Intelligent Medical Technologies and Biomedical Engineering:
Tools and Applications

Anupam Shukla
ABV – Indian Institute of Information Technology and Management, India

Ritu Tiwari
ABV – Indian Institute of Information Technology and Management, India

Medical Information Science Reference

MEDICAL INFORMATION SCIENCE REFERENCE

Hershey · New York

Director of Editorial Content: Kristin Klinger
Director of Book Publications: Julia Mosemann
Acquisitions Editor: Lindsay Johnston
Development Editor: David DeRicco
Publishing Assistant: Tom Foley and Jamie Snavely
Typesetter: Deanna Jo Zombro
Production Editor: Jamie Snavely
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by
Medical Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Intelligent medical technologies and biomedical engineering : tools and applications / Anupam Shukla and Ritu Tiwari, editors.
p. ; cm.
Includes bibliographical references and index.
Summary: "This book takes an innovative look at technology and engineering as they pertain to medicine (medical engineering), teaming them to facilitate new systems that have the ability to change the lifestyles and quality of life of people"--Provided by publisher. ISBN 978-1-61520-977-4 (hardcover) 1. Medical informatics. 2. Intelligent control systems. 3. Biomedical engineering. I. Shukla, Anupam, 1965- II. Tiwari, Ritu, 1977-
[DNLM: 1. Biomedical Technology. 2. Artificial Intelligence. 3. Biomedical Engineering--methods. 4. Biotechnology--methods. W 82 I61 2010]
R858.I557 2010
610.28--dc22
2009054319

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 9
Hybrid Intelligent Systems for Medical Diagnosis

Rahul Kala
Indian Institute of Information Technology and Management Gwalior, India

Anupam Shukla
Indian Institute of Information Technology and Management Gwalior, India

Ritu Tiwari
Indian Institute of Information Technology and Management Gwalior, India

ABSTRACT

The past few years have seen a lot of applications of Hybrid Soft Computing approaches that seem to have completely replaced the traditional uni-system approaches. The added abilities that come from the hybrid approaches motivate their use in every system. We find various new approaches being applied to the field of Bio-Medical Engineering as well as many new models being proposed. At this juncture, we study the effectiveness of various new hybrid approaches in the field of Bio-medicals. PIMA Indian diabetes database has been used for this purpose from the UCI Machine Learning Repository. The basic aim is to compare the various hybrid approaches from the recent literature and compare their performances. We study 3 major Hybrid Systems and standard Back Propagation Algorithm for this purpose. These are Adaptive Neuro Fuzzy Inference Systems, Ensembles and Evolutionary Artificial Neural Networks. We also try to explain the results from our theoretical understanding of the individual Hybrid Systems.

1. INTRODUCTION

Bio-Medical Engineering is a rapidly growing field as a result of the need and rise of automation. This field calls for the collaboration between the people from the medical background and the engineers to develop intelligent systems for the various tasks in bio-medicals. These systems are used for the detection of the various diseases. These act as Clinical Decision Support Systems (CDSS) in order to assist the doctors in their task of identification of the presence of the diseases. They hence act as valuable tools for the doctors in the analysis of the diseases. This is especially important considering the work load over the doctors and the vast presence of the diseases. The increasing health conscious-

DOI: 10.4018/978-1-61520-977-4.ch009

ness among the people has further resulted in a lot of emphasis in the development and use of such systems.

Here we make the use of Soft Computing techniques for the detection of diseases. This is essentially a classification problem where the task is to classify the given parameters into either of the two classes that stand for the presence or absence of diseases. A number of inputs or attributes are given to the system to carry out this classification. Suppose these attributes are graphically plotted in a graph with each class being marked with a different symbol. This graph has as many axes as the number of attributes called as the input space. The major task in the problem of classification is to fragment the input space distinctive regions such that each class belongs to some different segment. The boundaries separating the classes are called as decision boundaries. Every system tries to compute or predict these boundaries based on given information or data. The classification problems have always deserved a special mention from the scientific communities as they have their own issues and complexities. One of the most interesting facts with these problems is that despite the revolution in the methods and means of classification, the artificial systems are still far behind the classification powers of humans when compared with accuracies. A related terminology is pattern matching that deals with the ability to recognize any given pattern or set of attributes.

Machine Learning is an exciting field that deals with the learning of the historical data. In most of the problems a lot of data is available from the history. The systems are made to learn this data by the use of training algorithms that may be specific to the system. Learning involves the extraction of rules or patterns from the historic data. It is evident that well trained systems would be able to give correct results to the problems that they have been trained with. Further the time and memory requirement would be reasonably less as the system has already summarized the historic data into some patterns or rules. Generalization is the ability of the system to give correct outputs to the unknown similar problems. This happens by the application of extracted patterns or rules by the system to the unknown inputs. A system is considered effective if it shows a very high generalizing capability.

Soft Computing is a rapidly growing field that primarily incorporates Artificial Neural Networks, Fuzzy Inference Systems and Evolutionary Algorithms. Artificial Neural Networks (ANN) (Ciampi and Zhang, 2002) are an inspiration from the human brain. The brain consists of a large number of parallel processing elements called as neurons. Each neuron receives the information in the form of electrical signals, processes this information and further transmits the information for the processing of the other neurons. The ANNs consist of numerous artificially designed neurons that work in a layered architecture. Each neuron takes the weighted sum of the incoming information and then passes it by an activation function before giving it to the next neuron. Usually we make use of a 3-layered architecture that consists of a passive input layer, a hidden layer and an output layer. The ANNs form good means of learning from the past data (or machine learning) and generalizing the learnt trends into the unknown inputs. These networks hence undergo two separate stages of training and testing. One of the chief ways of training of the ANNs is Back Propagation Algorithm (BPA). This is a supervised learning model where the outputs must be known for every input while the training is being done. BPA has two separate passes called forward pass and backward pass. The forward pass is a conventional neural network where the inputs are applied and the corresponding outputs are calculated. The difference between the output and the target is calculated which is known as the error. The error is propagated in the backward direction from the output layer to the input layer. Through this all neurons adjust their weights and

biases. The entire process is applied for a number of times called as the epochs.

