Evolution of Modular Neural Network in Medical Diagnosis

Rahul Kala*, Harsh Vazirani, Anupam Shukla, Ritu Tiwari
Soft Computing and Expert System Laboratory
Indian Institute of Information Technology and Management Gwalior

Citation: R. Kala, H. Vazirani, A. Shukla, R. Tiwari (2010) Evolution of Modular Neural Network in Medical Diagnosis, *International Journal of Applied Artificial Intelligence in Engineering System* 2(1), 49 -58.

Abstract

Intelligent systems have been extensively used in the area of biomedical engineering for assisting the doctors in medical diagnosis. The inability of simple neural networks to solve the diagnosis problem, due to extensively large data size as well as complex mapping of inputs to outputs, has resulted in the growth of modular neural networks that try to exploit the modularity in the problem for better performance. Inability of humans to effectively set the training parameters of the neural networks has further resulted in the use of evolutionary algorithms that automatically compute the best configuration. In this paper we present a novel mechanism of combining the neural and the evolutionary approaches to build an evolutionary neural network classifier for solving the problem of diagnosis of PIMA Indian Diabetes. The modular neural network is in form of a mixture of exerts model with multi-layer perceptron, radial basis function network and self organizing maps as the experts. Results show that the proposed approach is better than the conventionally used models over the database used.

Keywords: Artificial Neural Networks, Evolutionary Neural Networks, Modular Neural Networks, Ensemble, Machine Learning, PIMA Indian Diabetes.

1. Introduction

There has been a growing need of computationally intelligent systems in the field of medical engineering. These systems are able to diagnose the disease automatically based on the attributes supplied. These systems hence become valuable tools for doctors for automatic disease diagnosis or for the computer assisted disease diagnosis. There has been a widespread use of these systems over the areas of detection of various diseases like Breast Cancer, Epilepsy, Heart Diseases etc.

The basic philosophy behind these systems is of machine learning. We try to figure out the attributes that may help in identification or diagnosis of the disease. We then design an intelligent system that takes the identified attributes as inputs and supplies the diagnosis result as output. The system formulates its own rules or logic that helps in the identification of the disease. Training input is supplied to the system to mine out the rules or logic in the training phase. It is expected that these rules may be generalized to new data or the testing data as well. In this manner the intelligent systems carry forward the task of diagnosis.

Since long, Artificial Neural Networks (ANN) are used for problem solving. A variety of models of ANNs exist like Multi-Layer Perceptron (MLP) with Back Propagation Algorithm (BPA), Self Organizing Maps (SOM), Learning Vector Quantization (LVQ), Radial Basis Function Networks (RBFN), etc. All of these use different types of fundamentals in learning of data and representation of knowledge. All these networks may be fundamentally classified into two basic types. The first type consists of the networks that try to solve problem as a curve fitting approach. The second type consists of the networks that try to solve the problem with a classificatory approach, where the applied input needs to be classified into either of the possible output classes.

This paper is organized as follows. In section 2 we give a background of the work and the basic terminologies used. Section 3 presents a brief literature survey in the domain of machine learning. Section 4 gives the Modular Neural network base of the algorithm. Section 5 presents the evolutionary neural network base of the algorithm. In section 6 we give the results. Conclusion remarks are given in section 7.

2. Background

In this paper we would be making frequent references to a number of terms that form the background of this work. Hence in this section we present all these terms and concepts. This would form the base of the algorithm and the rest of the discussion.

2.1 Evolutionary Neural Networks

Evolution is a major factor that guides the design of systems today. This class of algorithms carries the evolution of the artificial systems along with time or generations. Evolutionary algorithms mainly carry out the task of optimization. We are given a set of parameters and we are supposed to maximize the system output or the fitness value by finding the most optimal combination of the values of these parameters. A sophisticated use of evolutionary algorithms enables the evolution of the complete neural network or any system. Evolutionary algorithms are capable of maximizing the system performance by finding the correct set of weights and biases of the ANNs that constitute the modifiable parameters. These systems are also able to evolve the architecture of the neural networks as well. This forms a fascinating system that has benefits from the conventional neural networks where the parameters are set by humans. The humans usually make sub-optimal decisions in the setting of the number of neurons or other parameters. The evolutionary algorithms make use of computer program to do the same. Hence the evolved system is optimal in nature.

