MONNA: A multiple ordinate neural network architecture

Olivier Lezoray, Hubert Cardot, Dominique Fournier, Marinette Revenu

MONNA : a Multiple Ordinate Neural Network Architecture

Olivier Lezoray
Departement d'informatique
Campus II, Université de Caen
14000 Caen, FRANCE
Olivier.Lezoray@info.unicaen.fr

Hubert Cardot
LUSAC
Site universitaire,BP 78
501300 Octeville, FRANCE
Hubert.Cardot@greyc.ismra.fr

Dominique Fournier
GREYC,Université de Caen,Campus II
14000 Caen, FRANCE
Dominique.Fournier@info.unicaen.fr

Marinette Revenu
GREYC-ISMRA, Bd du Maréchal Juin
14000 Caen, FRANCE
Marinette.Revenu@greyc.ismra.fr

Abstract

This article aims at showing an architecture of neural networks designed for the classification of data distributed among a high number of classes. A significant gain in the global classification rate can be obtained using our architecture. This latter is based on a set of several little neural networks, each one discriminating only two classes. The specialization of each neural network simplifies their structure and improves the classification. Moreover, the learning determines automatically the number of hidden neurons. The discussion is illustrated by tests on data bases from the UCI machine learning database repository. The experimental results show that this architecture can achieve a faster learning, simpler neural networks and an improved performance in classification.

1. Introduction

The majority of the tasks of classification require a phase of training generating a model which associates with an input (represented by a vector of attributes) a class. In this article, we are interested in supervised classification of data. There is a great number of algorithms of classification of data. Mainly one can quote the decision trees [2, 11], the methods based on the theory of Bayes [3], the statistical methods [3] (dynamic clouds, K nearest neighbors) and the neural networks [6]. we are interested with this last type of method of classification. We propose an architecture based on neural networks in order to simplify on the structure of the networks used by an automatic and dynamic determination of the number of hidden neurons. The training is also simplified by limiting the task of the classifior while giving him to learn only the data relating to two classes. Our study is within a general framework, without a priori on the problem being treated. We will show that when the number of classes of objects to be distinguished in a base is high, our architecture tends to give better results (correctly classified data rate) than a single large multi-layer neural network (MLP).

2. MONNA : a neural network architecture

We propose to use an architecture MONNA (Multiple Ordinate Neural Network Architecture) made of several small neural networks to solve a problem of classification. The neural networks used are multilayer networks with retropropagation of the gradient error (MLP : Multi-Layer Perceptrons). Typically, a neural network consists of an input layer (parameters characterizing an object), of one or two hidden layers and of an output layer providing the class of the object. We choose to use an architecture of multiple neural networks in order to obtain better performance of classification compared to only one large MLP (in particular when the number of classes becomes high).

The architecture MONNA construction is done in three steps :

- The construction of the neural network architecture according to the number of classes of objects to be separated,

- The training of each neural network according to a method dynamically finding the optimal number of neurons of the hidden layer,
- The construction of a hierarchy of neural networks bringing to the classification of the objects.

2.1. Principle of construction

The construction of the architecture is supervised. It builds a set of neural networks (a forest of neural networks). The classification of data by only one large network can be difficult when the number of classes of objects to be separated is high : this neural network presents difficulties of generalization. What we propose consists in using only small neural networks : by small, we understand simple structure. Indeed, if the capacity of generalization of a network is weakened by the number of classes to discriminate, a simplification of the problem can be a reduction of the number of classes having to be recognized by this network. This obviously involves an increase in the number of networks having to discriminate several classes. Our architecture arises in the following way. For a problem of classification having to separate n classes, MONNA is composed of a set of unconnected networks, each one being intended to separate the elements from two distinct classes. If the problem has n classes, that leads us to have $(n*(n-1))/2$ neural networks being used for classification. The difficulty in separating n classes is simplified by the specialization of each network, because a network is interested only in separation of two classes. Consequently, at the time of its training, each network learns to recognize only examples of these two classes. The structure of the neural networks used is the following one : a layer of inputs containing as many neurons as the number of attributes associated with the object to be classified, a hidden layer containing a variable number of neurons and one output neuron. The value of the output neuron is in the interval $]-1,1[$. According to the sign of the result associated with this single neuron, an object is classified in one of the two classes that the network separates. This has several advantages. The simplicity of the task associated to each neural network simplifies the convergence of the training as well as the search for a simple structure as we will see further.