The BPA normally makes use of the gradient descent to compute the new value of the weight sand biases. It is hence quickly able to adjust the network weights for good performance. The graph or space denoting the error of the system for every combination of weights and biases is called as the error space. The aim of any training algorithm is to find the global optima in this search space. BPA may many a times get trapped in local minima. This is due to the absence of any global guiding strategy or the attempt to cover the entire error space which is highly complex and of high dimensionality.

Another effective problem solving systems include the Fuzzy Inference Systems (FIS) (Xiaoguang and Lilly, 2004). These make use of Fuzzy Set theory for the modeling of the problem. As per the Fuzzy Set theory every element belongs to some class by a certain degree which is known as the membership value. Hence an input attribute would belong to some associated class by a certain degree and there would be a certain degree by which it would not belong to the class. These systems are effectively able to map the inputs to the outputs by the applications of Fuzzy Rules. Fuzzy Rules are an inspiration from the conventional production systems that made use of rules to operate on state to decide the final state or output. These systems have fuzzified inputs, outputs and classes. The problem solving with FIS includes the fuzzification of the inputs, where the inputs are associated with some degree. The next step is the application of the fuzzy rules according to the fuzzy operators. We then perform a fuzzy aggregation of the individual rule output. At the end the task of defuzzification is carried out.

One of the major problems with fuzzy rules is that they run on human made parameters and rules. The human fixed values are prone to be erroneous or suboptimal. These systems need to be manually trained and tuned which is not always feasible for all the problems. This places a very big limitation over the effective use of these systems in practical applications. The continuous change of rules, fixing of system parameters, etc. to get the designed system perform as per the desired outputs may be very difficult manually.

The Evolutionary Algorithms (EA) form excellent algorithms for the purpose of search and optimization problems (Srinivasaa, Venugopala and Patnaikb, 2007). They are an inspiration from the natural evolution process. Life evolves in populations of individuals. Every population builds up the next generation of population. In this manner the evolution process goes on and on. At every generation the fittest set of individuals survive and go to the next generation as per the Darwinian theory of survival of the fittest. The stronger individuals carry more reproductive capability and affect the population by a larger degree as compared to the weak individuals. Similarly EAs model the various solutions of the problem as individuals. As the generations increase, the individuals or the whole population gets better or more optimal to solve the problem. At the end the best individual is regarded as the most optimal solution to the problem. Various Evolutionary Operators inspired from natural processes are employed to better aid in the generation of optimal child population from the parent population. These algorithms are very effective in searching for Global Minima in a finite and limited amount of time.

The application of these systems effectively solves problems, but their weakness result in sub optimality that needs to be addressed. The problems especially become important when the training data size is very large or the system is very complex. In most of these scenarios, the weaknesses of these systems enlarge and hence the problem cannot be optimally solved by using one system alone. Hence we make use of a combination of systems with an aim to combine the positive aspects of the individual systems and removing their negative aspects. The negative aspects of one system are removed by the

positive aspect of the other system. The Hybrid Approaches form a very exciting field of work and research. In these systems we combine ANN, FIS and EA and make a much more efficient system. A good collection of methods and applications can be found in the books by Mellin and Castillo (2005), and Bunke and Kendel (2002).

The increase in efficiency that they have brought to numerous problems in the various domains is a key reason behind their use and popularity. These systems make use of a collection of Artificial Intelligence and Soft Computing Systems for the purpose of problem solving. The benefits of one system make over for the limitations of the other systems. As a result the whole system has an added performance. In this chapter we hence stud the various hybrid systems and analyze their capability to diagnose diseases. We study the diagnosis of the PIMA Indian diabetes.

This chapter is organized as follows. Section 2 deals with the various approaches used in this chapter. In section 2.1 we deal with the first method of application i.e. ANFIS. We discuss about the Modular Neural Networks (MNNs) and Ensemble Techniques in Section 2.2. Section 2.3 is dedicated towards the use of a connectionist approach in Evolutionary Neural Networks. In section 3 we describe the database used along with the problem solving methodology. Section 4 gives the results. The conclusion remarks are given in section 5.

2. HYBRID INTELLIGENT SYSTEMS

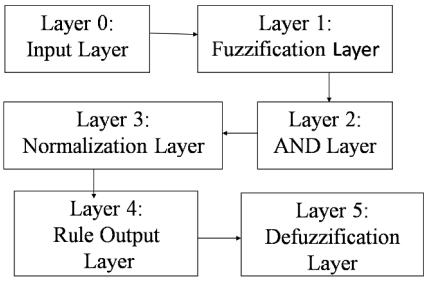
In this section we describe the various approaches used in this chapter for solving the diagnosis. The various approaches make extensive use of various Soft Computing techniques in a hybrid manner for the purpose of identification. Here we make use of 3 major techniques along with the standard Back Propagation Algorithm. These are ANFIS that is a combination of ANN with FIS, Modular Neural Network that make use of modularity in the problem and apply ANN for problem solving and Evolutionary ANN where the complete ANN is evolved with the use of GA.

2.1 ANFIS

Adaptive Neuro Fuzzy Inference Systems (ANFIS) (Vosoulipour, Teshnehlab and Moghadam, 2008; Temurtas, Yumusak, Temurtas, 2009) is the first hybrid method employed in solving the problem. This method is a fusion of the ANN with FIS. Essentially ANFIS is a Fuzzy Inference System (FIS) that is made over the Neural Network Architecture. This enables us to use Neural Network training algorithms in the training of FIS over a historical database. This tunes FIS and makes it capable to act according to the demands of the problem and give expected results to problems whenever it happens to be exposed towards them. This significantly contributes towards the minimization of error and achieving great performance in solving the problem. Automatic tuning of the fuzzy parameters are a problem in the conventional FIS that need to be done manually. These systems are hence very helpful in such scenarios as the parameter adjustment is done by the training algorithm itself. We just need to specify the training inputs along with their target outputs. The training algorithm does the rest of the task to tune the system as per the requirements.

