Feature Selection Forcing Overtraining May Help to Improve Performance

Enrique Romero
Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
eromero@lsi.upc.es

Josep M. Sopena
Lab. Neurocomputació
Universitat de Barcelona
jsopena@psi.ub.es

Gorka Navarrete
Lab. Neurocomputació
Universitat de Barcelona
gnavarga7@psi.ub.edu

René Alquézar
Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
alquezar@lsi.upc.es

***Abstract*— One of the main drawbacks of Machine Learning systems is the negative effect caused by overtraining. If the points in the dataset are perfectly fitted, the generalization performance is usually bad. We propose to take profit of overtraining, together with Feature Selection, to improve the performance of a learning system. The main idea lies in the hypothesis that when the dataset is as fitted as possible, the system is forced to use all the available variables as much as possible. Noisy and useless variables can be detected if generalization improves when the system is not allowed to use them. Forcing overtraining, noisy and useless variables should be more outstanding. In order to test this hypothesis, we performed several Feature Selection experiments using Feed-forward Neural Networks. The particular Feature Selection procedure used was *Sequential Backward Selection*. Experimental results with several real-world problems suggest that our hypothesis seems to be well-founded. Ironically, forcing overtraining may help to achieve good performance.**

I. Introduction

Suppose that someone wants to apply a Machine Learning (ML) technique to a certain problem. There exist many situations where one does not have *a priori* neither a model that could describe the phenomenon nor the knowledge of which variables are adequate to describe it. This is very common in Medicine or Psychology, for example. The expert may have several intuitions about the variables related to the problem, but by no means has neither the security that those are all the features needed to explain the phenomenon nor the confidence that all the features are useful. When important variables are missing, the problem cannot be solved. If some variables are useless, solutions that use them will probably have important performance problems. In addition, the number of available examples is usually small and they may be noisy or incomplete. In this situation, a Feature Selection (FS) procedure may be a very useful tool to select a good subset of variables. In addition of reducing the input dimension, FS may lead to a marked improvement in the performance of a ML system [5]. A justification for this assertion comes from the Bias/Variance decomposition [3], which suggests that the optimal performance is obtained when a tradeoff between the quality of the approximation to the training set and the variance of the solution is achieved. When too many variables are present the system can (surely) approximate very well the training set, but it is (probably) too complex, increasing the variance term. As far as the variables are eliminated, the complexity of the system is reduced (together with its capacity of approximation).

We will focus our work in classification tasks. In this paper we propose the use of Feed-forward Neural Networks (FNNs) to perform FS within the *wrapper* approach [6]. In particular, the *Sequential Backward Selection* (SBS) procedure was applied in our experiments (see Section II for a brief description of SBS and the wrapper approach). Our main motivation to use the SBS method was the expectation that it could be possible to identify noisy and useless variables as the features deleted in the first steps of the SBS procedure, while maintaining all the possible interactions among the initial set of variables. The *Sequential Forward Selection* (SFS) procedure, for example, does not satisfy this property. Ideally, there would exist an optimal point where the addition or elimination of any variable would lead to a worse performance.

In order to encourage the SBS procedure to eliminate noisy and useless variables in the first steps, we propose to fit the data as much as possible (that is, forcing overtraining). In theory, generalization improves when the system does not use noisy and useless variables. Our hypothesis is that this effect will be more evident if we try to fit the training set as much as possible (that is, when overtraining is highly present), since in this situation the variables in the system are forced to be as used as possible. Therefore, forcing overtraining, together with FS, may help to achieve good performance. This idea may be valid for both linear and non-linear classifiers.

The main motivation to use FNNs was their well-known universal approximation capability [7]. Using FNNs it is possible to fit enough the data to test our hypothesis.

After the selected variables are discarded, a different approach to the problem can be performed. In spite of the fact that the negative effect of overtraining may still be present, we expect that it will probably be lower. A standard technique that tries to control the overtraining (early stopping, for example) is expected to obtain better results with this reduced number of variables. We tested our proposal with several real-world problems. Experimental results suggest that our hypothesis seems to be well-founded.

