CHAPTER 4

SOFT COMPUTING TECHNIQUES

4.1 CONCEPT OF ARTIFICIAL NEURAL NETWORKS

Artificial neural networks consist of many interconnected neurons with familiar characteristics such as inputs, synaptic strengths, activation, outputs and bias. The neurodynamics of neural networks defines their properties, that is, how the neural network learns, recalls, associates and continuously compares new information with existing knowledge, how it classifies new information and how it develops new classifications if necessary. Hence, Artificial neural networks (ANNs) are excellent tools for analysis of complex problems that have many variables and complex interactions. In general, any neural network will have three layers, namely input layer, hidden layer and output layer. In the present work, artificial neural network is used to classify the faulty switch of multilevel inverter.

The field of neural networks can be thought of as being related to artificial intelligence, machine learning, parallel processing, statistics, and other fields. The attraction of neural networks is that they are best suited to solving the problems that are the most difficult to solve by traditional computational methods. Neurobiologists believe that the brain is similar to a massively parallel analog computer, containing about 10^10 simple processors which each require a few milliseconds to respond to input. With neural network technology, we can use parallel processing methods to solve some real-world problems where it is very difficult to define a conventional algorithm.

4.1.1 Justification for using Neural Network

The primary goal of the present work is to identify the faulty switch of cascaded H-Bridge multilevel inverter based on the features extracted from the output voltage signal. Therefore it is necessary to choose a suitable method capable of recognizing all the features obtained from the output voltage signal and identify the faulty switch of the multilevel inverter. Amongst the available pattern recognition methods, Artificial Neural Network (ANN) is a well-established and proven tool for addressing such pattern classification tasks. ANN is a parallel and highly adaptive learning system that can learn a task by generalizing from case studies of the tasks. If a problem can be posed as an input-output mapping problem, a supervised ANN can be used as a black box that learns the mapping from input-output examples from known cases of a task.

The significance of the neural network is the ability to learn from its environment and to improve the performance through further learning. In recent times, artificial neural networks are being applied to an increasing number of real world problems of various degrees of complexity. They are good pattern recognition engines and robust classifiers, with the ability for generalization and decision-making about imprecise and incomplete input data. They offer ideal solutions to a variety of classification problems such as speech, character and signal recognition, as well as functional prediction and system modeling, where the physical processes are not understood or are highly complex. The advantage of artificial neural networks lies in their resilience against distortions in the input data and their capability for learning from examples.

4.1.2 Model of a Neuron

Neural network consists of what are called neurons, information processing units, which process information dynamically in response to external inputs. The model of a neuron is shown in Figure 4.1. These neurons have a set of synapses or connecting links that are characterized by weights, an adder for summing up the input signals, and an activation function for linking the amplitude of the output of a neuron. The input and output of the neural network usually are connected through "hidden layers".

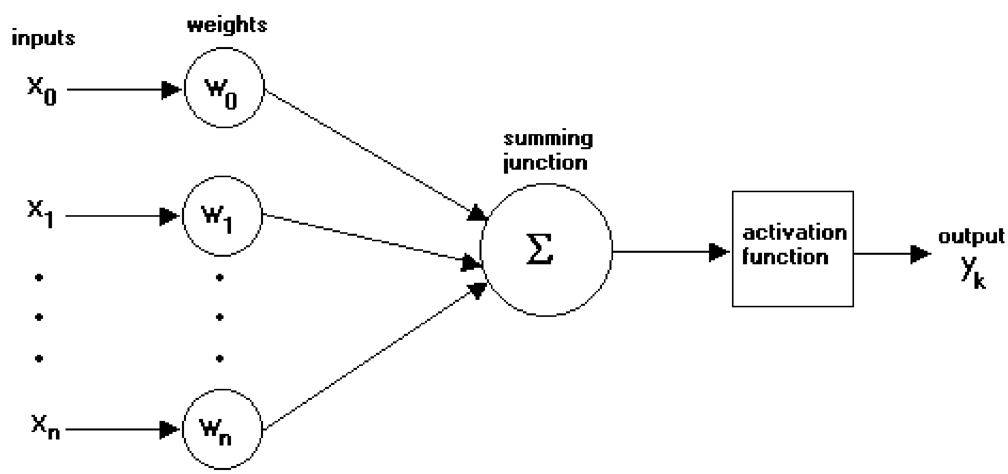

Figure 4.1 Model of a Neuron

The information about the relationship between the input and output is stored in the connecting weights during the training process. Before training, a suitable neural network topology has to be selected. The choice of activation function can change the behaviour of the neural network considerably. There are three basic types of activation functions, namely threshold functions, piece-wise linear function and the sigmoid function. In the present work, sigmoid function is used, because it assumes a continuous range of values from 0 to 1 and it is differentiable at each point, which is an essential requirement for using back propagation training algorithm.