Much like the ANN, the ANFIS has a layered architecture where the various layers do some task of the FIS. The various layers are connected by connections that connect the various modules of the ANFIS. The various layers used in the ANFIS are input, fuzzification, AND, normalization, rule output and defuzzification. The general layered structure is given in Figure 1. This is a multi-layer neural network model. Each layer acts on the data after the previous layer has finished its processing. It may b easily verified that these operations correspond to the general manner of problem solving with FIS. Hence the entire system may be viewed

Figure 1. The ANFIS layered architecture



in both respects i.e. as a neural network that has nodes arranged in various layers and each layer given a specific task to perform; or a FIS that performs the fuzzy operations to compute the outputs as per the inputs.

Problem solving in ANFIS consists of a series of steps. The first step is to make an initial Fuzzy model as per the requirements of the problem. This may be done manually by the user as per his understanding of the system, or automatically where all the combinations of rules may be generated. The initial fuzzy model would contain a set of membership functions for every input variable. The number of membership function plays a very big role in deciding the generalizing capability of the system. The lesser the number of membership function, the larger would be the generalizing capability. Higher number of membership functions makes the problem more localized in nature. It may be noted here that generalized systems try to find very general rules that fit the entire set of training data. This is in contrast to the localized systems try to frame rules for some part of the database. The entire training dataset may be said to be composed of multiple such sets. Generalization is specific to problem as well. All problems cannot be solved from the highest degree of generalization.

After the model has been formed, we need to optimize it by a training algorithm. For this we make use of a historical database of known inputs and outputs. This is a supervised learning technique used by the ANFIS. Two commonly used learning algorithms are Back Propagation Algorithm and Hybrid Learning Algorithm. The back propagation algorithm is same as that discussed in section 2.1. This tries to train the algorithm by the gradient descent of the errors in the error space. The testing involves giving the unknown inputs to the algorithm and then letting it classify the data as per the training performed. This step decides the efficiency of the algorithm.

2.2 Ensemble

The ensemble is the second Hybrid method used for the identification of diseases. The ensemble is a type of Modular Neural Network (MNN) (Shadabi, Sharma and Cox, 2006; Shukla *et al*, 2009). The MNN believes in exploiting the modularity in the problem by dividing the problem into a set of modules. The modules work independent of each other and contribute towards the problem solving. The conventional ANN has problems that result in a loss of performance when the training database is too large in size or the problem being concerned is very complex. This inhibits their performance and even leads to very large training times. This problem in the MNNs is solved by using modularity by MNNs.

The first task of problem solving by MNNs is to break the problem into a set of modules. This breaking is done at the time of design of the system itself. Whenever the problem is given to the system, it is first divided among the various modules. Then all the modules independently solve the problem or their part of the problem. All the solutions are calculated independently by the various modules and the results are supplied to a central integrator. The integrator does the task of combining all the individual results of the various modules and giving the final answer of the system to the supplied input.

One of the major tasks in the use of ensembles is the breaking up of the problem into a set of modules. The modules need to be cautiously di-

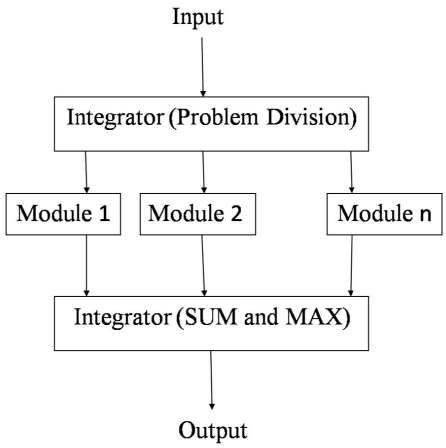
vided keeping the performance maximum without compromising much with the generality of the problem. It may be easily observed that as we increase the number of modules, the problem starts becoming more and more localized in nature. We know that localization is not a desirable thing in any system as it hinders the algorithm giving correct results when exposed to unknown inputs. But this is desirable till the added benefits of training time and performance give a boost to the system.

Ensembles are a popularly used type of MNNs especially for such classificatory problems (Kala, Shukla and Tiwari, 2009). We would only be discussing the particular model being used for problem solving in this chapter. Other approaches and techniques are not discussed.

The first step is training of the system. Each of the modules or the ANN is trained independently by all the training data present in the system. All the ANNs may be trained in parallel. Here the output of each ANN is the probability vector that denotes the presence or absence of each of the class. Say the system has n classes. Each ANN gives as output the probability vector $<p_1, p_2, p_3, ..., p_n>$ where each p_i denotes the probability of occurrence of the class *i*. Here we measure the probabilities in the scale of -1 to 1 with 1 denoting the maximum probability and -1 denoting the least.

In the implemented approach whenever the input is given to the system, it distributes it to all the various modules present in the system. Each module is a separate independent ANN. The modules analyze the problem and work over a solution. They then return a probability vector containing the probability occurrences of each of the classes. Say if the classificatory problem had *n* classes to which the input could belong to, each ensemble returns the probability vector $<p_1, p_2, p_3, ..., p_n>$ where each p_i denotes the probability of occurrence of the class *i*. Ideally only one out of all the numbers in this vector must be 1 and all others must have -1, but the same may not be the case due to the imperfection in the system.