The rest of the paper is organized as follows. In Section II, the FS problem is briefly described. The main ideas are explained in Section III. An algorithmic description of the proposed scheme is given in Section IV. The experimental

work is presented in Section V. Finally, some conclusions and future work are drawn in Section VI.

II. Feature Selection

The problem of FS can be defined as follows [8]: given a set of N_f candidate features, select a subset that performs the best under some evaluation criterion. From a computational point of view, the previous definition of FS leads to solve a search problem in a space of 2^{N_f} elements. In order to obtain a solution, we need to specify two components: the feature subset evaluation criterion and the procedure for searching through candidate subsets of features. Many different evaluation criteria have appeared in the literature, based on different measures, such as distance, information, consistency, dependence or accuracy, among others [8]. Concerning the search procedure there also exists a wide range of methods to avoid the computationally prohibitive (in the general case) exhaustive search. Some of them determine the optimal feature subset under certain assumptions, such as the *Branch and Bound* algorithm, which needs monotonicity of the evaluation criterion. Other methods seek for a suboptimal solution heuristically. Rather well-known methods of this type are the sequential ones, where features are deleted from (or added to) the partial solution at every step. The simplest ones are the SBS and the SFS procedures. SBS is a top-down process. Starting from the complete set of available features, one feature is deleted at every step of the algorithm, chosen on the basis of which of the available candidates gives rise (when deleted) to the best value of the selection criterion. SFS is a bottom-up process. The procedure begins by considering each of the variables individually and selecting the one which gives the best value for the selection criterion. At every step, the feature which gives rise (when added) to the best value of the selection criterion is added to the set. It is expected that performance may improve as far as features are deleted (added), but at some point the elimination (inclusion) of further features results in performance degradation [5].

Specially important is the wrapper approach (see, for example, [6]), where the feature subset selection is done using an induction algorithm as a black box (that is, no knowledge of the algorithm is needed, just the interface). The feature subset evaluation criterion is the accuracy of the induced classifiers (which is not necessarily monotone).

III. Overtraining, FNNs and SBS

One of the main drawbacks of ML frameworks in general is the poor generalization behaviour as a consequence of overtraining. If the points in the training set are perfectly fitted, the generalization performance is usually bad, specially in real-world problems. As previously said, in addition to the possible lack of information (including both missing values and missing variables), features in real-world datasets may be noisy or useless for the problem at hand. *A priori*, when many variables are present, there may be many different solutions capable of approximating the same training set. But only a few number of these solutions will lead to good generalization. There is no reason to think that a good one will be selected by our inducer. If the system gives some importance to noisy or useless variables in order to approximate the dataset, it will use this information for new data, probably leading to poor generalization even if we try to control the overtraining. The problem is that the relevance of the variables is not known a priori. Imagine that we have collected a database where we have a useless variable, say the color of the eyes, to predict a heart disease. Unfortunalely, there is no reason to think that this variable will not be used in the training procedure to approximate the dataset. Therefore, generalization will probably be poor, even if we try to control the overtraining. The existence of many solutions consistent with the data contribute to high variance in the Bias/Variance decomposition [3]. This behaviour is more probable to happen when only a small number of examples is available.

In this context, it may be convenient to use an FS procedure. Suppose that the system uses a (very) noisy or useless variable to approximate the dataset (the color of the eyes to predict a heart disease). Without this variable, generalization should improve (or, at least, should not worsen). Our hypothesis is that this effect will be more evident if we try to fit the training set as much as possible, that is, when overtraining is highly present. An intuitive justification of this statement could be the fact that, if we try to adjust perfectly the dataset, we are forcing all the variables to be as used as possible in the resulting solution. In this situation, noisy and useless variables should emerge more clearly if generalization improves when the system is not allowed to use them. Ironically, we want to improve generalization forcing overtraining. This reasoning may be valid for both linear and non-linear classifiers.