4.1.3 Multilayer Feed-Forward Neural Network

Among the various ANN architectures available in the literature, the multilayer Feed Forward network (as shown in Figure 4.2) with back propagation learning algorithm has been used for the present study because of its simple approach and good generalization capability.

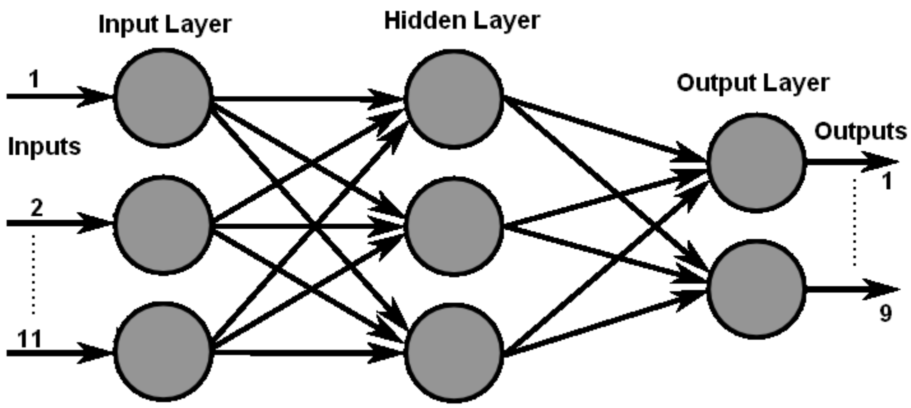

Figure 4.2 Schematic of Artificial Neural Network.

Multilayer feed forward neural network consists of an input layer, at least one hidden layer and an output layer. The author has adopted, the back propagation algorithm for training the multilayer neural network. Each layer is entirely related to next one. The number of neurons in the input and output layer is equal to the number of inputs desired and number of classification patterns required respectively. The training phase consists of the iterative presentation of the inputs with the estimated output, the adjustment of the weights and biases depend on the resultant error.

Stimulation is applied to the inputs of the first layer, and signals propagate through the middle (hidden) layer(s) to the output layer. Each link between neurons has a unique weighting value.

Inputs from one or more previous neurons are individually weighted, then summed as shown in Figure 4.3. The result is non-linearly scaled between 0 and +1, and the output value is passed on to the neurons in the next layer. Since the real uniqueness or 'intelligence' of the network exists in the values of the weights between neurons, we need a method of adjusting the weights to solve a particular problem. For this type of network, the most common learning algorithm is called Back Propagation (BP).

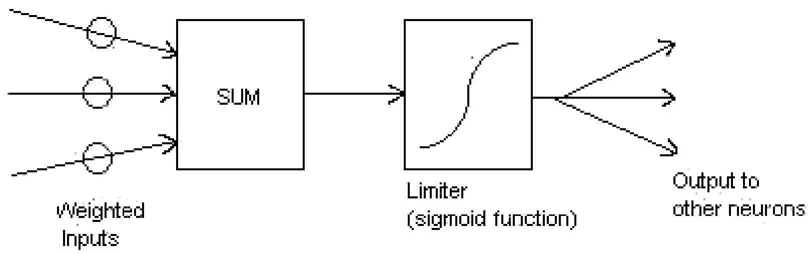

Figure 4.3 Processing of output of each neuron from sigmoid function

A BP network learns by example, that is, we must provide a learning set that consists of some input examples and the known-correct output for each case. So, we use these input-output examples to show the network what type of behavior is expected, and the BP algorithm allows the network to adapt.