A problem with the evolutionary algorithm is their time consuming nature. A very large number of computations often create a problem in systems where the solution may be generated after unbearably long span of time. Hence the use of a local search strategy is usually suggested that aids the evolutionary algorithm in its search operation. The local search strategy is very effective in finding the local minima in the vicinity of an individual in the search space. The evolutionary algorithms have unexceptional power to find the global optima in the entire search domain. The combination of these two strategies gives good results.

2.2 Modular Neural Networks

Another important concept is that of the Modular Neural Networks (MNN). The data set sizes are increasing drastically with the rise of computation. Also more and more problems are coming under the domain of machine learning and computational intelligence. The large amount of data and problem complexity demands the construction of very sophisticated neural networks for problem solving. It may not always be possible to engineer such large networks due to computational constraints where we can only make systems of a limited complexity. Further we can only handle a limited amount of data in these systems. This gives rise to the use of modularity in ANNs for problem solving.

The MNN is a novel concept in this regard. These systems try to exploit the modularity in the problem. The entire problem is first divided into parts. Each part is given to a separate module for solution computation. The different modules solve their part of the problem. The solutions are returned by these modules. These solutions come to a central integrator. The integrator does the task of combining the results of the different modules and giving the final result that is the output of the system. In this manner these systems perform the task of problem solving. There are two important terms here. The first term is the integrator. This is the unit that is responsible for the division of the problem into modules and the integration of the outputs of the different modules. The other term is the modules which are independent neural networks.

The division of problem into modules does result in some loss of generality. However as a result of this division the system is able to learn the data that would not have been possible without the module decomposition because of complex mapping or large volumes of data.

3. Literature Review

The basic motivation behind the paper is hybrid systems. A lot of work is being done in order to develop hybrid systems for various problems. Rutkowski [25] presented the flexible neuro fuzzy systems. This is a widely used hybrid system that has largely reformed the manner of working of Fuzzy Systems. Kuo et al. [14] proposed a hybrid model for learning fuzzy if-then rules. These have been applied to variety of problems [10, 15, 31]. A lot of work is going on in the various parts of these algorithms like clustering, genetic optimizations, rule forming, parameter updating etc. [6, 19, 27]. The algorithms try to optimize the performance in clustering by designing various models based on the architecture of neuro-fuzzy systems [4, 16, 17, 23, 33].

Adams [1] proposed the use of Genetic Algorithms to decide the connectivity of associative memory that can be generalized to artificial neural networks. Maravall et al. [18] also proposed a hybrid model of Genetic Algorithms with reinforcement learning for robotic controllers.

Classification is a major problem of study. People are trying to better adapt the present algorithms to make them more suitable for classificatory problems. Amin et al. [2] presented a model of single layered complex valued artificial neural network for classificatory problem. Here they had made use of new activation functions. Hu [12] used Choquet Integral with neural network and genetic algorithms for pattern classification. Estudillo et al. [7] proposed multiplicative nodes instead of additive in neural networks to build neural network classifiers. Ang et al. [3] used probability to genetically evolve a neural network from smallest size to larger sizes.

Genetic Algorithms are being constantly studied and modified for better performances at various situations. One of the major landmarks here are the Particle Swarm Optimizations. Nani used the Nearest Neighbor approach for prototype reduction using Particle Swarm Optimization and the found the Nearest Neighbor approach to be good [18]. Grammatical Evolution is another area of work. Tsoulus et al. [32] proposed grammatical evolution for neural network construction and training. Neill et al. [21, 22] has highlighted various developments in these fields in terms of representation and working of the algorithm. Ryann [26] proposed these algorithms for program generation in arbitrary language.

A lot of work is being done in the field of neural network for validating, generalizing and better training of the neural network [5, 8, 9, 29]. Classification also employs the use of Hidden Markov Models [11, 28, 30]. These are statistical models that can be used to predict the consequence of the unknown input. These models have found a variety of use in handwriting recognition, speech recognition etc. Instantaneously trained neural networks [13, 24] are a good approach for faster training with a smaller generalization capability. These networks require very less training time as the weights are decided just by seeing the inputs. Self organizing maps have also been used extensively for the problem solving. Various other mathematical models have been proposed. These employ mathematical techniques like point to point matching for solving the problem.

