Classification using Redundant Mapping in Modular Neural Networks

Yogesh Kumar Meena
Department of Information Technology,
B.M.A.S. Engineering College,
Keetham, Agra, India
kumarmeenaiiitm@gmail.com

Rahul Kala
School of Cybernetics
University of Reading,
Whiteknights, Reading,
Berkshire, UK
rkala001@gmail.com

Karam Veer Arya
Department of IT, Indian
Institute of Information
Technology and Management
Gwalior, Gwalior, MP, India
kvarya@iiitm.ac.in

Citation: Y. K. Meena, K. V. Arya, R. Kala (2010) Classification using Redundant Mapping in Modular Neural Networks, *Proceedings of the 2010 IEEE World Congress on Nature and Biologically Inspired Computing,* Fukuoka, Japan, pp 554 – 559.

Final Version Available At: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5716375

Abstract

Classification is a major problem of study that involves formulation of decision boundaries based on the training data samples. The limitations of the single neural network approaches motivate the use of multiple neural networks for solving the problem in the form of ensembles and modular neural networks. While the ensembles solve the problem redundantly, the modular neural networks divide the computation into multiple modules. A modular neural network approach is used where Self Organizing Map (SOM) selects the module which performs the computation of the output, whenever any input is given. In the proposed architecture, the SOM selects multiple modules for problem solving, each of which is a neural network. Then the multiple selected neural networks are used redundantly for computing the output. Each of the outputs is integrated using an integrator. The proposed model is applied to the problem of Breast Cancer diagnosis, the database of which is made available from the UCI Machine Learning Repository. Experimental results show that the proposed model performs better than the conventional approaches.

Keywords

Classification, Machine Learning, Ensembles, Modular Neural Networks, Self Organizing Maps, Breast Cancer, Medical Diagnosis

1. Introduction

Classification is a major problem of study attributed to the fact that a large number of applications that we encounter in daily lives. Examples include the problem of biometric recognition, object recognition, medical diagnosis, etc. In all these systems, the task is to select to which classes the applied input may map to. Usually it may be assumed that the input maps to one of the possible output classes (closed database, unary classification problems). It is then that the system returns as its output the label of a single class, to which the input belongs. Neural Networks are extensively used for the purpose of classification. The various models include Self Organizing Maps, Radial Basis Function Networks, Recurrent Neural Networks, etc [1]. These neural networks are trained using a historical dataset, where each of these models extracts out some trends or patterns that may be used for the classification of any new data, or the testing data. The networks may be stated to be having a very high generalizing ability, if they compute the correct output for the new applied data, unseen to the system during complete training.

The training data may be easily plotted onto a space called as the feature space [2]. Here we find that the various classes map to distinctive regions of the feature space, where the inter-class separation is high and the intra-class separation is low. This happens when the features are ideal in nature. Else the various classes may intermingle with each other across the feature space. In such a case it is usually difficult to obtain correct output to the data items that lie close to the decision boundaries. Further the model being implemented needs to be flexible enough to model ideal shape of the decision boundaries, keeping the generality as high as possible.

The single neural network approach to solve the problem of classification has numerous limitations. One of the limitations is the inability to model the ideal decision boundary. The decision boundaries with various parameters and various models may only be near perfect. This is due to the limitations of the model being implemented. Further many times model may get sub-optimally trained, and may be used by the user as the final system. In such a case as well there are possibilities of getting non-optimal system outputs. We try to remove these limitations by using multiple neural networks in place of one in neural network ensemble architecture.

The neural network ensembles redundantly solve the same problem by a variety of neural networks [3]. Each of the networks is given the same training dataset to learn the rules, and generalize them to give correct outputs using their own modeling scenario. The individual neural networks take the input, process it, and produce the outputs. The outputs of all the neural networks are integrated using an integrator. This becomes the final output of the ensemble.

The other problem of the neural networks is a large complexity. Many times the problem may be too complex for any neural network to solve. The network may hence require a significantly large training time. The system may be completely unable to learn because of the presence of such large complexity. In such a case we try to divide the major problem into some sub-problems using the concept of modularity.

The modular neural networks carry forward the task of division of computation, where a major problem is broken down into sub-problems [4]. Each sub-problem is carried forward by some different neural network. All the networks together solve the main problem. One of the common implementations of the concept is by the application of the partition of the input space [5]. Here the entire input space is partitioned into a set of regions. Each region is given its own neural network, which carries forward the task of giving the correct output for any of the applied inputs. The complete system consists of a neural network selector, which analyzes the applied input, and selects which network must process the input. The applied input is passed to the specific neural network for processing. The output produced by this network is the output of the system. Hence there is no integration required further.

