Instance or Prototype Selection for Function Approximation using Mutual Information

A. Guillen[1], L. J. Herrera[2], G. Rubio[2], A. Lendasse[3], H. Pomares[2], and I. Rojas[2] *

1- University of Jaen - Dept. of Informatics
Jaen - Spain

2- University of Granada - Dept. of Computer Technology and Architecture
Granada - Spain

3- Helsinki University of Technology - Lab. of Computer and Information Science
Granada - Spain

Abstract.
The problem of selecting the patterns to be learned by any model is usually not considered by the time of designing the concrete model but as a preprocessing step. Information theory provides a robust theoretical framework for performing input variable selection thanks to the concept of mutual information. The computation of the mutual information for regression tasks has been recently proposed providing good results in feature selection. This paper presents a new application of the concept of mutual information not to select the variables but to decide which prototypes should belong to the training data set in regression problems. The proposed methodology consists in deciding if a prototype should belong or not to the training set using as criteria the estimation of the mutual information between the variables. The novelty of the approach is to focus in prototype selection for regression problems instead of classification as the majority of the literature deals only with the last one. Other element that distinguish this work from others is that it is not proposed as an outlier identificator but as algorithm that determine the best subset of input vectors by the time of building a model to approximate it. As the experiment section shows, this new method is able to identify a high percentage of the real data set when it is applied to a highly distorted data sets.

1 Introduction

The task of selecting the correct subset of input vectors that are included in a training set when classifying, approximating or predicting an output is a relevant task that, if accomplished correctly, can provide storage and computational savings and improve the accuracy of the results.

Three main approaches have been used in order to optimize the set of inputs that the training algorithm will use: *incremental*, *decremental* and *batch*. The incremental approach starts from an empty set of input vectors and defines the training set including input vectors [1, 2]. The opposite perspective is taken in

*This work has been partially supported by the Spanish CICYT Project TIN2007-60587 and Junta Andalucia Project P07-TIC-02768.

the *decremental* approach that starts considering all the input vectors available and, following a prefixed criteria, proceeds to remove the non desired instances [3, 4]. The batch method iterates several times before deleting the instance from the training set, setting a flag on the instances that could be removed in next iterations [5]. Recently, many other approaches have been proposed such as evolutive algorithms [6, 7, 8], boosting-based algorithms [9, 10], and pruning techniques [11].

The work developed in this paper is framed within the decremental approach since it considers the whole data set at the beginning. The criteria to remove the input vectors has been taken from the method used to perform feature selection. The problem of finding the adequate set of variables is quite important by the time of designing models to predict, approximate or classify input data. If the set of input data has redundant or irrelevant data, the training can result in overfitted model with poor generalization capabilities [12, 13, 14]. Furthermore, if the dimensionality is not reduced, some local approximator models suffer the curse of dimensionality, making it impossible to design accurate models.

To tackle the feature selection problem. two main streams have been followed: $filter$ and $wrapper$ methods. The filter approach consists in a preprocessing of the input data so the model is built after. The wrapper approach attempts to design the model at the same time that performs the variable selection. The concepts of entropy and mutual information make the Information theory an interesting framework for filtering approaches.

The majority of the research in prototype selection has been focused in classification problems [8], although few works aimed at solving problems for continuous output. For example [15], presents a method to select the input vectors when calculating the output using the k-Nearest Neighbors algorithm (k-NN), however, this methodology does not permit the selection of the input vectors before designing more complex models such as neural networks. In [6], a genetic algorithm is used to select both the feature and the input subsets, however, it is only suitable for linear regression models. The main problem of genetic algorithms that use binary encoding to determine if an input vector should be included considered, is that the higher the number of instances is, the longer becomes the individual, making difficult to find a good solution.

As commented before, in regression problems, the input and the output values are real and continuous values so additional techniques have to be used to estimate the probability distribution [16].Although there exists a variety of algorithms to calculate the Mutual Information (MI) between variables, this paper uses the approach presented in [17] which is adapted for continuous variables thanks to the use of the MI estimator based on the k-NN algorithm [18].

2 Prototype Selection Based on the Mutual Information

This section firstly describes the mutual information and the methodology to calculate it, then, the algorithm to perform the prototype selection is introduced.