2.2. Training

As the neural networks used by our architecture are very simple (only one hidden layer, only one neuron of output), the generalization of their structure can be made in a dynamic way very easily. Some research studies were already undertaken in order to build neural networks dynamically. Those include the dynamic creation of the nodes [1], the "cascades correlation" algorithm [4], the "tilling" algorithm [8], the algorithm "self-organizing" [13] and the "upstart" algorithm [5]. These algorithms are used to eliminate the need to determine in advance (before the training of a network) the number of neurons of the hidden layer. This is very useful because a simpler network having less hidden neurons reduces the complexity of calculations. However, the fact of fixing this number of neurons of the hidden layer can be penalizing if not suitable. We thus propose to modify the structure of a neural network during the training while thus allowing to build dynamically a neural network. At the time of its training a neural network starts with a very simple structure : only one neuron on the hidden layer. Neurons are then added one by one in the hidden layer in order to improve the rate of classification of the network on a validation set. Each time a new neuron is added, the network proceeds to a new training by the method of retropropagation of the gradient. The initial weights are obtained in a random way at each addition of a new neuron to the hidden layer (figure 1). The initial training data (S_T) are splitted in two subsets : a learning set (S_L) and a validation set (S_V). This latter consists in 20% of S_T and the learning set in 80% of S_T. The learning of a neural network is performed on S_L and the validation set S_V is used to evaluate the classification rate of the network. The validation set is not learned by the neural networks.

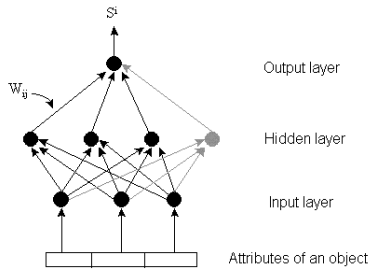

Figure 1. An example of neural network with an adaptive structure : when a neuron is added to the hidden layer (in gray), the links are updated and the weights all initialized in a random way.

The training ends either when the network reaches a satisfying rate of classification on the validation set, or when the maximum number of hidden neurons is reached. It is noted here that during the training one always stores the structure of the network which gave

the best rate of classification, because this latter can increase and decrease with the addition of neurons (figure 2). The training algorithm can be described in the following way for only one neural network. Let us denote h as the number of hidden neurons, h_{max} the maximum number of hidden neurons, Q_{max} the satisfying rate of classification, $R(h)$ the network having h hidden neurons, $Q(h)$ the rate of classification of network $R(h)$ on the validation data, R_{best} the network which the structure gives the best rate of classification and Q_{best} the rate of classification of network R_{best}.

1. $h = 1$, $R_{best} = R(1)$ and $Q_{max} = 100\%$
2. Randomly initialize the weights of the network $R(h)$.
3. Find the weights which minimize the function of error by retropropagation of the gradient.
4. If $Q(h) >= Q_{max}$ then end the training and go to step 7.
5. If $Q(h) > Q_{best}$ then $Q_{best} = Q(h)$ and $R_{best} = R(h)$.
6. Add a neuron to the hidden layer : $h = h + 1$. If $h <= h_{max}$ go to step 2.
7. The final structure of the network is that of R_{best}.

The algorithm allows to obtain in a reliable way, the neural network which best learns with less neurons in the hidden layer. Figure 2 gives an illustration of the influence of the number of hidden neurons on the training of a network ($Q_{max} = 95\%$ and $h_{max} = 50$) on the "Glass" database. Only the networks which did not quickly reach the Q_{max} quality are presented. One generally reaches an optimal configuration with a small number of hidden neurons.

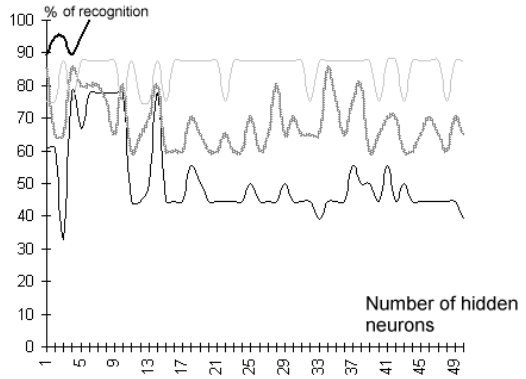

Figure 2. Variation of the rate of classification according to the number of neurons of the hidden layer on the validation database of the "Glass" database (see results).

