2.1 NEURO FUZZY SYSTEMS

Neural network learning provides a good way to adjust the knowledge of expert and automatically generate additional rules and membership functions to meet certain specifications. Fuzzy logic enhances the generalization capability of a neural network system by providing more reliable output when extrapolation is needed beyond the limits of the training data (Mirzai, 1990) (Rich et al 1991) (Russell et al 1995). Fuzzy sets are an aid in providing information in a more human comprehensible or natural form and can handle uncertainties at various levels.

In the field of artificial intelligence, neuro-fuzzy refers to combinations of artificial neural networks and fuzzy logic. Neuro-fuzzy hybridization results in a hybrid intelligent system that synergizes these two techniques by combining the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks. It is widely termed as Fuzzy Neural Network (FNN) or Neuro-Fuzzy System (NFS) in the literature which incorporates the human-like reasoning style of fuzzy systems through the use of fuzzy sets and a linguistic model consisting of a set of IF-THEN fuzzy rules. The main strength of neuro-fuzzy systems is that they are universal approximators with the ability to solicit interpretable IF-THEN rules.
Neuro Fuzzy approach gets the benefits of neural networks as well as of fuzzy logic systems and removes the individual disadvantages by combining them on the common features such as distributed representation of knowledge, model-free estimation, ability to handle data with uncertainty and imprecision etc., (Lin et al 1996, Medsker 1995 and Jana et al 1996). It consists of the components of a conventional fuzzy system except that computations at each stage are performed by n layer of hidden neurons and neural network’s learning capacity is provided to enhance the system knowledge. Even it is described as a fuzzy system that uses a learning algorithm derived from or inspired by neural network theory to determine its parameters by processing data samples. A model of neurofuzzy system can be depicted as in Figure 2.1

Neuro Fuzzy system combine the advantage of fuzzy systems, which deal with explicit knowledge that can be explained and understood and neural networks, which deal with implicit knowledge that can be acquired by learning (Pal et al 1999) (Wang, 1994). Neuro Fuzzy computing makes one to build more intelligent decision making systems. Both neural network and fuzzy systems are dynamic and parallel processing systems that estimate input output functions. They estimate a function without any mathematical model and learn from experience with sample data. A fuzzy system is capable of making fuzzy associations from representative numerical samples. On the other hand neural networks can generate and refine fuzzy rules from training data (Liu et al 1997) (Mitra, et al 2000).
2.1.1 Membership Functions and Fuzzy Rules

A fuzzy set is an extension of a classical set. If $X$ is the universe of discourse and its elements are denoted by $x$, then a fuzzy set $A$ in $X$ is defined as a set of ordered pairs as

$$A = \{ x, \mu_A(x) \mid x \in X \}$$

(2.1)

where $\mu_A : x \rightarrow [0, 1]$ is called the membership function defined as $\mu_A(x) \subseteq [0, 1]$, for each $x \in A$. For example the membership function maps each element of $x$ to a membership value between 0 and 1 as shown in Figure 2.2.
Figure 2.2 Membership function for the fuzzy variable “temperature”

A membership function is a mathematical function which defines the degree of information contained in a fuzzy set. It is described by the membership function which is useful to develop a lexicon of terms to describe various special features of this function. It is a graphical representation of the magnitude of participation of each input. It associates a weighting with each of the inputs that are processed, define functional overlap between inputs, and ultimately determines an output response. The rules use the input membership values as weighting factors to determine their influence on the fuzzy output sets of the final output conclusion. Once the functions are inferred, scaled, and combined, they are defuzzified into a crisp output which drives the system (Hisao, et al 1993). There are different membership functions associated with each input and output response and categorized as triangular, trapezoidal and Gaussian functions. The simplest membership functions are formed using straight lines. Of these, the simplest is the triangular membership function. The triangular curve is a function of a vector $x$, and depends on three scalar parameters $a$, $b$, and $c$ as...
\[ f(x; a, b, c) = \max \left( \min \left( \frac{x-a}{b-a}, \frac{c-x}{c-b} \right), 0 \right) \] (2.2)

