Ensembles of Multilayer Perceptron and Modular Neural Networks for Fast and Accurate Learning

R.M. Valdovinos
Lab. Reconocimiento de Patrones
Instituto Tecnológico de Toluca
Av. Tecnológico s/n, 52140 Metepec (Mexico)
li_rmvr@hotmail.com

J.S. Sánchez
Dept. Llenguatges i Sistemes Informàtics
Universitat Jaume I
Av. Sos Baynat s/n, E-12071 Castelló de la Plana (Spain)
sanchez@uji.es

Abstract

In this paper, we analyze the performance of a classifier ensemble consisting of small-sized neural networks for accurate and fast learning. To this end, three topologies of neural networks are evaluated: the multilayer perceptron with two different configurations in the hidden layer, and the modular neural network. The experiments here carried out illustrate the effectiveness of the ensembles of neural networks in terms of average predictive accuracy and processing time, as compared to the single classifiers.

1. Introduction

Currently, the combination of multiple classifiers (also known as ensemble of classifiers, committee of learners, mixture of experts, etc.) constitutes a well-established research field in Pattern Recognition and Machine Learning. It is recognized that in general, the use of multiple classifiers instead of a single classifier leads to an increase in the overall predictive accuracy [7, 19]. Even in some cases, although the ensemble might not result better than the "best" single classifier, it could diminish or eliminate the risk of picking an inadequate single classifier [19].

Let $\mathcal{D} = \{D_1, \ldots, D_h\}$ be a set of h classifiers. Each classifier D_i $(i = 1, \ldots, h)$ gets as input a feature vector $x \in \Re^n$, and assigns it to one of the c problem classes. The output of an ensemble of classifiers is an h-dimensional vector $[D_1(\mathbf{x}), \ldots, D_h(\mathbf{x})]^{\mathrm{T}}$ containing the decisions of each of the h individual classifiers. For combining the individual decisions, the most popular (and simplest) method is the majority voting rule [18], although there exist other more complex schemes (e.g., average, minority, medium, product of votes) [3, 14, 17].

In general, an ensemble is built in two steps, that is, training multiple individual classifiers and then combining their predictions. According to the styles of training the base classifiers, current ensemble algorithms can be roughly categorized into two groups: algorithms where base classifiers must be trained sequentially (e.g., AdaBoost [9], Arc-x4 [8], MultiBoost [23], LogitBoost [10]),

and algorithms where base classifiers could be trained in parallel (e.g., Bagging [7], SEQUEL [2], Wagging [6]).

Briefly, an ensemble of neural networks employs a set of neural networks trained for the same task [11]. In the present paper, we address both the computational cost and the predictive accuracy when using ensembles of neural networks. To this end, the individual networks include a small-sized structure and then, the whole training set is equally distributed among the neural networks. By this way, each individual network is expert on a (small) part of the problem to solve. The two models here employed for the construction of the ensembles are the multilayer feedforward perceptron and the modular neural network.

From now on, the rest of the paper is organized as follows. Section 2 provides a description of the neural network ensembles here proposed. Section 3 briefly describes the two resampling methods employed in the present paper. Next, the experimental results are discussed in Section 4. Finally, Section 5 gives the main conclusions and points out possible directions for future research.

2. Ensembles of neural networks

The present section provides the main characteristics of the ensembles that will be further tested in the experiments. Correspondingly, we here describe the particular topology of the neural networks.

2.1 Multilayer perceptron

Probably, the multilayer perceptron is one of the most popular connectionist models. It organizes the representation of knowledge in the hidden layers and has a very high power of generalization. The typical topology consists of three sequential layers: input, hidden and output [15].

For the ensemble of multilayer perceptrons proposed in this study, all the networks have n input units and c output units corresponding to the number of input features (or attributes) and data classes, respectively. Also, each network has only one hidden layer with two different structures: MPa and MPb. In the former, the amount of neurons in the hidden layer is set to $(n + 1)$ [21] whereas in the latter, it is set to 2. The initial values of the connection weights are randomly picked out in the range $[-0.5, 0.5]$.

The networks will be here trained by using the backpropagation algorithm with a sigmoidal activation function for both the hidden and output layers. The backpropagation algorithm is simple and easy to compute. It often converges rapidly to a local minimum, but it may not find a global minimum and in some cases, it may not converge at all [20]. To overcome this problem, a momentum term can be added to the minimization function, and a variable learning rate can be applied. In our experiments, the learning rate and momentum are set to 0.9 and 0.7, respectively, whereas the number of training iterations is 5000.