Figure 2. The ensemble using probability based approach

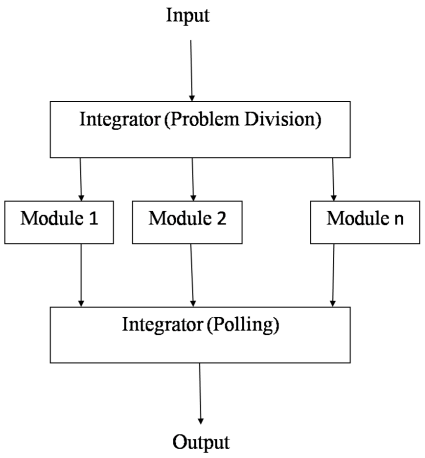


All the various probability vectors are given to the central integrator as per the principles of the MNNs. The integrator receives all the probability vectors and does the task of deciding the final output of the system. For this the first task is to average the probabilities of the various probably vectors. This makes a final probability vector that combines the results of all the modules. This is a mechanism where the integrator collects the individual responses of the modules to the problem to get the average response. Then the integrator selects the class that gets the maximum response or has the maximum probability of occurrence. This class is returned as the final class. This concept is given in Figure 2.

Another classical method to solve the problem is by making use of polling or voting mechanism. Here each module returns the class which it believes is the class to which the input belongs. The integrator gets all the various classes by different modules and then carries a voting in between the various modules. The class that gets the highest votes is decaled as the final winner. This concept is given in Figure 3.

The key aspect of the ensembles is the use of various models or ANNs in parallel. We know

Figure 3. The ensemble using polling approach



that it may not always be possible for every ANN to train itself well as per the problem. This may be due to the large complexity of the problem or data size and sometimes even noise. It may further not be wise to keep adding the neurons to the system as they have a negative impact in the generalizing capability of the system. Hence we make use of several ANNs. Now each ANN may be at some other configuration. The differences may be in the weights, biases, architectures or even the model of ANN that is used altogether. As a result they showcase different capabilities and different performances. They all may be said to specialize in different aspects. Now we need to design an integrator in a manner that makes the best use of the capability of each and every module or ANN. The sub optimality in the training of the ANNs are removed by the other ANNs. Hence the whole system gives a better performance to the problem. This makes the ensembles valuable tools for solving problems that gave poor performance with the use of single ANNs.

2.3 Evolutionary Artificial Neural Networks

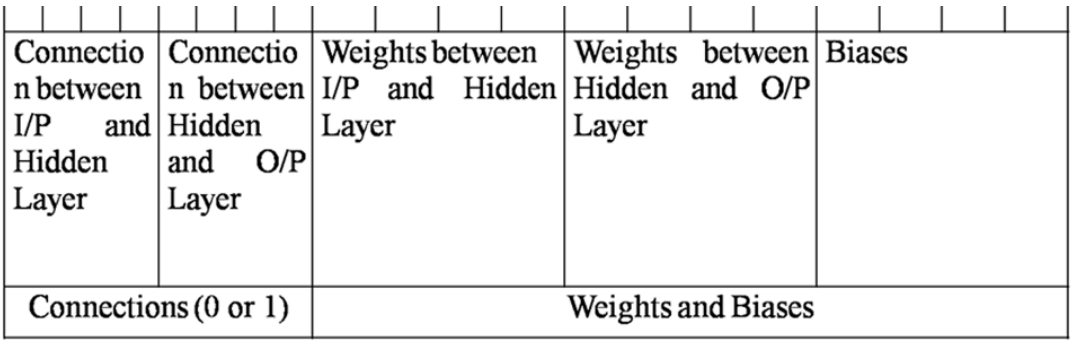
The last hybrid approach that we use to solve the problem is Evolutionary Neural Networks (He, Wu and Saunders, 2006; Augusteijn and Harrington, 2004). In this approach we try to train the ANN using GA. The BPA used for the training of ANN has many limitations that are solved by the use of GA. The ANN may get trapped in local minima. Also we have to specify the architecture at the start of the algorithm which may not be optimal in accordance with the problem.

The Evolutionary ANN technique tries to evolve and train the ANN by using the optimization property of GA. It may easily be seen that ANN training is essentially an optimization problem where the performance or the error function acts plays the role of objective function. The weights, biases and even the architecture plays the role of the modifiable parameters. These systems may be used only for the ANN training with fixed network architecture or for evolving the architecture as well. The former is a much simpler problem with a limited size of the search space as compared to the later which is more complex a problem with the search space spanning to a lot of dimensions.

Here we discuss the model of Evolutionary ANN that we follow for solving the problem of disease identification. This is a connectionist approach. Here we tried to evolve the ANN weights as well as the correct connectionist architecture. The ANN that we generally take for problem solving in the problems is a fully connected model where every neuron of every layer is connected to all the neuron of the next layer. This architecture leads to a great demand of computation that increases the overall training time. Subtracting neuron may not be that vital as the network may not train for smaller number of neurons. If the network fails to train, the general approach is to add a neuron. But imagine the immense increase in dimensionality as a result of this addition of a single neuron. This expands the dimensionality by a big amount. This may make it very difficult for the training algorithm to train the network as a result of the same.

Hence we make use of the concept of a connectionist approach which tries to inhibit or stop

Figure 4. The chromosome representation



certain connections to limit the computation time as well as overall complexity of the problem. This makes the calculations very fast and even the further training can take place easily as the connections are limited. In this problem we use GA to evolve connectionist architecture as well as to fix the ANN weights. The maximum number of neurons however needs to be defined at start.

The most important part in the implementation of the GA for the problem is the individual representation. The individual representation deals with the manner in which the solution of a problem would be represented. We represent the solution which is an ANN in our case as a set of numbers. This is divided into two halves for this problem, comprising of two and three sections respectively.

The first section of the chromosome consists of a 0 or 1 depending upon the existence of connection between the input layer and the hidden layer neurons. A 1 depicts the existence of connection and a 0 depicts an absence of connection. Similarly the next section depicts the existence if weight between a hidden layer neuron and an output layer neuron. The bits have their same meaning. The other half of the chromosome contains the actual values of weights between the various connections. Here the first section of the second half contains weights between input and hidden layer neurons and the second section between hidden and output layer neurons. These are the actual values of connections that hold if the connection exists or not. If the physical connection is not there, the corresponding weight value may be ignored. The last part contains the values of the biases. The general structure of the chromosome is shown in Figure 4.