2.1 Mutual Information

Given a single-output multiple input (MISO) function approximation or classification problem, with input variables $X = [x_1, x_2, \ldots, x_n]$ and output variable $Y = y$, the main goal of a modelling problem is to reduce the uncertainty on the dependent variable Y. According to the formulation of Shannon, and in the continuous case, the uncertainty on Y is given by its entropy defined as

$$H(Y) = -\int \mu_Y(y) \log \mu_Y(y) dy, \tag{1}$$

considering that the marginal density function $\mu_Y(y)$ can be defined using the joint PDF $\mu_{X,Y}$ of X and Y as

$$\mu_Y(y) = \int \mu_{X,Y}(x, y) dx. \tag{2}$$

Given that we know X, the resulting uncertainty of Y conditioned to known X is given by the conditional entropy, defined by

$$H(Y|X) = -\int \mu_X(x) \int \mu_Y(y|X = x) \log \mu_Y(y|X = x) dy dx. \tag{3}$$

The joint uncertainty on the $[X, Y]$ pair is given by the joint entropy, defined by

$$H(X, Y) = -\int \mu_{X,Y}(x, y) \log \mu_{X,Y}(x, y) dx dy. \tag{4}$$

The mutual information (also called cross-entropy) between X and Y can be defined as the amount of information that the group of variables X provide about Y, and can be expressed as

$$I(X, Y) = H(Y) - H(Y|X). \tag{5}$$

In other words, the mutual information $I(X, Y)$ is the decrease of the uncertainty on Y once we know X. Due to the mutual information and entropy properties, the mutual information can also be defined as

$$I(X, Y) = H(X) + H(Y) - H(X|Y), \tag{6}$$

leading to

$$I(X, Y) = \int \mu_{X,Y}(x, y) \log \frac{\mu_{X,Y}(x, y)}{\mu_X(x)\mu_Y(y)} dx dy. \tag{7}$$

Thus, only the estimate of the joint PDF between X and Y is needed to estimate the mutual information between two groups of variables.

2.2 Prototype Selection using Mutual Information

The idea that motivates this paper is: since the MI is able to let us know how much information from the output can be retrieved using the different variables starting from a set of input vectors (prototypes), it would be possible that if a significant prototype is removed from the set of input vectors, the amount of MI that could be retrieved would be decreased. On the other hand, if an insignificant prototype is deleted from the original set, the amount of MI should not be decreased. These two sentences are correct, however, there are situations where they might not be completely true. For example, if there are outliers, they will probably provide a significant amount of MI but they should not be considered. On the other hand, if the output of the system remains constant, the amount of information will not fluctuate if similar prototypes are removed.

Thus, in order to make an objective evaluation of how relevant an input vector is, it is necessary to consider the lost of MI relatively to its neighbors in such a way that, if the lost of MI is similar to the prototypes near $\vec{x}_i$, this input vector must be included in the filtered data set. The algorithm proposed to calculate the reduced set of prototypes is described below:

---

calculate the k nearest neighbors in the input space of $\vec{x}_i$ for $i = 1...n$
estimate the mutual information MI between X and the output Y
$diff$=0;
for i=1...n
  calculate the mutual information MIf_i when removing $\vec{x}_i$ from X
end
normalize MIf_i in [0,1]
for i=1...n
  for $cont$=1...α_2
    $diff$= $|MIf_i| - |MIf_{cont}|$
    if $diff > \alpha_1$
      $Cdiff$=$Cdiff$+1
  end
  if $Cdiff < \alpha_2$
    discard prototype
  else
    select prototype
  end
end

---

where α_1 is a predefined threshold that determines how different the MI should be respect the neighbors and α_2 is the number of neighbors to be considered in the comparisons.

When calculating how much of MI was lost, two approaches could be taken: 1) to calculate the MI between the complete set of variables and the output or 2) to compute the MI between each variable and the output. With the MI

RBF centers	radii	weights
0.5463	0.1698	0.4829
0.6366	0.1787	0.2096
0.5709	0.2435	-0.3246
0.9271	0.7518	0.3583
0.8638	0.1991	0.4094

Table 1: Parameters for the function f_1

estimator used in the experiments, no difference between those two approaches could be seen, however, other implementations should be analyzed.

3 Experiments

This section presents the results of applying the new algorithm to highly distorted data sets. These data sets were generated syntheticlly son it was possible to know excatly which elements were the originals and which the noisy ones.

The first experiment was performed using a one dimensional function (Figure 1 a)) that was generated using a gaussian Radial Basis Function Neural Network (RBFNN) ($e^{-\frac{||\vec{x}_k - \vec{c}_i||^2}{r_i^2}}$) using the randomly chosen parameters in Table 1.

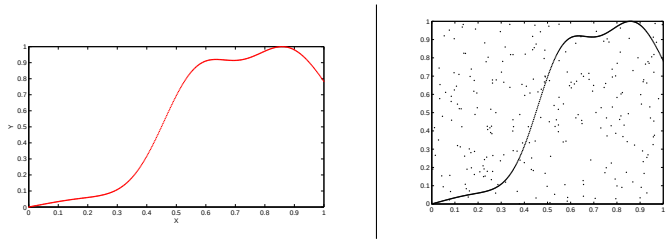

Fig. 1: a) Original target function and b) distorted data set

The original data set consisted in 400 prototypes with their corresponding output. This data set was modified adding a set of 250x2 random values in [0,1] from an uniform distribution, obtaining a new data set of 650 prototypes of dimension 1 with one output. This data set is represented in Figure 1 b).