The parameters \( a \) and \( c \) locate the "feet" of the triangle and the parameter \( b \) locates the peak as in the Figure 2.3

![Triangular Membership Function](image)

**Figure 2.3 Triangular Membership Function**

The trapezoidal membership function has a flat top and really is just a truncated triangle curve. These straight line membership functions have the advantage of simplicity. The trapezoidal curve is a function of a vector \( x \), and depends on four scalar parameters \( a, b, c, \) and \( d \), as

\[ f(x; a, b, c ,d) = \max \left( \min \left( \frac{x-a}{b-a}, \frac{d-x}{d-c} \right), 0 \right) \] (2.3)

The parameters \( a \) and \( d \) locate the "feet" of the trapezoid and the parameters \( b \) and \( c \) locate the "shoulders" as in the Figure 2.4
Because of their smoothness and concise notation, Gaussian and bell membership functions are popular methods for specifying fuzzy sets. Both of these curves have the advantage of being smooth and nonzero at all points. The symmetric Gaussian function depends on two parameters $\sigma$ and $r$ as given by

$$f(x; \sigma, r) = e^{-\frac{(x-r)^2}{2\sigma^2}}$$  \hspace{1cm} (2.4)

where $x$, $r$, $\sigma$ represents the input values, centre and width respectively as shown in Figure 2.5
There are possibly more ways to assign membership values or functions to fuzzy variables than there are to assign probability density functions to random variables (Dubios et al 1980). The assignment process is intuitive or it is based on some algorithmic or logical operations. Some of the techniques used to assign membership values to fuzzy variables are intuition, inference, rank ordering, angular fuzzy sets, genetic algorithms etc., (Zadeh, 1978).

Intuition method needs little or no introduction which is simply derived from the capacity of humans to develop membership functions through their own innate intelligence and understanding. Intuition involves contextual and semantic knowledge about an issue which can also involve linguistic truth values about the knowledge. In the inference method, knowledge is used to perform deductive reasoning where conclusion is inferred using body of facts and knowledge. Rank ordering method can be implemented with assessing preferences by a single individual, a committee, a poll and other opinion methods to assign membership values to a fuzzy variable. Even fuzzy membership functions may be created for fuzzy classes of an input data (Takagi et al 1991). Genetic algorithms can be used to compute membership functions (Karr et al 1993). Given some functional mapping for a system, some membership functions and their shapes are assumed for the various fuzzy variables defined for a problem. These membership functions are then coded as bit strings that are then concatenated (Timothy, 1995).

Using fuzzy sets associated with each parameter for identification of a disease, general If-Then rules can be formulated as shown in the equation 2.5. A fuzzy rule base contains fuzzy rules \( R \), which can be expressed as
$R_i: IF (PA_1 \text{ is } A_{i1}) \text{ AND } (PA_2 \text{ is } A_{i2}) \text{ AND } \cdots \text{ AND } (PA_n \text{ is } A_{in})$

$\text{THEN } (Y \text{ is } B_i)$  \hspace{1cm} (2.5)

where $A_{ij}, j : 1..n$ and $B_i$ are fuzzy sets,

$PA_j$ and $Y$ are fuzzy inputs and output respectively.

2.2 SUPERVISED AND UNSUPERVISED LEARNING ALGORITHMS

2.2.1 Back Propagation Algorithm

Back Propagation is a systematic method for training multi layer artificial neural network. It provides a computationally efficient method for changing the weights in a feed forward network with differentiable activation function units, to learn a training set of input-output examples (Rumelhart et al 1986). Being a gradient descent method, it minimizes the total squared error of the output computed by the net. The network is trained by supervised learning method to achieve a balance between the ability to respond correctly to the input patterns that are used for training and the ability to provide good responses to the input that are similar. This network consists of an input layer, one or more hidden layers and an output layer. Here, both the output units and the hidden units have bias. The bias acts like weights on connection from units whose output is always ‘1’. The training algorithm of back propagation involves four stages of Initialization of weights, Feed forward, back propagation of errors and updation of weights and biases. The output of the network is determined by the activation of the units in the output layer as in Equation (2.6) and the most used activation function to is the sigmoid which is represented as

$$X_o = f \left[ \sum_h x_h \cdot w_{ho} \right]$$ \hspace{1cm} (2.6)
where $f$ is the activation function, $x_h$ activation of $h$th hidden layer node and $w_{ho}$ is the interconnection between $h$th hidden layer node and $o$th output layer node, and

$$X_o = \frac{1}{1 + \exp(-\sum_h x_h w_{ho})}$$

(2.7)