4. Modular Neural Network Framework

In this section we discuss about the modular approach used by this algorithm. As per the modular approach, we first need to divide the problem into modules. Each module is an ANN. We use three different models of ANNs here. The entire problem is given to each of the modules or ANNs. The first model is Multi-Layer Perceptron (MLP), the second model is Self Organizing Maps (SOM) and the third model is Radial Basis Function Network (RBFN). Each of these networks may solve their part of the problem using their standard methodology. After solving the problem, the results are given to the central integrator. The integrator has a voting mechanism. Here voting is done in favor of the classes. The class getting the largest votes is declared as the winner. This may be represented by figure 1.

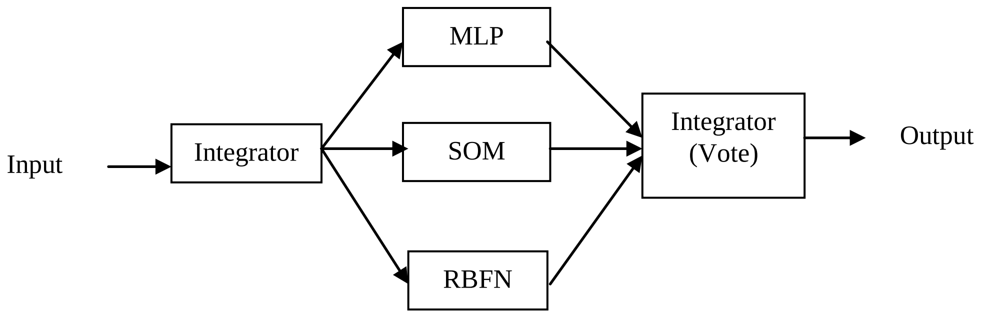

Fig. 1. General Architecture of the Modular Neural network used

4.1 Problem Division

The problem division simply gives the entire input vector to all the modules in the system. Each of the modules or ANNs hence works over the same data item and tries to figure out the final output class. Each module uses its own methodology to process the input and decide the output class to which the applied input may belong.

This type of working has a great relevance in classification. Every system makes its own decision boundaries to separate the various classes from each other. The various systems have different issues, advantages and disadvantages in the making of these boundaries. Hence they all perform well in some part of the feature space and do not perform well in some other part of the feature space. The use of different modules or ANNs assures that some modules would have a good idea about most of the data that may come in real life. These would contribute towards the decision making regarding the final output class.

4.2 Polling

A polling mechanism is used in his algorithm as the integrator. The output from each of the modules is a class that was computed as the final output class by these modules. The integrator assesses these outputs for deciding the final output. A voting mechanism is used. Each module or ANN output is a vote in favor of its computed class. The class getting the maximum votes is declared as the winner. In case of a tie, any of the classes may be randomly selected and returned as the output.

4.3 Training Algorithms

Each of the ANNs used above is trained by their own training mechanisms. It is true that EA is supposed to train these systems, but these training algorithms aid the EA in locating the nearest local optima in the vicinity to any genetic individual. This leaves the EA to only be concerned about locating the global optima. These training algorithms carry forward the task of local search.

5. Evolutionary Framework

In this section we open a completely new paradigm to the algorithm by giving it an evolutionary nature. The evolution is carried independently for all the three modules. The evolutionary nature of the different algorithms is however quite similar to each other. Each of these algorithms makes the same number of generations at the same instance of time. Hence there is a dependency between the various parallel running evolution processes that are otherwise independent. Another dependency added is in terms of the fitness evaluations where every evolutionary ANN uses the best networks of the other evolutionary ANN for fitness computation. This gives the entire framework a cooperative evolution nature. The general framework is described in figure 2.

5.1 Individual Representation

The individual representation is different for all the three ANNs. In MLP the individual simply stores the different neurons along with weights and biases. The activation functions are assumed to be fixed. It is assumed that there is only 1 hidden layer. The number of neurons in this layer can be variable. Hence the total size of the evolutionary individual is variable.

In SOM, the individual holds the number of neurons in each of the two dimensions of the 2-dimensional feature map. It further stores their positions in the feature space.

The other network is the RBFN. Here also we store the locations of the various neurons in the feature space along with the spreads. The weights from the hidden to the output layer are assumed to be constant and unity. All the networks have variable size of the individual.