The BP learning process works in small iterative steps: one of the example cases is applied to the network, and the network produces some output based on the current state of it's synaptic weights (initially, the output will be random). This output is compared to the known-good output, and a mean-squared error signal is calculated. The error value is then propagated backwards through the network, and small changes are made to the weights in each layer. The weight changes are calculated to reduce the error signal for the case in question. The whole process is repeated for each of the example cases, then back to the first case again, and so on. The cycle is repeated until the overall error value drops below some pre-determined threshold. At this point we say that the network has learned the problem "well enough" - the network will never exactly learn the ideal function, but rather it will asymptotically approach the ideal function.

The convergence is reached when the error between the measured and the desired output is less than fixed value (convergence criteria). In this network, error is back propagated and weights and biases are conveniently readjusted with an algorithm used in order to minimize the Mean Square Error (MSE), which is the average of sum of the errors for all set of inputs and corresponding outputs, which is calculated as follows,

$$MSE = \frac{1}{m}\sum_{k}^{m}\left(S_k - Y_k\right)^2 \tag{4.1}$$

where S_k and Y_k are respectively the desired and measured output for the k^{th} input set and m is the total number of input sets.

4.1.4 Neural Network Parameters

The important factors governing the convergence and the learning time of the neural network are network topology, size, learning rate, number of training sets, convergence criterion, number of iterations and the like. The learning rate is known to damp out oscillations to some extent, during the training phase. Higher values of learning rate may result in fast convergence, but it may result in oscillation. Higher values of training sets and training cycles will increase the training time of the neural network. It is necessary to arrive at an optimal neural network topology for better classification results. Therefore, in order to arrive at an optimal topology, the performance of the neural network for different values of the learning rate, training sets, convergence criterion, iterations and different number of neurons in the hidden layer are to be studied in detail and the performance is to be evaluated.

4.2 CONCEPT OF ANFIS

ANFISs represent a neural network approach to the design of fuzzy inference systems. Since their introduction, ANFIS networks have been widely considered in the technical literature and effectively applied to classification tasks, rule-based process controls, and pattern recognition problems and so on. ANFIS combines the inference ability of fuzzy logic and the self-learning ability of artificial neural networks (Jun Li 2012). It is one of the most successful Neuro-Fuzzy systems developed by Jang (Jorge & Antonio 2013), which applies neural learning rules to identify and tune the parameters and structure of a fuzzy inference system, based only on the available data. ANFIS utilizes the Sugeno-type fuzzy inference system, in which a combination of least-squares method and back propagation gradient descent method is applied for training the fuzzy inference membership function parameters to emulate a given training data set. ANFIS is widely used in function approximation, classification, pattern recognition and intelligent control applications.

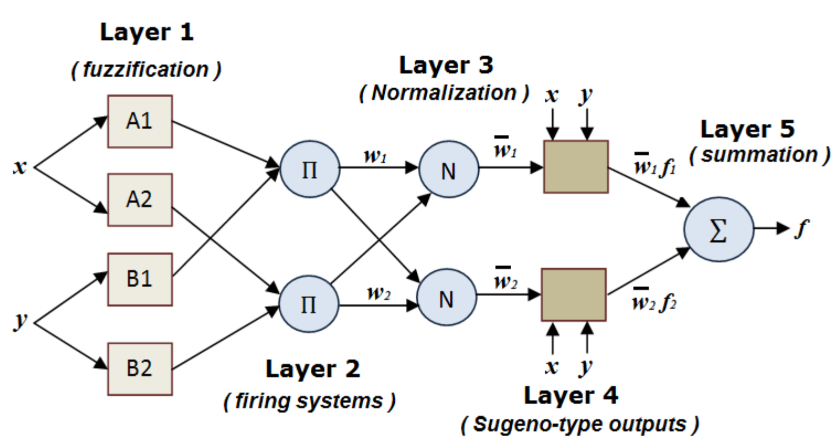

Figure 4.4 Schematic of ANFIS for Sugeno-type fuzzy inference system

A typical architecture of a ANFIS network for Sugeno-type fuzzy inference system is shown in Figure 4.4, in which a circle indicates a fixed node, whereas a square indicates an adaptive node. In adaptive node, node parameters are dependent on other nodes and in fixed nodes, node parameters are independent. ANFIS requires a set of input data and output data. It presents a multilayered feed-forward network, in which each layer has a particular function on the input data. It provides a set of membership functions for mapping the input data to output. The membership function chosen should be able to minimize the error between the actual output data and ANFIS input-mapped output data. The parameters associated with the membership functions will change through the learning process.