We conjecture that this idea will allow to detect useless and (very) noisy variables for the problem at hand. After these variables have been discarded, a different approach to the problem can be performed, since we can consider (of course, with a certain probability of error) that all the remaining variables are quite useful. It does not necessarily imply that the system will not present the negative effect of the overtraining with the selected variables, but it will probably be lower. A standard technique that tries to control the overtraining (early stopping, for example) is expected to obtain better results with this reduced number of variables.

Since the evaluation criterion is the performance of the system, we decided to use a wrapper approach in order to select the resulting feature subset. Therefore, we only needed to specify the induction algorithm and the search procedure. In order to fit the dataset as much as possible, we decided to use FNNs. As it is well-known, FNNs have been shown to be universal approximators [7]. Thus, they are an appropiate induction framework to fit the data as required. Among all the existing FS techniques, we decided to use a standard SBS procedure, so that it could be possible to identify noisy and useless variables as the features deleted in the first steps of the SBS procedure, while maintaining all the possible interactions among the initial set of variables (standard SFS, for example, does not satisfy this property).

Algorithm

Let V_1 the full set of N_f features
for $N = 1$ **up to** $N_f - 1$ **do**
  for each $v \in V_N$ **do**
    Set $V = V_N - \{v\}$
    Train the network with V and keep its generalization performance. The network is overtrained, trying to fit the data as much as possible.
  end for
  Set $V_{N+1} = V_N - \{v^*\}$ where v^* corresponds to the best performance of the network in the previous loop
end for
Return V_{N^*} where N^* corresponds to the best performance of the network in the previous loop
end Algorithm

Fig. 1. The SBS procedure with FNNs forcing overtraining at every step.

Dataset	#Var.	#Cla.	#Exa.	Missing	Source
Australian Credit	14	2	690	yes	UCI/Statlog
Ionosphere	33	2	351	no	UCI/Statlog
Sonar	60	2	208	no	UCI/Statlog
Hepatitis	19	2	155	yes	UCI/Statlog
Cleveland Heart	13	2	303	yes	Statlog
Statlog Heart	13	2	270	no	Statlog
Bupa Liver	6	2	345	no	UCI
Lung Cancer	56	3	32	yes	UCI
IIM Dataset	25	2	62	no	New

TABLE I

DESCRIPTION OF THE DATASETS. THE COLUMN '#VAR.' INDICATES THE NUMBER OF VARIABLES, THE COLUMN '#CLA.' THE NUMBER OF CLASSES, AND THE COLUMN '#EXA.' SHOWS THE NUMBER OF EXAMPLES.

IV. ALGORITHM

The proposed SBS procedure with FNNs forcing overtraining can be seen in Figure 1. It works roughly as follows. First, the parameters of the network are adjusted so as to achieve a low value of the total squared error in a reasonable number of epochs N_e. Then, the SBS procedure starts. For every variable, we train the network N_e epochs without this variable, so that the training set is "as approximated as possible" with the selected parameters. The variable such that, when deleted, gives rise to the best generalization performance, is permanently removed. This loop is repeated until only one variable remains. Typically, it is expected that performance will improve until some point where the elimination of further features results in performance degradation. This is the subset of features returned by the algorithm.

V. EXPERIMENTS

We now present the experiments performed in order to test the hypothesis presented in Section III.

A. Datasets description

1) UCI and Statlog Benchmarks: We selected several datasets from two well-known ML repositories: UCI [2] and Statlog [9]. A wide variety of problems is represented by these benchmarks, as can be seen in Table I. When the range of inputs was not normalized, we performed a linear scale transformation in $[0, 1]$. For real-valued variables, missing values were substituted by the average within the class. For discrete ones, they were substituted by the most frequent value in the class.