We thus have an algorithm allowing to dynamically build a neural network at the time of the training. Our architecture MONNA uses this algorithm to carry out the training of each neural network constituting it. However, when one of them learns how to differentiate two classes, only the objects belonging to these two classes are presented to the neural network. This implies, on the one hand to simplify the training once again and on the other hand to facilitate discrimination between these two classes since the network learned how to recognize only those.

The error function that we used is quadratic error $E = (S-Y)^2/2$ in a synchronous version (the weights are modified with each presentation of an object of the learning base). Y belongs to $\{-1, 1\}$ and indicates the desired value of the output neuron (the class of the object to be learned), S indicates the value of the output neuron $S = f(W_i O_i)$ where W_i indicates the weights between the output neuron and neurons of the hidden layer (i). O_i is the value of the neuron i of the hidden layer. The transfer transfer function is $f(y) = 2/(1 + e^{-y\alpha}) - 1$.

It will be noted that thereafter if a neural network separates two classes C_1 and C_2 an object is regarded as class C_1 if $S_i < 0$ and C_2 if $S_i >= 0$

2.3. Hierarchy of neural networks

We now have a set of neural networks separating each one two classes, as well as a training algorithm allowing to dynamically find the optimal number of neurons of the hidden layer. We must now specify how to use the results coming from each network in order to classify an object. With this intention we build a hierarchy of neural networks. The fact of using several neural networks separating each one only two classes allows to have a hierarchical vision not only of the data, but also of the decision of classification which will be refined gradually at the time of successive interrogations to the networks. The hierarchy of neural networks defined by our architecture is articulated around three points :

- One has $(n * (n-1))/2$ networks discriminating each one two classes,
- The networks are ordered according to their quality of classification,
- The decision of classification is made by elimination.

The first point is the base of architecture MONNA, each network is used only to separate two classes. We specify here that an object belonging to none of the

classes discriminated by a network will be classified without significance in one or the other class.

The second point rests on the quality of classification of the neural networks. By quality of classification, we understand the rate of classification obtained by the network at the end of the training. This quality of classification is evaluated on a validation set and measures the capacity of a network to correctly classify an object between two classes (and only between these two classes since the network learned how to recognize only those). One thus can, at the end of the training, arrange the networks according to their quality of classification, the first network being that which best learned i.e. that for which the decision of classification is most reliable.

The third point defines the decision of classification. We have a hierarchy of neural networks classified according to their quality of classification. However, each network discriminating only two classes, the problem of the choice of the final class of an object to be classified arises. We set up with this intention a selection by elimination. Two criteria were defined, the Sequential-Elimination criterion and the QualityOutput-Elimination criterion. Some notations first of all. Let us denote $E = \{C_1, C_2 ..., C_N\}$ as the set of the classes of the objects in the training base, C_l indicates the class of an object and n the number of classes. R_k indicates the k^{th} neural network in the hierarchy, R_k^p indicates the class p discriminated by the network, $R_k^p \in E$ and $p \in \{1, 2\}$ since a network discriminates only two classes. Finally one defines $S(R_k)$ as the value of the neuron of output of network R_k , $T = \{R_1, ..., R_{(n*(n-1))/2}\}$ as the set of neural networks and $Q(R_k)$ as the rate of classification of the k^{th} neural network.

Sequential-Elimination Criterion :

$k = 0$
While $Card(E) \neq 1$
  If $S(R_k) < 0$ **then** $E = E - R_k^1$
  else $E = E - R_k^2$
  $k = k + 1$
EndWhile

This criterion uses one by one the networks according to the hierarchy and eliminates progressively the classes according to the output of the neural networks.

The second criterion used is the product of the output of a network by the quality of this one. Indeed, the decision related to the classification made by a network depends on two things : its potential of classification and its decision. Thanks to this criterion, a network for which the output is close to 0, which characterizes a dubious decision, but which belongs to the first in the hierarchy, will not be inevitably used in first.

