General purpose Input Variables Extraction: A Genetic Algorithm based Procedure GIVE A GAP

Silvia Cateni, Valentina Colla, Marco Vannucci
Scuola Superiore S.Anna
Viale Rinaldo Piaggio, 34. 56025 Pontedera (PI)

s.cateni@sssup.it, colla@sssup.it, mvannucci@sssup.it

Abstract— **The paper presents an application of genetic algorithms to the problem of input variables selection for the design of neural systems. The basic idea of the proposed method lies in the use of genetic algorithms in order to select the set of variables to be fed to the neural networks. However, the main concept behind this approach is far more general and does not depend on the particular adopted model: it can be used for a wide category of systems, also non-neural, and with a variety of performance indicators. The proposed method has been tested on a simple case study, in order to demonstrate its effectiveness. The results obtained in the processing of experimental data are presented and discussed**.

Keywords-genetic algorithm; variables selection; neural network.

I. INTRODUCTION

The variables selection is an important task in the design of systems for the simulation and control of real processes. In function approximation applications, when a parametric model is applied to reproduce the relationship between a set of input variables and some output variables and an experimental database is provided for model identification purposes, it is essential to find the subset of input variables that are actually correlated with the variable(s) to predict. The issue of variables selection has been deeply investigated in literature for-prediction problems [1-2], but also for classification [3-7] and clustering [8] tasks.
A new approach based on the fusion of delta test method and genetic algorithm is presented in [9]. This method assigns to each variable a parameter to determine a subset of variables which is representative of a given function.
The selection of the relevant input variables is an important step to build a model that is capable to provide satisfactory performance [10].
On the other hand, this issue strictly deals with the knowledge acquisition on a process or a phenomenon that is not yet deeply understood. For instance, in the industrial field, commonly a huge number of measurements related to the process and to the partially manufactured product are usually collected by the sensors distributed along the production chain as well as through specific analyses and tests, and it is of utmost interest to point out which variables actually affects e.g. the features and the quality of the final product or the occurrence of malfunctioning [11-12].
In this paper an automatic approach to variables selection is described, that exploits genetic algorithm and artificial intelligence techniques. The proposed method is called : GIVE A GAP (General purpose Input Variables Extraction: A Genetic Algorithm based Procedure). The genetic algorithm is used to generate several combinations of the input variables through the mechanisms of natural selection and genetic recombination [13], by adopting a fitness function aimed at pointing out the input variables that mainly affects a target to be predicted. It is important to underline that this approach is far more general and does not depend on the particular adopted model: it can be used for a wide category of systems (also for non-neural systems) and with a variety of performance indicators.
The main aim of the proposed method is not only the development of a system providing acceptable performance in the prediction or classification task it is designed for, but also the detection of the input variables that mainly affect the considered process or problem, thus it is a data mining algorithm, in the sense that it actually extracts knowledge from data.
The paper is organized as follows: Sec 2 is devoted to a literature survey on the subject of variables selection. Sec 3 is describes the proposed method while in Sec 4 the results obtained in an exemplar case-study are presented and discussed. Finally Sec.5 presents some final comments and perspectives for the future work.

II. Background on Variable Selection

In applications where the correlation among a (usually large) number of independent variables and a set of variables that are supposed to depend on the previous ones is investigated, an important task is the selection of the relevant input variables. This problem can be faced with two different approaches [14-15]:

- *Feature extraction*
- *Variables selection.*

Feature extraction consists in a transformation of the original variables to create other features that are more significant.

A popular technique which is widely used for feature extraction is the Principal Component Analysis (PCA) [16]. PCA consists of a linear transformation of the variables. A new reference system is obtained where the transformed variables are sorted in decreasing order by respect to their influence on target variance.

The feature or variables selection consists in the selection of the a subset of the available initial variables.

Variable subset selection methods can be divided in three main different categories [17]:

- Wrappers.
- Filters.
- Embedded.