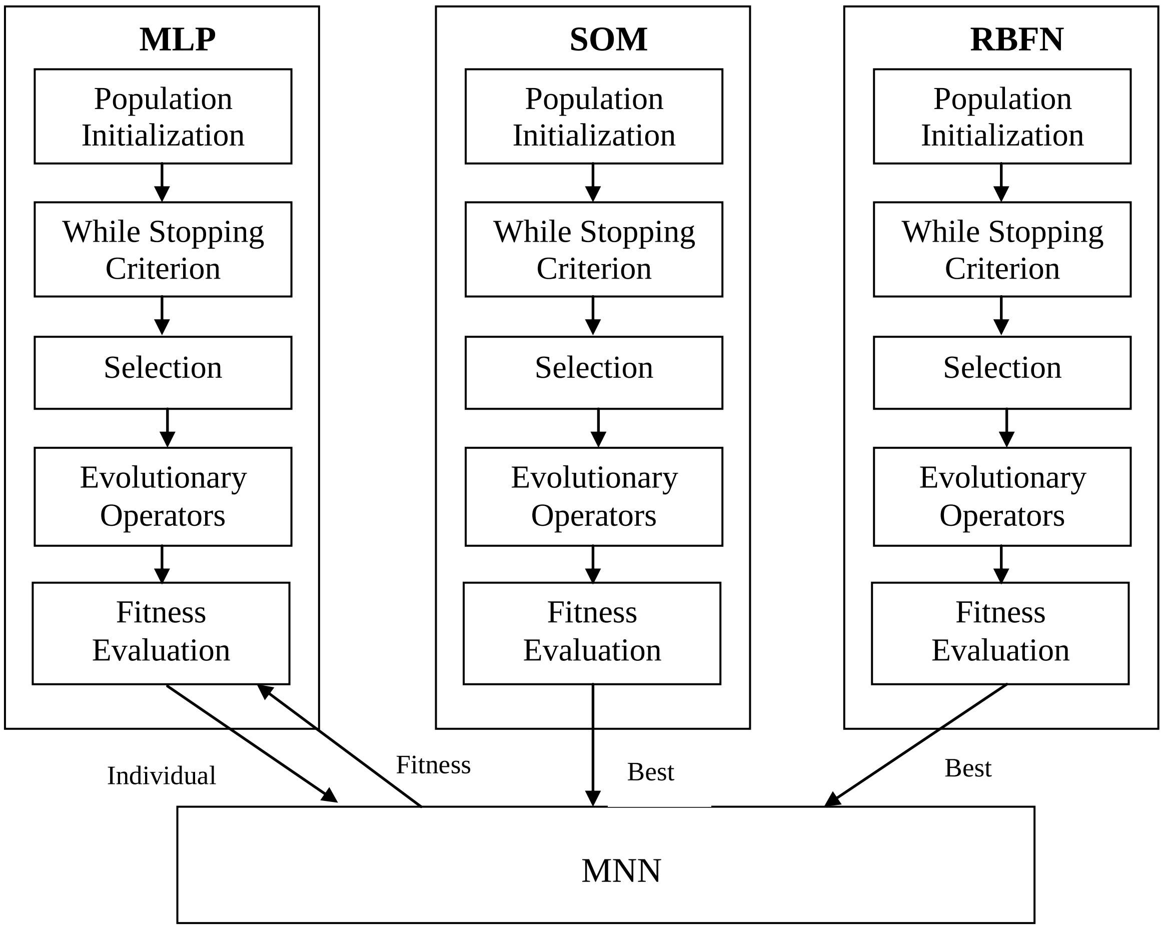

Figure 2: The general architecture of the algorithm. Figure assumes evaluation of MLP. The various modules would carry evaluation in a cyclic manner.

In this paper we have used an incremental evolution technique. The maximum number of neurons in any model (*L*) is set by the system. This number is initially kept to a very low value. This restricts the entire evolutionary process to generate larger networks. As a result the evolutionary process tries to solve the problem given using smaller number of neurons. As the generations increase, this number is increased. This allows the system to develop larger networks with high number of neurons in the hidden layer. The increase in the maximum number of neurons *L* at any generation g is given by equation (1)

$$L(g)=L(0)+L_{max}\, g/G \qquad (1)$$

Here *L(0)* is the least number of neurons, which may normally be kept as 1 and *G* is the maximum number of generations. L_{max} is the maximum change in *L*.

5.2 Evolutionary Operators

Evolutionary operators are used to generate higher generation population from a lower generation population. Again all three ANNs have a different methodology of evolution.

The first major operator applied is crossover. This generates two children from two parents. The number of neurons of the first child is the same as the first parent and the number of neurons of the second child is the same as the second parent. The individual weights are taken randomly from either of the two parents. This is similar to the application of scattered crossover per neuron in a sequence. In this way we get the two new individuals of the next generation ready.

