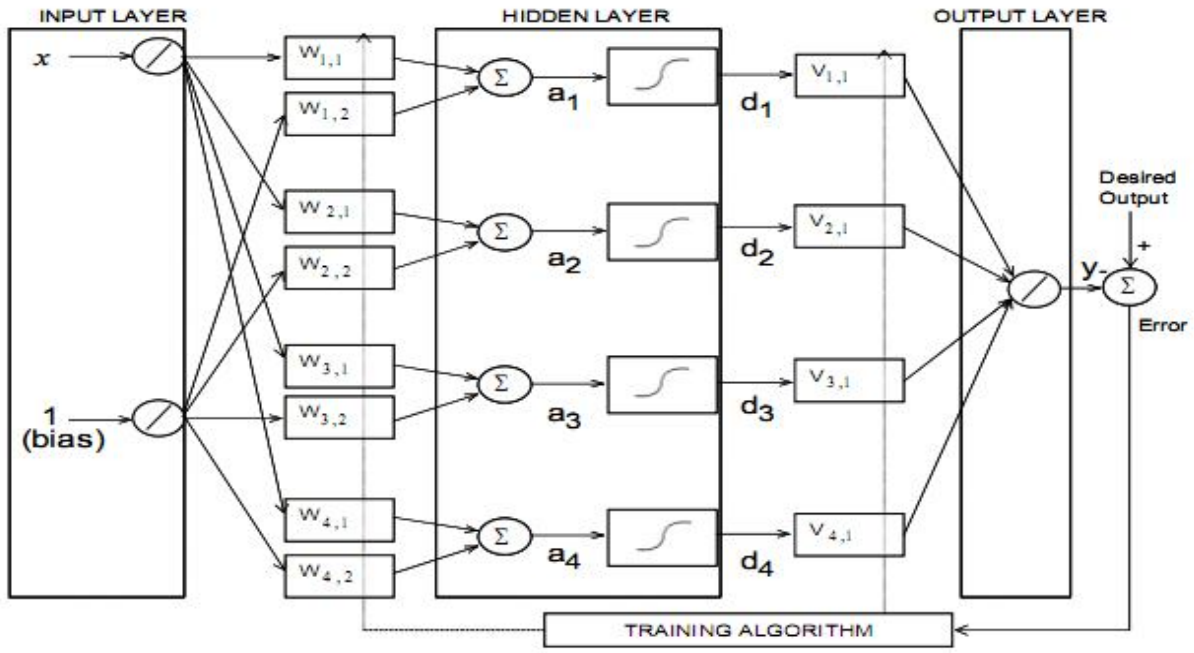

Figure (4.4): Feed Forward Neural Network with One Hidden Layer (Chakraborty, 2010).

An MLP is a powerful system, often capable of modeling complex, relationships between variables. It allows prediction of an output object for a given input object. The architecture of MLP is a layered feed forward neural network in which the non-linear elements (neurons) are arranged in successive layers, and the information flow unit-directionally from input layer to output layer through hidden layer(s). The characteristics of Multilayer Perceptron are as follows:

- Has any number of inputs
- Has one or more hidden layers with any number of nodes. The internal layers are called “hidden” because they only receive internal input (input from other processing units) and produce internal output (output to other processing units). Consequently, they are hidden from the output world.
- Uses linear combination function in the hidden and output layers.
- Uses generally sigmoid activation function in the hidden layers.

- Has any number of outputs with any activation function.
- Has a connection between the input layer and the first hidden layer, between the hidden layers, and between the last hidden layer and the output layer.

An MLP with just one hidden layer can learn to approximate virtually any function to any degree of accuracy. For this reason MLPs are known as universal approximates and can be used when we have litter prior knowledge of the relationship between input and targets.

Each interconnection in an ANNs has a strength that is expressed by a number referred to as weight. This is accomplished by adjusting the weights of given interconnection according to some learning algorithm. Learning methods in neural networks can be broadly classified into three basic types (i) supervised learning (ii) unsupervised learning and (iii) reinforced learning. In MLP, the supervised learning will be used for adjusting the weights.

4.7 Building Blocks of Artificial Neural Network

The basic building blocks of the artificial neural network are:

4.7.1 Network Architecture

The arrangement of neurons into layers and the pattern of connection within and in between the layers are generally called as the architecture of the network. Feed forward, feedback, fully interconnected network, competitive network, etc., Are the various types of network architectures.

4.7.1.1 Single Layer Network: It is one of the feed forward networks. It has only one layer of weighted interconnections. The inputs may be connected fully to the output units. In some cases, none of the input and output units are connected with other input and output units respectively. In a single layer network, the weights from one output unit do not influence the weights of other output units. See figure (4.5).