2) IIM Dataset: We had the opportunity to work on a real-world problem of medical diagnosis. The dataset contained the data of 62 patients suffering from Idiopathic Inflamatory Myopathies (IIMs). IIMs, specially dermatomyositis, are associated with an increased risk of cancer. Evaluation of patients for the presence of an occult malignancy is worrisome, deserves time consumption and patients are often subjected to extensive invasive investigations. Although some factors as age, sex, refractory or recurrent disease and some types of myositis specific antibodies (such as antisynthetase or anti-Mi-2) have been proposed to be related to the risk of cancer in IIM patients, conclusive studies are lacking [4].

On average, the 62 patients diagnosed of IIM in our study were followed up for 8 years in the Hospital de la Vall d'Hebron, Barcelona. The diagnosis of inflamatory myopathy was based on a strict clinical definition and histologic criteria. The input consisted of 25 variables containing clinical and laboratory data. Fortunately, there was no missing value and the values of the variables had (presumably) little noise. The target was the presence or absence of cancer. All the malignancies were registered and pathologically confirmed in the hospital. The number of patients diagnosed of cancer was 11. This low number of examples in our dataset is due to the fact that IIMs are extremely rare diseases.

Neither the noise in the data nor the absence of information were seen as severe drawbacks in the IIM dataset. In contrast, the existence of useless variables was considered our outstanding problem. As explained previously, the reason is that many of the variables were gathered without knowing exactly their importance (although guessing that they could help to give an insight of the problem). Real-valued variables were normalized with mean 0 and variance 1, whereas discrete ones were codified in a unary 1-of-C scheme. In this problem the two classes are clearly unbalanced. It was considered a major error to predict absence of cancer when this was not the case. Therefore, the sum-of-squares error function was modified to assign equal importance to every class, as in [10].

B. Experimental Setting

The training of the networks was performed with standard Back-propagation (BP) [12] in pattern learning mode (weights are modified after the presentation of each example). We used both linear and non-linear FNNs. For non-linear FNNs, and in order to reduce the computational cost, we decided to use Multi-layer Perceptrons (MLPs) with one hidden layer of units, with the sine as the activation function in the hidden layer and

Benchmark	Lin (All)	Sin (All)	Lin (SBS)	Sin (SBS)
Australian Credit	85.9%	86.0%	87.4% (8)	87.2% (7)
Ionosphere	86.9%	89.5%	92.4% (11)	92.6% (11)
Sonar	77.4%	86.1%	90.6% (25)	91.6% (20)
Hepatitis	85.3%	76.0%	92.3% (3)	94.0% (6)
Cleveland Heart	83.4%	80.7%	83.8% (5)	82.5% (3)
Statlog Heart	83.3%	81.3%	85.2% (4)	84.5% (3)
Bupa Liver	68.3%	72.4%	69.1% (4)	71.8% (4)
Lung Cancer	46.9%	36.3%	87.5% (14)	87.5% (9)
IIM Dataset	73.6%	74.5%	93.2% (12)	94.2% (9)

TABLE III

GENERALIZATION RESULTS BEFORE AND AFTER THE APPLICATION OF THE SBS PROCEDURE, EXPRESSED AS THE AVERAGE OF 5 RUNS OF A CROSS-VALIDATION PROCEDURE. THE NUMBER OF SELECTED VARIABLES IS INDICATED BETWEEN BRACKETS.

Dataset	#Hidd.	Weights Range	Epochs
Australian Credit	20	[-1.0,+1.0]	1000
Ionosphere	10	[-0.5,+0.5]	300
Sonar	35	[-1.0,+1.0]	300
Hepatitis	15	[-2.5,+2.5]	500
Cleveland Heart	20	[-1.0,+1.0]	500
Statlog Heart	20	[-0.5,+0.5]	600
Bupa Liver	20	[-2.0,+2.0]	1500
Lung Cancer	20	[-1.0,+1.0]	100
IIM Dataset	15	[-0.5,+0.5]	150

TABLE II

DESCRIPTION OF THE PARAMETERS OF THE SINUSOIDAL ARCHITECTURES.

the hyperbolic tangent in the output layer, as in [13]. Units in the hidden layer had no bias, and the momentum term was set to 0. For every dataset, the number of hidden units, the initial range of weights in the hidden layer and the number of epochs trained can be seen in Table II. The learning rates were adjusted for every particular dataset to fit the data as much as possible.