Wrapper techniques, diffused by Kohavi and John [18], directly optimize the predictor ability and does not depend on the knowledge about the learning machine. This technique uses the predictive accuracy of the applied learning machine to calculate the suitability of the selected variables [19]. The disadvantage of this approach lies in the fact that, when the available dataset include many input variables, the computational burden becomes very relevant [20]. In 2005 Marono et al. developed a new wrapper method for feature selection based on ANOVA decomposition [21].

The filter methods are applied in the pre-processing phase and do not depend on the learning algorithm. The inputs are chosen by evaluating the relationships between each subset of input variables and the target [22]. A feature selection filter method is proposed in [23]. This algorithm exploits a correlation-based heuristic in order to select the suitability of the considered subset. Moreover its effectiveness is determined by using three traditional Maximum Likehood (ML) methods .

In embedded methods the variable selection is not separated from the learning phase [24] but they are directly connected. Embedded and wrapper methods have the same advantages concerning the interaction between the variable selection and the learning machine but the embedded method is computationally less complex because the selection of the variables is included into the learning phase. A novel variable selection method is shown in [25], where the most relevant features are incrementally added.

III. Method Descriptionì

In the present paper a variables selection method based on genetic algorithm [26-27] is proposed, that is applied to function approximation problems, but, in principle, can also be used for classification or clustering purposes. The model whose input variables need to be determined can be implemented with any kind of feed-forward neural network [28] and also with other artificial intelligence approaches (such as fuzzy or neuro-fuzzy inference systems).

The final aim of the GIVE A GAP approach is to optimize the prediction performance, that is evaluated in terms of the Normalized Square Root Mean Square Error (NSRMSE). This index has been introduced by [29] and is given by the following formula:

$$NSRMSE\ \% = \frac{1}{\sigma_y}\sqrt{\frac{\sum_{i=1}^{N}(\hat{y}(i)-y(i))^2}{N}}\cdot 100 \qquad (1)$$

Where σ_y is the standard deviation of target in the test set (y), $\hat{y}$ is the output of the model and N is the number of samples in the test set.

It must be underlined that any other kind of performance function can be adopted, depending on the particular application.

The chromosomes of the genetic algorithm are binary strings whose length is fixed and equal to the number of potential input variables: each gene is associated to an input variable. A unitary value of a gene means that the associated variable is part of the subset of input variables that is represented by the chromosome and is selected to be fed as input to the model.

The fitness function (1) is evaluated for each chromosome of the population and standard genetic operators (such as mutation and crossover) are applied.

The mutation operator randomly extracts a bit of the selected chromosome and switches it from 0 to 1 or vice versa. An example is shown in Figure 1.

The crossover operator selects each gene of the son chromosome by randomly taking genes from each one of the two parent chromosomes as shown in Figure 2 where the first two chromosome are the parents while the third one represents the son. The dotted arrows correspond to the first parent while the continuous ones correspond to the second one.

The available dataset has been divided in three subsets: 60% of the data are used for the network training, 20%

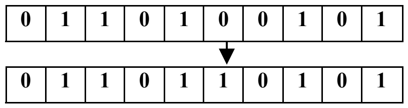

Figure 1. Mutation operator example

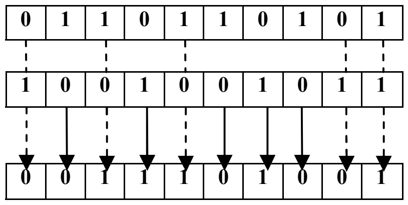

Figure 2. Crossover operator example: the line identifies the origin of each son gene.

as validation set and the remaining 20% of observations is used for testing the network. The use of a validation set drastically lowers the training time of the network and aims at the improvement of its generalization capabilities by avoiding the overfitting. The fitness function to minimize is the average NSRMSE achieved on the test set at the end of at least five different and independent training procedures. This repetition aims at counterbalancing the lack of repeatability in the training of neural networks even if they are characterized by the same structure, the same number of neurons and are trained on the same dataset. It is therefore an attempt to avoid the possibility to award a wrong variable combination that provided by chance a good result or, vice versa, to punish a good combination due to an unlucky result.

Figure 3 represents the whole flow diagram of the developed genetic algorithm.