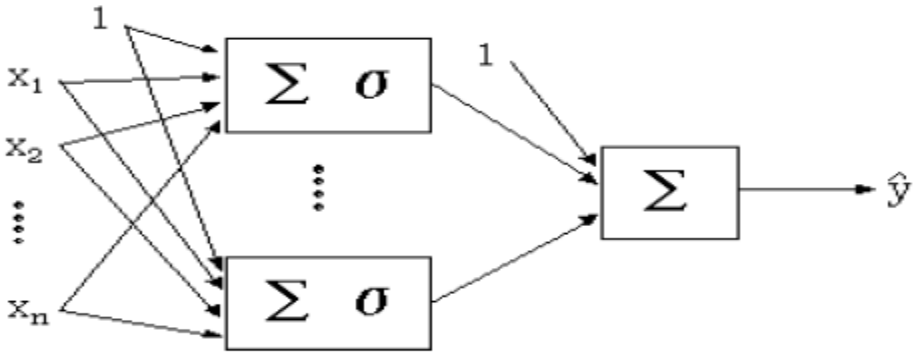

Figure (4.5): Single Layered Feed Forward Network (Chakraborty, 2010).

4.7.1.2 Multilayer Network: It is also feed forward net i.e., the network where flow the signals from the input units to the output units in a forward direction. The multi-layer network pose one or more layers of nodes between the input and output units. See figure (4.6).

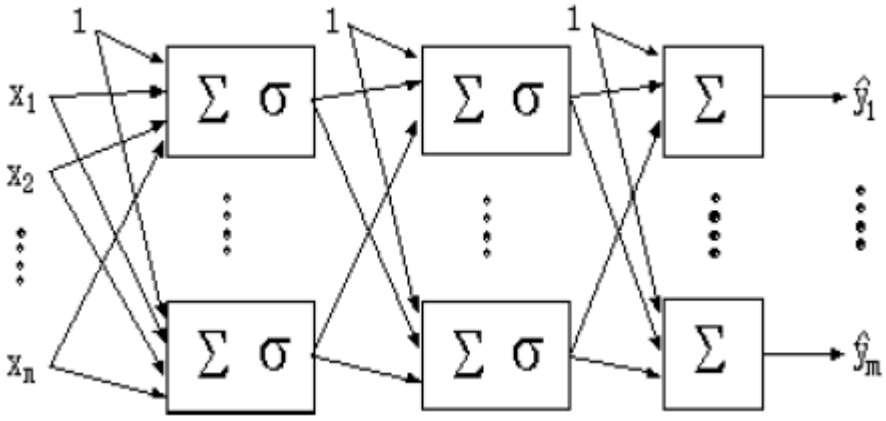

Figure (4.6): Multi Layered Feed Forward Network (Chakraborty, 2010).

4.7.2 Weights Settings

The process of learning or training depends on setting the value for the weights. The process of modifying the weights in connection between network layers with the objective of achieving the expected outputs is called training a network. Learning is the internal process that takes place when a network is trained.

4.7.3 Activation Function

The activation function is used to calculate the output response of neuron. The sum of the weighted input signal is applied with an activation to obtain the response. Same activation functions are used for neurons in the same layer. The non-linear activation functions are used in a multilayer network. The output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). In general, there are many types of activation functions; in this research the sigmoid function was used to build ANNs model.

The Sigmoid Function: This function can range between 0 and 1. The Sigmoid Function is generally used in multi-layered networks and Single Layer Network that is trained using back propagation algorithm.

4.8 Neural Network Learning Algorithms

Figure (4.7) summarizes the three basic types of neural network learning methods.

Learning law describes the weight vector for the i^{th} processing unit at time instant (t+1) in terms of the weight vector at time instant (t) as follows:

$$wi(t+1) = wi(t) + \Delta wi(t) \qquad \text{Eq. (4.2)}$$

Where $\Delta wi(t)$ is the change in the weight vector.

The network adapts as follows: change the weight by an amount proportional to the difference between the desired output and the actual output. As an equation:

$$\Delta Wi = \gamma * (D - Y).Ii \qquad \text{Eq. (4.3)}$$

where γ is the learning rate, D is the desired output, Y is the actual output, and Ii is the i^{th} input. This is called the Perceptron Learning Rule. The weights in an ANN, similar to coefficients in a regression model, are adjusted to solve the problem presented to ANN. Learning or training is term used to describe process of finding values of these weights. The supervised and unsupervised learning methods are the most popular forms of learning compared to reinforced learning.

Supervised learning which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the

learning process global information may be required. An important issue concerning supervised learning is the problem of error convergence, i.e. the minimization of error between the desired and computed unit values. The aim is to determine a set of weights which minimizes the error.

Unsupervised learning uses no external teacher and is based upon only local information. It is said that neural network learns off-line if the learning phase and the operation phase are distinct. A neural network learns on-line if it learns and operates at the same time. Usually, supervised learning is performed off-line, whereas unsupervised learning is performed on-line.

Reinforced learning; a teacher is present but does not present the expected or desired output but only indicated if the computed output is correct or not current.