C. Results

The generalization results for every dataset can be seen in Table III as the average of 5 runs of a cross-validation procedure. For the Lung Cancer and the IIM datasets a leave-one-out method was applied, whereas the rest of the datasets were tested with a 10-fold cross-validation.

For every dataset, the following experiments were performed. First, an early stopping procedure was run with the whole set of variables. These results can be seen in the columns 'Lin (All)' for linear networks and 'Sin (All)' for sinusoidal ones. Second, SBS was applied to every dataset (both for linear and sinusoidal FNNs), as explained in Section III and Section IV. The results with the set of variables selected by SBS are shown in the columns 'Lin (SBS)' for linear networks and 'Sin (SBS)' for sinusoidal ones. The number of selected variables is indicated between brackets in the same columns.

To the best of our knowledge, the results obtained for UCI and Statlog datasets are as good as most of previous published results for FS procedures with these benchmarks. For the IIM dataset, the results are also excellent. In order to have only a brief reference, the best results and the results of BP in [9] (when cross-validation tests were available) are included in Table IV. Although these results are probably out of date, they were performed with the same methodology than ours. It should also be noted that all our results are obtained with MLPs trained with BP, although there may be problems better suited for other kind of classifiers or learning methods. In [9] some of the datasets were tested with more than 20 different methods, and the results of BP were far from the best, as can be seen in Table IV. It is not clear, however, whether a FS procedure has been applied in the results shown in [9] or not.

Benchmark	Best (Statlog)	BP (Statlog)
Australian Credit	86.9%	84.6%
Sonar	87.5%	84.7%
Hepatitis	92.9%	82.1%
Cleveland Heart	85.1%	81.3%
Statlog Heart	83.6%	65.6%

TABLE IV

STATLOG RESULTS [9] FOR THE DATASETS WHERE CROSS-VALIDATION TESTS WERE AVAILABLE.

Anyway, the important point is the fact that there has been, on average, a great improvement after the FS procedure has been applied, leading to a very good performance. In our opinion, this means that the system has eliminated noisy and useless variables. Specially remakable is the behaviour of linear networks, which are able to obtain very good results, although usually with more variables than non-linear networks. This supports the independence of our hypothesis from a particular learning model.

Figure 2 shows, for every dataset, how varies the percentage of correct examples in the training and test set with respect to the number of eliminated variables in the SBS procedure. Basically, two kinds of behaviour can be observed. There exist