2.2 Modular neural network

The second neural model used in this paper is the modular neural network (or mixture of experts, ME) [1, 12, 16]. A modular network solves a complex computational task by dividing it into a number of simpler subtasks and then combining their individual solutions. Thus, a modular neural network consists of several expert neural networks (modules), where each expert is optimized to perform a particular task of an overall complex operation. An integrating unit, called gating network, is used to select or combine the outputs of the modules (expert networks) in order to form the

final output of the modular network. In the more basic implementation of these neural networks, all the modules are of a same type [5, 13], but different schemes could be also used.

There exist several implementations of the modular neural network, although the most important difference among them refers to the nature of the gating network. In some cases, this corresponds to a single neuron evaluating the performance of the other expert modules [13]. Other realizations of the gating network are based on a neural network trained with a data set different from the one used for training the expert networks [5]. In this work, all the modules (the experts and the gating network) will be trained with a unique data set [16, 24] (see Figure 1).

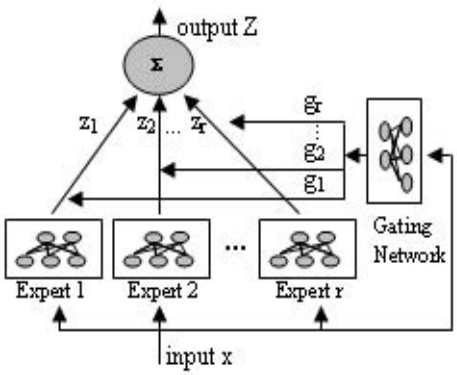

Figure 1. Graphical representation of the modular neural network architecture. Each module (including the gating network) is a feedforward network and receives the same input vector. The final output of the whole system is the sum of $z_j g_j$

All modules, including the gating network, have n input units, that is, the number of features. The number of output units in the expert networks is equal to the number of classes c, whereas that in the gating network is equal to the number of experts, say r. The learning process is based on the stochastic gradient algorithm, where the objective function is defined as:

$$-\ln\left(\sum_{j=1}^{r} g_j \exp\left(-\frac{1}{2}\left|\left|s-z_j\right|\right|^2\right)\right) \tag{1}$$

where s is the output desired for input x, $z_j = xw_j$ is the output vector of the j'th expert network, $g_j = \frac{\exp(u_j)}{\sum_i \exp(u_j)}$ is the normalized output of the gating network, u_i is the total weighted input received by output unit j of the gating network, and g_j can be viewed as the probability of selecting expert j for a particular case.

Each individual component in the ensemble corresponds to a modular neural network, each one with the same structure: five expert networks and one gating network. As in the case of the multilayer perceptron, the decisions of the base classifiers in the ensemble are finally combined by simple majority voting.

3. Resampling methods

In this work, two different methods for designing the training samples are used in the ensembles of neural networks: Bagging [7] and random selection without replacement [4]. These are here

implemented by means of the class-dependent strategy [22], which consists of picking instances in a way that the initial class distributions are preserved in each of the h subsamples generated. The rationale behind this strategy is to reduce the possibly high computational complexity (computing time and storage loads) of the base classifiers induced by each of the subsamples. In general, the smaller the subsample size, the lower the classifier complexity.

Briefly, in a class-dependent Bagging ensemble, each base classifier is trained on a set of N/h training examples randomly drawn, by replacement, from the original training set (of size N). Such a set is called a bootstrap replicate of the original training set. By this technique, many examples may appear multiple times, while others may be left out. This negative effect can be overcome by using the random selection without replacement method, in which each example will be selected only once.

4. Experimental results

In this section, we present the results corresponding to the experiments carried out over 10 real and artificial data sets taken from the UCI Machine Learning Database Repository (`http://www.ics.uci.edu/~mlearn`). Table 1 summarizes the main characteristics of each data set: number of classes, attributes, and patterns. For each database, we estimate the average predictive accuracy and processing time by 5-fold cross-validation: each data set is divided into five blocks, using four folds as the training set and the remaining block as the test set.

Table 1. A brief summary of the UCI databases used in this paper.