The proposed method was applied using the value 0.01 for the threshold α_1 and 1 for α_2, obtaining a filtered data set of size 400. From the 250 elements removed, 208 were added prototypes and 42 were original prototypes, this is, the algorithm discriminated the 83.2% of the incorrect prototypes and identified the 89.5% of the original prototypes. Figure 3 depicts the results, where the circles represent the original prototypes and the stars represent the prototypes selected from the distorted data set. If a star is included in a circle, it means that the original prototype was chosen correctly.

To evaluate the utility and effectiveness of the proposed approach, several RBFNNs were designed using the three different data sets: original, distorted

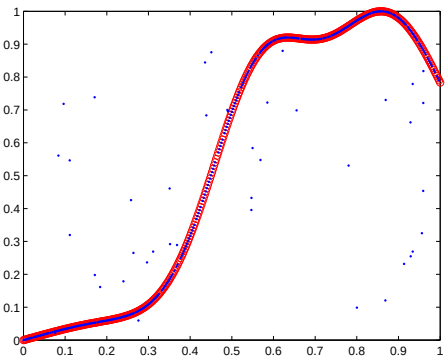

Fig. 2: Filtered data (stars) and original data (circles)

Data set	error	error over original data set
original	0.2124	0.0024
distorted	0.7425	0.4020
selected	0.3265	0.1068

Table 2: Approximation errors (Normalized Root Mean Squared Error) obtained when training the networks using the different data sets.

and filtered. The methodology to design the RBFNN was: first, initialize the centers with the ICFA algorithm [19], then apply k-NN to get a first value for the radii and then, apply a local search to make a fine tuning of these parameters. As it was expected, thanks to the prototype selection, the approximation errors (Table 2) that can be obtained are much smaller than if no prototype selection was made. Figure 3 shows the approximations of the original function by the RBFNNs generated using the distorted data a) and using the data after the prototype selection b).

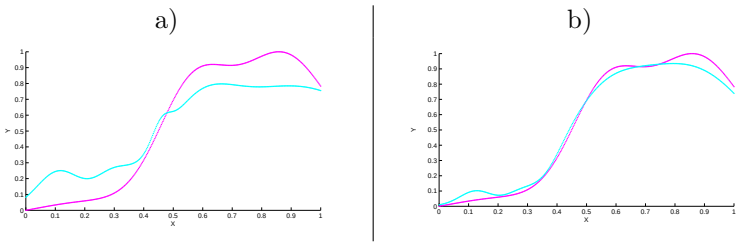

Fig. 3: Approximation made by the RBFNN trained with the data before de prototype selection (a) and after the selection (b)

Secondly, a two dimensional synthetic function f_4 (Figure 4 a)), defined in

Equation 8, was used. First, 400 input vectors were generated, then, 200 input vectors were generated using random values in [0,1] from an uniform distribution, remaining the complete data set as it is depicted in Figure 4 b).

$$f_4(x_1, x_2) = 1.9[1.35 + e^{x_1} sin(13(x_1 - 0.6)^2) e^{-x_2} sin(7x_2)] \; x_1, x_2 \in [0, 1] \quad (8)$$

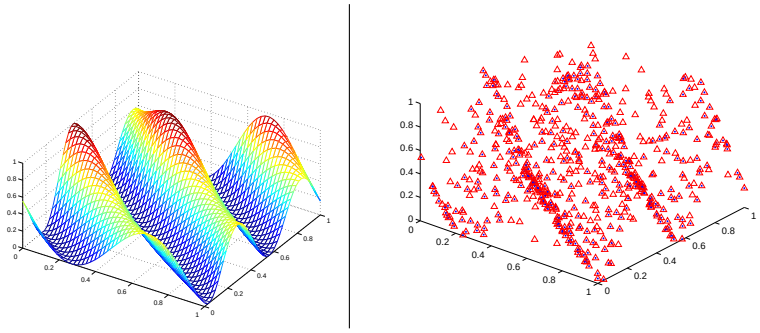

Fig. 4: a) Original target function and b) distorted data set (diamonds) with the original training data (dots)

The algorithm was applied in the same way than in the previous case using $\alpha = 0.015$ and $\alpha_2 = 2$, Figure 5 shows the results. In this occasion, the algorithm identified the 82.75% of the real input vector and the 50% of the noise ones, demonstrating again the good behavior of the proposed approach.