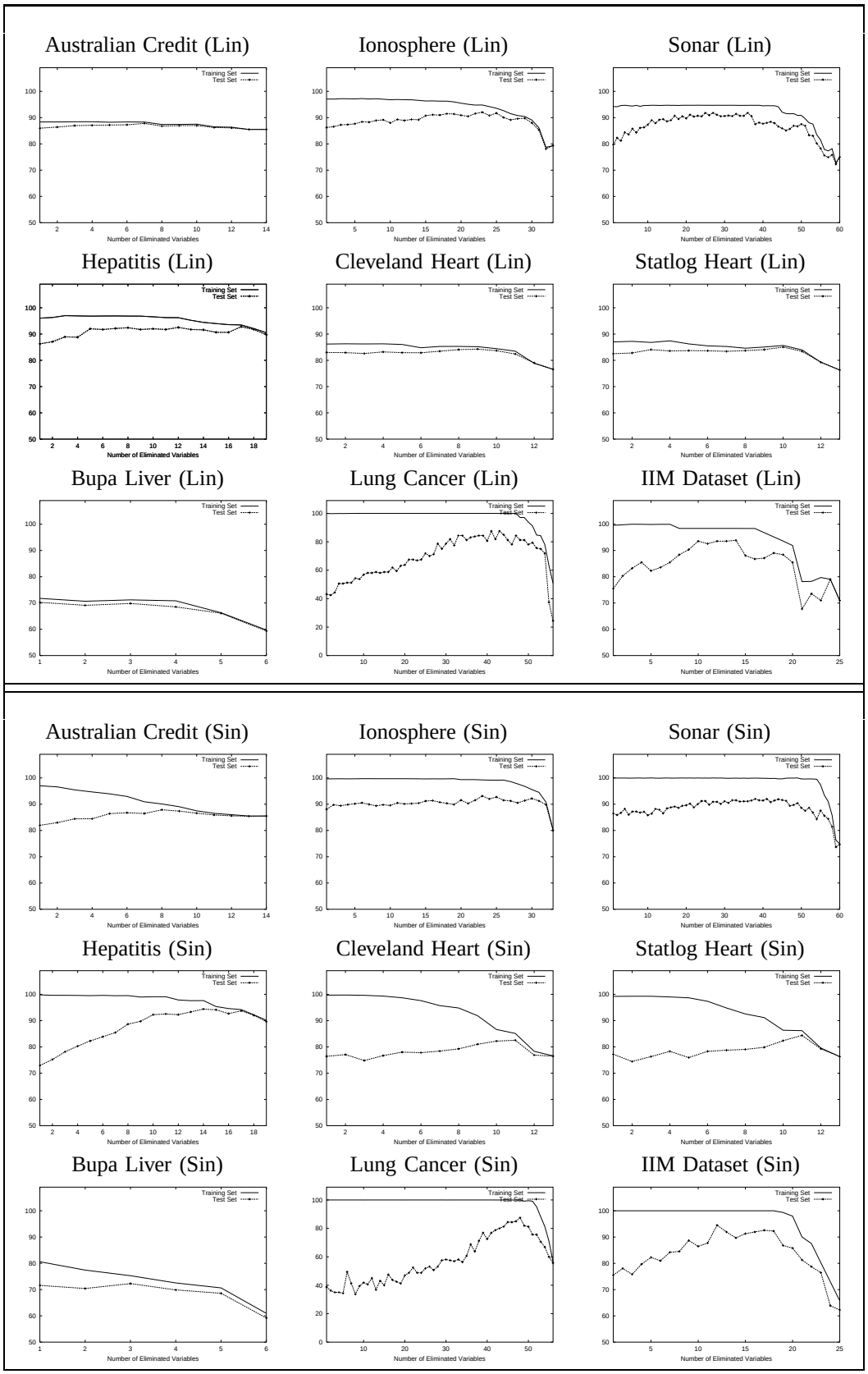

Fig. 2. Percentage of correct examples in the training and test set with respect to the number of eliminated variables in the SBS procedure for linear (top) and sinusiodal (bottom) FNNs.

problems where the performance hardly improves. The Bupa Liver dataset is probably the most representative example of this type. On the other hand, there exist datasets where the curve of performance has a more desired behaviour, as the Lung Cancer or Hepatitis datasets: test error usually improves as far as variables are being eliminated, up to a point where starts to degenerate. It is interesting to note that these two datasets are very different with respect to the number of examples and variables.

There is a surprising coincidence when comparing linear and non-linear networks in Figure 2. The respective test curve shapes are very similar for the same problem. This fact could also may be indicating that the method is quite independent of the particular learning system used.

For some problems (Bupa Liver or Australian Credit, for example), it could be possible that important variables are lacking, since the system fails to approximate the training set from the first steps of the SBS procedure.