	No. Classes	No. Features	No. Patterns
Heart	2	13	272
Pima	2	8	770
Sonar	2	60	210
Iris	3	4	150
Wine	3	13	180
German	2	24	1002
Phoneme	2	5	5406
Waveform	3	21	5001
Segment	7	19	2212
Satimage	6	36	6437

In all experiments, the ensembles consist of nine individual classifiers built on nine different training sets. For each ensemble, the corresponding training sets have been designed by using the class-dependent Bagging and random selection without replacement techniques, as described in Sect. 3. Three configurations for the base classifiers, MPa, MPb, and ME, have been tested. The experiments have been carried out using a personal computer with Intel Centrino 1.3 GHz and 512 MB of RAM.

Figure 2 and Figure 3 illustrate the accuracy results for the ensembles with MPa and MPb structures, respectively. As a baseline, we have also included the results given by a single (neural network) classifier. Firstly, one can see that in general, as expected from theory, the accuracies of ensembles are higher than those obtained by the single classifiers. When comparing both ensemble

configurations (MPa and MPb), it seems that the MPb ensembles generally provide better results than the MPa, although differences are not statistically significant.

Within the comparison of MPa and MPb, the Satimage database constitutes a very particular case, in which the gains with MPb are much more important: the accuracy obtained by the MPb ensemble with Bagging is 28% higher than that given by MPa. On the other hand, in both configurations, the resampling method that produces the best results (in terms of predictive accuracy) corresponds to the random selection without replacement.

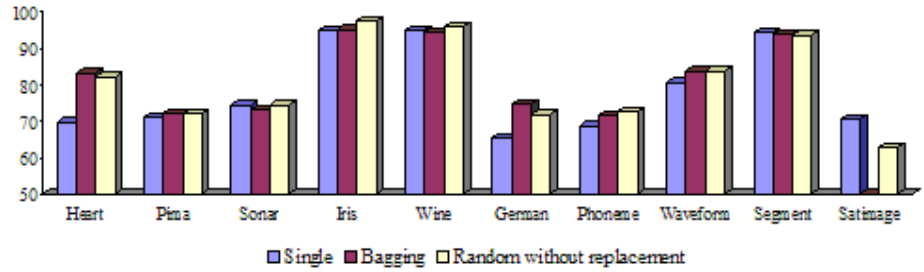

Figure 2. Overall accuracy with MPa ensembles (the number of neurons in the hidden layer is set to $n + 1$).

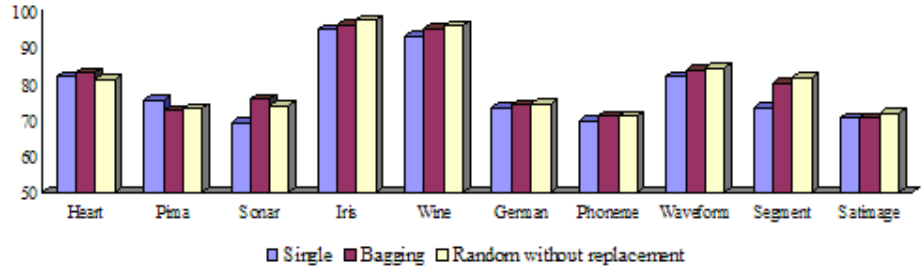

Figure 3. Overall accuracy with MPb ensembles (two neurons in the hidden layer).

The ensemble ME (mixture of experts) is the second scheme used in this work. Each individual component of the ensemble corresponds to a mixture of five experts and the integrating network, all of them with the same structure. Figure 4 shows the results obtained with the ME configuration. Although the present results are similar to those corresponding to the multilayer perceptron (MPa and MPb), it is worth remarking that the increase in performance of the ensembles with respect to the single network is now statistically significant. In fact, for Sonar and German databases, differences in accuracies are about 10%, and for Satimage data set, the average predictive accuracy of the ensemble (with Bagging) is 24% higher than that of the single classifier. Finally, like in the case of the multilayer perceptron, the best resampling method seems to be the random selection without replacement.