The activation level of the nodes in the hidden layer is determined in a similar way which is based on the differences between the calculated output and the target value. An error is defined as

$$E = \frac{1}{2} \sum_i \sum_o (t_o - x_o)^2$$

(2.8)

where $N$ is the number of pattern in dataset and $L$ is the number of output nodes. Here the aim is to reduce the error by adjusting the interconnections between layers using gradient descent back propagation algorithm.

### 2.2.2 Kohonen’s Self Organizing Maps (KSOM):

KSOM is similar to a single-layered feed forward network except that there are connections, usually negative between the output nodes and the architecture is shown in Figure 2.6. In this network, during the initial learning process of ordering phase, the learning rate parameter should be set close to unity and then gradually decreased. While in the convergence phase of the learning process, the learning rate parameter attains relatively small values for a long time (Kohonen, 1990).
During training, the net determines the output unit that is the best match for the current input vector and the weight vector for the winner is then adjusted with respect to the net’s learning algorithm. Initially the weights and learning rate are set. The input vectors to be clustered are presented to the network and once the input vectors are given, based on the initial weights, the winner unit is calculated either by Euclidean distance method or sum of products method as represented in equation (2.9). Based on the winner unit selection, the weights are updated for that particular winner unit using competitive learning rule as shown in equation (2.10). In this unsupervised training algorithm, the process is continued for particular number of epochs or the learning rate reduces to a very small rate.

For each j, Squared Euclidean distance is calculated as

$$D (j) = \sum (W_{ij} - X_i)^2$$  \hspace{1cm} (2.9)$$  

where $W_{ij}$ represents weight vector and $X_i$ represents input patterns.

Using standard competitive rule, the change $\Delta W_{ij}$ is given as

$$...$$
\[ \Delta W_{ij} = \begin{cases} \alpha (X_j - W_{ij}) & \text{if neuron } i \text{ wins the competition} \\ 0 & \text{if neuron } i \text{ loses the competition} \end{cases} \quad (2.10) \]

where \( \alpha \) represents the learning rate.

### 2.2.3 Radial Basis Function Algorithm

A radial basis function neural network (RBF) solves a nonlinear problem by casting input samples into a higher dimensional space in a nonlinear way. It is based on supervised learning which is good in modeling nonlinear data and also helps in learning the given application quickly. The architecture of RBF network consists of three layers like input, hidden and the output layers as shown in Figure 2.7.

Figure 2.7. RBF networks gets rapidly trained than that of back propagation networks where the weights are the centers of a set of basis function calculated using K means clustering. All hidden units in the RBF networks have the same width or degree of sensitivity to inputs. Calculation of individual width increases the performance of the RBF network.

![Architecture of Radial Basis Function Networks](image)
RBF networks are feed-forward networks trained using a supervised training algorithm. They are typically configured with a single hidden layer of units whose activation function is selected from a class of functions called basis functions (Moody et al. 1989). The structure of RBF neural networks consists of layer of neurons where each hidden layer has two parameters like center and the width of Gaussian basis function which determines the output of the units shown in Equation (2.11).

Gaussian basis function has the following form

$$\Phi_j(x) = \exp \left( -\frac{\|x - \mu_j\|^2}{\sigma_j^2} \right)$$  \hspace{1cm} (2.11)

where $x$ and $\mu$ are the input and the center of RBF unit respectively. $\sigma_j$ is the spread of the gaussian basis function. The output of $i^{th}$ neuron in the output layer of RBF is determined by the linear combination of the output of the RBF units in the hidden layer as

$$y_i(x) = \sum_{j=1}^{M} W_{ij} \Phi_j(x) + b_i$$  \hspace{1cm} (2.12)

where $W_{ij}$ the weights between RBF units and the output node of RBF neural network and $b_i$ is the bias term. The weights are optimized using least mean square LMS algorithm once the centers of RBF units are determined.