QualityOutput-Elimination Criterion :

While $Card(E) \neq 1$
  $max = -1$
  For $l = 1$ **to** $(n * (n-1))/2$ **do**
    If $R_l \in T$ **then**
      If $S(R_l) < 0$ **then** $Tmp = R_l^1$
      Else $Tmp = R_l^2$
      If $|S(R_l) * Q(R_l)| > max$ and $Tmp \in E$
        $max = |S(R_l) * Q(R_l)|$
        $class = Tmp$
        $used = l$
  EndFor
  $T = T - R_{used}$
  $E = E - class$
EndWhile

At this stage, we did not specify which elimination criterion to use. It appears obvious that the criterion to use is that which gives the best classification rate. However, after having carried out various tests on several bases, it did not prove to have of great difference (on the level of the rate of classification) between the two criteria, the QualityOutput-Elimination criterion being slightly higher.

The only difference between the two criteria lies in the choice of the network used to eliminate a class. The Sequential-Elimination criterion uses one by one networks according to the hierarchy whereas the QualityOutput-Elimination criterion chooses among all the networks that which maximizes $|S(R_l) * Q(R_l)|$. This implies to use all of them because one must know the output of each one of them and must propagate the data for each network. However, it is noted that the sequence of the networks leading to the decision of classification is made in a more hierarchical way for this last criterion. We will see thereafter on various bases that this is always checked and that the criterion to be retained is rather the QualityOutput-Elimination criterion to have a more hierarchical point of view of the decision of classification.

3. Experimental results

The data bases for which results will be given here are real data bases coming from "The Machine Learning Data Repository of the University of California At Irvine (UCI)" [9] and also from our own works (Serous base) [7]. These data bases are used in various articles on classification. This will enable us to compare the performances of our architecture with other neural networks approaches which use traditional or improved methods.

3.1. Description of the data bases

Table 1 summarizes the characteristics of each data base. Their source is varied and corresponds to very different problems (medical, segmentation of images, character recognition, wines, cars, etc.). Each data base is characterized by the number of classes to discriminate, the number of attributes describing an object and the number of instances in the base (table 1).

We used 10 data bases coming from the base of the UCI. Our data base relating to classification of the cells was retained because it uses a great number of classes (18) and illustrates well the utility of our neural network architecture MONNA. This base comes from our research studies [7] on the automation of the sorting of cells from serous cytology (cavities of the human body such as the peritoneum). It was obtained by a tagging of objects (the cells) carried out by three different experts. This enabled us to have a reliable base in the sense that it is representative of the data to classify (even if the problem to be solved is difficult).

Database	Number of classes	Number of attributes	Number of instances
Liver Bupa	2	6	345
Pima	2	8	768
Ionosphere	2	34	350
Cancer	2	9	683
Wine	3	13	178
Vehicle	4	18	846
PageBlocks	5	10	5473
Glass	7	9	214
Segmentation	7	19	210
OptDigits	9	64	3823
Serous	18	46	4837

Table 1. Data bases used for the tests.

3.2. Results analysis

For each base, 20% of the data are used as a test base and 80% as training base (S_T) further splitted in a learning base (S_L) and in a validation base (S_V). The rate of classification used for comparison is measured on the test base (S_T). We first of all compared in term of rate of classification our architecture MONNA with a traditional MLP neural network. It will be noted that we used our algorithm of dynamic determination of the number of neurons of the layer hidden for the training of the MLP. For the bases where the number of classes is equal to two, the performances are identical, (it is normal since only one neural network is used). For the bases presenting a number of classes to be distinguished higher than two, our architecture proved to be better in all the cases, even simple (few classes) or more difficult (many classes). MONNA is, on the basis of table 1, between 3% and 15% better than a traditional MLP (table 2).

Base	MLP	MONNA
Liver Bupa	71	71
Pima	76.6	76.6
Ionosphere	90.1	90.1
Cancer	97.8	97.8
Wine	97.3	100
Vehicle	71.9	78.4
PageBlocks	85.4	90.1
Glass	66.7	82.2
Segmentation	82.4	87.8
OptDigits	78.1	81.6
Serous	55.9	65.8

Table 2. Comparison between the rate of classification of a MLP and MONNA on the data base of Table 1.