Apart from accuracy, the time required for training a given classification system is another important factor to take into account when analyzing the performance of the learning process. Thus, a certain trade-off between predictive accuracy and processing time should be pursued to decide which model will be used. Correspondingly, Table 2 reports processing times (relative to training

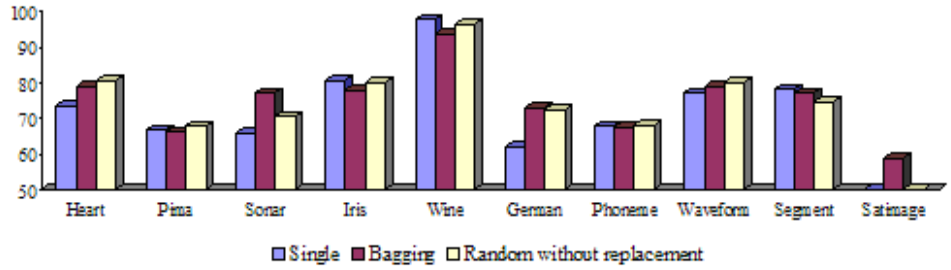

Figure 4. Average predictive accuracy with ME ensembles.

and classification phases) required by each learning method.

Table 2. Processing time (in minutes).

	Single			Bagging			Random w/o replac.		
	MPa	MPb	ME	MPa	MPb	ME	MPa	MPb	ME
Heart	2.6	2.5	5.5	0.2	0.2	3.7	0.2	0.2	2.5
Pima	8.5	3.6	9.8	5.7	4.1	9.6	6.8	3.9	9.8
Sonar	1.2	3.1	9.1	0.5	0.2	0.3	0.5	0.2	0.3
Iris	1.0	0.4	2.3	0.1	0.1	1.8	0.1	0.1	1.8
Wine	0.3	0.1	0.1	0.1	0.1	0.2	0.2	0.1	0.2
German	104.4	13.3	31.4	4.0	9.2	25.2	3.1	8.3	22.9
Phoneme	60.5	21.3	66.8	39.4	22.6	57.4	41.9	23.7	59.0
Waveform	437.9	65.1	174.5	167.3	44.8	133.0	131.2	41.1	127.2
Segment	210.9	35.0	1.4	81.8	24.4	2.1	63.1	23.0	2.1
Satimage	134.4	134.5	593.2	913.9	88.3	420.6	829.4	82.4	409.0

The significant differences in processing times between the three ensembles and the single networks result quite evident (see Table 2). For example, in the case of the Iris data, the time required by the single MPa is one minute, whereas the corresponding ensembles (Bagging and random selection without replacement) need only 10 seconds for training and classification.

Analogously, in the case of the ME configuration, the corresponding time goes down 0.5 minutes when using an ensemble. Besides this considerable reduction in time, it has to be remarked that the ensembles obtain even better overall prediction accuracy (see Figures 2–4), thus producing the "best" results in terms of both performance measures (accuracy and processing time).

When comparing the results given by the three ensembles, in most cases the MPb ensembles need much less processing time than the other two schemes. In fact, it should be pointed out that the MPa configurations consist of $(n + 1)$ neurons in the hidden layer, while the MPb schemes are formed by only two neurons. Obviously, this supposes a considerable difference in computational cost. Also, it is important to remind the classification results obtained with the three ensemble configurations. Taking into account all these results, it seems possible to conclude that in general, the MPb ensemble provides a well-balanced trade-off between time requirements and predictive accuracy, thus becoming somewhat superior to MPa and ME.

5. Concluding remarks

The design of ensembles with multilayer perceptron and modular neural networks has been here empirically analyzed. All ensembles employed in this paper consist of nine individual classifiers (neural networks). In the case of the multilayer perceptron, the number of neurons in the hidden layer has been determined according to two criteria: $(n + 1)$ in the first case, and 2 for the second configuration. On the other hand, for the modular neural network structure, we have employed five expert networks and a unique gating network.

The experimental results allow to validate the performance of these three ensemble models, in terms of processing time and predictive accuracy. In general, the alternative of using an ensemble of neural networks appears as an effective and efficient classification algorithm, excelling the results of the single neural networks. When comparing the three ensembles, it has been empirically demonstrated that the employment of a perceptron with only two neurons in the hidden layer results in the best performance: higher accuracy and lower computational cost.

The results presented in this paper should be viewed as a first step toward a more complete understanding of how to exploit the benefits of using an ensemble of neural networks. In this sense, other topologies and different parameters in the neural network configurations have to be analyzed in the future. Also, it should be further investigated the relationship between the individual classifiers of the ensemble and the resampling methods in order to determine the "best" combination.

Acknowledgements

This work has been partially supported by grants SEP-2003-C02-44225 from the Mexican CONACYT and TIC2003-08496 from the Spanish CICYT (Ministry of Science and Technology).