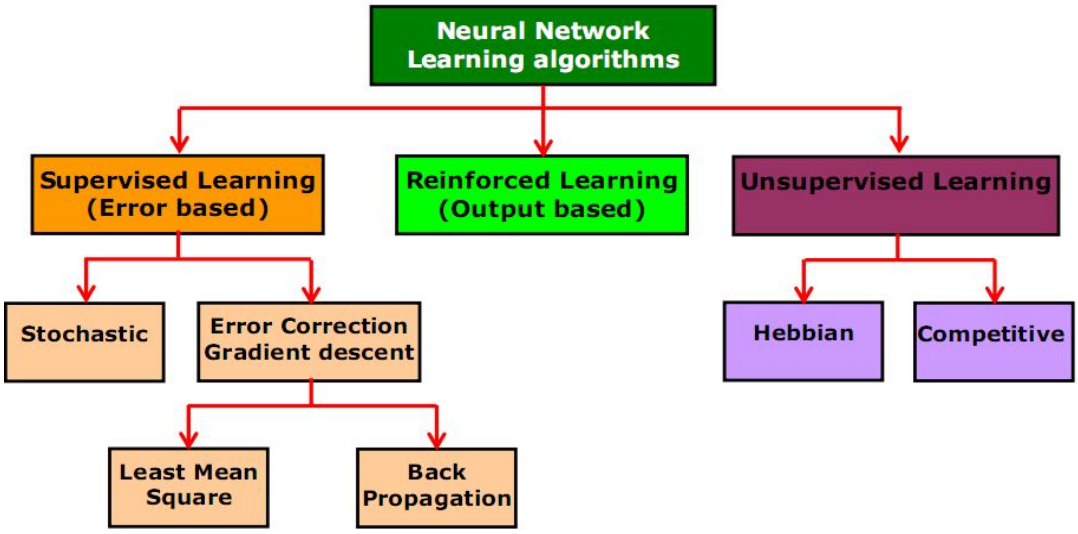

Figure (4.7): Classification of Learning Algorithms (Chakraborty, 2010).

4.9 Back Propagation Algorithm

The MLP network is trained using one of the supervised learning algorithms of which the best known example is back propagation, which uses the data to adjust the network's weights and thresholds so as to minimize the error in its predictions on the training set. It is denoted by W_{ij} the weight of the connection from unit u_i to unit u_j. It is then convenient to represent the pattern of connectivity in the network by a weight matrix W whose elements are the weights W_{ij} . The pattern of connectivity characterizes the architecture of the network. A unit in the output layer determines its activity by following a two-step procedure (Sargur, 2016).

- First, it computes the total weighted input x_j, using formula;

$$X_j = \sum_i y_i W_{ij} \qquad \text{Eq. (4.4)}$$

  Where y_i is the activity level of the j^{th} unit in the previous layer and W_{ij} is the weight of the connection between the i^{th} and the j^{th} unit.

- The next, the unit calculates the activity y_j using some function of the total weighted input.

Typically, the sigmoid function is used in such way:

$$y_j = (1 + e^{-x_j})^{-1} \qquad \text{Eq. (4.5)}$$

From (0 — 1) when $y_j = 0$

Once the activities of all output units have been determined, the network computes the errors E, which is defined by the expression;

$$E = \sum_i (y_i - d_i)^2 \qquad \text{Eq. (4.6)}$$

Where y_j is the activity level of the j^{th} unit in the top layer and d_j is the desired output of the j^{th} unit.

The back propagation algorithm consists of four steps:

(i) Compute how fast the error changes as the activity of an output unit is changed. This error derivative (EA) is the difference between the actual and the desired activity.

$$EA_j = \frac{\partial E}{\partial y_j} = y_i - d_i \qquad \text{Eq. (4.7)}$$

(ii) Compute how fast the error changes as the total input received by an output unit is changed. This quantity (EI) is the answer from step (i) multiplied by the rate at which the output of a unit changes as its total input is changed.

$$EI_j = \frac{\partial E}{\partial X_j} = \frac{\partial E}{\partial y_j} \times \frac{\partial y_j}{\partial x_j} = EA_j y_j (1\text{-}y_j) \qquad \text{Eq. (4.8)}$$

(iii) Compute how fast the error changes as a weight on the connection into an output unit is changed. This quantity (EW) is the answer from step (ii) multiplied by the activity level of the unit from which the connection emanates.

$$EW_{ij} = \frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial X_j} \times \frac{\partial X_j}{\partial W_{ij}} = EI_j y_j \qquad \text{Eq. (4.9)}$$

(iv) Compute how fast the error changes as the activity of a unit in the previous layer is changed. This crucial step allows back propagation to be applied to multilayer networks. When the activity of a unit in

the previous layer changes, it affects the activities of all the output units to which it is connected. So to compute the overall effect on the error, it should be added together all these separate effects on output units. But each effect is simple to calculate. It is the answer in step (iii) multiplied by the weight on the connection to that output unit.

$$EA_i = \frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial x_j} x \frac{\partial x_j}{\partial y_i} = \sum_j EI_j W_{ij} \qquad \text{Eq. (4.10)}$$

By using steps (ii) and (iv), we can convert the EAs of one layer of units into EAs for the previous layer. This procedure can be repeated to get the EAs for as many previous layers as desired. Once we know the EA of a unit, we can use steps (ii) and (iii) to compute the EWs on its incoming connections.