Architecture MONNA thus proves to be useful for the bases presenting a number of classes higher than 2. But one of the strength of our architecture is also to use networks having simple structures, because if the discrimination between two classes is simple, the best structure of the network is quickly found and the neural network generalizes well.

If one compares the number of hidden neurons of a MLP and of the architecture MONNA, overall the networks used by MONNA are simpler. Table 3 summarizes these results : for the MLP the number of hidden neurons is presented and for MONNA the minimal and maximum number, the average and the standard deviation of the number of hidden neurons and this for all the networks used. The structure of the networks used by MONNA is simpler, but only when each network can generalize sufficiently easily. If it is not the case, that can involve the creation of large networks : for the data base " Serous ", a network has 46 neurons on its hidden layer, 11 more than one MLP.

However, even if this network used by MONNA is complex, its training is faster because it takes place only on two classes and not on the totality of the classes. On the data base " vehicle ", one notices (table 3) that a neural network has 28 neurons on its hidden layer, which can appear high. This net-

Base	MLP	MONNA			
	Nb.	Min	Max	Avg.	St. Dev.
Liver Bupa	3	3	3	3	0
Pima	23	23	23	23	0
Ionosphere	7	7	7	7	0
Cancer	1	1	1	1	0
Wine	2	1	1	1	0
Vehicle	47	1	28	6.6	9.9
PageBlocks	31	1	17	2.8	4.7
Glass	25	1	11	1.8	2.3
Segmentation	18	1	1	1	0
OptDigits	47	1	1	1	0
Serous	35	1	46	4.5	10.3

Table 3. Comparison between the number of neurons of the hidden layer between a MLP and MONNA.

work is dedicated to the separation of classes 0 and 1 (table 4), but these classes are difficult to separate so that involves a more complex network. Indeed, if the classes are difficult to separate, that induces a weak rate of classification (lower than 60%) and thus a longer training since Q_{max} quality is never reached and the algorithm determining the optimal structure of the network will continue until the maximum number of neurons is reached before stopping.

In general, certain networks require a longer training because they have covered the whole possible set of the networks ($h = 1$ to 50) so that in certain cases, small networks are stored and in other cases larger. This training uses, for a neural network, only the objects relating to the classes which it separates, which accelerates convergence being given the simplicity of the task allotted to the network.

Number of hidden neurons	Quality	Network Classes i<->j
1	96.4	2<->3
1	96.4	1<->3
5	96.5	0<->2
4	94.3	1<->2
1	93.9	0<->3
28	58.6	0<->1

Table 4. Quality and a number of neurons of the layer hidden of the networks of architecture MONNA on the "vehicle" base.

If one is interested in the influence of the criterion used to choose by elimination the final class, one notes that the criterion Sequential-Elimination is as good as the QualityOutput-Elimination criterion in term of rate of classification (even if the last criterion is slightly better than the first : table 5). However, from the point of view of the grading of the decision of classification, the QualityOutput-Elimination criterion makes easier interpretation because it chooses more judiciously the networks bringing to the elimination of a class. Table 6 presents for each base, the number of networks used to take a decision of classification. It is noted that for each of the two criteria, the number networks used grows as the number of classes increases. But the QualityOutput-Elimination criterion reduces this increase to a significant degree (Table 6).

Base	Rate for Sequential	Rate for Quality Output
Wine	100	100
Vehicle	78.4	78.4
PageBlocks	90.1	90.1
Glass	82.2	82.2
Segmentation	87.8	87.8
OptDigits	81.6	83.2
Serous	65.8	66.3

Table 5. Comparison of the rates of classification for the two criteria of selection per elimination.

To finish these comparisons, we will compare the results obtained by our architecture with those of other works and more precisely those of Richeldi [12] and Yang [15, 14]. These works are based on neural networks. ADHOC [12] is a traditional neural network using a selection of characteristics. DistAI [15] uses genetic algorithms for the training, and GA-DistAI [14] is an improved version of DistAI [15] using a selection of characteristics, this selection also based upon genetic algorithms. On the bases common to our study, our architecture is better on all the bases presenting more than 2 classes (table 7). This fact is accentuated when the number of classes increases. For a number of classes equal to two our architecture does not differ from the structure of a traditional MLP, which explains lower results compared to using MLP with improvement techniques (genetic algorithms and selection of characteristics).