The present analysis is limited to the prediction of a single output variable. The adopted neural prediction model is a classical two layer perceptron [30] with *k* inputs, *n* neurons in the hidden layer, that are characterized by a sigmoidal activation function, and a single linear neuron in the output layer (see Fig. 4).

The number of neurons in the hidden layer is automatically determined for each combination of input variables on the basis of the number such: in particular, a well-known empirical formula [31] is applied, which states that the number of free parameters cannot be greater than fourth or fifth part of the number N_c of input-output patterns contained in the training dataset. For the selected neural structure, the number of free parameters is equal to *(kn+2n+1)*. Therefore the following inequality:

$$(kn + 2n + 1) \le \frac{N_c}{5} \tag{2}$$

implicitly provides an upper bound for *n*. Once k is determined as the sum of unitary bins in the selected chromosome, the system automatically fixes the number of neurons in the hidden layer according to the following formula:

$$n = \text{int}\left(\frac{N_c}{5k+10}\cdot 0,7\right) \tag{3}$$

where *int(•)* means the biggest integer value lower or equal to the value within parenthesis and the factor 0.7 is used to reduce the number of neurons in the hidden layer with respect to the empirically estimated maximum number, according to what is normally done in the practice to the aim of improving the generalization capabilities of the network and to reduce the training process.

The genetic algorithm stops when either a fixed maximum number of iterations is reached (the maximum number of iterations before the algorithm halts is setted to 500) or the a plateau of the fitness function is achieved. The winner chromosome provides the subset of input variables associated to the best value of the fitness.

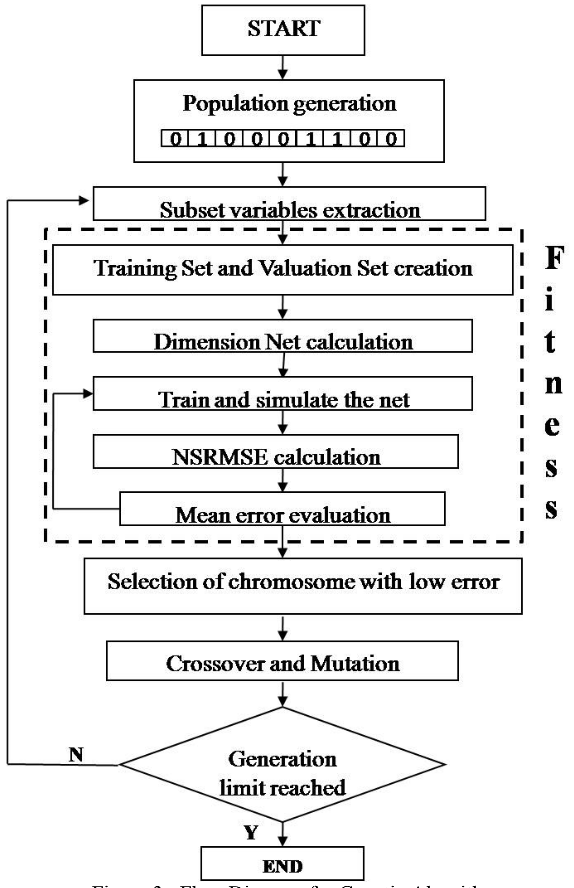
START
Population generation
0 1 0 0 0 1 1 0 0
Subset variables extraction
Training Set and Valuation Set creation
Dimension Net calculation
Train and simulate the net
NSRMSE calculation
Mean error evaluation
Fitness
Selection of chromosome with low error
Crossover and Mutation
Generation limit reached
N
Y
END

Figure 3. Flow Diagram for Genetic Algorithm

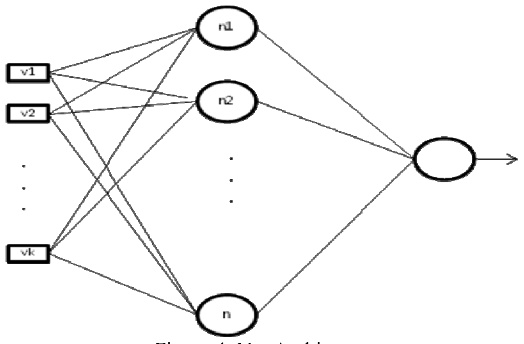

Figure 4. Net Architecture

However the problem of genetic algorithm is that the initial population is randomly chosen and also some selection steps include a component of randomness. Thus it can happen that a variable is included in the winner chromosome although it is not correlated with the target to predict.