### 2.2.4 Non Linear Hebbian Learning Algorithm

The Hebbian paradigm is perhaps the best-known unsupervised learning theory in connectionism (Papageorgiou, et al. 2006). The non linear hebbian learning algorithm focus in the domain of artificial neural network field and it embodies properties such as locality and the capability of being
applicable to the basic weight-and-sum structure of neuron models. Since fuzzy cognitive maps have non-linear structure of their concepts, the non-linear hebbian learning is used to train FCM. Here in this algorithm, the learning rule for FCMs integrates a learning rate parameter $\eta_k$, weight decay parameter $\gamma$, and the input/output concepts. The value of each concept of FCM is updated through Equation (2.13) whereas the value of weight is calculated using Equation (2.14).

$$A_i^{(k-1)} = f \left[ A_i^{(k)} + \sum_{j \neq i}^N A_j^{(k)} \cdot w_{ji}^{(k)} \right]$$  \hspace{1cm} (2.13)$$

or

$$A^{(k)} = f( A^{(k-1)} + \sum A^{(k-1)} \cdot w )$$

where $A_i^{(k+1)}$ denotes the value of concept $C_i$ at simulation step $k+1$, $A_i^{(k)}$ denotes the value of concept $C_j$ at simulation step $k$, $w_{ji}^{(k)}$ is the weight of interconnection between concept $C_j$ and concept $C_i$ and $f$ is the sigmoid threshold function which is calculated as $f(x) = \frac{1}{1 + e^{-x/\lambda}}$ where $\lambda$ is a positive constant which takes value as 1 or 5 and $f(x)$ lies between 0 and 1 (i.e.)

$$w_{ji}^{(k)} = \gamma \cdot w_{ji}^{(k-1)} + \eta_k A_i^{(k-1)} ( A_j^{(k-1)} - ( w_{ji}^{(k-1)} \cdot w_{ji}^{(k-1)} ) w_{ji}^{(k-1)} A_i^{(k-1)} )$$  \hspace{1cm} (2.14)$$

When the NHL algorithm is applied, only the initial non zero weights suggested by the experts are updated for each iteration step. All the other weights of weight matrix $W_{ji}$ remains zero which is their initial value. There are two termination conditions for the NHL algorithm. The first termination condition is the minimization of function $F1$ which uses decision of concepts(DOC) as defined by experts and a target value $Ti$ which represents a desired value (or) the mean value when DOC represents a
concept as shown in Equation (2.15). The second termination condition $F2$ is the minimization of the variation between two subsequent values of DOC as shown in Equation (2.16) and helps to terminate the iterative process of the learning algorithm.

\[
\text{Condition 1: Calculate } F1 = \sqrt{\sum_{i=1}^{m} (DOC_i - T_i)^2} \quad (2.15)
\]

where $m$ is the number of DOCs and $T_i$ defines the target value, which can be calculated as

\[
T_i = \frac{(T_{i_{\text{min}}} + T_{i_{\text{max}}})}{2}, \quad i = 1 \text{ to } 3
\]

\[
\text{Condition 2: Calculate } F2 = \left| DOC_i^{(4)} - DOC_i^{(4)} \right| < e \quad (2.16)
\]

where $e$ takes a value of 0.001

### 2.3 CLASSIFICATION ALGORITHMS

*K*-nearest neighbour (KNN) is one of the most fundamental and simple classification methods. *KNN* was developed in order to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine. The *KNN* classifier is commonly based on the Euclidean distance between a test sample and the specified training samples. Let $x_i$ be an input sample with $p$ features ($x_{i1}, x_{i2}, \ldots, x_{ip}$), $n$ be the total number of input samples ($i = 1, 2, \ldots, n$) and ‘$p$’ the total number of features ($j=1, 2, \ldots, p$). The Euclidean distance between sample $x_i$ and $x_k (k=1, \ 2, \ldots, \ n)$ is defined as