The other important factor in the use of the GA is the genetic operators. They enable the generation of individuals from a lower generation to a higher generation. Selection is the genetic operator that deals with the selection of individuals from one generation that go to the higher generation. Selection follows Darwin's theory of survival of the fittest and hence the fittest individuals are selected. The solutions are first ranked and then stochastic selection is applied over the ranks as weights. This prohibits over-domination of very fit individuals in the population pool.

The other important operator is crossover. This operator tries to fuse two parent solutions and generate a child solution or chromosome. The crossover is modified to match the requirement of the problem. The crossover used is a 2-point crossover which was applied in such a way that the 1st point always lay in the sections that constitute the architecture and consisted of the bits. The second part always lay on the section containing the values of weights and bias. These were the real values.

Mutation is the genetic operator that tries to add new characteristics in the individual. It is an attempt to explore the search space by the GA. If the new added characteristics are good, they would be retained by the algorithm and the newer individuals may try to add those characteristics. If these characteristics are bad, the individual may get deleted in the next run by the phenomenon of the survival of the fittest. Similarly mutation is modified to ensure that the section reserved for bits always contains only one of the 2 valid inputs i.e. 0 and 1.

The fitness function is a function that measures the goodness or the optimality of any solution. The population is optimized by the GA in its run when the optimality is measured by the fitness function. Here the individual is an ANN with some architecture and corresponding weights and biases. The fitness is the accuracy of this ANN in predicting the presence of the disease. This may be measured as a percentage of the total number of correct predictions by the system out of all the given test cases.

In order to determine the extreme values or ranges of weights and bias, we see some trained ANNs of fully connected types and then the extreme values of the weights and bias that they take. The selected range is one that comfortably covers the observed range at the same time not resulting in the search space or the configuration space from becoming very vast in nature.

3. METHODOLOGY

The aim of the system is to solve the problem of detection of PIMA Indian Diabetes. For this we make use of the database of UCI Machine Learning Repository (Sigillito, 1990). The PIMA Indian Diabetes data set consists of a total of 8 attributes. These decide the presence of diabetes in a person. This database places several constraints on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

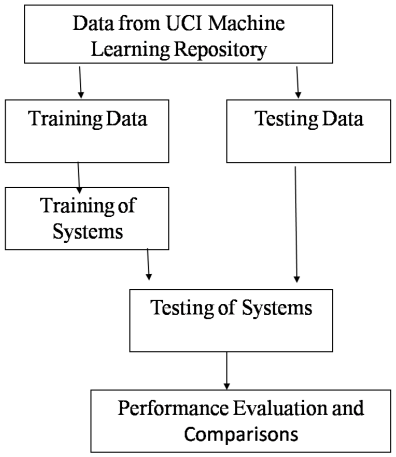
The first attribute is the number of times the women was pregnant. The next attribute is Plasma glucose concentration a 2 hours in an oral glucose tolerance test. We further have the attributes Diastolic blood pressure (mm Hg), Triceps skin fold thickness (mm), 2-Hour serum insulin (mu U/ml), Body mass index (weight in kg/(height in $m)^2$), Diabetes pedigree function and Age (years).

The whole methodology may be divided into two parts. These are training and testing. The database contains a total of 768 instances of data. 535 (~70%) instances of data of the data were used for the training purposes and the rest 233 (~30%) of the data was used for the testing purposes.

In the training phase the various hybrid systems are given the historical database and trained according to the individual training method. The data used is the training database. The purpose of training is to expose the system to new instances so that it can tune its parameters. Training is performed by the training algorithm that is specific to the individual system. All the systems try to extract the rules or patterns out of the given database while training. The aim is to train the system such that is has the highest degree of generality. Training time may again depend upon the problem in hand and the network design. The training may be carried out again and again with different parameters of the designed system to see the performance.

The next stage involved is testing. Here the system is given unknown data from the testing data set. The system output is collected and compared with the standard output. This helps us in the determination of the performance of the system. The overall performance of the system is its performance at the testing phase. This needs to be as high as possible. Here it is important to note that increasing the number of neurons in ANN or membership functions in FIS may generally give a high system performance at the time of training. However, this may not be true in case of testing data. We would

Figure 5. The general working methodology



observe that as we increase the number of neurons in ANN or membership functions in FIS, the training performance keeps increasing and the testing performance keeps reducing. This is due to the fact that increased number makes these systems more localized in nature and generality is lost. Hence the system suffers. At the same time it should be noted that too few neurons or membership functions may again not be able to solve the problem. Every data needs to have sufficient neurons or membership functions to be able to model the problem as per the data available.

The general methodology (Shukla, Tiwari and Kaur, 2009a and 2009b) may be summarized by Figure 5.

4. RESULTS

The basic objective is to analyze the behavior of various hybrid intelligent systems for the detection of PIMA Indian diabetes. In section 2 we discussed the various methods from a theoretical point of view. We saw the way in which the different systems work and solve this classificatory problem. Further the data set chosen consists of numerous attributes that help in the diagnosis of the problem. In this section we experimentally see the behavior of each of the discussed systems for solving the problem. We compare the various methods with respect to their ability to train, learn and generalize the data. The ultimate aim is to have a larger generalizing capability that would mean an effective detection of the disease in the final system. In the next sub-sections we present and compare the results of the various algorithms and hybrid systems that we use.