In order to solve this problem, the genetic algorithm is run several times (typically ten) and all winner chromosomes are stored. In a final step, the presentation frequency of each variable in the winning chromosomes is calculated and only the variables that are presented in more than the 90% of the winning chromosomes are considered as actually affecting the output variable.

IV. Results

In order to test the effectiveness of the proposed approach, a database including 1000 samples and 15 independent variables v_i (with $1 \leq i \leq 15$) has been created. Three different targets, as non-linear combinations of input variables, have tested, moreover random noise with gaussian distribution was added to the target variable in different proportions with respect to average value of the target to predict as specified in Tab. 1.

Table 1 shows for each target the selected variables and the prediction error in term of NSRMSE. It is clear that the error increases with increasing levels of the additive noise.

Table 1 . Results of different functions and addive noise levels.

Theoretical function	Addive Noise level	Selected Variables	NSRMSE (%)
$\sin(2 V_2) + (V_4)^2$	2%	2, 4	4 %
$\sin(2 V_2) + (V_4)^2$	5%	2, 4	12%
$\sin(2 V_2) + (V_4)^2$	10%	2, 4	23%
$e^{V_{10}} \cdot 3V_6 + V_8$	2%	6, 8, 10	3%
$e^{V_{10}} \cdot 3V_6 + V_8$	5%	6, 8, 10	9%
$e^{V_{10}} \cdot 3V_6 + V_8$	10%	6, 8, 10	17%
$\sqrt{V_1} \cdot e^{(V_7 + V_{13})} + V_4^2$	2%	1, 4, 7, 13	4,8%
$\sqrt{V_1} \cdot e^{(V_7 + V_{13})} + V_4^2$	5%	1, 4, 7, 13	10%
$\sqrt{V_1} \cdot e^{(V_7 + V_{13})} + V_4^2$	10%	1, 4, 7, 13	19%

Table 2 shows the details of the variables which are selected by each complete run of the genetic algorithm considering the example reported in the first row of Tab.1. The columns correspond to the iterations while the rows correspond to the input variables. For each iteration, a cross has been put in correspondence of the selected variables. It clearly appears that only variables v_2 and v_4 are present in all the winning subsets.

The prediction result obtained with the two layer neural network with 35 hidden neurons, according to eq (3), and the two extracted variables as inputs is shown in Figure 5 and Figure 6.

In particular, Figure 5 represents the plot of the predicted versus measured target while Figure 6 shows the surface of the predicted target function with respect to the selected input variables.

Table 2. Results of 10 GA runs

	1	2	3	4	5	6	7	8	9	10
V_1									X	X
V_2	X	X	X	X	X	X	X	X	X	X
V_3	X				X	X		X	X	
V_4	X	X	X	X	X	X	X	X	X	X
V_5				X	X	X	X	X	X	X
V_6	X	X	X					X		
V_7	X	X	X		X	X	X		X	X
V_8	X						X		X	
V_9		X			X	X	X	X	X	X
V_{10}	X	X	X	X		X	X	X		
V_{11}					X	X		X		X
V_{12}		X				X				
V_{13}				X	X	X			X	X
V_{14}	X		X	X		X	X	X	X	X
V_{15}	X		X	X				X	X	

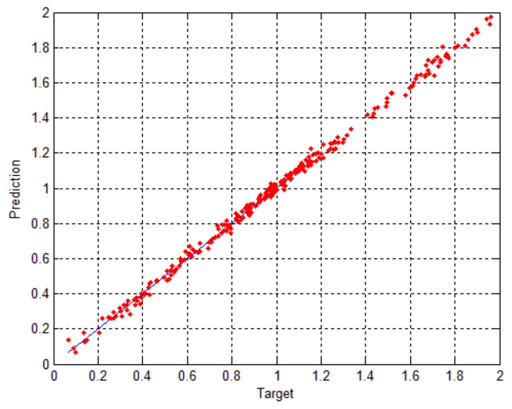

FIGURE 5. PREDICTION RESULT USING THE SELECTED VARIABLES.