4. Conclusion

We showed the interest of the neural network architecture MONNA for the problems of classification pre-

Base	Number of classes	Sequential	Quality Output
Wine	3	2	2
Vehicle	4	5	3
PageBlocks	5	8	4
Glass	7	19	6
Segmentation	7	13	6
OptDigits	9	31	9
Serous	18	130	17
Average	7	30	7

Table 6. The average number of neural networks associated with the classification decision according to the criterion used.

Base	Monna	Monna SFFS	Adhoc	Distai	Gadistai
Pima	76.6	77.9	73.2	76.3	79.5
Ionos	90.1	95.8	-	94.3	98.6
Cancer	97.8	98.5	-	97.8	99.3
Wine	100	100	-	97.1	99.4
Vehicle	78.4	85.5	69.6	65.4	68.8
Glass	82.2	84.4	70.5	70.5	80.8
Serous	65.8	77.1	-	-	-

Table 7. Comparison between various neural approaches.

senting a great number of classes. This architecture is based on three steps : its construction, establishment of a hierarchy among the neural networks according to their quality of classification and the use of a selection by elimination to obtain the decision of classification. We also proposed a method of dynamic determination of the number of hidden neurons of each network during the learning. The superiority of our architecture MONNA was shown compared to different neuronal approaches, in the case of more than two classes. We also saw that in addition to its strength of classification, the neural networks used in MONNA have a very simple structure allowing a fast and powerful training to separate two classes.

It can be interesting to select the relevant attributes for the separation of two classes. It is noted that some networks have only one neuron on the hidden layer, which suppose a very simple task of classification. One could then still simplify the structure of a network by selecting its input attributes. This is all the more interesting since one can select the relevant attributes for each one of the networks used in MONNA. We have tried different algorithms for the selection of relevant attributes and the wrapper SFFS [10] algorithm was retained. This enables to increase once again the classification rate of our architecture MONNA with SFFS (Table 7).

References

[1] T. Ash. Dynamic node creation in backpropagation networks. *Connection Science*, 1(4):365–375, 1989.

[2] L. Breiman, J. Friedman, and R. Olshen. *Classification and regression trees.* Wadsworth and Brooks, California, 1984.

[3] E. Diday. *Optimisation en classification automatique.* Tomes 1 et 2, INRIA Ed., 1979.

[4] S. Fahlman and C. Lebiere. *The cascade-correlation learning architecture,Advances in Neural Information Processing Systems I.* D. Touretzky Editions, Morgan Kaufmann, San Mateo, CA, 1989.

[5] M. Frean. The upstart algorithm : a method for constructing and training feedforward neural networks. *Neural Computation*, 2(2):198–209, 1990.

[6] J. Hérault and C. Jutten. *Réseaux neuronaux et traitement du signal.* Traité des nouvelles technologies (série traitement du signal), Ed. Hermès, 1994.

[7] O. Lezoray, A. Elmoataz, H. Cardot, and M. Revenu. Arctic : An automatic system for cellular sorting by image analysis. In *Proceedings of Vision Interface 99*, 1999.

[8] M. Mezard and J. Nadal. Learning in feedforward layered networks : the tilling algorithm. *Journal of Physics A*, 2(12):2191–2203, 1989.

[9] P. Murphy, D. Aha, and G. Robinson. Uci repository of machine learning databases : Machine-readable data repository. http://www.ics.uci.edu/~mlearn/MLRepository.html, 1992.

[10] P. Pudil, F. Ferri, J. Novovicovà, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125, 1994.

[11] J. Quinlan. *C4.5 : programs for machine learning.* Morgann Kauffman, San Mateo, 1993.

[12] M. Richeldi. Performing effective feature selection by investigating the deep structure of the data. In *Proceedings of the second international conference on Knowledge Discovery and Data Mining*, pages 379–383, 1996.

[13] M. Tenorio and W. Lee. Selforganizing network for optimum supervised learning. *IEEE Transactions on Neural Networks*, 1(1):100–110, 1990.

[14] J. Yang and V. Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.

[15] J. Yang, R. Parekh, and V. Honavar. Distai : An inter-pattern distance based constructive learning algorithm. In *Proceedings of the international joint conference on neural networks*, 1998.