Mutation is applied as a genetic operator. This causes small changes in the various weights and locations that are determined by a Gaussian Random mutation technique. Grow is an operator that adds a new neuron to the network. The addition of the neuron takes place with randomly with some probability. Add is another operation that is applied in this evolutionary technique. This model adds random individuals of the maximum allowable size as per the present generation. Elite transfers the best individual from one generation to the other generation without any modification. This saves the best individuals from being killed in the genetic process.

5.3 Fitness Evaluation

The fitness evaluation is simply the performance of the network over the training data. Here performance is the correct recognition rate of the overall system. Based on the discussions in section 4, we know that the overall system is made up of three ANNs. At any generation we measure two performances that contribute towards the final fitness of the system. The first is the performance of the individual or the ANN alone. The second is performance in the MNN architecture. For the MNN evaluation the individual is taken along with the best individuals or other parallel running evolutionary ANNs. The performance of the entire MNN so formed is measured. This judges the capability of the individual to complement the other ANNs.

We know that very large networks may give a good accuracy in the training phase but a poor accuracy in the testing phase. We hence need to penalize the larger networks by assigning some penalty per neuron in the network. The penalty is proportional to the number of neurons in the network. This penalty is different for all the three networks.

The resultant fitness may hence be given by equation (2)

$$\mathrm{Fit}(I) = P_1(I) + P_2(I, B_1, B_2) + \alpha\, N(I) \tag{2}$$

Here *I* is the individual whose fitness is to be computed. $P_1(I)$ is the recognition by this individual alone. $P_2(I, B_1, B_2)$ is the performance by this network along with the best networks of the other evolutionary algorithms. α is the penalty constant. *N(I)* is the number of neurons in *I*.

6. Results

The above discussed model was coded and implemented in JAVA. The database used for the experimentation was of PIMA Indian Diabetes [34]. The data was taken from the UCI Machine Learning Repository. The aim is the detection of Diabetes in a patient given the various attributes. 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 general methodology involves the division of database into training and testing data sets. The training data set is used for training the system and the testing data set is used to measure the performance. The entire database of the disease has a total of 768 instances of data. This was divided into two parts. The first part consisted of the training database containing 534 (~70%) instances of data. The second part was the testing database consisting of 234 (~30%) instances of data. In the training phase the proposed algorithm was given the training instances. The next stage was testing. Here the evolved system was given unknown data from the testing data set. The output was collected and compared with the standard output. This determined the performance or the classifying ability of the system.

For each module, the evolutionary parameters were fixed to same value. The number of individuals was 100. The execution was carried over 1000 generations. At any generation crossover contributed 40% of the population and Mutation contributed 20% of the population. The contributions of Add, Grow and Elite operators were 24%, 15% and 1% respectively. The training algorithm was executed for 20 epochs. The maximum number of neurons was 30 for MLP, 28 for RBFN and 25 for SOM. The various parameters are given in table 1.

Table 1. Evolution and Network Parameters for all modules

S. No.	**Parameter**	**MLP**	**SOM**	**RBFN**
1.	No. of individuals	100	100	100
2.	Generations	1000	1000	1000
3.	Crossover	40%	40%	40%
4.	Mutation	20%	20%	20%
5.	Add	24%	24%	24%
6.	Grow	15%	15%	15%
7.	Elite	1%	1%	1%
8.	α	0.01	0.01	0.01
9.	Maximum Neurons	30	25	28

The algorithm gave an accuracy of 84.07% in the training data and 82.37% in the testing data using the mentioned constants values. The accuracies are summarized in table 2. Hence we may easily conclude that the system was able to affectively solve the problem of detection of diabetes in the patients.

Table 2. The performance of the algorithm

S. No.	**Data Set**	**Instances**	**Correctly Identified**	**Accuracy**
1.	Training	534	449	84.07%
2.	Testing	234	193	82.37%

We further applied a few commonly used methods to the same data set to compare the efficiency of the proposed algorithm against these methods. The first method applied was of Artificial neural Network (ANN) trained with Back Propagation Algorithm. The second method applied was ensembles. Here we had used 4 modules each being ANN with BPA with different architecture and training constants. The third method applied was of ANFIS. The fourth system was evolutionary ANN. Here we tried to evolve the ANN weights as well as the correct connectionist architecture. The chromosome stored the presence or absence of connections in between neurons (along with weights) to eliminate a fully connected architecture. The performance of the various models is analyzed in table 3. We can easily see that the proposed system gave the best performance than all the commonly known methods. We may hence generalize the results to other similar classificatory problems as well. The algorithm can effectively solve the problem of diagnosis in reasonably small times.