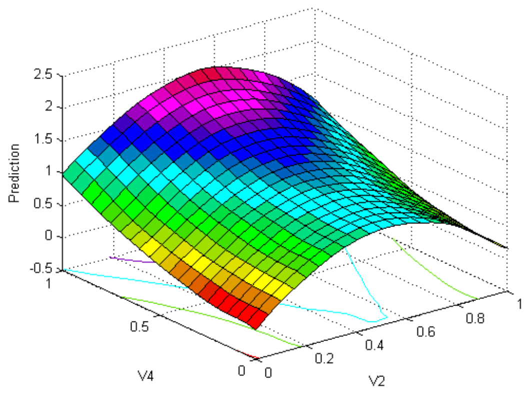

FIGURE 6. PREDICTION RESULT USING THE SELECTED VARIABLES

The achieved value of the NSRMSE on the validation set is about 4%.

By running only once the genetic algorithm and by a-priori fixing the neural network dimension (as in [1]), the order of the obtained NSRMSE decreases to 5% but, most of all, the winning subset of input variables does not contain only v_2 and v_4.

In fact, by considering the first column of Tab. 2, the first run of the GA selects as inputs the following variables: v_2, v_3, v_4, v_6, v_7, v_{10}, v_8, v_{10}, v_{14}, v_{15} the variables that really affect the target prediction are not pointed out.

If the purpose of the developed work is not only to design a good and reliable predictor of the target variable, but also to acquire knowledge on the input variables that actually affect the target, this latter aspect is essential and the proposed method showed to be more effective than [1].

The price to be paid for the achievement of this result is an increase in the computation time (in the proposed examples the computation time is about 20 minutes). However this method has been elaborated to be performed *una tantum* off line to the aim of extracting the variables that mainly affect the target. For this reason this limit is acceptable but future work will concern the algorithm optimization in order to reduce the computation time and improve its on-line implementation.

V. CONCLUSION AND FUTURE WORK

A very effective algorithm for feature selection is presented in a context of function approximation applications (although its main idea is applicable to several common contexts such as classification or clustering).

This approach is capable not only to improve prediction accuracy but also to select only the variables that really affect the phenomenon under consideration, by contributing to improve its knowledge.

The future work is focused on the algorithm optimization, in order to improve its computational efficiency especially for very large databases used in real applications.

VI. REFERENCES

[1] D. A. Sofge and D.L. Elliot, “Improved Neural Modeling of Real-World Systems using Genetic Algorithm based Variable Selection”, *Proocedings Conference on Neural Networks & Brain*, Oct 1998.

[2] D.A. Sofge, “Using Genetic Algorithm Based Variable Selection to Improve Neural Network Models for Real-World Systems”, Proocedings of the 2002 International Conference on Machine Learning & Applications, 2002.

[3] N. Kwak, C.H. Choi, “Input feature selection for classification problems”, IEEE *Trans. On Neural Networks*, Vol. 13, pp. 143-159, 2002.

[4] R.Battiti, “Using mutual information for selecting features in supervised neural net learning”, IEEE *Trans. On Neural Networks*, Bruges,Belgium, 2005.

[5] L. Li, R.D. Cook and C.J. Nachtsheim, “Model-free variable selection”, *J.R. Statist. Soc.*B, 67, Part2, pp. 285-299, 2005.

[6] L. Xu and W.J. Zhang, “Comparison of different methods for variable selection”, *Analytica Chimica Acta*, 446, pp.477-483, 2001.

[7] J.Y. Lin, H.R Ke, B.C. Chien, W.P. Yang, “Classifier design with feature selection and feature extraction using layered genetic programming*”, Expert System with Applications*, N. 34, pp. 1384-1393, 2008.