VI. Conclusions and Future Work

In this paper it is experimentally shown how overtraining can be used, together with FS, to improve generalization performance in many cases. The idea lies in the hypothesis that when overtraining is present, the variables are forced to have too much importance in the resulting solution. In this situation, the effect of noisy and useless variables should be more evident. The proposed methodology is based on perform FS forcing overtraining, so that it can be easier to detect noisy and useless variables. In our experiments, we used FNNs to perform SBS within the wrapper approach.

There exist several issues that can be improved. For example, the parameters (number of hidden units, learning rates, etc) of the network should be readjusted after the elimination of every variable in the SBS procedure. Although it is expected that the new parameters should be quite similar to the previous ones, they may not be necessarily equal. In this sense, an automatic selection of the parameters would be desirable [11]. A different training algorithm may also be used in order to reduce the computational cost. Larger datasets with a larger number of variables could need a different treatment.

More experiments are needed to confirm the hypothesis. For example, suppose that we select the variable to eliminate in the SBS procedure performing early stopping instead of forcing overtraining. According to our hypothesis, our method would detect noisy and useless variables better than this approach.

In addition, there exist several issues that can be modified. Although we have used FNNs in our experiments, we think that our hypothesis is independent of a particular learning model. The only requirement is the capacity of fitting the training set. In a similar way, other FS procedures, instead of SBS, could be used, such as hybrid or bidirectional methods applying SBS and SFS alternatively or simultaneously.

Acknowledgment

This work was supported by Consejo Interministerial de Ciencia y Tecnología (CICYT), under project DPI2002-03225.

References

[1] Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press Inc., NY.

[2] Blake, C.L. and Mertz, C.J. (1998). UCI Repository of Machine Learning Databases. University of California, Irvine, Department of Information and Computer Science. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[3] Geman, S., Bienenstock, E. and Doursat, R. (1992). Neural Networks and the Bias/Variance Dilemma. *Neural Computation* 4, 1-58.

[4] Hill, C.L, Zhang, Y., Sigurgeirsson, B., Pukkala, E., Mellemkjaer, L., Airio, A., Evans S.R. and Felson, D.T. (2001). Frequency of Specific Cancer Types in Dermatomyositis and Polymyositis: A Population-based Study. *Lancet* 357, 96-100.

[5] Kittler, J. (1986). Feature Selection and Extraction. In *Handbook of Pattern Recognition and Image Processing*, Academic Press, New York, 59-83

[6] Kohavi, R. and John, G.H. (1997). Wrappers for Feature Subset Selection. *Artificial Intelligence* 97 (1-2), 273-324.

[7] Leshno, M., Lin, V.Y., Pinkus, A. and Schocken, S. (1993). Multilayer Feedforward Networks With a Nonpolynomial Activation Function Can Approximate Any Function. *Neural Networks* 6, 861-867.

[8] Liu, H. and Motoda, H. (1998). *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers.

[9] Michie, D., Spiegelhalter, D.J. and Taylor, C.C. (1994). Machine Learning, Neural and Statistical Classification. Ellin Horwood. Results available at http://www.phys.uni.torun.pl/kmk/projects/datasets-stat.html.

[10] Parikh, C.R, Pont, M.J. and Jones, B. (1999). Improving the Performance of Multi-layer Perceptrons when Limited Training Data are Available for Some Classes. *9th International Conference on Artificial Neural Networks*, 227-232.

[11] Romero, E. and Alquézar, R. (2002). A New Incremental Method for Function Approximation using Feed-forward Neural Networks. *International Joint Conference on Neural Networks*, vol 2, 1968-1973.

[12] Rumelhart, D.E., Hinton, G.E. and Williams, R.J. (1986). *Parallel Distributed Processing*, Vol 1. MIT Press.

[13] Sopena, J.M., Romero, E. and Alquézar, R. (1999). Neural Networks with Periodic and Monotonic Activation Functions: A Comparative Study in Classification Problems. *9th International Conference on Artificial Neural Networks*, 323-328.