Table 3. The Accuracies with various methods

S. No.	**Algorithm**	**Training Accuracy**	**Testing Accuracy**
1.	**Proposed Algorithm**	**84.07%**	**82.37%**
2.	ANN with BPA	77.33%	77.73%
3.	Ensemble	78.72%	76.98%
4.	ANFIS	88.97%	66.52%
5.	Evolutionary ANN	77.38%	73.81%

7. Conclusions

In this paper we proposed a novel method for classification where we used an evolutionary model of Modular Neural Networks. Three different neural network models were used in a mixture of experts' architecture. These three networks performed independently in solving the problem. Each of these gave some result that was finally integrated using an integrator. The integrator adopted a voting methodology in trying to figure out the final solution to the problem. The various experts casted their votes in favor of the class they computed. Each of these neural networks had its own training algorithm that served as a local search strategy in the working of the entire algorithm.

The algorithm was implemented upon an evolutionary approach where evolution of the various networks was carried out. The evolutionary algorithm had various evolutionary operators that aided in the evolution of the concerned systems or ANNs.

This algorithm was applied over the problem of diagnosis of PIMA Indian diabetes. It was observed that the system so evolved was easily able to solve the problem and gave an accuracy that was much better than the congenitally used systems used in literature. Hence this can be an effective approach for disease diagnosis in particular and classification in general.

The algorithm discussed was applied over the problem of PIMA Indian Diabetes. This database is not a very extensive in terms of the number of attributes and the number of data instances. A better experimentation over the other databases may be done that may exploit the modularity factor of the algorithm to an even greater extent. Similar works in the fields of biometrics may be carried out where dimensionality is a major problem.

References

[1] Adams, R., Calcraft, L. & Davey, N. (2009). Using a genetic algorithm to investigate efficient connectivity in associative memories, Neurocomputing, 72(4-6) 732-742

[2] Amin, M. F. & Murase, K. (2009). Single-layered complex-valued neural network for real-valued classification problems, *Neurocomputing*, 72(4-6) 945-955

[3] Ang, J. H., Tan, K.C. & Al-Mamun A (2008). Training neural networks for classification using growth probability-based evolution, *Neurocomputing*, 71(16-18) 3493-3508

[4] Chaudhuri, B.B. & Bhattacharya U. (2000). Efficient training and improved performance of multilayer perceptron in pattern classification, *Neurocomputing* 34, 11-27

[5] Draghici, S. (1997) A neural network based artificial vision system for license plate recognition, International Journal of Network Security, *International Journal of Neural Systems*, 8 (1)

[6] Er, M.J., Zhou, Y. (2008) A novel framework for automatic generation of fuzzy neural networks, *Neurocomputing*, 21(4-6) 584-591

[7] Estudillo, F.J. Martinez, Martinez, C. H., Gutierrez, P.A. & Estudillo, A.C. Martinez (2008) Evolutionary product-unit neural networks classifiers, *Neurocomputing*, 72(1-3) 548-561

[8] Gernot, A., Fink, P. & Thomas (2006). Unsupervised Estimation of Writing Style Models for Improved Unconstrained Off-line Handwriting Recognition, *Proc. Tenth International Workshop on Frontiers in Handwriting Recognition*

[9] Graves, A., Fernandez, S., Liwicki, M., Bunke, H. & Schmidhuber, J. (2008). Unconstrained Online Handwriting Recognition with Recurrent Neural Networks, *Advances in Neural Information Processing Systems*

[10] Grigore, O. & Gavat, I. (1998) Neuro-fuzzy Models for Speech Pattern Recognition in Romanian Language, *Proc. European Symposium on Intelligent Techniques CONTI'98*. Timisoara, Romania, pp. 165–172

[11] Hewavitharana, S., Fernando, H. C. & Kodikara, N.D. (2002). Off-line Sinhala Handwriting Recognition using Hidden Markov Models

[12] Hu, Yi-Chung (2008). Nonadditive grey single-layer perceptron with Choquet integral for pattern classification problems using genetic algorithms, *Neurocomputing*, 72(1-3) 331-340