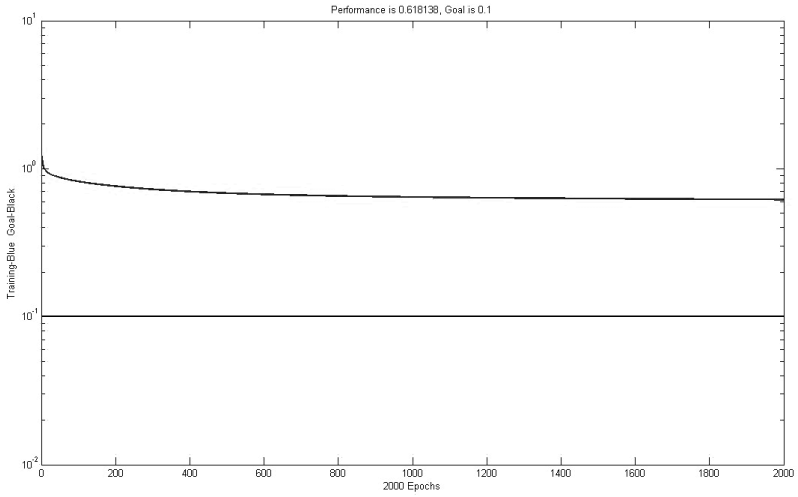
4.1 ANN with BPA

The first method used for the problem was ANN with BPA. MATLAB was used as a platform for the implementation. Here we used a single hidden layer which consisted of 12 neurons. The activation functions for the hidden layer was *tansig* and *purelin*. The training function used was *traingd*. The other parameters were a learning rate of 0.05 and a goal of 10^{-1}. Training was done till 2000 epochs.

It may be seen that ANN with BPA is primarily a good regression agent. It tries to form smooth functions that behave as per the trends in the training data set. The entire network tries to form a generalized function that acts as needed. Ideally the network must always output 0 or 1 denoting the absence or presence of the disease. However this may not be practically possible due to numerous reasons. The system always gives continuous outputs. The output is closer to 1 for affected cases and closer to 0 for the non-affected cases. We place a threshold here that helps in the decision of the presence or absence of diseases. Any value of the network above this threshold is taken to be 1 and values less than this threshold are taken to be 0. In this problem we fix the threshold as 0.5.

After the network was trained and tested, the performance of the system was found out to be 77.336% for the training data set and 77.7358% for the testing data set. The training curve of the ANN is given in Figure 6.

Figure 6. The training curve for ANN with BPA



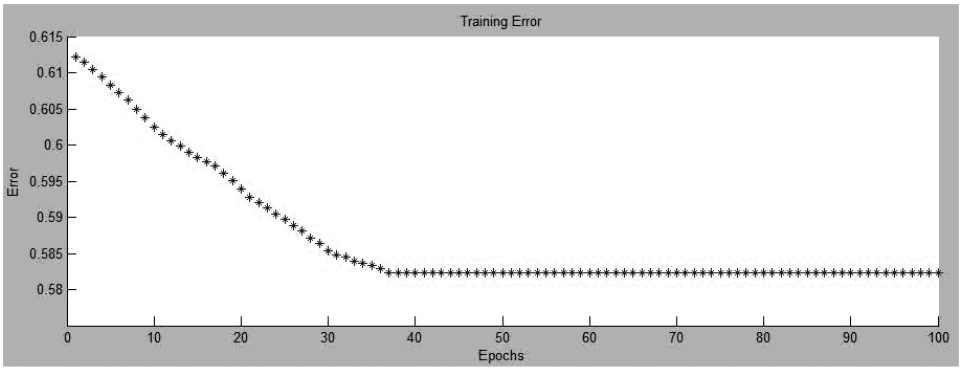
4.2 Ensemble

The second experiment was done on ensembles. Here we had used 4 modules or ANNs. Each one of them was trained separately using the same training data set. MATLAB was used as an implementation platform. All 4 ANNs were first trained independently one after the other. Then all these ANNs were used to make an integrator which gave the inputs to each of these ANNs, collected their outputs and then gave the final result of the system. The 4 ANNs were more or less similar with small changes. The first was exactly the same as discussed in section 4.1. The other ANNs only had changes made in the number of neurons in the hidden layer and the number of epochs. These had 14, 10 and 12 neurons respectively. The numbers of epochs were 2500, 200 and 4000. The four ANNs were trained separately.

Making the architecture of the ANNs different makes them behave differently and lie at different locations in the error space. This difference helps a lot in creating disparity in the modules. We know that exactly same modules may not be helpful to us as they would essentially convey the same information and behave as same networks. Another interesting fact to note is that even if the architectures of two ANNs are different, they would behave differently after training. This is because the ANN with BPA algorithm starts the training by randomly setting the weights and the biases. This makes the ANN place at some random location in the entire error space. As the training goes on, the error is tried to be minimized. This happens by the movement of the ANN in the error space in exploration of the global minima. Even now time is a major factor that decides the final position after a set number of epochs. Hence the ANNs behave as different modules in the final system that emerges.

Here we had used a probabilistic polling in place of the normal poling. The resulting system had a total performance of 78.7276% for the training data and 76.9811% for the testing data. It may be noted that the performance was 78.33% for the training data and 76.2264% for the testing data in the use of conventional ensemble where each module votes for some class.