In the typical schematic of ANFIS as shown in Figure 4.4, it consists of two inputs (x,y) and one output f. It has five different layers. Layer 1 is for fuzzification of the input variables (x,y). In this layer, every node is a square node and the nodes (A_i, B_i) are the fuzzy sets, which contains the membership function (MF) assigned to each input. In the Sugeno fuzzy model, fuzzified output is a multi-valued vector, which is the result of the expression $x_i \in A_i$, which is the grade of membership that the input x_i has within the fuzzy set A_i. As set A_i contains m membership functions distributed over the dynamic range of each input x_i, the corresponding fuzzy value is an m-length vector.

In layer 2, each node is a circle node labeled Π and its output represents the firing strength of a rule. T-norm operators are used to compute the rule antecedent part. Each node Π multiplies incoming signals and sends the product out.

$$w_i = \mu_{A_i(x)} \mu_{B_i(y)} \qquad i = 1,2 \tag{4.2}$$

Hence, as shown in Figure 4.4, in this layer, the system identifies the two rules (rule 1: $x=A_1$, $y=B_1$ and rule 2: $x=A_2$, $y=B_2$). Table 4.1 lists the corresponding fuzzy if-then rules of ANFIS.

Table 4.1 Fuzzy If-Then rules of ANFIS

Rules	**If**		**Then**
Rule Number	**x**	**y**	**f**
1	A_1	B_1	$f_1 = p_1x + q_1y + r_1$
2	A_2	B_2	$f_2 = p_2x + q_2y + r_2$

Layer 3 normalises the rule strengths, i.e. each circle node N computes the ratio of the i_{th} rule's firing strength to the sum of all rule's firing strengths

$$\hat{w}_i = \frac{w_i}{w_1 + w_2} \quad i = 1,2 \tag{4.3}$$

In layer 4, the consequent parameters of the rule are determined in the square node. Layer 4 consists of the linear functions f_1 and f_2 of the input signals taking into account the normalized firing strength of each rule (*w1,w2*) calculated in the precedent layer.

$$f_1 = p_1x + q_1y + r_1 \tag{4.4}$$

$$f_2 = p_2x + q_2y + r_2 \tag{4.5}$$

Where f_1 and f_2 are the outputs within the fuzzy region specified by the fuzzy rule and p_i, q_i, r_i are the design parameters that are determined during the training process. In the layer 5, the overall output as the summation of all incoming signals is computed in the circle node.

$$f = \hat{w}_1 f_1 + \hat{w}_2 f_2 \tag{4.6}$$

where *f1* and *f2* are the outputs of layer 4. The final error is defined by the root-mean-square (RMSE) difference between the current and desired outputs.

ANFIS can be classified into two main types based on the input space partitioning method: grid and scattering partitioning. Grid partition (GP) generates rules by enumerating all possible combinations of membership functions of all input parameters. This leads to an exponential explosion even when the number of inputs is moderately large. For instance, for ANFIS with seven inputs, each with three membership functions, the grid partitioning creates 37 rules. Conversely, scattering partition (also known as SC) generates the number of rules equal to the number of clusters.

Similar to the concept of Neural Networks, ANFIS modeling starts by partitioning the data sets (input–output data pairs) into training and testing (or validating) data. The training data set is used to find the initial premise parameters for fuzzy membership functions by equally spacing each membership function. When the values of the premise parameters are fixed, the overall predicted output functions can be expressed as a linear combination of consequent parameters.

4.2.1 Advantages of ANFIS

Since the Neuro-Fuzzy systems combine the inference ability of Fuzzy Logic like a human and the learning and parallel data processing abilities of ANNs, it has many advantages when compared with other classification techniques. With these systems, the development time is very much reduced and the accuracy of the fuzzy model is improved. The major advantages of ANFIS methodology are as follows,

a) Implementation is easy

b) Learning is fast and accurate

c) It has strong generalization skills

d) Fuzzy rules make easier its understanding

e) It is easy to incorporate both linguistic and numeric knowledge for problem solving.

4.3 CONCLUSION

In this chapter concept of Artificial Neural Networks with back propagation training algorithm and adaptive neuro fuzzy inference system was discussed in detail for the effective implementation of classification of signals.