In this paper we implement both ensembles and modular neural network. The aim is to select a number of neural networks using the modular approach. All these selected neural networks are then used for redundantly solving the problem using the concept of ensembles. In such a manner we benefit from the advantage of the ensembles as well as the modular neural networks. The proposed algorithm may hence be better than both the approaches, and may hence be used for effective classification.

A variety of methods have been tried to solve problems from various domains using both ensembles and modular neural networks that include speech and speaker recognition [6], medical diagnosis [7], etc. In [8] the concept is used for solving the problem of biometric recognition using a combination of face and speech along with probabilistic sum integration. Another application may be seen into the works [9, 10]. Here three modules are used independently for each of the three biometric modalities i.e. face, speech and fingerprint. Each module is itself an ensemble. Evolutionary neural networks do classification at root level.

Another approach makes use of an algorithm inspired by neuro-fuzzy systems for better classification [11, 12]. Here the training inputs are first clustered as per their classes, and then individual cluster representatives are used for problem solving. A technique of evolution of neural networks using co-evolution is presented with the name of COVNET [13]. Here various modules that make up the modular neural network cooperate with each other so that the resultant neural network that evolves is optimal.

The paper is organized as follows. Section 2 presents the algorithm. The experimentation results over the field of breast cancer database are provided in section 3. Conclusion remarks are given in section 4.

2. Algorithm

The use of modular neural networks using mechanism of division of the input space is conventional in neural network literature. In this paper we primarily extend one of our earlier algorithms [5] that carried out this division of the input space using Fuzzy C Means clustering. The network selector of the algorithm simply mapped any input to the cluster with the nearest cluster center.

Multi-Layer Perceptron with Back Propagation Algorithm was used for the individual networks.

The basic problem with these systems is that the feature space (or the input space) is divided into discrete regions. Each region has its own expert that makes the decisions over the same region. Hence some experts may not get required amount of data, and may get undertrained due to data shortage. Of course a mechanism may be to decrease the number of modules, but it may give too many elements to some modules, or clusters, that happen to be overloaded with heavy amount of training data instances, with complex rules. The other problem associated is usually a poor generalizing capability. The data items that lie close to the decision boundary are especially poor performing that do not get easily mapped, as the network is trained with smaller number of data instances. The inclusion of modularity indeed benefits the system, as the complex rules may now be learnt, but this may be with an extra cost of loss of generality. Further the partition many times results in individual modules in possession with too less characteristics that make possible for any neural network to train.

For solving all these problems we propose the inclusion of redundant mapping in this particular architecture. The redundant mapping means that the network selector would select a number of networks in place of a single network. It may be verified that this solves all the problems mentioned above. Each network not only gets the training data instances of the inputs that fall closest to it, but also of inputs that are somewhat apart from it. Further the entire system would not suffer even if a single network is poorly trained due to the limitations of the region. Every data item is processed by a number of networks, which means the limitations of a single network would be averaged out by the other networks.

The concept of discrete division of the input space is shown in figure 1(a). Figure 1(b) shows the change in the partition of the space with redundant mapping. Each of the regions shown in this figure is a neural network. Figure 1(b) has overlapping regions representing redundant mapping. The magnitude of redundancy decides the magnitude of overlap. Each point here is associated with multiple networks which represents the level of redundancy.
The entire algorithm may be studied in three steps. We first develop a network selection mechanism. We would then study how the individual networks solve the problem. We later see the mechanism of integration. The complete algorithm is summarized in Figure 2.