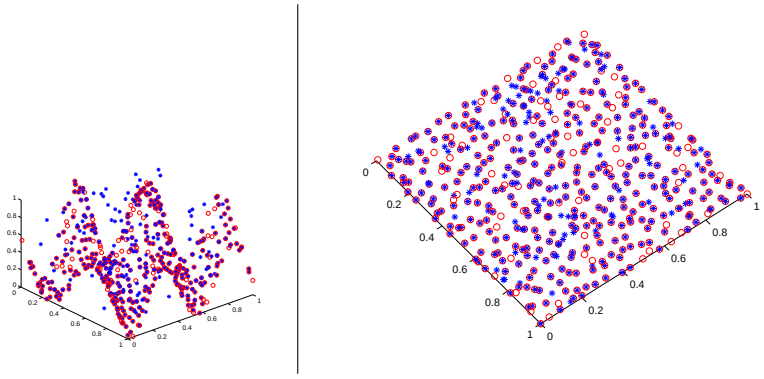

Fig. 5: Filtered data (stars) and original data (circles)

4 Conclusions and Further Work

This paper has presented a possible approach to solve the problem of selecting adequate inputs before using any model to approximate a function. This new

method is based on the concept of MI which was used before for feature selection. The main difference between the already existing approaches and the proposed one is that is oriented to data sets with a continuous output value instead of a predefined set of labels and with the global perspective that the MI provides of the complete data set. As the experiments have shown, the method seems quite effective selecting the correct prototypes with a high accuracy. Further work could be done regarding the influence of the two parameters the algorithm has, how to estimate their values building models to evaluate the quality of the selection, and also a comparison among the different ways of calculating the mutual information.

References

[1] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Mach. Learn.*, 6(1):37–66, 1991.

[2] David W. Aha. Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *Int. J. Man-Mach. Stud.*, 36(2):267–287, 1992.

[3] D. R. Wilso and T. Martinez. Reduction techniques for instance based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.

[4] D. R. Wilso and T. Martinez. Instance prunning techniques. In *Proceedings of the 14th International Conference on Machine Learning*, pages 404–411. Morgan Kaufmann Publishers, 1997.

[5] I. Tomek. An experiment with edited nearest neighbor rule. *IEEE Transactions on Systems, Man and Cybernetics*, 6:448–452, 1976.

[6] J. Tolvi. Genetic algorithms for outlier detection and variable selection in linear regression models. *Soft Computing*, 8(8):527–533, 2004.

[7] R. Baragona, F. Battaglia, and C. Calzini. Genetic algorithms for the identification of additive and innovation outliers in time series. *Computational Statistics & Data Analysis*, 37(1):1–12, July 2001.

[8] H. Ishibuchi, T. Nakashima, and M. Nii. Learning of neural networks with ga-based instance selection. *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, 4:2102–2107 vol.4, 25-28 July 2001.

[9] Richard Nock and Marc Sebban. Advances in adaptive prototype weighting and selection. *International Journal on Artificial Intelligence Tools*, 10(1-2):137–155, 2001.

[10] M. Sebban, R. Nock, and S. Lallich. Stopping criterion for boosting-based data reduction techniques: from binary to multiclass problems. *Journal of Machine Learning Research*, 3:863–865, 2002.

[11] V. B. Zubek and T. G. Dietterich. Pruning improves heuristic search for cost-sensitive learning. pages 27–34, 2002.

[12] S. Haykin. *Neural Networks*. Prentice Hall, New Jersey, 1998.

[13] E. Liitiäinen, F. Corona, and A. Lendasse. Non-parametric residual variance estimation in supervised learning. In *IWANN 2007, International Work-Conference on Artificial Neural Networks, San SebastiÃ¡n (Spain)*, Lecture Notes in Computer Science. Springer-Verlag, June 20-22 2007.

[14] E. Eirola, E. Liitiäinen, A. Lendasse, F. Corona, and M. Verleysen. Using the delta test for variable selection. In *European Symposium on Artificial Neural Networks, Bruges (Belgium)*, April 2008.

[15] J. Zhang, Y. Yim, and J. Yang. Intelligent selection of instances for prediction functions in lazy learning algorithms. *Artificial Intelligence Review*, 11:175–191, 1997.

[16] B.V. Bonnlander and A.S. Weigend. Selecting input variables using mutual information and nonparametric density estimation. In *Proc. of the ISANN*, Taiwan, 2004.

[17] L.J. Herrera, H. Pomares, I. Rojas, M. Verleysen, and A. Guillen. Effective Input Variable Selection for Function Approximation. *Lecture Notes in Computer Science*, 4131:41–50, 2006.

[18] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physics Review*, June 2004.

[19] A. Guillén, J. González, I. Rojas, H. Pomares, L.J. Herrera, O. Valenzuela, and A. Prieto. Improving Clustering Technique for Functional Approximation Problem Using Fuzzy Logic: ICFA algorithm. *Neurocomputing*, DOI:10.1016/j.neucom.2006.06.017, June 2007.