\[
D(x_i, x_k) = \sqrt{(x_{i1} - x_{k1})^2 + (x_{i2} - x_{k2})^2 + \ldots + (x_{ip} - x_{kp})^2} \quad (2.17)
\]
In pattern recognition, the $K$ nearest neighbour algorithm (KNN) is a method for classifying objects based on closest training examples in the feature space. It is a type of instance-based learning, or lazy learning where the function is only approximated locally and all the computation are deferred until classification. The $K$ nearest neighbor algorithm is amongst the simplest of all machine learning algorithms. An object is classified by a majority vote of its neighbors, with the object being assigned to the class, most common amongst its $K$ nearest neighbors. ‘$K$’ is a positive integer, typically small. If $K = 1$, then the object is simply assigned to the class of its nearest neighbor. In binary (two class) classification problems, it is helpful to choose $K$ to be an odd number as this avoids tied votes.

The best choice of ‘$K$’ depends upon the data; generally, larger values of $K$ reduce the effect of noise on the classification, but makes boundaries between classes less distinct. A good ‘$K$’ can be selected by various heuristic techniques, for example, cross-validation. The special case where the class is predicted to be the class of the closest training sample (i.e. when $K = 1$) is called the nearest neighbor algorithm. The accuracy of the KNN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance. Much research work has been put into selecting or scaling features to improve classification.

2.4 FUZZY COGNITIVE MAPS

Fuzzy Cognitive maps (FCM) are fuzzy–graph structures for representing causal reasoning. Their fuzziness allows hazy degrees of causality between hazy causal objects (concepts). Their graph structure allows systematic causal propagation, in particular forward and backward chaining, and it allows knowledge base to be grown by connecting different FCMs. Political Scientist Robert Axelrod(1976) introduced cognitive maps in the
1970s for representing social scientific knowledge. Axelrod’s cognitive maps are signed digraphs. Nodes are variable concepts like social instability and edges are causal connections. Even cognitive maps facilitate constructing symbolic representations of expert documents. Causal conceptual centrality in cognitive maps can be defined with adjacency-matrix components $e_{ij} = e(C_i,C_j)$ is the causal edge function value, the causality causal node $C_i$ imparts to $C_j$, $C_i$ causally increases $C_j$ if $e_{ij} = 1$, causally decreases $C_j$ if $e_{ij} = -1$, and imparts no causality if $e_{ij} = 0$ as shown in Figure 2.8.

\[
\begin{pmatrix}
C1 & C2 & C3 & C4 & C5 & C6 \\
C1 & 0 & -1 & 1 & 0 & 0 & 0 \\
C2 & 0 & 0 & 0 & 1 & 0 & 0 \\
C3 & 0 & 0 & 0 & 0 & 1 & 0 \\
C4 & 0 & 0 & 0 & 0 & 0 & -1 \\
C5 & 0 & 0 & 0 & -1 & 0 & -1 \\
C6 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

**Figure 2.8  Adjacency Matrix**

The methodology for developing FCMs is described analytically and it is based on a group of experts who are asked to define concepts and describe relationships among concepts (Papageorgiou, et al 2003). Every expert describes each interconnection with a fuzzy rule and the inference of the rule is a linguistic variable, which determines the grade of causality between the two concepts. Figure 2.9 illustrates the graphical representation of a simple FCM, where each interconnection $e_{ji}$ between two concepts $C_i$ and $C_j$ has a weight, belonging to the interval [-1, 1]. The sign of weight indicates whether the relation between the two concepts is direct or inverse.
Figure 2.9 A simple Fuzzy Cognitive map