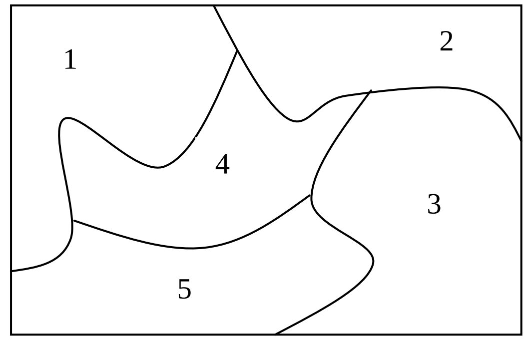

Fig 1(a): The discrete partition of input space

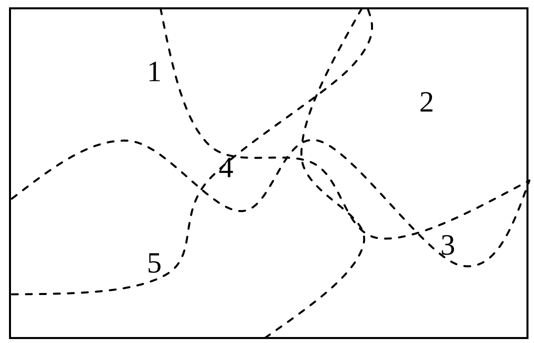

Fig 1(b): The partition of input space with redundant mapping

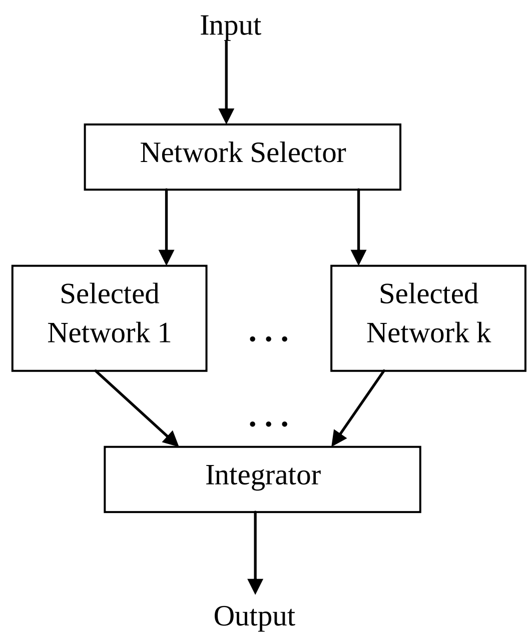

Fig 2: Algorithm description

A. Network Selector

The basic motivation behind the algorithm is to first partition the input space into regions. This is an implementation of the concept of modularity of the problem. This produces different spaces with an expert of each space. The network selector used in this algorithm is a Self Organizing Map (SOM). We take a total of *m x n* hidden layer neurons in the SOM. The network is trained using the training dataset. The SOMs are known natural clustering agents that cluster the data they are presented with. Hence after training the different neurons assume positions at the centers of the *m x n* clusters.

The next phase to plan is the testing phase where the SOM would be given unknown inputs. As per the conventional SOM working, every input is mapped to the neuron that lies closest to it, and it is assumed to have the same output. However in this approach we produce *k* number of outputs, where *k* is the known as the redundancy degree. Every input is mapped to *k* neurons that lie closest to it.

B. Modules

Each neuron of the SOM is basically a neural network. In this algorithm we use a Multi-Layer Perceptron with Back Propagation Algorithm. These process the applied input as per own problem modeling. There are hence a total of *m x n* networks in the system. The first task to be done is the training of the individual networks. For this we need to divide the entire training database in-between the various networks. Since the mapping is redundant, every input is given to multiple (*k* in total) neural networks. In order to do the division we make use of the network selector. The network selector produces *k* outputs each of which is a neuron, which is mapped to a neural network. In this mechanism the distribution of the training data items may take place.

The other task associated with the individual neurons is in testing. Whenever an input is applied, the network selector selects and invokes *k* neural networks. Each of these has been trained using its own training database. These networks use the conventional working to produce an output that is an indicative to the complete system.

C. Integration

The integrator does the task of integration of the results of the individual neural network. The network selector, for any input selects *k* networks, which in turn produces *k* outputs. The integrator does the task of integrating those *k* outputs and producing the final output of the complete system. We use a probabilistic sum integrator for the task of integration. This integrator assumes that each neural network or expert gives as its output a vector denoting the matching scores of the various classes. The matching scores are normalized to lie between 0 and 1, such that the sum of all matching scores for all the classes is unity. The integrator computes the net matching score vector, which is the summation of all the matching vectors of the individual experts. The class getting the maximum matching score is regarded as the final output of the system.

3. Results

The proposed algorithm was implemented in MATLAB. Experimentation of the algorithm is done on the Breast Cancer database available from UCI Machine Learning Repository [14]. The problem is detection of the type of breast cancer i.e. Malignant or Benign. Hence the problem is binary classification in nature, where the system outputs either of two classes as the identified cancer types. This is based on some attributes that are given as inputs in the database. This database consists of 30 real valued inputs. These correspond to the following features for each cell nucleus: radius (mean of distances from center to points on the perimeter), texture (standard deviation of gray-scale values), perimeter, area, smoothness (local variation in radius lengths), compactness ($perimeter^2$/area-1.0), concavity (severity of concave portions of the contour), concave points (number of concave portions of the contour), symmetry, fractal dimension (coastline approximation - 1). The entire data set consists of a total of 357 benign and 212 malignant cases, totaling to 569 instances in the database.