References

[1] R. Anand, K. Mehrotra, C.K. Mohan, S. Ranka. Efficient classification for multiclass problems using modular neural networks, *IEEE Transactions on Neural Networks*, 6(1):117–124, 1995.

[2] L. Asker, R. Maclin. Ensembles as a sequence of classifiers, in: *Proceedings of 15th Int. Joint Conference on Artificial Intelligence*, Nagoya, Japan, 1997, pp. 860–865.

[3] D. Bahler, L. Navarro. Methods for combining heterogeneous sets of classifiers, in: *Proceedings of 17th Natl. Conference on Artificial Intelligence, Workshop on New Research Problems for Machine Learning*, 2000.

[4] R. Barandela, R.M. Valdovinos, J.S. Sánchez. New applications of ensembles of classifiers, *Pattern Analysis and Applications*, 6(3):245–256, 2003.

[5] C. Bauckhage, C. Thurau. Towards a fair'n square aimbot — Using mixture of experts to learn context aware weapon handling, in: *Proccedings of GAME–ON*, Ghent, Belgium, 2004, pp. 20–24.

[6] E. Bauer, R. Kohavi. An empirical comparison of voting classification algorithms: bagging, boosting, and variants, *Machine Learning*, 36(1/2):105–139, 1999.

[7] L. Breiman. Bagging predictors, *Machine Learning*, 26(2):123–140, 1996.

[8] L. Breiman. Arcing classifiers, *The Annals of Statistics*, 26(3):801–823, 1998.

[9] Y. Freund, R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, *Jorunal of Computer and System Sciences*, 55(1):119–139, 1997.

[10] J. Friedman, T. Hastie, R. Tibshirani. Additive logistic regression: a statistical view of boosting, Technical Report, Stanford University, 1998.

[11] G. Giacinto, F. Roli. Design of effective neural network ensembles for image classification purposes, *Image and Vision Computing*, 19(9/10):699–707, 2001.

[12] B.L.M. Happel, J.M.J. Murre. Design and evolution of modular neural architectures, *Neural Networks*, 7(6/7):985–1004, 1994.

[13] P. Hartono, S. Hashimoto. Ensemble of linear perceptrons with confidence level output, in: *Proceedings of 4th Intl. Conf. on Hybrid Intelligent Systems*, Kitakyushu, Japan, 2004, pp. 186–191.

[14] T.K. Ho. Complexity of classification problems and comparative advantages of combined classifiers, in: *Proceedings of 1st Intl. Workshop on Multiple Classifier Systems*, Cagliari, Italy, 2000, pp. 97–106.

[15] K. Hornik, M. Stinchcombe, H. White. Multilayer feedforward networks are universal approximators, *Neural Networks*, 2(5):359–366, 1989.

[16] R. Jacobs, M. Jordan, G. Hinton. Adaptive mixture of local experts, *Neural Computation*, 3(1):79–87, 1991.

[17] L.I. Kuncheva, J.C. Bezdek, R.P.W. Duin. Decision templates for multiple classifier fusion, *Pattern Recognition*, 34(2):299–314, 2001.

[18] L.I. Kuncheva, R.K. Kountchev. Generating classifier outputs of fixed accuracy and diversity, *Pattern Recognition Letters*, 23(5):593–600, 2002.

[19] L.I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley & Sons, Hoboken, NJ, 2004.

[20] R. Nath, B. Rajagopalan, R. Ryker. Determining the saliency of input variables in neural networks classifiers, *Computers & Operations Ressearch*, 24(8):767–773, 1997.

[21] Y.H. Pao. *Adaptive Pattern Recognition and Neural Networks*. Addison-Wesley, Reading, MA, 1989.

[22] R.M. Valdovinos, J.S. Sánchez. Class-dependant resampling for medical applications, in: *Proceedings of 4th Intl. Conf. on Machine Learning and Applications*, Los Angeles, CA, 2005, pp. 351–356.

[23] G.I. Webb. MultiBoosting: a technique for combining boosting and wagging, *Machine Learning*, 40(2):159–196, 2000.

[24] R. Zaman, D.C. Wunsch III. TD methods applied to mixture of experts for learning 9x9 Goevaluation function, in: *Proceedings of IEEE/INNS Intl. Joint Conf. on Neural Networks*, Washington DC, 1999, pp. 3734–3739.