Figure 7. Training curve for ANFIS



4.3 ANFIS

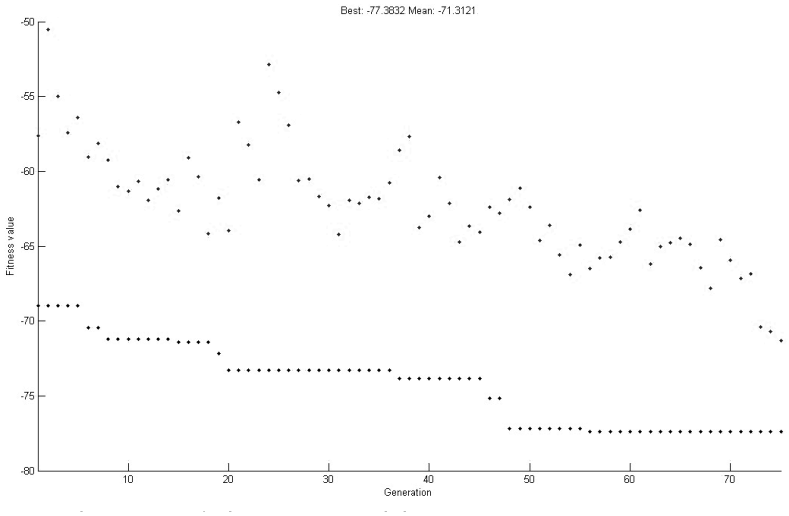
The next experiment was done using ANFIS as a classifier. Here we used the same training as well as testing data sets. The FIS was generated using a grid partitioning method. Each of the attributes had 2 MFs with it. The system was allowed to be trained for a total of 100 epochs. We discussed in section 2 earlier that the number of membership function play a major role in deciding the generalizing capability of the system. The higher number of membership function show less generalizing behavior. Another major problem with large number of MFs is the training time. The larger number of MFs would mean a larger number of modifiable parameters. This is because of the fact that more rules may be formed by these membership functions which add to the system complexity and the overall parameters in the system. Now training of larger number of parameters requires more computational time. Also more complex system would require more computation for calculation of the outputs at every input. This greatly limits the training of the overall system. Further the larger number of modifiable parameters needs a larger database for the training. This further increases training time. In many problems it might also not be possible to increase the database.

The resulting training graph is shown in Figure 7. The final system so obtained had a performance of 88.9720% for the training data and 66.5236% for the testing data.

4.4 Evolutionary ANN

The last hybrid system that was applied to the same problem was evolutionary ANN. Here we tried to evolve the ANN weights as well as the correct connectionist architecture. The first job was to make a fitness function that would be later on used by the Genetic Algorithm for optimization. The fitness function creased a neural network. The connectivity between the various neurons is provided as parameters of the function. The unconnected components are kept as zero. The weights and biases are also the parameters of the function. After the formulation of the network, the task was to compute the efficiency of the system. This was done by simulating the training data by the GA. The performance of the GA in the training data was noted which denoted the fitness of the ANN or GA individual. The larger number of connections was penalized by the fitness function to encourage simpler networks. We then build GA to optimize the various parameters of the fitness function using the MATLAB toolkit. The GA hence fixed the values of the various parameters

Figure 8. Evolutionary ANN training curve



that consisted of the connections, weights and the biases. At the end of the required number of generations, the best individual was used as the final most optimized ANN. Then testing data set was used to test the performance of this ANN.

The parameters of the GA were a maximum number of 25 neurons, 25 as the population size with an elite count of 2. The creation function was uniform and double vector representation was chosen. Rank based fitness selection was used. Stochastic Uniform selection method was used. Crossover ratio was 0.8. The algorithm was run for 75 generations. The training curve is shown in Figure 8.

The final system had a performance of 77.38% for the training data set and 73.819% in the testing data set.

5. FUTURE RESERCH DIRECTIONS

Based on the work and ideas presented in the chapter, a lot many directions are laid open. The presented work and comparisons talk about the application of hybrid methods for the detection of PIMA Indian diabetes, no comparison is made on the basis of other diseases. These diseases may give different trends or reveal different results. Databases to a large number of diseases are already available in numerous public data repositories. These may be worked over by the readers. The different diseases and the associated datasets differ in the number and type of attributes. Many times this makes them showcase different characteristics altogether.

Many more hybrid methods are available in literature. These may be tried for better results to the problem. Large data set size is another upcoming problem that is being studied in many ways. The challenge is to design systems that can handle large complex data in computationally finite time without loss of generalization. Also the present disease does not have any missing value. Medical data is usually filled with a lot of noise, uncertainties and altogether missing values. This requires systems that can handle uncertainties and extract the maximum out of the available data. The use of better machine learning techniques holds the key for greater accuracies. All these may be done in the future.

The completely multi-disciplinary nature of the field puts a serious limitation over its growth and development. A lot many diseases are yet to come under the hood of computational intelligence. The task of making expert systems for these diseases involves the identification of the attributes, making of database with these attributes, machine learning, training, testing and validation. The iterative nature of correcting the model at various stages to achieve greater diagnosis efficiency is a major task. This may be another focus for the readers for contribution and work in this field.

6. CONCLUSION

In this chapter we saw the working of four different Hybrid methods for the problem of detection of PIMA Indian diabetes. In all the cases we were able to solve the problem with fine accuracies using the intelligent hybrid approaches. The first experiment done was of the use of ANN with BPA. These systems used the generalizing capabilities of the ANN for problem solving. We were able to get sufficient accuracies using these systems. The ANN with BPA was a simple system. Various limitations of the BPA motivated the use of hybrid systems for solving the same problem. The next implementation was using Ensemble. These systems removed the shortcomings of ANN with BPA. This resulted in a rise in accuracy of the systems. A number of ANNs were applied in a modular manner and final integrator took the results of all these ANNs to get the final output. Next we applied ANFIS to the same problem. The results reveal that the system performed very well on the training data set but did not do that well on the testing data set. This may be attributed to the failure of the system to generate rules that could be generalized over the network. At the training phase, the system must have adjusted the MFs so as to meet the requirements, but the rules so generated failed to generalize. This may be even because of the selection of the wrong data that essentially belonged to some parts of the network. The last approach was using the Evolutionary ANNs. These systems gave good performances where they could decide their own architecture besides the values for the weights and biases.

All the systems were implemented on the PIMA Indian Diabetes database. The database records the presence or absence of diabetes in the Indian female patients. The ultimate aim is to make an expert system that would assist the doctors in diagnosis of the disease. This would prove to be a very useful system considering the present scenario where diseases are on a hike and there is a lack of availability of specialized doctors. Such a system learns from the past data which is a collection of a lot of information in itself. The system tries to extract this hidden information to make a generalized system for the detection of the disease.