The first step towards the implementation of the algorithm was to divide the dataset into inputs and outputs, followed by the normalization of the inputs. The complete dataset was randomly divided into training and testing data sets. The training dataset was used for the training purposes, while the testing dataset was used for finding out the net accuracy of the resulting system.

The entire training dataset was initially given to the SOM, which acted as the network selector. The SOM used in the experiments had a network size of 4 *x* 4 neurons in the hidden layer. The network was trained for a total of 100 iterations. This gave the resultant positions of all the neurons, or the cluster centers.

The next task was the division of the database of between all the 16 (4 *x* 4) neural networks. This was done by computing the shortest distance of all the training inputs to all of the hidden layer neurons. The closest *k* neurons were selected. The value of *k* was fixed to 4 in the experiments.

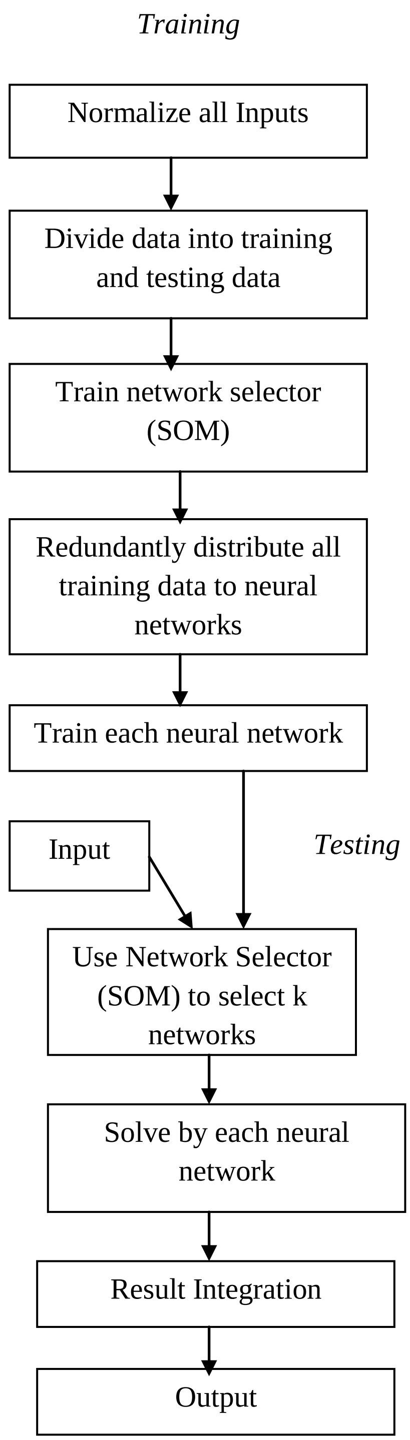

Fig 3: The working methodology of the algorithm

The next step was the actual training of all the neural networks. For simplicity we assumed that all the networks had the same architecture and same training parameters. All the neural networks had a single hidden layer which consisted of 18 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, momentum of 0.7 and a goal of 10^{-3}. Training was done till 3500 epochs. These parameters were taken from earlier experiments of the authors on a single neural network.

The next task was testing. For testing we applied the inputs and used SOM for the selection of the k neural networks. Each of these were invoked and corresponding outputs were computed. The integration was done using the probabilistic sum rule. The outputs were matched against the targets to compute the final accuracy of the system. The complete system working methodology is summarized in figure 3.

The resultant system had an accuracy of 98.32% for the training dataset, and an accuracy of 96.87% for the testing dataset. The summary of the parameters and the results is given in table 1.

Table 1: Parameters and performance of the system

S. No.	Parameter	Value
1.	m	4
2.	n	4
3.	k	4
4.	SOM epochs	100
5.	MLP hidden layer	1
6.	MLP neurons	18
7.	Learning Rate	0.05
8.	Momentum	0.7
9.	Goal	10^{-3}
10.	Max Epochs	3500
11.	Training Accuracy	98.32%
12.	Testing Accuracy	96.87%

The high accuracies of diagnosis motivate the use of the proposed algorithm in devising medical expert systems. We saw that using both modularity and ensemble technique, we were able to achieve a very high performance.