[8] S. Wang and J. Zhu, “Variable selection for model-based high dimensional clustering and its application to microarray data”, *Biometrics* N.64, pp.440-448, June 2008.

[9] A. Guillèn, D.S. Lendasse, F. Mateo and I. Rojas, "Minimising the delta test for variable selection in regression problems", International Journal High Performance Systems Architecture, Vol. 1, N°4, 2008.

[10] J. Hao, "Input selection using mutual information – applications to time series prediction", Helsinki University of Technology, M.S. thesis, Dep. Of Computer Science and Engineering, September 2005.

[11] V. Colla, R. Valentini, G. Bioli: "Mechanical properties prediction for Aluminium-Killed and Interstitial-Free steels," Revue de Metallurgie, Special Issue JSI Dicembre 2004, pp.100-101.

[12] V. Colla, M. Vannucci, S. Fera, R. Valentini: "Ca-treatment of Al-Killed steels: inclusion modification and application of Artificial Neural Networks for the prediction of clogging,", Proc. 5th European Oxigen Steelmaking Conference EOSC'06, 26-28 June 2006, Aachen, Germany, pp. 387-394.

[13] L.Y. Zhai, L.P. Khoo and S.C. Fok, "Feature extraction using rough set theory and genetic algorithms – an application for the simplification of product quality evaluation", *Computers and Industrial Engineering*, N.43, pp. 661-676, 2002.

[14] G. Castellano and A.M. Fanelli, "Variable selection using neural-network models", *Neurocomputing,* 31, pp.1-13, 2000.

[15] T.S. Lin and J.Meador, "Statistical feature extraction and selection for IC test pattern analysis", ISCAS, May 1992.

[16] I.T. Jolliffe, "*Principal Component Analysis",* Springer Series in Statistics, 2nd ed. Springer, New York, 2002.

[17] I.Guyon and A. Elisseeff, "An introduction to Variable and Feature Selection", *Journal of Machine Learning Research,* N. 3, pp.1157-1182, 2003.

[18] R. Kohavi and G. John, "Wrappers for feature selection", *Artificial Intelligence,* N. 97, pp.273-324, December 1997.

[19] B.M Vidyavathu and C. N. Ravikumar, "A novel hybrid filter feature selection method for data mining", *Ubliquitous Computing and Communication Journal,* Vol. 3, N° 3, 2008.

[20] A. Blum and P.Langley. "Selection of relevant features and examples in machine learning", *Artificial Intelligence,* pp.245-271, 1997.

[21] N.S. Marono, A.A. Betanzos and E. Castillo, "A new wrapper method for feature subset selection", European Symposium on Artificial Neural Network, Bruges, 27-29 April 2005.

[22] R. N. Khushaba, A. Al-Ani and A. A Al-Jumaily, "*Differential Evolution based Feature Subset Selection",* IEEE, 2008.

[23] M. A. Hall and L. A. Smith, "*Feature Subset Selection : A Correlation Based Filter Approach*", Springer, 1997.

[24] Holland, J.H. "*Adaptation in Natural and Artificial Systems*", University of Michigan press, Ann Arbor, MI, 1975.

[25] Z.Xiao, E. Dellandrea, W. Dou and L. Chen, "ESFS: A new embedded feature selection method based on SFS", Rapports de recherché, September 2008.

[26] M. L. Schmitt, *Theory of Genetic Algorithms*, Theoretical Computer Science 259: 1-61, 2001.

[27] Goldberg, D.E. "*Genetic Algorithms in Search Optimization and Machine Learning*", Addison-Wiley Pubblishing Company, Reading, MA, pp.412, 1989.

[28] Patterson, D., "*Artificial Neural Networks*", Prentice Hall, Singapore, 1996.

[29] Q. Zhang, A. Benveniste: "Wavelet Networks," *IEEE Transactions on Neural Networks*, Vol. 3, No. 6, pp. 889–898, November 1992.

[30] Fausett, L., "*Foundamentals of Neural Networks",* Prentice Hall, New York, 1994.

[31] Haykin, S., "*Neural Networks: a Comprehensive Foundation",* MacMillman Publishing, New York, 1994