This chapter reveals the functioning of the various hybrid approaches in problem solving and disease detection. We saw that all the methods were competitive and hence any generalization cannot be made regarding the effectiveness of a method over the other which is well supported by literature which does not lay any preference to any one of the hybrid approach for problem solving. The only exception was in the use of ANFIS which could not form generalized rules. It may be seen that the problem solution depends also upon the division of data between the training and testing data. Besides, there is a lot of dependence on the choice and measurement of attributes and the diseases at large. While some methods may look better for the problem of PIMA Indian diabetes, it cannot be guaranteed that the results would observe same behavior for other diseases. This again necessitates on the knowledge of both theoretical and practical aspects of the various hybrid methods and an iterative design approach to get a good soft computing system for problem solving.

It may again be noted that though we have built effective systems, we were still not able to get an accuracy of 100%. In a field like bio-medical engineering wrong decisions can greatly harm a patient's health. This imposes a big limitation in the use of these systems for completely autonomous diagnosis of disease. No matter how better our systems are engineered, for a very long time we would require the supervision of doctors for the final verdict. We have a very long way to go before the accuracy can be made 100%, enabling systems to perform autonomously.

REFERENCES

Augusteijn, M. F., & Harrington, T. P. (2004). Evolving transfer functions for artificial neural networks. *Neural Computing & Applications*, *13*(1), 38–46. doi:10.1007/s00521-003-0393-9

Bunke, H., & Kandel, A. (Eds.). (2002). *Hybrid Methods in Pattern Recognition*. New York: World Scientific.

Ciampi, A., & Zhang, F. (2002). A new approach to training back-propagation artificial neural networks: empirical evaluation on ten data sets from clinical studies. *Statistics in Medicine*, *21*, 1309–1330. doi:10.1002/sim.1107

He, S., Wu, Q. H., & Saunders, J. R. (2006). A Group Search Optimizer for Neural Network Training. In *Proceedings of Computational Science and Its Applications* (pp. 934–943). ICCSA. doi:10.1007/11751595_98

Kala, R., Shukla, A., & Tiwari, R. (2009). Fuzzy Neuro Systems for Machine Learning for Large Data Sets. In *Proceedings of the IEEE International Advance Computing Conference*, (pp. 541-545), Patiala, India.

Melin, P., & Castillo, O. (2005). *Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing*. New York: Springer.

Shadabi, F., Sharma, D., & Cox, R. (2006). Learning from Ensembles: Using Artificial Neural Network Ensemble for Medical Outcomes Prediction, *Innovations in Information Technology*, 1-5

Shukla, A., Tiwari, R., & Kaur, P. (2009a). Diagnosis of Epilepsy disorders using Artificial Neural Networks. In *Proceedings of the Sixth International Symposium on Neural Networks*, (Advances in Intelligent and Soft Computing, Volume 56, pp. 807-815). Berlin / Heidelberg, Germany: Springer/Verlag.

Shukla, A., Tiwari, R., & Kaur, P. (2009b). Knowledge Based Approach for Diagnosis of Breast Cancer, In *Proceedings IEEE International Advance Computing Conference (IACC)*, (pp. 06-12). Patiala, India.

Shukla, A., Tiwari, R., Meena, H., & Kala, R. (2009). *Speaker Identification using Wavelet Analysis and Modular Neural Networks, Journal of Acoustic Society of India*. JASI.

Sigillito, V. (1990). UCI *Machine Learning Repository*, The Johns Hopkins University. Retreived from http://archive.ics.uci.edu/datasets/Pima+Indians+Diabetes

Srinivasaa, K. G., Venugopala, K. R., & Patnaikb, L. M. (2007). A self-adaptive migration model genetic algorithm for data mining applications. *Information Sciences*, *177*(20), 4295–4313. doi:10.1016/j.ins.2007.05.008

Temurtas, H., Yumusak, N., & Temurtas, F. (2009). A comparative study on diabetes disease diagnosis using neural networks. *Expert Systems with Applications*, *36*(4), 8610–8615. doi:10.1016/j.eswa.2008.10.032

Vosoulipour, A., Teshnehlab, M., & Moghadam, H. A. (2008). Classification on Diabetes Mellitus Data-set Based-on Artificial Neural Networks and ANFIS. In *IFMBE: Proceedings of the 4th Kuala Lumpur International Conference on Biomedical Engineering 2008*, 27-30

Xiaoguang, C., & Lilly, J. H. (2004). Evolutionary design of a fuzzy classifier from data. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, 1894–1906.

KEY TERMS AND DEFINITIONS

Artificial Neural Networks: Networks consisting of a number of artificial neurons arranged in a layered manner, where each neurons takes input, processes it and gives it to the out neuron for further processing, or as system output. They are excellent agents for problem solving where some inputs are mapped to outputs in an unknown manner.

Classification: Problems consisting of a number of inputs or attributed which need to be mapped to a set of predefined classes. Each input is said to be belonging to some output class out of all the available classes.

Evolutionary Algorithms: Algorithms that generate a random set of solutions for the problem, which keep getting optimized along with iterations or generations. Each population of a generation aids in the formation of a fitter higher generation population with the help of specially designed genetic operators.

Error Space: A multi-dimensional space plotting the error values of the Artificial Neural Network (or any other system) for every combination of weights and biases (or other system parameters).

Expert Systems: Systems that behave like experts in performing any action. They can read the inputs, comprehend them, compute the necessary outputs and respond to the inputs. They can be trained for desired performance as well as they can learn from their experience.

Fuzzy Inference Systems: Logic based systems that use fuzzy logic and fuzzy rules to compute the output to any given input.

Generalization: The capability of a system to give correct outputs to the inputs that were not presented to it any time during training. The ability of a system to form rules or assume behaviors that are valid for a very large (or all) cases that may be presented to the system.

Hybrid Systems: Systems that are engineered by a combination of soft computing approaches along with heuristic techniques. The advantages of one overcome the disadvantages of the other method.

Input Space: A multi-dimensional space plotting the output values of the Artificial Neural Network (or any other system) for every combination of inputs for a specific system state.

Machine Learning: The act of reading the historical inputs (and outputs); forming patterns, trends or rules from them; and storing them in a summarized manner in the knowledge base.