4. Conclusion

The neural networks are extensively used for solving numerous real life applications that we encounter in everyday life. Many of these applications usually face problems which present the limitations of the monolithic neural networks. Hence there is an abundant use of neural network ensembles and modular neural networks for problem solving. In this paper we presented an interesting approach where the neural networks were the modular neural network architecture was used for problem solving. Here we divided the problem as per its location in the input space. In place of a discrete division, we proposed the use of redundant mapping of the inputs to the neural networks. This ensured that every input is processed by a variety of neural network experts. The integration of these experts was carried out using probabilistic sum integration technique. This provided the final output of the system.

Experiments were performed over the breast cancer dataset available from the UCI Machine Learning Repository. Experiments confirmed that the proposed algorithm was easily able to carry out the task of classification and achieve a higher training accuracy.

The proposed method was presently tested against a single dataset. Extensive experimentation over a variety of other databases may be required to fully justify the advantages of the proposed algorithm. Further the algorithm assumes that there is a large amount of datasets available with multiple items in the training dataset, with simple or complex mapping. The algorithm may not perform well if there are too less training data items available, as the individual networks may not have enough training data items for them to train well. A major task presently is to decide the number of modules into which the division of problem would take place as well as the total degree of redundancy. Adaptive means to carry forward this task may prove to be very valuable to make the complete system perform optimally. All this may be carried forward into the future.

References

[1] A. Konar, *Artificial Intelligence and Soft Computing: Behavioral and Cognitive Modeling of the Human Brain*, CRC Press, Boca Raton, 2000

[2] A. Shukla, R. Tiwari, R. Kala, *Real Life Applications of Soft Computing*, Boca Raton, CRC Press, 2010

[3] L. K. Hansen, P. Salamon, "Neural Network Ensembles", *IEEE Transaction on Pattern Analysis and Machine Learning*, Vol 12, No. 10, pp 993-1001, 1990

[4] F. Matera, "Modular Neural Networks", *Substance Use and Misuse*, Vol. 33, No. 2, pp 307-315, 1998

[5] R. Kala, A. Shukla, R. Tiwari, "Fuzzy Neuro Systems for Machine Learning for Large Data Sets", *Proceedings of the IEEE International Advance Computing Conference, ieeexplore*, pp 541-545, 2009, Patiala, India

[6] A. Shukla, R. Tiwari, H. K. Meena, R. Kala, "Speaker Identification using Wavelet Analysis and Modular Neural Networks", *Journal of Acoustic Society of India*, Vol 36, No. 1, pp 14-19, 2009

[7] R. Kala, A. Shukla, R. Tiwari, "Comparative analysis of intelligent hybrid systems for detection of PIMA Indian diabetes" , *Proceedings of the IEEE 2009 World Congress on Nature & Biologically Inspired Computing, NABIC '09*, pp 947 - 952, 2009, Coimbatote, India

[8] R. Kala, H. Vazirani, A. Shukla, R. Tiwari, "Fusion of Speech and Face by Enhanced Modular Neural Network", *Proceedings of the International Conference on Information Systems, Technology and Management, ICISTM 2010*, CCIS 54, pp. 363-372, 2010, Bankok, Thailand

[9] P. Melin, O. Castillo, *Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing*, Springer-Verlag Heidelberg, 2005

[10] P. Melin, C. Gonzalez, D. Bravo, F. Gonzalez, G. Martinez, "Modular Neural Networks and Fuzzy Sugeno Integral for Face and Fingerprint Recognition", In: *Applied Soft Computing Technologies: The Challenge of Complexity*, Springer-Verlag Heidelberg, pp 603-618, 2006

[11] R. Kala, A. Shukla, R. Tiwari, "A Novel Approach to Classificatory problem using Neuro-Fuzzy Architecture", *International Journal of Systems, Control and Communications (IJSCC)*, 2010

[12] R. Kala, A. Shukla, R. Tiwari, A Novel Approach to Classificatory Problem using Grammatical Evolution based Hybrid Algorithm, *International Journal of Computer Applications*, Vol 1, No 28, pp 61-68, 2010

[13] N. G. Pedrajas, C. H. Martinez, J. M. Perez, "Multi-objective cooperative coevolution of artificial neural networks (multi-objective cooperative networks)", *Neural Networks*, Vol. 15, 2002, pp 1259–1278

[14] W. H. Wolberg, O. L., Mangasarian, D. W. Aha, UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html], University of Wisconsin Hospitals, 1992