[13] Kak, S. C. (1998). On generalization by neural networks, *Information Sciences*, 111, 293-302

[14] Kuo, R.J., Hong, S.M., Lin, Y. & Huang, Y.C. (2008). Continuous genetic algorithm-based fuzzy neural network for learning fuzzy IF–THEN rules, *Neurocomputing*, 71(13-15) 2893-2907

[15] Lee, S. G. et al. (1999) A Neuro-Fuzzy Classifier for Land Cover Classification, *Proc. 1999 IEEE International Fuzzy Systems Conference Proceedings*, Seoul, Korea

[16] Lin, C. J. & Hong, S. J. (2007) The design of neuro-fuzzy networks using particle swarm optimization and recursive singular value decomposition, *Neurocomputing* 71(1-3) 297–310

[17] Lin, C. J., Chung, I.F. & Chen, C. H. (2007). An entropy-based quantum neuro-fuzzy inference system for classification applications, *Neurocomputing*, 70(13-15) 2502–2516

[18] Maravall, Dario, Lope, Javier de, Martin & Jose Antonio H (2009). Hybridizing evolutionary computation and reinforcement learning for the design of almost universal controllers for autonomous robots, *Neurocomputing*, 72 (4-6) 887-894

[19] Mu, C. S., Chien, H. C., Eugene, L. & Jonathan, L. (2006) A new approach to fuzzy classifier systems and its application in self-generating neuro-fuzzy systems, *Neurocomputing* 69 (4-6) 586–614

[20] Nani, L. & Lumini, A. (2009) Particle swarm optimization for prototype reduction, *Neurocomputing*, 72(4-6) 1092-1097

[21] O' Neill, M. & Ryan, C. (2001) Grammatical Evolution, *IEEE Transactions on Evolutionary Computing* 5(4)

[22] O' Neill, M., Brabazon, A. (2005). Recent Adventures in Grammatical Evolution, *In Proc. of CMS 2005 Computer Methods and Systems.* Vol. 1, pp. 245-253

[23] Pinero, P. et al. (2004) Sleep stage classification using fuzzy sets and machine learning techniques, *Neurocomputing,* 58–60, pp 1137 – 1143

[24] Rajagopal, P. (2003) The Basic Kak Neural Network with Complex Inputs, Master of Science in Electrical Engineering Thesis, *University of Madras*

[25] Rutkowski, L. (2004). *Flexible Neuro-Fuzzy Systems, Structures, Learning and Performance Evaluation*, Kluwer Academic Publishers

[26] Ryan, Conor, Collins, JJ & Neill, Michael O, Grammatical Evolution - Evolving Programs for an Arbitrary Language

[27] Sandhu, P. S., Salaria, D. S. & Singh, H. (2008) A Comparative Analysis of Fuzzy, Neuro-Fuzzy and Fuzzy-GA Based Approaches for Software Reusability Evaluation, *Proc of Worlds Academy of Science, Engineering and Technology* Vol. 29

[28] Shi, D., Shu, W. & Liu, H. (1998). Feature Selection for Handwritten Chinese Character Recognition Based on Genetic Algorithms, *In Proc. Intl' Conf. on Systems, man and Cybernetics*, Vol. 5, No 5, pp 4201-4206

[29] Som, T. & Saha, S. (2008) Handwritten character recognition by using Neural-network and Euclidean distance metric, *Social Science Research Network*, Available at SSRN: http://ssrn.com/abstract=1118755

[30] Soryani, M. & Rafat, N. (2006). Application of Genetic Algorithms to Feature Subset Selection in a Farsi OCR, *Proceedings of World Academy of Science, Engineering and Technology*, Vol. 18, ISSN 1307-6884

[31] Taur, J.S. & Tao, C.W (2000) A New Neuro-Fuzzy Classifier with Application to On-Line Face Detection and Recognition, *Journal of VLSI Signal Processing* 26, pp 397–409

[32] Tsoulos, I., Dimitris, G.& Glavas, E. (2008) Neural network construction and training using grammatical evolution, *Neurocomputing*, 72(1-3) 269-277

[33] Vieira, A & Barradas, N. (2003) A training algorithm for classification of high-dimensional data, *Neurocomputing*, 50, 461 – 472

[34] Sigillito, Vincent, 1990, UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html], The Johns Hopkins University. Available At: http://archive.ics.uci.edu/datasets/Pima+Indians+Diabetes