FCM is used to model and simulate the behaviour of any system. It consists of concepts (representing the main domain aspects such as variables, states, inputs, outputs) and of interconnections among concepts that represent their relationships. The most important element in describing the system is the determination of the degree of influence among the concepts. There are three possible types of causal relationships among concepts that express the type of influence from one concept to the others. The weight of the interconnection between concept \( C_i \) and concept \( C_j \), denoted by \( W_{ij} \), could be positive if \( W_{ij} > 0 \) for positive causality or there is negative causality \( W_{ij} < 0 \) or there is no relationship between concept \( C_i \) and concept \( C_j \), thus \( W_{ij} = 0 \). The causal knowledge of the dynamic behaviour of the system is stored in the structure of the map and in the interconnections that summarize the correlation between cause and effect. The main objective of building a fuzzy cognitive map around a problem is to predict the outcome by letting the relevant issues interact with one another. These predictions can be used in a decision support system (DSS) for finding out whether a conclusion arrived at, is consistent with the whole collection of stated causal assertions. In FCM model for each step, the value \( A_i \) of a concept is calculated, computing the influence of the interconnected concepts to the specific concept according to equation (2.13)

In this work we use \( m=\frac{1}{2} \), because this value showed best results in previous works (Miao et al., 2000). A concept is turned ‘on’ or activated by
making its vector element 1 or -1. New state vectors showing the effect of the activated concept are computed using method of successive substitution, i.e., by iteratively multiplying the previous state vector by the relational matrix using standard matrix multiplication \( A^k = A^{k-1} + A^{k-1} \cdot W \). The iteration stops when a limit vector is reached, i.e., when \( A^k = A^{k-1} \) or when \( A^{t} - A^{t-1} \leq e \); where \( e \) is a residual value, which depends on the application type (and in most applications is equal to 0.001). Equation (2.13) includes the previous value of each concept, and so the FCM possesses memory capabilities and there is a smooth change after each simulation step. The forward inference process of FCM starts with a stimulus event vector and it is inserted into the FCM. Multiplying the stimulus vector to the FCM matrix which eventually yield the following:

**Fixed point:** if the FCM equilibrium state of a dynamical system is a unique state vector, the state vector remains unchanged for successive iterations, then it is called the fixed point.

**Limit cycle:** if the FCM settles down with a state vector repeating in the form \( A1 \rightarrow A2 \rightarrow \ldots \rightarrow Ai \ldots \rightarrow A1 \ldots \) then this equilibrium is called a limit cycle.

**Chaotic attractor:** the FCM state vector keeps changing at every iteration and repeating states are never found.

The development and design of the appropriate fuzzy cognitive map for the description of a system requires the contribution of human knowledge. The experts develop fuzzy cognitive maps using an interactive procedure to present their knowledge on the operation and behaviour of the system. Experts are asked to determine the concepts that best describe the model of the system, since they know the factors that are the key principles
and functions of the system operation and behaviour. They introduce a concept for each one. Experts have observed the operation and behaviour of the system during its operation, since they are the operators and supervisors of the system, who control it using their experience and knowledge. They have stored in their mind the correlation among different characteristics, states, variables and events of the system and in this way they have encoded the dynamics of the system using fuzzy rules. The procedure described in this thesis is based on earlier works for constructing FCMs (Papageorgiou et al 2003, 2006, 2008).

When the FCM model has been developed, the non linear hebbian algorithm is applied to adjusting the weights of the FCM interconnections and modifying them according to the specific problem characteristics. The NHL algorithm adapts synchronously the non-zero weights of the FCM model using the unsupervised learning approach. The main advantage of the NHL algorithm is that it can modify the initial FCM causal links between the concepts in order to increase classification capabilities of the FCM. In this way, the NHL algorithm increases the FCMs’ effectiveness, flexibility and robustness, and creates advanced FCMs with dynamic behaviour and greater modeling abilities.

2.5 SUMMARY

This chapter has presented different state of the art approaches based on supervised, unsupervised, machine learning algorithms and Fuzzy Cognitive maps. Each approach has been discussed with algorithm, working principle and its architecture. In this thesis work, it is proposed to utilize Neuro Fuzzy model, hybrid model of supervised and unsupervised algorithms and Fuzzy cognitive maps using non linear hebbian learning algorithm.